Exploration à base ontologique de données issues de patients atteints de maladies rares

Quentin Riché-Piotaix

École doctorale n° 610 :
Sciences et Ingénierie des Systèmes,
Mathématiques, Informatique

THÈSE

pour l'obtention du Grade de

DOCTEUR DE L'UNIVERSITÉ DE POITIERS

Secteur de Recherche "Bio-informatique"

présentée et soutenue publiquement par

Quentin Riché-Piotaix

le 20 décembre 2019

Exploration ontologique de données issues de patients atteints de maladies rares

Directeur de thèse : **Professeur Patrick Girard**
Co-encadrant de thèse : **Docteur Frédéric Bilan**

Jury		
Gaëlle Calvary,	Professeur	Rapporteur
Christèle Dubourg,	Maître de conférence	Rapporteur
Ladjel Bellatreche,	Professeur	Examinateur
Annie Choquet-Geniet,	Professeur	Examinateur
Sophie Lepreux,	Maître de conférence	Examinateur
Jean Muller,	Maître de conférence	Examinateur

Université de Poitiers
Laboratoire d'Informatique et d'Automatique pour les Systèmes
Téléport 2 - 1 avenue Clément Ader BP40109 86961 Futuroscope Chasseneuil Cedex - France
EA3808 NEUVACOD & Laboratoire de Génétique Biologique
service de Génétique, CHU de Poitiers

THESE

Remerciements

Je tiens tout d'abord à remercier mon directeur de thèse Patrick pour toute l'aide fournie, sans laquelle l'élaboration de ces travaux aurait été impossible, et pour m'avoir guidé et conseillé tout au long de ces 4 années.

Je remercie particulièrement mon co-directeur Fred, de m'avoir épaulé, supporté et inspiré au quotidien, et plus que tout d'avoir cru à ce projet quand moi-même je n'y croyais plus.

Mes remerciements vont aussi à Christèle Dubourg et Gaëlle Calvary de me faire l'honneur de rapporter ce travail. Je remercie également Ladjel Bellatreche, Annie Choquet-Geniet, Sophie Lepreux et Jean Muller d'avoir accepté de composer mon jury et de venir examiner ma soutenance.

Le laboratoire de génétique à été un endroit idéal pour réaliser ces travaux, j'en remercie tous les acteurs que j'ai eu le plaisir de côtoyer et plus particulièrement mes camarades de café, Barbara, Sylvie, Vincent et Valérie, pour avoir contribué significativement à mon réveil chaque matin. Un autre merci particulier à Montse pour son efficacité et sa gentillesse. Je suis très heureux de pouvoir rester parmi vous quelques années de plus.

Toujours au sein du laboratoire, je remercie tous ceux qui ont prit de leur temps pour tester mes différentes productions, m'expliquer des pratiques obscures et m'apporter de précieux conseils : Gwenaël, Tanguy, Fabienne, Dominique, et un merci spécial à Lucile pour les goûters et les salades.

Merci à mes parents d'avoir été d'un soutien sans faille et d'une aide précieuse, comme depuis toujours. Merci à Pierre et Mathilde pour leurs soutiens, aux royaumes d'Hyrule et de l'enfer. Merci à Audessa, pour la motivation quand je n'en avais plus et pour des leçons de vie que je n'oublierai jamais. Merci à Ortie pour son soutien moral et à N pour son énergie.

Enfin et surtout, Amandine, la vie à tes côtés est un tel bonheur que tout me paraît plus facile. Je t'aime.

Table des matières

Liste des figures

Liste des tableaux

Chapitre 1

Introduction

Depuis l'arrivée du séquençage nouvelle génération ou *Next Generation Sequencing* (NGS), auparavant appelé séquençage haut-débit, la génétique est entrée dans une nouvelle ère. Il est désormais possible de séquencer un génome pour un coût proche de 1000 dollars. Malgré ce coût en réactif assez faible, le génome n'est que très rarement utilisé dans un but diagnostic en France, dû à l'investissement nécessaire dans le matériel dédié au séquençage et à l'analyse, qui est de l'ordre du million d'euros, ainsi qu'à la très grande complexité d'interprétation des données générées. En revanche, l'analyse de l'exome, séquençage de l'ensemble des parties codantes des 23 000 gènes du génome humain, est désormais pratiquée en routine par les laboratoires de génétique, malgré le retard des structures ministérielles qui considèrent encore cette analyse comme étant de la recherche scientifique. Ce compromis permet d'obtenir une quantité importante de données interprétables pour un investissement matériel bien moindre. Cependant, même si la quantité de données générées par le séquenceur est bien plus petite que pour un génome complet, l'analyse d'un exome requiert tout de même des étapes bio-informatiques longues pour rendre les données interprétables. Ces étapes sont souvent condensées par les bio-informaticiens dans des "*pipelines*" d'analyse, qui consistent en une suite d'outils et d'opérations permettant de passer des séquences brutes à une liste de variations analysables par le médecin généticien, qui va en extraire la/les variation(s) causant la pathologie du patient. L'ensemble du cheminement de l'analyse d'un exome est résumé dans la figure 1.1.

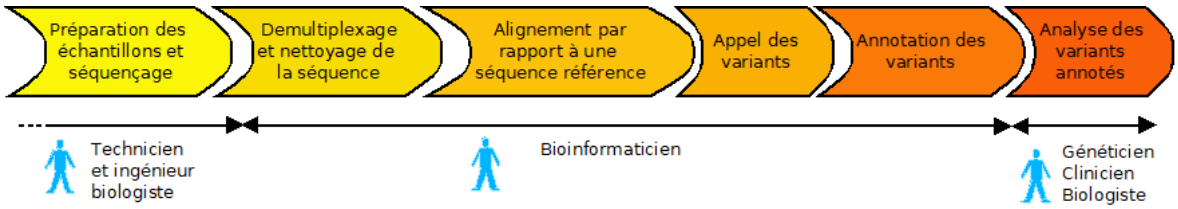


FIGURE 1.1 – Processus d'analyse d'un exome

Le séquençage est une technique longue et complexe, qui peut être robotisée. Il s'agit de préparer les échantillons d'Acide DésoxyriboNucléique (ADN) des patients afin de ne garder que les parties du génome que l'on souhaite séquencer. Les séquences sont identifiées en ajoutant un index qui joue le rôle de "code barre" sur chaque fragment d'ADN, ce qui permet de reconnaitre le patient d'origine pour chacune des séquences. L'ADN de plusieurs patients ainsi identifiés est rassemblé dans un seul tube de façon à ce que la quantité d'ADN de chacun des patients soit équivalente. Ce tube contenant l'ensemble des ADN est ensuite mis dans le séquenceur, dans une *flow cell* (que l'on pourrait traduire par "cassette de séquençage"), qui sert de support à la réaction d'amplification et de séquençage. Cet automate a pour charge de générer la séquence d'ADN sous forme de fichiers plats numériques. Ces fichiers sont très volumineux, plusieurs dizaines de Gigaoctets, d'où le besoin de structures informatiques capables de les traiter. Les serveurs informatiques utilisés présentent souvent des quantités de *Random Access Memory* (RAM) très importantes, un nombre élevé de processeurs et surtout un espace de stockage de grande taille : plusieurs dizaines de Téraoctets.

Les *pipelines* bio-informatiques effectuent 4 grandes étapes. La première étape est le démultiplexage et le nettoyage des séquences. Le séquenceur ne produit pas un résultat sous la forme d'une unique chaîne d'ADN, mais sous la forme de millions de petits fragments de 150 nucléotides appelés "lectures" en français, mais on préférera le terme bien plus employé de *reads*. Les données sont dupliquées de nombreuses fois, les *reads* se recouvrant les uns les autres dans le but que chaque région d'intérêt soit lue par le plus grand nombre de *reads* possibles. Le nombre de fois où une base est séquencée est appelée profondeur de séquençage. Une profondeur de séquençage élevée permet d'assurer une stabilité statistique et donc d'augmenter la confiance dans les résultats et de limiter les artéfacts de séquençage, qui sont des erreurs imputables au fonctionnement de la technique. Certains peuvent être évités, d'autres sont

éliminés par des logiciels dédiés, mais il est important d'en tenir compte dans les protocoles d'analyses. Un schéma récapitulatif adapté de Letienne (Letienne, 2019) est présenté figure 1.2.

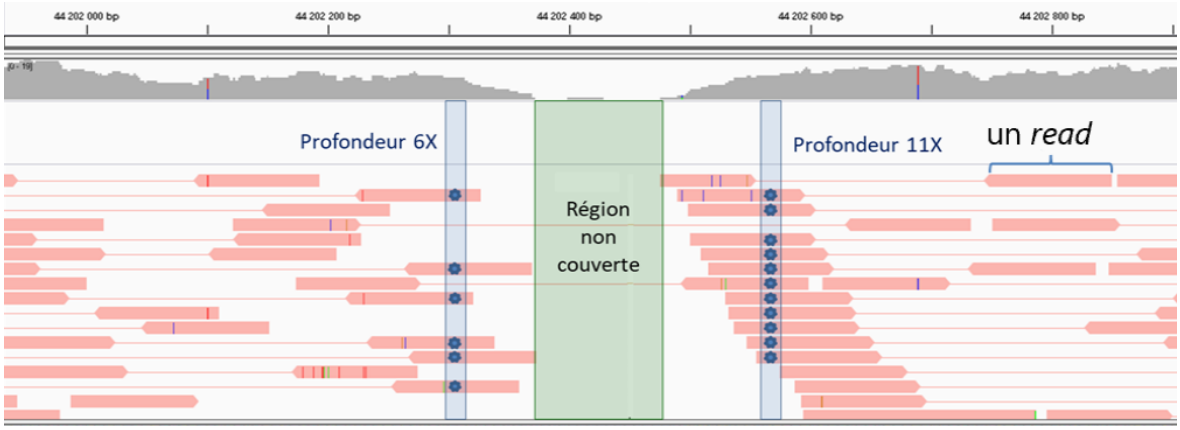


Figure 1.2 – Schéma illustrant la notion de profondeur de séquençage

1.1 Principales étapes de l'analyse

L'analyse bio-informatique peut se diviser en 4 étapes principales.

Étape 1 La première étape bio-informatique est de répartir les *reads* par patients, car tous les patients sont séquencés ensemble. Chacun des *reads* est préfixé par son index, permettant donc de reconnaitre à quel patient il appartient. Un premier tri de l'ADN par patient est ainsi réalisé. Les séquences sont ensuite nettoyées, cela signifie que les identifiants désormais inutiles sont enlevés, ainsi que d'autres séquences artificielles nécessaires au séquençage et ajoutées au cours de l'expérimentation en laboratoire. Cette phase de nettoyage est longue car les fichiers sont encore très volumineux, les données étant encore très dupliquées.

Étape 2 L'étape suivante est une étape d'alignement. Chaque *read* est aligné contre la séquence d'un génome humain de référence afin de déterminer dans quelle région génomique il se situe. C'est une étape coûteuse en calculs : il faut retrouver où, parmi 3 milliards de paires de bases, s'aligne le mieux une suite de 150 nucléotides et il faut réaliser cette opération plusieurs millions de fois. Cependant, cette étape s'est beaucoup raccourcie au cours des années avec la perfection des outils informatiques et notamment l'apport des graphes de De Bruijn (Compeau, Pevzner et Tesler, 2011) ou encore des accélérations *Graphics Processing Unit* (GPU) (Aluru et Jammula, 2013), qui utilisent les ressources des processeurs propres aux cartes graphiques pour accélérer le processus en profitant de l'architecture très parallèle de ces composants qui s'adapte parfaitement à ce type de calcul. Cet alignement permet de mieux compresser les informations, les fichiers sont donc considérablement allégés en taille (quelques Gigaoctets contre plusieurs dizaines auparavant).

Étape 3 L'objectif de la troisième étape est alors de détecter les variations entre les séquences des patients et l'ADN de référence. Cette étape s'appelle le *Variant call* qui pourrait se traduire par "l'appel des variations". Des outils dédiés existent, qui listent toutes les différences entre la séquence produite et celle de référence. Cette étape n'est pas aussi simple qu'elle peut le paraître, car il faut aussi réduire le nombre d'artefacts de séquençage tout en s'assurant de ne pas manquer de variations.

Étape 4 Enfin, la quatrième grande étape est l'annotation des séquences. Cette étape est bien plus médicale que les précédentes et les généticiens commencent à s'impliquer à cette

étape là. Le but est d'ajouter des informations sur les données obtenues par les étapes précédentes, ces informations sont stockées dans des bases de données biologiques. Ces données viennent compléter les informations sur le gène et sur la variation dans le but de faciliter la prise de décision des généticiens.

Pour comprendre l'organisation des bases de données génétiques, il est important de comprendre que la génétique est un domaine très vaste et très actif en particulier au niveau de la recherche (fondamentale ou appliquée). Cette activité est d'ailleurs très liée aux avancées des techniques de séquençage évoquées précédemment. Ces deux points rendent illusoire la création d'une base de données complète représentant tout le domaine, comme cela peut être le cas dans des domaines plus restreints ou plus stables temporellement. Un exemple de domaine restreint peut être une base de données de pièces aéronautiques ou d'écrivains d'une période définie, ce sont des ensembles finis où il n'y a pas ou peu de nouvelle création d'items. En conséquence, les bases de données génétiques se sont organisées selon trois principaux modèles :

— Les bases de données complètes de données stables

— Les bases de données centrées sur des gènes

— Les bases de données centrées sur des phénotypes

Les bases de données complètes visent à maintenir une information à jour quelle que soit la région du génome, ce qui n'est possible que sur des données stables, c'est-à-dire des données que ne varient pas dans le temps. Ces bases de données s'appuient sur des informations elles-mêmes stables, qui n'évoluent pas ou peu, qu'elles stockent. C'est par exemple le cas des données de fréquence alléliques dans la population générale, qui désormais ne changent pas beaucoup au fil des ans. Ainsi les projets comme 1000 génomes (Thousand Genomes Project Consortium et al., 2015) ou ExAC (Karczewski, Weisburd et al., 2016) ont choisi cette approche. Il est cependant nécessaire d'avoir accès à des informations moins stables et pour autant le plus à jour possible. Pour cela, certaines bases ont choisi de se restreindre à des gènes connus et documentés, sur lesquels les informations sont disponibles. Cette approche est celle de la base de données *Online Mendeleian Inheritance in Man* (OMIM) (Hamosh et al., 2005) qui stocke 4086 gènes associés à leurs phénotypes détaillés, ce qui correspond à moins de 20% des gènes humains. Cette approche reste cependant pertinente puisqu'elle permet d'avoir une information d'une grande fiabilité, car les connaissances évoluent moins vite sur ces gènes déjà bien connus. Elle est complémentaire avec la troisième approche, qui se centre sur un ou plusieurs phénotypes et associent à ces phénotypes un ensemble de gènes. Dans ce cas, un très petit pourcentage de gènes sera concerné, mais ces gènes ne seront pas forcément que des gènes répertoriés dans la base de données OMIM. Le très faible nombre de gènes permet de maintenir des données à jour sur les dernières découvertes. Les données seront donc moins stables, mais beaucoup plus réactives. Dans la Déficience Intellectuelle (DI), *Systems biology approaches to Intellectual Disability* (SysID) (Kochinke et al., 2016b) est l'un des projets qui ont choisi cette approche. Les mises à jour y sont très fréquentes.

Les concepts stockés dans les bases de données génétiques sont assez nombreux, mais peuvent se regrouper en plusieurs grandes catégories :

— Les informations sur l'impact de la variation nucléotidique, ce sont toutes les informations décrivant la variation, sa localisation, son type, le gène impacté,... Ce sont des informations générales, qui ne permettent pas de conclure quant à la pathogénicité d'une variation mais qui servent de base à la réflexion du généticien.

— Les informations de qualité, qui comprennent le pourcentage de présence de la variation, la profondeur du séquençage et un score de qualité (le plus utilisé étant le *Phred* score). Ces informations ne permettent pas non plus de conclure à la pathogénicité d'une variation, l'objectif est plutôt de distinguer les vraies variations des artéfacts de séquençage.

— Les informations de fréquences, c'est-à-dire le nombre de fois où cette variation particulière a été identifiée, dans une population générale contrôle, non malade, principalement (mais cela peut aussi être dans des cohortes de patients) et selon l'origine ethnique. Les fréquences dans la population générale permettent d'éliminer des variations candidates (au moins dans le cadre des maladies rares) si elles sont trop répandues.

— Les prédictions informatiques. Ce sont des scores générés par des ordinateurs qui décrivent la probabilité de pathogénicité de chaque variation. Ils sont très nombreux, mais leur efficacité est très variable d'un score à l'autre et d'un contexte à l'autre.

— Les informations à l'échelle protéique, qui englobent les changements d'acides aminés, le type de protéine produite, les éventuelles voies métaboliques impactées, les domaines protéiques affectés par la variation. Ce savoir est important pour évaluer les dommages potentiels que peut entraîner une variation.

— Les informations à l'échelle du gène, par exemple le mode de transmission, le taux des variations faux-sens et non-sens attendus et constatés, son appartenance à des bases de données, ou encore ses liens connus avec des pathologies. Ces données sont capitales, elles peuvent changer complètement le regard du généticien sur une variation, la rendant très probablement pathogène, ou à l'inverse très probablement bénigne.

— Les informations cliniques, ce sont les descriptions cliniques de patients présentant des variations proches, la façon dont a précédemment été rendue la variation, la classe de la variation. Ces informations seront directement comparées au compte rendu clinique du patient.

Le processus d'annotation consiste donc à préciser ces différentes informations pour chacune des variations nucléotidiques.

Le processus bio-informatique se termine par la phase d'annotation, c'est à sa suite que commence la phase d'analyse des données annotées. Cette phase est sous la responsabilité du généticien, bien que certains laboratoires utilisent des pré-tris bio-informatiques définis avec l'aide des généticiens pour ne plus fournir qu'une liste de quelques variations. Devant les risques de faux-négatifs élevés de cette méthode, la variation pathogène pouvant être éliminée du résultat par un filtre, les généticiens ont souvent accès à l'ensemble des données, soit environ 60000 variations pour une analyse d'exome. Ils appliquent alors différents filtres successifs sur différents critères issus de l'annotation pour ne garder que les meilleurs variations candidates. Ils étudient ces variations en détail en se basant sur leurs expériences, le descriptif clinique détaillé du patient ainsi que sur les outils bibliographiques disponibles. Si aucune des variations candidates n'est retenue, le processus d'analyse recommence avec des filtres un petit peu moins stricts. Ce processus est brièvement présenté dans la figure 1.3.

Avec le développement des outils bio-informatiques, ces premières étapes ne posent plus de problèmes majeurs. Les difficultés se situent maintenant sur la phase d'analyse par le généticien, où le grand nombre de données rend l'extraction des variations pathogènes complexe. La méthode utilisée par certains laboratoires de ne laisser que quelques variations revient au même problème, les filtres effectués par le bio-informaticien sont plus rapides car automatisés, mais les seuils restants fixés avec le généticien, les risques de faux-négatifs sont encore élevés et il est fréquent de devoir ré-analyser les données avec des filtres moins stringeants.

La phase d'analyse cristallise donc les problèmes d'efficacité, avec des variations pathogènes manquées, que ce soit à cause de filtres trop stricts éliminant la variation ou trop lâches, la noyant dans une quantité importante de données. A ces problèmes d'efficacité se joignent d'importants problèmes d'efficience. Il est malaisé de manipuler une liste aussi conséquente et aussi dense d'informations. Il en résulte que la tâche du généticien est longue et fastidieuse, augmentant le risque d'erreur. Cette difficulté a mené à la création de scores de prédictions *in silico* pour assister les généticiens en prédisant pour chaque variation son potentiel pathogène.

Les scores bio-informatiques sont très nombreux et il n'est pas rare de voir une trentaine de scores de prédiction différents dans un fichier annoté, ce qui n'est qu'une petite fraction

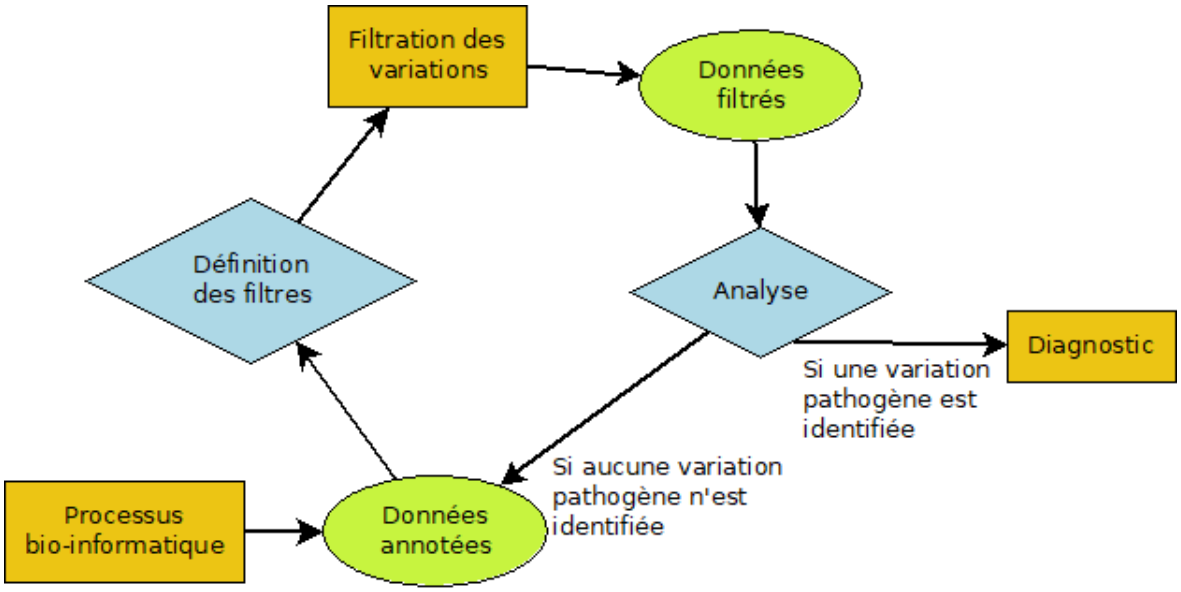


FIGURE 1.3 – Récapitulatif du processus d'analyse réalisé par le généticien

de l'ensemble des scores présents dans la littérature scientifique. Les généticiens les utilisent, mais ne s'appuient pas entièrement sur eux, car ils restent prudents quant à leur efficacité. Ces scores servent donc principalement de points d'appui, d'indices supplémentaires de pathogénicité. Cette prudence vient de la grande variabilité de l'efficacité des scores, qui est de plus, dépendante du contexte. Les généticiens s'appuient souvent sur un nombre restreint de scores qu'ils connaissent bien, plutôt que d'utiliser les prédictions de l'ensemble des outils.

La phase d'analyse dépend grandement de la phase d'annotation : plus les informations seront précises, fiables et surtout pertinentes, plus le travail du généticien en sera facilité. Le généticien dépend donc très fortement du bio-informaticien pour que ce dernier mette à sa disposition les bases de données les plus adaptées. Cette dépendance est également un problème dans le processus, car elle limite très fortement la possibilité du généticien de tester la pertinence de l'annotation fournie par une nouvelle base de données, de ré-annoter d'anciennes données, de mettre à jour les bases de données. Le généticien restera bien sûr dépendant du bio-informaticien pour tout le reste du *pipeline*, mais les conséquences sont bien moindres étant donné la relative stabilité des *pipelines* d'analyse.

Dans ce contexte, nous avons concentré nos efforts sur deux points principaux, la gestion des bases de données d'annotation et l'interprétation des variants. Après un chapitre permettant de contextualiser nos travaux, nous décrirons l'état des connaissances dans les domaines de la génétique médicale, la bioinformatique, la création d'ontologies ainsi que de l'Interaction Homme Machine. Cet état des connaissances nous permettra d'aborder dans le chapitre IV nos travaux concernant la création d'un gestionnaire de bases de données à destination des généticiens. Ces travaux permettront enfin d'enchainer sur le chapitre V, traitant de notre contribution à l'interprétation des variations génétiques, en proposant une nouvelle approche d'assistance pour les généticiens permettant d'améliorer leurs conditions de travail.

Chapitre 2

Contexte

2.1 Cadre de la recherche

Ces travaux prennent lieu dans un cadre bio-informatique, c'est donc un travail informatique très fortement inscrit dans des thématiques biologiques. Dans notre cas, ces thématiques biologiques se concentrent sur la génétique médicale, il est important d'en comprendre les problématiques afin de mettre en perspective ce travail.

2.1.1 Contexte médical

Cette étude se place dans le contexte de la génétique des maladies rares, c'est dans ce contexte que nous allons présenter les travaux, même si les conclusions et contributions sont transposables à d'autres champs de recherches. Les recherches ont été effectuées et validées sur des patients atteints de Déficience Intellectuelle (DI). Ce chapitre descriptif du contexte de l'étude se concentre donc sur ce domaine particulier de la génétique, les maladies rares, et en particulier la DI.

Les maladies génétiques rares forment un ensemble volumineux (environ 6 000 selon Orphanet, 2019) mais chaque entité est par définition rare, leurs prévalences individuelles sont faibles, inférieures à 1 pour 100 000. Cependant, la prévalence faible est en partie liée à leur grande variété, il faut donc regarder la prévalence de l'ensemble qui avoisine alors les 4 % (Alliance maladies rares, 2016) pour mieux cerner leur impact épidémiologique. Elles proviennent d'un défaut dans l'Acide DésoxyriboNucléique (ADN) sur une ou plusieurs bases ou d'une modification de sa structure, qui engendre l'absence ou l'altération de certaines protéines, causant ainsi un grand panel de conséquences phénotypiques. Si les maladies génétiques sont par définition incurables, l'impact de certaines maladies peut être diminué avec des traitements limitants les symptômes. Il est également important de connaître l'origine de la maladie pour pouvoir apporter un conseil génétique approprié. Ce conseil génétique permet notamment d'avoir un accompagnement adapté pour la famille du malade, en cas de nouvelles grossesses en proposant un dépistage pré-natal ou une procréation médicalement assistée par exemple. Ces maladies étant complexes à diagnostiquer, il est très fréquent de voir en consultation de génétique médicale des patients en errance diagnostique, le phénotype seul ne permettant pas toujours de poser un diagnostic.

Pour identifier précisément la maladie, il faut réussir à mettre en évidence la variation causale, c'est-à-dire identifier l'altération de la séquence d'ADN qui a un effet délétère. Certaines maladies génétiques sont très bien connues et les variations qui les causent également. Ainsi, la mucoviscidose par exemple, est une maladie causée par un seul gène (*CFTR*) dont la variation prépondérante est p.Phe508del, c'est-à-dire une délétion enlevant seulement le $508^{ème}$ acide aminé du canal CFTR. Le diagnostic de la DI est plus complexe car plus de 1200 gènes sont connus pour être concernés d'après la base de données *Systems biology approaches to Intellectual Disability* (SysID) (Kochinke et al., 2016b) et de nouveaux gènes candidats apparaissent de façon très régulière dans la littérature scientifique.

Les nouvelles techniques de séquençage de l'ADN ont révolutionné l'approche diagnostic en baissant drastiquement les coûts en ressources et en temps pour le séquençage de l'ADN d'un patient. Le séquençage du génome n'est aujourd'hui pas encore un outil diagnostique fréquemment utilisé en France, car il génère des quantités de données trop importantes et trop difficiles à interpréter à cause du peu d'informations à leur sujet. Les généticiens préfèrent utiliser deux autres stratégies plus adaptées : l'analyse des panels de gènes et l'analyse de l'exome. Le principe est de restreindre les régions séquencées à des zones d'intérêt, afin de faciliter l'interprétation des données et de réduire la charge de travail. Dans le cas de l'analyse du panel de gènes, ce sont les créateurs du panel qui vont choisir quels gènes ils vont inclure. Les panels sont souvent centrés sur un type de maladie. Par exemple, un laboratoire peut disposer d'un panel permettant d'explorer les surdités. Ces panels contiennent une centaine de gènes qui sont les plus fréquemment atteints dans les cas de surdités génétiques. Le grand nombre de gènes liés aux déficiences intellectuelles donne naissance à des panels souvent plus

important qui contiennent de 300 à 600 gènes. L'exome correspond comme nous l'avons vu à toutes les parties codantes des gènes du génome ainsi qu'à leurs régions introniques flanquantes, soit environ 23000 gènes, il génère donc une quantité de données beaucoup plus importante qu'un panel. Ce dernier est une analyse plus ciblée, centrée sur un type de pathologie très spécifique, où quelques dizaines à quelques centaines seulement de gènes seront analysés. Les résultats d'un panel sont par conséquent plus facile à interpréter, mais n'apportent qu'une réponse partielle. Il est donc prescrit par les généticiens cliniciens, qui examinent les patients en consultation, lorsqu'ils ont une hypothèse assez précise sur le gène atteint. L'étude de l'exome permet, lui, une approche où diagnostic et recherche médicale seront intimement liés, menant parfois à des découvertes concernant l'implication de nouveaux gènes dans des pathologies.

Dans ces travaux, l'analyse de l'exome est principalement prise en exemple car il est plus complexe à analyser qu'un panel de gènes et parce que cette analyse est de plus en plus prescrite par les généticiens cliniciens. De plus, beaucoup de tâches qui peuvent encore être effectuées manuellement sur des résultats d'analyse de panels de gènes doivent être automatisées à l'échelle de l'exome. Enfin, le domaine de la génétique tend vers l'analyse du génome, qui produit environ dix fois plus de données qu'une analyse d'exome, l'informatisation et l'automatisation seront alors indispensables.

2.1.2 Processus de diagnostic

Pour montrer les spécificités du domaine de la génétique médicale et de l'analyse d'exome, nous pouvons analyser le processus complet de diagnostic, qui part d'une consultation de génétique clinique et de la prescription d'une analyse d'exome par le clinicien. Après cette prescription, un échantillon d'ADN est prélevé sur le patient, classiquement *via* une prise de sang. Des étapes de préparation de l'échantillon ont lieu dans le laboratoire afin, entre autres, d'extraire l'ADN, de sélectionner uniquement les régions d'intérêts et d'ajouter des séquences index permettant d'identifier le patient. Lorsque l'ADN du patient est prêt, il est soumis au séquençage, dans une machine dédiée que l'on appelle un séquenceur, qui produira en sortie des fichiers informatiques de grande taille (plusieurs dizaines de Gigaoctets) contenant la séquence de l'ADN du patient. S'ensuivent plusieurs étapes bio-informatiques permettant d'extraire de ces données une liste de variations brutes. Toutes les étapes du processus sont décrites plus profondément dans le chapitre III.

La liste de variations contient toutes les différences entre l'ADN du patient et la séquence d'ADN de référence. Ces variations sont décrites sous la forme "*Chromosome : coordonnées génomiques référence > altération*". Par exemple "*3 :102568C>A*" signifie que la 102 568ème base du chromosome 3, normalement un C, est ici muté en A. Chaque exome correspond à environ 60 000 variations, dont plus de 99,99 % n'ont pas de conséquences pathologiques. Il est évidemment impossible de déterminer la ou les variations pathogènes avec ces seules informations. C'est pour cela qu'une autre étape bio-informatique est nécessaire : l'annotation. Cette étape consiste à annoter chacune des 60 000 variations avec du savoir contenu dans des bases de données ou des résultats de logiciels de prédictions. C'est une étape clé, car un problème survenant lors de cette étape peut entraver le diagnostic. Il faut donc que les bases de données soient fiables, variées, pertinentes et qu'elles soient le plus à jour possible.

Pourquoi doivent-elles être fiables ? Les bases de données contiennent assez souvent des données erronées car celles-ci ne sont pas toujours vérifiées par un expert indépendant. La plupart utilise des structures collaboratives, où chaque utilisateur saisit ses données. Cela peut entrainer des problèmes si des utilisateurs font des erreurs à la saisie. Cela peut aussi entrainer des conflits entre différentes bases de données avec des désaccords de spécialistes sur l'interprétation de certaines variations.

Pourquoi doivent-elles être variées ? L'information doit être recoupable, il est souvent pertinent d'aller la chercher dans plusieurs sources afin de s'assurer de sa consistance. Ce point fait directement suite

au point précédent, cela permet d'augmenter la robustesse de la démarche et donc d'augmenter la confiance dans les résultats. Cependant, ce point a un revers, il s'agit de conflits de nomenclatures entre les bases de données, qui peut rendre l'interprétation difficile à cause de dénominations différentes d'une base à l'autre par exemple.

Pourquoi doivent-elles être pertinentes ? L'annotation doit être adaptée à son sujet, il existe de nombreuses bases de données spécialisées sur un type de maladie particulier. L'adéquation entre la thématique de la base et les indications phénotypiques du patient est primordiale pour une analyse et une interprétation les plus justes et efficaces possibles. Elle permet de plus une rapidité et une efficience plus importante au généticien, en présentant plus facilement les informations pertinentes.

Pourquoi doivent-elles être à jour ? La génétique Humaine est un domaine de recherche actif, de nouveaux gènes et de nouveaux liens gènes-maladies apparaissent fréquemment. Des informations bibliographiques récentes peuvent ainsi modifier l'interprétation d'une variation et donc le diagnostic.

Les bio-informaticiens utilisent généralement des outils dédiés, appelés annotateurs, qui sont spécialisés dans cette étape. Ces outils vont enrichir les variations en ajoutant des informations qu'ils extraient de requêtes sur différentes bases de données de références. Les modules de requêtes sont spécifiques à chaque base de données de références et sont construits à façon par l'équipe de développement de l'annotateur. Par conséquent, le recours à ces outils peut limiter la pertinence des bases, qui resteront très générales, les développeurs choisissant les bases utiles au plus grand nombre d'utilisateurs. Il peut aussi exister des problèmes de mises à jour, qui sont généralement soumises au travail des développeurs de l'annotateur, puisque le module de requête doit être mis à jour également. À titre d'exemple, pour ces travaux, trois annotateurs différents sont utilisés, ainsi qu'une dizaine de bases de données en direct, c'est-à-dire sans annotateur, le module de requête étant créé localement par le bio-informaticien. Les contraintes sont les mêmes en termes de pertinences et de mises à jour, mais l'équipe de développement étant plus accessible, leur portée est diminuée. Ces bases utilisées directement le sont pour plusieurs raisons : soit parce qu'elles ne sont pas disponibles *via* des annotateurs, soit parce que la version utilisée par l'annotateur est trop vieille par rapport à la version actuelle de la base de données. Le résultat prend la forme d'un tableau avec une ligne par variation et une colonne par annotation. Il y a environ 50 000 lignes en moyenne pour une analyse d'exome et souvent plus d'une centaine de colonnes.

La dernière étape du processus diagnostic est de trier les variants en fonction de la probabilité de pathogénicité. Cette probabilité est complexe à déterminer, car elle dépend de nombreux facteurs. Ces facteurs peuvent être par exemple le type de variation, sa localisation, des prédictions de pathogénicité *via* des algorithmes, les données cliniques issues de la littérature ou encore l'adéquation génotype-phénotype. Le généticien va s'appuyer sur la centaine de colonnes disponibles et définir des seuils sur des colonnes cibles, pour éliminer des variations et réduire le champ d'investigation. Les généticiens commencent par réduire le nombre de variations en appliquant des filtres sur des critères techniques, comme la profondeur et la qualité du séquençage, ils utilisent ensuite des critères de fréquence allélique qui permettent d'éliminer des variations trop fréquentes pour être impliquées dans des pathologies génétiques rares. Les filtres suivants sont très variables en fonctions des généticiens, des pathologies recherchées ainsi que des hypothèses de transmission, notamment en cas de pathologies familiales.

Cette méthode de filtres successifs peut être vue comme une suite de postulats du généticien. Par exemple, il émet le postulat que la variation qu'il recherche n'est pas vue plus de 2 fois dans la population générale, il peut donc éliminer toutes les variations qui sont présentent deux fois ou plus dans les banques de données alléliques. Il fait ensuite le postulat que les variations dont la profondeur de séquençage est inférieure à 20 lectures ne sont pas interprétables, il peut donc les éliminer. Les généticiens peuvent proposer des postulats différents, donc des filtres différents, en fonction de plusieurs facteurs comme le type de pathologie recherchée, la présence d'un forme familiale, ou tout simplement de leurs habitudes et de leurs propres

connaissances ou expériences.

Ce premier ensemble de postulats généraux permet de concentrer les variations intéressantes, mais il reste encore plusieurs centaines de variations à analyser. Les généticiens appliquent alors des tris ou des filtres temporaires pour prioriser les variations à analyser. Ils peuvent par exemple commencer par n'étudier que les variations de type tronquantes (décalage du cadre de lecture, apparition prématurée d'un codon stop) car elles ont un impact potentiellement très élevé sur la protéine. Dans ce cas, un tri sur le type de variation permet de ne conserver que les variations tronquantes, qui sont ensuite analysées une par une. Le processus est ensuite répété avec un autre type de variations, qui sont à nouveau étudiées, et ainsi de suite jusqu'à ce que la variation pathogène soit identifiée ou que toutes les variations soient examinées. Si aucune variation n'est retenue, les généticiens reprennent alors les postulats généraux et tentent d'élargir les filtres progressivement, afin d'augmenter l'exhaustivité du résultat. Tout comme les premiers filtres, l'étape de priorisation plus fine est d'une grande variabilité d'un généticien à l'autre, en fonction de leurs préférences, de leur spécialité, de leur aisance avec certains critères ou encore de leur connaissance de la clinique du patient.

La complexité de la tâche est donc double : il faut être capable de déterminer des premiers seuils adaptés pour minimiser le nombre de variations tout en maximisant les chances d'avoir des variants intéressants, puis il faut également être capable de gérer les centaines de variations restantes. Le processus d'analyse peut donc durer plusieurs heures par patient, c'est une tâche lourde et complexe qui demande méthode, expertise et précision. De plus, les exomes où aucune variation causale n'a été identifiée sont régulièrement réanalysés après les mises à jour des bases de données d'annotation, qui peuvent ainsi permettre de faire ressortir une variation précédemment mise à l'écart. Ceci illustre le besoin de pouvoir réutiliser la procédure d'analyse, en effet, en plus d'une nouvelle annotation, de nouvelles informations cliniques peuvent également entrainer une réanalyse. Le processus d'analyse doit dans ce cas être repris depuis le départ, ce qui n'est pas optimal.

Un exemple de processus de filtration est détaillé ci-dessous, pour un patient dont le phénotype est caractérisé par des malformations cardiaques, des dysmorphies au niveau du visage, notamment un hypertelorisme, des oreilles postérieures, une très petite bouche et des lèvres fines, ainsi qu'un retard global de développement, des troubles de l'apprentissage et des troubles alimentaires.

L'ouverture du fichier et la mise en place des premiers filtres demande 64 clics et environ 2 minutes à un utilisateur entrainé. Ces étapes permettent de supprimer les variants présents de trop nombreuses fois dans la base de données du laboratoire. En effet, on recherche ici des maladies rares, les postulat est donc qu'il est extrêmement peu probable que la variation causant cette maladie soit vue plus de 2 fois sur les 81 patients répertoriés dans la base. Les autres filtres permettent de supprimer les variants présents de trop nombreuses fois dans une base de données de la population générale, selon la même logique. Deux autres filtres sont appliqués afin de s'assurer une profondeur de séquençage et un pourcentage de présence satisfaisants, par exemple 20 lectures et 10% de présence aux minimums. Ces deux grandeurs reflètent la qualité du séquençage à cette position et correspondent respectivement au nombre de fois ou le nucléotide est séquencé et au pourcentage de *reads* dans lequel l'altération est présente. Si ces valeurs sont trop faibles, la variation est très probablement un artéfact de séquençage. Il est important de remarquer que les principaux filtres se basent sur l'expérience du laboratoire. Ces 6 filtres permettent de réduire à 924 le nombre de variations. Ces étapes correspondent à l'ensemble de postulats généraux classiques, qui sont appliqués par la très grande majorité des généticiens. Ensuite, des différences d'approches peuvent être constatées d'un généticien à l'autre, une approche classique est de commencer par mettre en évidence les variations ayant le plus fort impact sur les protéines. Ce sont des variations tronquantes ou provoquant des décalages du cadre de lecture causés par des insertions ou délétions de quelques nucléotides.

Dans notre fichier exemple, il y en a 38. Parmi elles, 2 ressortent, car elles sont notées

potentiellement pathogènes dans ClinVar, une base de données spécialisée dans la classification des variants dans un contexte clinique donné. Cependant, ces deux variations sont localisées sur des gènes dont le mode de transmission est de type autosomique récessif, comme aucune autre variation n'est présente sur ces mêmes gènes, ces variations peuvent être éliminées. En effet, les deux allèles d'un gène dont la transmission est dite récessive doivent être affectés pour que la maladie soit déclenchée, ce qui n'est pas possible avec une seule variation hétérozygote. Trois variations sont également intéressantes car les gènes impliqués ont un rôle dans la DI, mais les phénotypes associés ne mentionnent pas de caractère dysmorphique au niveau du visage ni de malformation cardiaque. Le généticien analyse alors les 33 autres variations, mais elles ne sont pas probantes, les gènes sont peu connus et peu documentés. Ensuite, le généticien étudie les variants faux sens, ce sont des substitutions d'un nucléotide qui va changer un acide aminé de la séquence protéique. Dans cette analyse d'exome, 148 variations faux sens ont passé les filtres. L'une est notée pathogène selon ClinVar et on observe une corrélation génotype / phénotype, le gène correspondant est *CDK13*, dont l'atteinte qui en est décrite correspond parfaitement à la clinique de notre patient pris ici en exemple. L'étape suivante est l'étude de sa ségrégation familiale. Si la variation est survenue de manière *de novo*, c'est-à-dire qu'elle n'est pas présente chez les parents, elle sera rendue pathogène. Si en revanche elle est héritée de l'un des parents qui eux, ne présentent pas un phénotype similaire, celle-ci ne permet probablement pas d'expliquer le phénotype, l'explication se trouve donc très certainement ailleurs.

Cet exemple était très facile, la variation était assez évidente à identifier car elle était répertoriée pathogène dans la base de données ClinVar, sur un gène connu en pathologie humaine dont la clinique est bien documentée et de transmission dominante. Elle a été identifiée comme causale en une dizaine de minutes par les généticiens. Dans des cas plus complexes, de nombreuses autres hypothèses restent à tester, notamment les variants susceptibles d'affecter l'épissage, les variants liés au chromosome X et les variants affectant un gène dont la transmission est récessive. Ces différentes hypothèses peuvent prendre plusieurs heures aux généticiens, avec un risque non négligeable de manquer la variation importante. Cela demande également au généticien une concentration importante pour se souvenir des différentes hypothèses testées afin de n'oublier aucun cas. Enfin, si aucune variation n'est identifiée à la fin du processus, l'analyse sera de nouveau effectuée dans quelques mois, après une mise à jour des bases de données par le bio-informaticien, en espérant faire mieux ressortir une variation manquée.

C'est à cause de ces risques de manquer la variation et des différences d'approches de chaque généticien que ce travail de recherche du variant causal se termine souvent par une synthèse collective des résultats, où les différents généticiens qui ont analysé le patient (analyses clinique et biologique) vont mettre en commun les variants candidats et en discuter. Il est fréquent que certains variants aient été délibérément écartés par des généticiens et pas par d'autres. Cette réunion de concertation permet de confronter les arguments de chacun et de ne garder que des variants consensus. Il est aussi fréquent qu'un des généticiens soit le seul à avoir mis en évidence un variant intéressant et le propose aux autres. En effet, si l'approche décrite plus haut est l'approche classique, par laquelle tous les généticiens commencent, si aucun variant évident n'apparaît, ils vont élargir leurs critères de recherches et à ce moment-là, les méthodes diffèrent très fortement. La métaphore de Gregory Cooper à ce sujet "Needles in stack of needles : finding disease-causal variants in a wealth of genomic data" (Cooper et Shendure, 2011) est assez révélatrice de la difficulté du processus. La tâche consiste à chercher une aiguille inconnue parmi 60000 aiguilles.

2.1.3 Bilan des contraintes

Cet aperçu du processus permet d'identifier les différentes contraintes du domaine. Ces contraintes sont principalement liées à la grande spécialisation du domaine, à sa grande diversité et à son évolution constante à un rythme rapide.

Les approches sont différentes pour chaque généticien, qui sont souvent spécialisés dans

une pathologie particulière. Les laboratoires ont souvent des spécialités multiples, par exemple ils peuvent être spécialisés sur la DI, mais des analyses d'exomes peuvent être ponctuellement réalisées sur d'autres thématiques comme le syndrome CHARGE (Coloboma, Heart defect, Atresia choanae, Retarded growth and development, Genital hypoplasia, Ear anomalies/deafness), la myopie forte ou le diabète familial. Les ressources du domaine, en particulier les bases de données sont également concernées et sont souvent spécifiques à une pathologie particulière. C'est par exemple le cas de la base de données SysID (Kochinke et al., 2016b) qui suit et décrit les gènes impliqués dans la DI. Des projets comme *PanelApp* (England, 2018), visant à catégoriser les gènes en fonction des pathologies qui y sont liées montre également cette spécialisation. Enfin, de nombreux laboratoires utilisent des bases de données locales permettant de concrétiser leur expérience en gardant une trace de chaque variant identifié et de pouvoir rapidement comparer de nouvelles variations à celles qu'ils ont déjà classées et qui ont donné lieu au rendu du résultat au patient.

De cette grande spécialisation découle une grande diversité, car le domaine est étendu. Les approches peuvent être très différentes en fonction des généticiens, de la pathologie visée, des ressources disponibles sur cette pathologie, de l'histoire familiale ou encore des mécanismes de transmission. Ainsi, la recherche de variations récessives à l'état homozygotes dans une famille consanguine est une approche complètement différente de la recherche d'une pathologie liée au chromosome X ou d'une pathologie dominante. La quantité de connaissances disponible sur le gène (celle personnelle du généticien ou celle disponible dans la littérature scientifique) peut également induire de grands changements dans les approches.

Enfin, le domaine de la génétique est encore en mouvement suite à l'apparition et à l'amélioration constante des techniques de séquençage nouvelle génération. Beaucoup de nouvelles données sont produites, beaucoup de nouveaux liens sont créés, ce qui entraine des mises à jour très fréquentes des ressources par le bio-informaticien, que ce soit *via* l'apparition de nouveaux logiciels et outils de prédictions ou par la mise à jour voire la création de bases de données. Cela fonctionne aussi en sens inverse, car certaines ressources disparaissent ou ne sont plus utilisées, à cause d'un arrêt de la maintenance par l'équipe de développement (souvent des équipes de recherches avec de fortes contraintes de financement) par exemple, mais aussi de la pollution de certaines bases par des données non relues, risquant de fausser les résultats.

Ces contraintes demandent des développements *ad hoc* par les bio-informaticiens. Cela crée également une très grande dépendance du généticien vis à vis du bio-informaticien, car beaucoup de tâches nécessitent un travail informatique préalable, tel que la mise à jour d'une base de données, ou le développement d'un script dédié pour intégrer une base de donnée à l'annotation. Les outils doivent être créés et maintenus pour chaque base de données utilisée par le généticien. Cette dépendance nuit forcément à la réactivité des généticiens, qui doivent attendre la réalisation des scripts informatiques ou le déploiement d'un outil avant de profiter de la ressource souhaitée. Cela limite également la capacité d'exploration des généticiens, qui ne peuvent pas forcément tester toutes leurs hypothèses au risque d'entrainer des délais importants du côté bio-informatique. Cela tend aussi à normaliser les approches entre les différents généticiens, en imposant des ressources et des processus.

Enfin, de nombreuses contraintes techniques secondaires pèsent sur le domaine, au premier plan desquelles la taille importante des données. Les bases de données ont une taille qui peut vite dépasser plusieurs dizaines de Go, ce qui demande une gestion relativement fine de leur mise en place, au risque de voir survenir des temps de réponses trop longs.

2.2 Problématique et esquisse générale

Le résumé de ces contraintes permet de constater à quel point le fonctionnement actuel est problématique : la complexité de la tâche associée au besoin de coopération avec un bio-informaticien baissent drastiquement l'efficacité et le confort de travail des généticiens.

C'est à partir de cette question que nous avons construit notre question générique :

Comment peut-on améliorer l'efficience des généticiens ?

Notre objectif de recherche est donc de mettre au point une méthode augmentant l'efficience du travail du généticien. Pour augmenter l'efficience, il faut augmenter l'efficacité, le confort de travail ou les deux. Nous ne pouvons évidemment pas agir sur la complexité de la tâche. Nous avons donc centré notre approche sur la coopération bio-informatique. La question de recherche qui guide ce travail est donc :

Comment augmenter l'indépendance du généticien vis à vis du bio-informaticien ?

Notre approche de réponse se construit sur l'utilisation d'environnements de type bac à sable pour le généticien, lui permettant de facilement explorer et exploiter ses données dans un contexte graphique. Nous avons concentré notre approche sur deux phases particulières du processus qui seront traitées indépendamment comme illustré dans la figure 2.1. Une première partie se situe au niveau de l'étape d'annotation, qui demande l'utilisation de nombreuses bases de données. Notre approche est de construire une méthode et un environnement permettant au généticien de gérer lui-même les bases de données de références utilisées pour l'annotation, leur mise à jour et leur intégration au processus d'annotation. La dépendance du généticien est actuellement très marquée sur cette étape. Le détail de cette partie se situe dans le chapitre IV.

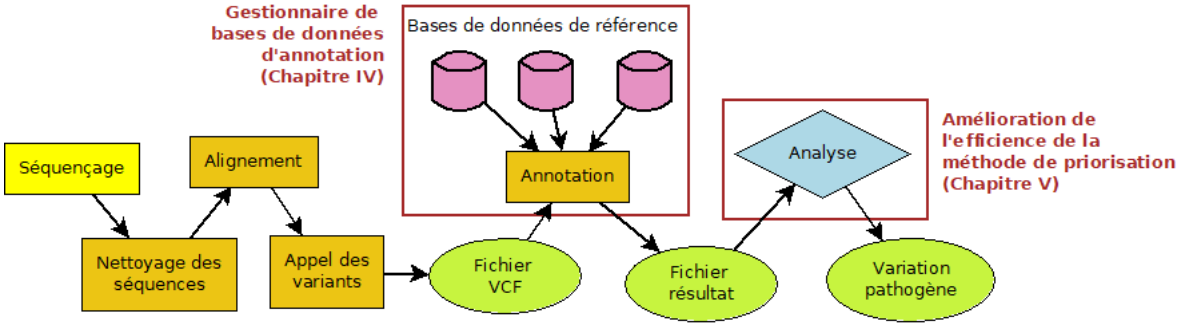


FIGURE 2.1 – Schéma du positionnement de nos contributions sur le processus global de l'analyse de données d'exomes

La seconde partie se situe au niveau de l'analyse des variations, ou l'efficience est actuellement assez faible. L'objectif est d'améliorer cette efficience en prenant en compte les résultats de la première partie. Nous proposons ainsi la création d'une méthode de priorisation des variations efficace, fiable et adaptable. Cette approche doit aussi être réalisée dans un environnement graphique. Le détail de cette partie se situe dans le chapitre V.

Chapitre 3

Etat de l'art général

3.1 Cadre théorique

Dans un premier temps il nous faut définir le cadre théorique de notre étude. Comme nous l'avons décrit dans la partie précédente, nos travaux prennent place dans un contexte bio-informatique centré sur la génétique médicale. Il nous faut donc décrire ce domaine et ces outils pour en comprendre précisément les enjeux.

3.1.1 Contexte génétique

Avant de passer en revue les pratiques médicales actuelles, il nous faut revoir rapidement quelques notions théoriques basiques. Nous allons commencer par quelques rappels génériques sur la structure de l'Acide DésoxyriboNucléique (ADN), puis les variations génétiques avant de décrire la génétique dans un contexte médical.

3.1.1.1 L'ADN

L'ADN est une molécule composée de deux chaînes qui s'enroulent l'une autour de l'autre pour former une structure en double hélice. Elle est le support de l'information génétique qui permet le développement, le fonctionnement et la reproduction des organismes vivants. Chacun des brins est composé d'une suite de nucléotides. Les nucléotides sont des molécules construites à partir de l'une des 4 bases nucléiques suivantes : l'adénine (A), la cytosine (C), la guanine (G) ou la thymine (T). L'adénine et la guanine sont regroupées dans la catégorie des purines, la cytosine et la thymine sont dans celle des pyrimidines. Les deux chaînes, aussi appelées brins, sont complémentaires, cela signifie que chacun des nucléotides d'un brin est apparié avec un nucléotide de l'autre brin. Il y a deux appariements possibles : celui des adénines avec les thymines ainsi que celui des guanines avec les cytosines, ce qui correspond à chaque fois à une liaison entre une purine et une pyrimidine. On peut noter ici que la séquence d'un brin suffit donc à déduire la séquence de l'autre brin. L'ordre de ces nucléotides forme la séquence de l'ADN, c'est cette séquence qui donne du sens à l'information génétique. Certaines séquences particulières peuvent être transcrites en Acide RiboNucléique (ARN), ces séquences sont des gènes. Les ARN formés à partir des gènes peuvent ensuite être traduits en protéine, ou avoir une fonction propre en tant qu'ARN.

Le génome humain contient 3 milliards de paires de bases, parmi lesquelles 1.5 % appartiennent à des gènes (Wolfsberg, McEntyre et Schuler, 2001). Le nombre de gènes est estimé à environ 23000 chez les humains. Ce nombre varie selon la définition du mot "gène", certaines sources en comptent moins (Ezkurdia et al., 2014), quand certains sont disposés à inclure les gènes non codants et donc à obtenir des comptes bien supérieurs (Salzberg, 2018). Les principales bases de données répertorient autour de 20 000 gènes codants pour des protéines, mais l'inclusion de gènes permettant la création de *long non coding RNA* (lncRNA), de longs ARN non codants, peut augmenter ce nombre à environ 50 000. Dans ces travaux, nous utiliserons le terme "gène" pour parler des séquences particulières de nucléotides codants pour des protéines. Les protéines sont les molécules qui effectuent toutes les fonctions essentielles à l'organisme en assurant toutes les tâches nécessaires au bon fonctionnement de la cellule. Ces fonctions comprennent par exemple le transport de nutriments, la structuration des cellules, la catalyse de réactions chimiques ou encore la réplication de l'ADN et la fabrication de nouvelles protéines. On peut donc voir les gènes comme les plans sur lesquels sont bâtis les organismes.

Pour passer de la séquence génétique à la protéine, deux étapes sont nécessaires. La première étape est la transcription, qui consiste à recopier la séquence codante du gène sur un ARN messager (ARNm). Lors de cette étape, les thymines sont remplacées par des uraciles, notées U. La seconde étape est la traduction, qui consiste à transformer cet ARNm en protéine. Les protéines sont composées d'acides aminés ; chacun des acides aminés est codé par trois nucléotides. Comme il y a 64 possibilités de trio de nucléotides et seulement 20 acides aminés,

certains sont présents plusieurs fois, on dit que le code génétique est redondant. Par exemple les séquences d'ARNm ***GUU*** et ***GUG***, qui correspondent donc aux séquences ***GTT*** et ***GTG*** sur le génome, codent toutes les deux pour l'acide aminé valine. Il existe trois codons stop, ***UGA***, ***UAA*** et ***UAG***, qui permettent d'indiquer la fin de la traduction en protéine.

3.1.1.2 Variations de l'ADN

Les variations génétiques, aussi appelées variants, sont des altérations de la séquence de l'ADN. Elles sont causées principalement par des erreurs lors de la réplication de l'ADN, notamment lors des multiplications cellulaires des gamètes. Elles peuvent être des insertions de nucléotides supplémentaires, des délétions ou encore des erreurs de nucléotides. Ces erreurs peuvent être de deux types en fonction du type de changement de nucléotide : s'il s'agit du remplacement d'un nucléotide par un autre de la même catégorie, purine ou pyrimidine, le changement est appelé "transition". Si les deux nucléotides ne sont pas de la même catégorie, on parle alors de "transversions". Les transitions sont environ deux fois plus fréquentes que les transversions, car la proximité chimique des deux nucléotides rend le changement plus facile. Ce rapport s'accentue encore dans les parties codantes de l'ADN, car les transversions étant plus susceptibles de modifier l'acide aminé et la protéine, elles sont également plus susceptibles d'être éliminées dans le processus de sélection naturelle. Les variations ont un rôle biologique, elles permettent notamment la diversité génétique nécessaire à l'évolution et elles jouent un rôle majeur dans la mise en place du système immunitaire. Si ces deux aspects sont incontestablement bénéfiques, le mécanisme de variations génétiques implique aussi des aspects négatifs, les deux principaux étant les cancers et les maladies génétiques.

Une variation peut avoir des conséquences plus ou moins importantes en fonction de sa localisation et de son type. Certaines variations peuvent toucher des régions importantes pour les gènes, mais qui ne sont pas converties en protéines, ce sont des variants affectant les régions régulatrices des gènes, des promoteurs ou des séquences régulant l'épissage. Ce type de variations peut avoir des effets conséquents, notamment sur la production de transcrits alternatifs pouvant donner des protéines non fonctionnelles. Dans les régions codantes, les variations peuvent être de plusieurs types. Pour les illustrer, nous allons définir une séquence exemple, ***GTTTGGACCTAG***, qui donne en ARN la séquence ***GUUUGGACCGUAG***, ce qui correspond en acide aminé à la séquence **Valine-Tryptophane-Thréonine-Stop**

- Faux-sens, c'est le type le plus simple et le plus commun, il s'agit du remplacement d'un acide aminé par un autre. Dans notre exemple, le remplacement du premier ***G*** par un ***T*** changerait la **Valine** en **Phényl-alanine**, modifiant donc la séquence de la protéine finale.
- Non-sens, il s'agit de l'apparition d'un codon stop suite à la mutation. Par exemple, la mutation de la troisième guanine, de ***G*** en ***A***, donne la séquence protéique **Valine-Stop-Thréonine-Stop**, la protéine produite sera donc tronquée au premier codon stop et amputée d'une partie de sa séquence, ce qui peut la rendre complètement inopérante. Ce type de mutation provoque un gros impact sur la protéine et donc des conséquences potentiellement importantes sur l'organisme.
- Perte de stop, c'est l'inverse d'un non-sens, il s'agit d'une variation transformant un codon stop en une autre séquence, entraînant potentiellement des protéines trop longues. Ce type de variation est rare, mais son impact peut être important.
- Synonyme, ce sont des variations silencieuses sur les protéines à cause de la redondance du code génétique. Dans notre séquence d'exemple, une mutation de la deuxième thymine ***T*** en ***G*** par exemple, donnerait toujours une **Valine** dans la séquence protéique. Si elles sont synonymes pour la suite d'acides aminés constituant la protéine, elles peuvent néanmoins impacter les mécanismes d'épissage en créant de nouveaux sites donneurs ou accepteurs de l'épissage, ou encore en modifiant des séquences de régulation de l'épissage comme les *Exonic Splicing Enhancer* (ESE) ou les *Exonic Splicing Silencer* (ESS). *Via*

ces mécanismes, il est donc possible qu'une variation synonyme ait des effets, certes indirects, mais néanmoins conséquents sur les protéines.

— Décalage du cadre de lecture, ce type de mutation est plus complexe. Il est causé uniquement par des insertions ou des délétions d'une ou plusieurs bases non multiples de 3. Comme les nucléotides sont lus dans l'ordre et groupés trois par trois, l'insertion d'un nucléotide supplémentaire va entrainer un réarrangement des groupes de trois nucléotides, donc modifier la séquence. Dans notre séquence type, insérons un ***A*** après le premier triplet pour former cette séquence ***GTTATGGACCTAG***. Le premier triplet ne change pas, ***GTT*** donne toujours une **Valine**. Le triplet suivant est maintenant ***ATG***, qui code une **Méthionine**, puis ***GAC*** qui code l'**Acide Aspartique**, puis ***CTA*** qui code une **Leucine**. La nouvelle séquence, **Valine-Méthionine-Acide Aspartique-Leucine** n'a plus rien à voir avec l'ancienne. Ce type de variation entraîne dans la plupart des cas l'apparition d'un codon stop prématuré.

— Les insertions ou délétions d'un multiple de trois bases ne causent pas de décalage du cadre de lecture, mais simplement l'ajout ou la suppression d'un ou plusieurs acides aminés. C'est le cas par exemple de la mutation prépondérante de la mucoviscidose, p.Phe508del.

Le type de mutation joue un rôle dans la pathogénicité de celle-ci, mais n'est pas le seul facteur responsable. Il est par exemple possible d'avoir des décalages du cadre de lecture qui ne donnent pas de maladies génétiques. En effet, le produit du gène affecté peut ne pas avoir d'impact sur le développement de l'individu, ou bien la quantité de protéine normale (i.e. non affectée) produite par la cellule est suffisante pour assurer une fonction (notion d'allèle dominant ou récessif). Chaque gène est présent en deux copies appelées allèles chez les individus, car les chromosomes fonctionnent par paire, l'une provenant de la mère et l'autre du père. Dans le cadre d'un caractère génétique dominant, l'allèle affecté est dominant et cause le phénotype, même si l'autre allèle est sain. Dans le cadre d'un caractère génétique récessif, il est nécessaire que les deux allèles soient affectés pour que le phénotype soit exprimé. Les organismes présentant un allèle sain et un affecté sont des porteurs sains, qui peuvent transmettre le génotype à leurs enfants mais ne présentent pas eux-mêmes le phénotype. Enfin, les gènes situés sur les chromosomes sexuels obéissent à des règles plus complexes. On peut néanmoins rappeler que la présence d'un seul chromosome X chez les garçons entraine dans la grande majorité des cas des phénotypes plus marqués que chez les filles dans la plupart des maladies génétiques liées au chromosome X.

Concernant les variations créant des codons stop, donc les variations non-sens et les décalages du cadre de lecture, deux mécanismes physiopathologiques peuvent expliquer les effets : il s'agit de l'haploinsuffisance ou de la création d'une protéine tronquée. Les protéines tronquées peuvent perdre leurs fonctions, voire devenir toxiques pour l'organisme en ne répondant plus aux mécanismes de régulation par exemple. L'haploinsufficance est un mécanisme de manque, la protéine est présente dans la cellule grâce à l'allèle sain, mais la quantité produite par ce seul allèle ne suffit pas aux besoins de la cellule. Dans ce cas, les ARNm produits par l'allèle affecté sont dégradés rapidement dans la cellule *via* le mécanisme du *Nonsense Mediated mRNA Decay* (NMD). Il est important de tenir compte de ces modes d'actions, car le mécanisme physiopathologique peut être l'un des critères déterminants dans l'approche d'analyse choisie par le généticien.

3.1.1.3 Contexte médical

Dans un contexte médical, plusieurs raisons poussent les médecins à chercher la variation causale, celle qui est responsable de la maladie du patient. La première est l'errance diagnostique. Les maladies génétiques rares sont difficiles à diagnostiquer, car elles sont très peu fréquentes, très nombreuses et que les tableaux cliniques sont très variables. Une étude française de 2016 (Alliance maladies rares, 2016) réalisée sur mille patients français

atteints de maladies rares montre que le délai pour avoir un diagnostic est d'au moins un an et demi dans la majorité des cas et se prolonge au delà de 5 ans pour un quart des patients. De plus, ce délai ne tient pas compte du temps passé avant que la recherche ne commence, qui peut aussi se compter en années. Enfin, les erreurs de diagnostic sont également fréquentes, ce qui augmente encore cette errance et peut mener à des erreurs de prise en charge graves. En particulier, l'errance diagnostique retarde un conseil génétique indispensable dans un tiers des cas, ce qui peut avoir de graves conséquences. Un diagnostic permet de poser une reconnaissance de la maladie pour l'individu et son entourage mais aussi de mettre en place les moyens pour faire face à ces difficultés, par exemple des orthophonistes, des psychologues, des associations de patients ou encore l'indispensable conseil génétique.

Ce conseil génétique est la seconde raison poussant au diagnostic précis des maladies rares car c'est une consultation de génétique spécialisée et adaptée à la pathologie désormais identifiée. Elle est à destination des patients et de leur entourage et permet d'orienter le patient vers la meilleure prise en charge possible, *via* notamment un suivi médical adapté. Il permet aussi de définir les risques pour les apparentés, voire de proposer un suivi plus approfondi pour les grossesses à venir.

Enfin, le dernier point majeur de cette recherche est le traitement du patient. La maladie est présente dans l'ADN du patient et ne peut en 2019 pas être éliminée. Des recherches notamment sur la thérapie génique sont en cours, accélérées avec l'arrivée de la technique *CRISPR/CAS9* (Ran et al., 2013), mais sont encore très préliminaires. De nos jours, nous pouvons dans certains cas améliorer l'état de santé du patient *via* des traitements et ce domaine est également en évolution rapide, offrant de plus en plus de possibilités et améliorant parfois très nettement le confort de vie des patients, mais un traitement reste impossible à mettre en place dans la très grande majorité des cas.

L'intérêt de l'identification des variants est donc multiple, il permet une meilleure compréhension de la pathologie et une meilleure prise en charge du patient et de sa famille. C'est également un pas de plus dans la compréhension du domaine de la génétique et cela peut amener à des découvertes scientifiques sur les rôles de certains gènes.

3.1.2 Le séquençage Nouvelle Génération

Le séquençage haut débit est une technologie récente qui a considérablement modifié la génétique, en particulier la génétique médicale. Après une brève mise en contexte historique, nous expliquerons le principe général du séquençage nouvelle génération, avant de détailler le protocole de l'analyse de l'exome.

3.1.2.1 Historique

Depuis la découverte de la structure de l'ADN par Rosalind Franklin, publié par James Watson et Francis Crick en 1953 (Watson et Crick, 1953), la capacité à lire et comprendre ce support de l'information génétique est un enjeu primordial du domaine de la génétique. Plusieurs événements ont marqué le domaine, le premier étant la mise en place de la première méthode de séquençage par Frederick Sanger en 1977 (Sanger, Nicklen et Coulson, 1977). Le second pourrait être le projet "*human genome*" visant à séquencer entièrement le génome humain. Ce projet fut lancé en 1990 pour une publication finale de résultats au début des années 2000 et pour un coût d'environ 3 milliards de dollars (Venter et al., 2001). Depuis, le prix du séquençage a chuté de façon importante, pour se stabiliser autour des 2000 dollars pour un génome humain complet en 2019 (Schwarze et al., 2018) avec les techniques dites NGS, traduit par séquençage nouvelle génération en français.

3.1.2.2 Principe

Les séquenceurs modernes utilisent différentes méthodes pour obtenir la séquence d'ADN, mais la philosophie globale reste la même. La technologie utilisée pour la génération des

données utilisées dans le cadre de ce travail est fournie par *Illumina*, c'est donc ce principe qui est détaillé ici.

La première étape est de découper l'ADN en petits fragments, puis de sélectionner les fragments correspondant aux régions d'intérêt, si seule une partie du génome est désirée. C'est par exemple le cas dans les analyses d'exomes et de panel de gènes, où seules les régions codantes sont capturées. Les fragments sélectionnés sont ensuite multipliés par une technique de *Polymerase Chain Reaction* (PCR) pour augmenter la force du signal. Enfin, le brin complémentaire est synthétisé et fixé sur un support de séquençage, que l'on appelle *flow cell*. C'est à ce moment que le séquençage à proprement parler commence avec l'ajout des nucléotides marqués. Ces derniers sont associés à des fluorochromes différents, un pour chaque type de nucléotide (Adénine, Guanine, Cytosine et Thymine), qui empêchent la fixation d'un second nucléotide à leur suite, ce qui garantit un séquençage base par base. Après chaque cycle d'ajout de nucléotide, la *flow cell* est lavée pour enlever les nucléotides excédentaires, puis un laser excite les nucléotides ajoutés, cela provoque le clivage du fluorochrome. Ce clivage a deux effets, il permet de libérer la séquence pour permettre un nouvel ajout de nucléotide au prochain cycle et il provoque l'émission de lumière *via* le fluorochrome libéré, une photographie est prise à cet instant. Sur celle-ci, des centaines de points lumineux correspondent aux séquences, la couleur du point étant déterminée par le type du fluorochrome émis et donc le type du nucléotide incorporé à la séquence. La *flow cell* est lavée pour enlever les fluorochromes libérés, puis le processus recommence pour un nouveau cycle permettant de déterminer un nucléotide de plus. L'enchainement des couleurs pour chaque séquence correspond à l'enchainement des nucléotides ajoutés, donc à la séquence du fragment. Ces étapes sont récapitulées dans la figure 3.1.

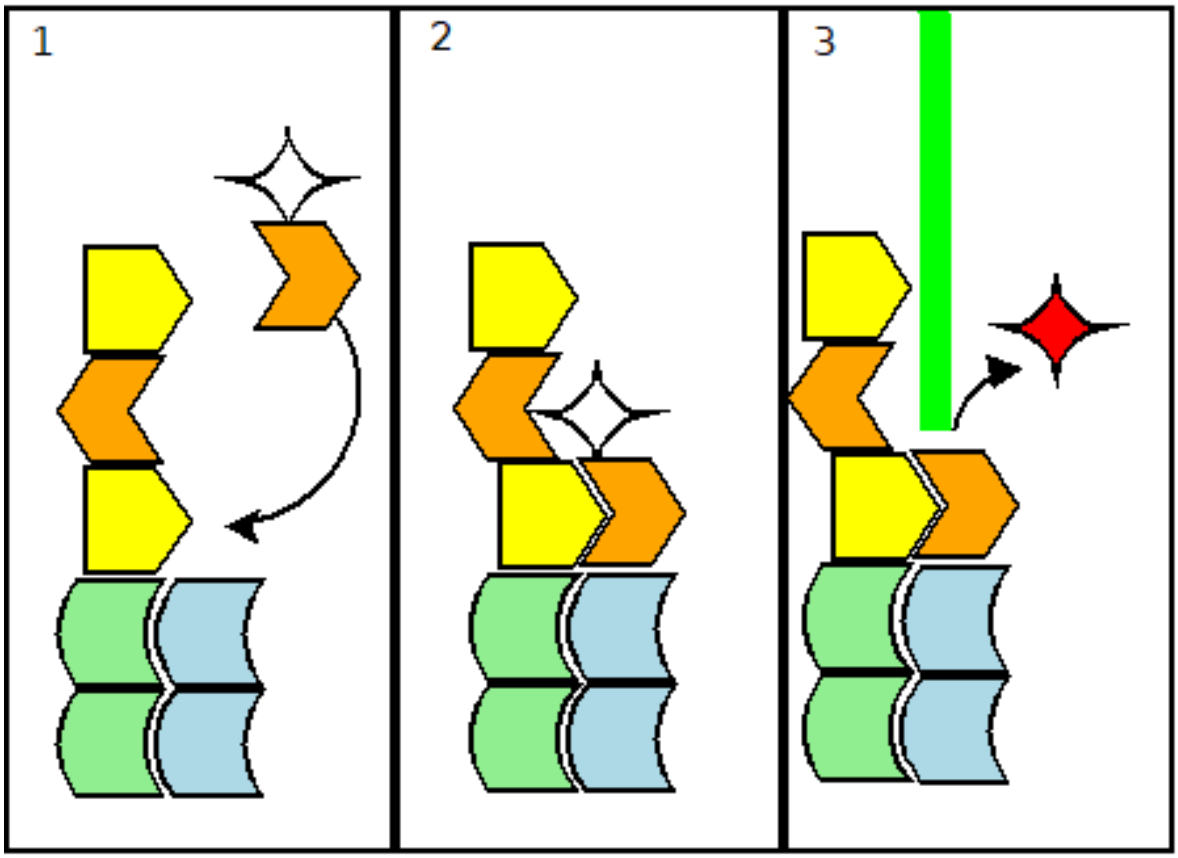


FIGURE 3.1 – Schéma du principe de séquençage selon Illumina **1)** Ajout du nucléotide marqué **2)** Fixation et blocage de la séquence par le fluorochrome **3)** Excitation au laser, libération de la séquence et émission du flurochrome excité

D'autres principes existent, basés sur l'hybridation de sondes fluorescentes de petite taille,

par le passage de brins d'ADN dans des nanopores ou encore par la libération d'ions H^+.

3.1.2.3 La technique de séquençage l'exome

La technique de séquençage de l'exome commence par l'extraction de l'ADN des patients à partir de prélèvements sanguins. Cette partie est réalisée par un automate qui capture l'ADN avec des nanoparticules magnétiques. La concentration et la pureté de l'ADN sont alors contrôlées. En effet, la quantité d'ADN est cruciale tout au long de l'expérimentation, elle est ici mesurée à l'aide d'un spectrophotomètre à micro-volume ainsi qu'avec un fluorimètre permettant la quantification d'ADN. Chacune de ces deux techniques de quantification présente des avantages et des inconvénients, le premier est moins précis mais permet de déceler les contaminations, ce que ne permet pas le second, cependant plus précis, c'est pourquoi elles sont utilisées couplées ainsi. L'étape suivante est la préparation des librairies. Cela commence par la fragmentation de l'ADN *via* des transposases. Une fois l'ADN fragmenté, des ajouts de séquences d'adaptateurs sont réalisés aux extrémités. Les fragments sont ensuite purifiés afin de ne conserver que ceux liés à un adaptateur.

Une phase d'amplification par PCR permet d'augmenter la quantité de matériel disponible et de s'assurer de meilleurs résultats finaux. L'ADN est à nouveau purifié après cette amplification pour les mêmes raisons que précédemment. Les quantités, qualités et tailles des fragments sont évaluées par un dosage effectué après une séparation électrophorétique des acides nucléiques. Les échantillons sont dilués pour obtenir une concentration identique d'ADN pour chaque patient. Les fragments sont alors hybridés avec des librairies composées des séquences d'intérêt à étudier dans le génome, donc pour un exome, contenant l'ensemble des exons des gènes et leurs régions introniques flanquantes. Les séquences hybridées sont récupérées *via* des billes magnétiques. Une nouvelle phase d'amplification par PCR vient en suivant, puis les billes sont enlevées de la solution. Enfin, une dernière phase de purification est effectuée, puis la qualité des fragments d'ADN est évaluée une nouvelle fois.

Les librairies se préparent généralement pour huit patients à la fois pour des raisons techniques de quantité de réactifs disponibles dans les kits et de contraintes liées à la manipulation par un humain. Lorsque 12 librairies, donc 12 patients, sont réalisées, c'est-à-dire que les étapes précédentes ont été effectuées deux fois pour obtenir 16 librairies, 12 échantillons sont mélangés de façon équimolaire. Les 4 autres échantillons sont temporairement congelés et seront utilisés lors du prochain séquençage. Pour réaliser le mélange équimolaire, les quantités d'ADN sont mesurées sur le fluorimètre *Qubit*. L'ADN contrôle est ajouté dans le mélange équimolaire, il s'agit d'un ADN de virus, le *PhiX* qui a la particularité d'être composé à 25 % de chacune des bases (***A***, ***T***, ***C*** et ***G***). Il joue le rôle de témoin positif, qui permet de vérifier qu'il n'y a pas eu de problème lors du séquençage, en particulier une mauvaise distribution des nucléotides marqués, puisqu'on sait devoir obtenir 25 % de chaque nucléotide. Enfin, la cartouche de réactif est préparée, il s'agit d'une boîte adaptée au séquenceur dans laquelle des puits sont percés par l'utilisateur pour accueillir les différents réactifs. Ceux-ci doivent être mis dans les puits correspondants pour permettre le séquençage, ainsi que bien sûr le mélange équimolaire. Le séquençage est réalisé en "*paired-end*", ce qui signifie qu'il est réalisé sur chacun des brins de l'ADN. Ce principe de séquençage permet une meilleure couverture, puisqu'on double le nombre de lectures, mais aussi un meilleur alignement des séquences lors des étapes bio-informatiques. La *flow cell* est chargée dans l'appareil, il s'agit d'une plaque de verre qui sert de support à la réaction chimique du séquençage et particulièrement de support aux fragments d'ADN à séquencer.

Avant le séquençage à proprement parler, les fragments d'ADN de la librairie vont se fixer sur la *flow cell via* une ligation aléatoire aux extrémités de séquences d'attache. Des cycles d'ajouts de nucléotides puis de dénaturation vont permettre la multiplication de chacun des fragments localement pour former des *clusters* composés de plusieurs milliers de fragments identiques très proches sur la *flow cell*. Il est important que les *clusters* soient bien individualisés, sinon le séquençage ne pourra pas fonctionner, c'est pour cela que les

quantités d'ADN sont très contrôlées tout au long de l'expérimentation. Lorsque les *clusters* sont formés, le séquençage peut commencer comme décrit dans la section 3.1.2.2 ci-desssus. Au bout de 150 cycles, le séquençage est terminé. Chaque *cluster* correspond donc à une lecture (appelé *read* en anglais). On obtient cette séquence en analysant informatiquement la suite des 150 photographies l'ordre des couleurs émises, qui permettent de déduire la séquence nucléotidique.

3.1.3 Analyse bio-informatique de données NGS

L'analyse des données de séquençage haut débit est réalisée dans un premier temps par des techniques bio-informatiques pour extraire des données brutes des informations facilement interprétables par les généticiens. Après avoir globalement décrit le principe, nous décrirons les différents étapes selon leur ordre d'apparition dans un processus classique d'analyse.

3.1.3.1 Principe

Les fichiers de résultats en sortie de séquenceurs sont au format *.bcl*, un format propriétaire d'*Illumina*. Un logiciel fourni par *Illumina* permet de rapidement les convertir au format *fastq*, qui est un format libre répandu dans le monde de la bio-informatique et présenté figure 3.2. Ce format fonctionne par groupe de 4 lignes, chaque groupe étant associé à une séquence, appelée *read*. Ce sont des fichiers plats, non compressés ; comme chaque séquence est présente un grand nombre de fois, ce sont des fichiers qui peuvent être très volumineux. Ils sont analysés par des outils bio-informatiques. Ces outils sont souvent organisés en *pipelines*, c'est-à-dire en enchaînement d'outils dont le processus de fonctionnement est le suivant : le résultat d'un outil donné est directement utilisé par l'outil suivant. Ces outils varient en fonction du projet étudié, l'approche n'est par exemple pas la même lors du séquençage de génomes de bactéries, d'un génome humain, ou d'ADN tumoral. Dans notre contexte d'étude, la génétique médicale des maladies rares *via* l'analyse d'exomes, on peut distinguer quatre étapes majeures décrites ci-dessous.

```
@NB551159:101:HJKN7AFXY:1:11101:23879:1095 1:N:0:AGGCAGAA+NTAGAGAG
AAGATGATCAAATGTAAACTATTTTCAGTAAGTTTGCTTAATGACCCCGATACCAGATGATAAGCTAGACTTTAGTAACCTAGAGGTTTAATGATAAACCAGATTCTTCCAATGTTTAATTGGCTTAATAGTGATAAAAGAAGCCATTGTA
+
<E<EA<EEAAEE/AE<</E/EAEAA//AEEEEEEE<EEAAEAA//////E<<EA/E<E<AEEE6/E6E<EEEE/EEA/<EEEEE</EEEE/E<EEE/E/EEEEEAEEEEEE/EEEEAEAEEEEAEEEEEEEEEEEEEEE/EEAE/AAAAA
@NB551159:101:HJKN7AFXY:1:11101:12811:1095 1:N:0:AGGCAGAA+NTAGAGAG
TACCATGATTGTCCACTTTCTGTTTTCTGCCTCTGGGAACACTAGTGATCGAGGAGAATAAAAGACTTCATCATCCACTTCTTCTGGCTTTCTGGGCAAGCAGTGTAAAAATGTCCAGTCAGAGGCAGCAACTTTAGCAGTCTAAAGAGAT
+
EEAEAEEEEEEEAAEE<EEAEEEEEEEEEEE<EEEEEEEEEEEEEEAEEEEAEEEEEEEEAEEEEEEEAEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEAEEEEEAEEEEE/EEEEEEEEEEEEEEEEEEEEEEEEEAAAA6
```

FIGURE 3.2 – 2 *reads* au format *fastq*

3.1.3.2 Nettoyage et qualité

La première étape, commune à toutes les méthodes et les types de données, consiste au nettoyage des données ainsi qu'en la mesure de quelques indicateurs de qualité. Lors du séquençage, les données sont associées à des séquences supplémentaires pour permettre leur sélection, leur identification ou leur fixation : ce sont les index, les séquences d'adaptateurs ou encore l'ADN du *PhiX* mentionné plus haut. On utilise des outils appelés *trimmers*, pour reconnaître et couper ces séquences.

Des contrôles qualités ont également lieu à cette étape, ce qui va permettre de vérifier que les séquences sont issues d'un séquençage propre, c'est-à-dire contenant peu d'erreurs et d'artéfacts. La qualité est mesurée dans un fichier *fastq* par un score appelé *Phred*. Ce score donne le taux d'erreur probable pour chaque base. On peut donc dire pour chaque base, la probabilité qu'elle soit vraiment la base présente dans la séquence d'ADN. Un second nettoyage peut avoir lieu pour supprimer les bases dont la qualité est inférieure à un certain seuil. Il est en effet bien souvent préférable de raccourcir les séquences à cette étape plutôt que d'avoir des données impropres qui risquent d'engendrer des problèmes conséquents lors des étapes suivantes.

3.1.3.3 Alignement

Cette étape prend deux formes en fonction du type de recherche et du type d'organisme séquencé. Si l'organisme séquencé a déjà un génome complet, on aura affaire à une étape d'alignement classique. C'est ce qu'il se passe en génétique humaine puisque l'on possède la séquence d'un génome humain de référence depuis 2001 (Venter et al., 2001). Les génomes de références sont construits à partir de l'ADN de plusieurs individus et servent de base commune. Néanmoins, des problèmes de nomenclatures et de versions nuisent à cet aspect de socle commun. Deux organismes se partagent la création des génomes humains de références, avec 2 nomenclatures différentes, il s'agit du *Genome Reference Consortium* (GRC), dont la version actuelle est le *GRCh38*, bien que la version *GRCh37* soit encore la plus utilisée. L'autre organisme est l'*University of Carolina Santa-Cruz* (UCSC). La version actuelle de ce génome est le *hg38*, qui correspond au *GRCh38*, mais la version précédente est encore très utilisée, il s'agit de la version *hg19*, qui correspond à la version *GRCh37* du GRC. Ces données sont résumées dans le tableau 3.1.

Date de sortie	Nomenclature GRC	Nomenclature UCSC
2013	GRCh38	hg38
2009	GRCh37	hg19
2006	NCBI Build 36	hg18

Tableau 3.1 – Tableau récapitulatif des différentes versions des génomes humains de référence

Un outil bio-informatique va analyser chacune des séquences contenues dans les fichiers *fastq* afin de trouver sur le génome de référence la région correspondante. C'est une étape longue, mais les outils et les algorithmes dédiés à cette étape sont de plus en plus performants. Ils utilisent notamment de plus en plus les ressources des cartes graphiques pour séparer le problème. Le résultat de cette étape est un fichier *sam*, qui peut être compressé en un fichier *bam*. Ces fichiers sont de plus petites tailles que les fichiers *fastq* car ils sont compressés et prennent en charge les duos de fichier *paired-end* en un seul fichier. Un exemple du contenu de ces fichier est présenté figure 3.3. Ils conservent le score de qualité *Phred* et ajoutent des informations utiles, telles que la séquence de référence où a eu lieu l'alignement, la position génomique sur cette séquence et enfin la qualité de l'alignement dans un score appelé *Compact Idiosyncratic Gapped Alignment Report* (CIGAR).

Ce score présente 9 possibilités pour chaque base, qui sont ensuite regroupées. Les 9 possibilités sont :

M pour ***Match***, l'alignement s'est fait normalement, mais la séquence peut être un *match* (le nucléotide correspond à la séquence de référence, un appariement) ou un *mismatch* (le nucléotide n'est pas celui de la séquence de référence, un mésappariement)

I pour ***Insertion***, une insertion par rapport à la référence

D pour ***Deletion***, une délétion par rapport à la référence

N pour ***Not***, une région absente incluse dans la séquence, qui est plutôt utilisée en *RNAseq* pour définir les introns

S pour ***Soft clipping***, des séquences coupées aux extrémités du *read* mais présentes dans le *bam*, souvent pour des raisons de qualité

H pour ***Hard clipping***, des séquences coupées aux extrémités du *read* absentes du *bam*

P pour ***Padding***, utilisé uniquement lors de l'alignement à plusieurs séquences pour ajouter des blancs plus visuels

= pour désigner un *match* de séquence, c'est-à-dire une correspondance

X pour désigner un *mismatch* de séquence, une absence de correspondance

Il faut garder à l'esprit que seul l'alignement est coté par les opérateurs traditionnels, qui sont les 7 premiers. Les opérateurs de séquences sont rarement utilisés. Ainsi, un score CIGAR **140M10S** indique que l'alignement a été correctement réalisé sur 140 nucléotides et que 10 ont été supprimés en fin de chaîne, mais ne renseigne pas sur les *Single Nucleotide Variation* (SNV), c'est-à-dire une variation qui ne modifie qu'un seul nucléotide, qui pourraient être présents. Un autre exemple de score CIGAR, **87M1D63M** indique que 87 bases ont été alignées, suivies d'une délétion, puis d'un nouvel alignement réussi de 63 bases. On ne sait pas non plus si des SNV sont présents dans cette séquence.

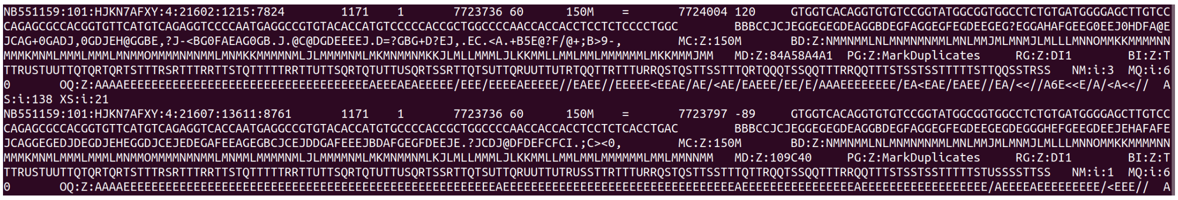


FIGURE 3.3 – Visualisation de 2 *reads* au format *bam*

Lorsque le génome de l'organisme n'est pas connu, cette approche est impossible et c'est donc une approche d'assemblage qui est utilisée. Le but est alors d'identifier les zones de chevauchement de chaque *read* pour établir la séquence complète. C'est également une étape coûteuse en ressources informatiques et en temps de calcul, qui repose sur des techniques telles que les graphes de De Bruijn (COMPEAU, PEVZNER et TESLER, 2011). Elle n'est que peu utilisée en génétique humaine, du fait de la bonne documentation du génome humain. Le seul domaine médical où elle est utilisée est le microbiote intestinal, où les génomes bactériens de toutes les bactéries ne sont pas forcément bien définis.

Enfin, les outils d'alignement peuvent faire des erreurs en particulier en cas de modification de structure, ainsi qu'en cas de faibles profondeurs de séquençage et de séquences répétées ; il est donc important d'avoir des étapes de nettoyages et de recalibrations après l'alignement, notamment pour mieux prendre en compte les variants de type *indels*, qui correspondent à des insertions ou des délétions de matériel. En effet, en modifiant la séquence de façon importante, ces variations baissent la similarité entre le génome de référence et le *read* analysé, ce qui rend plus difficile la tâche du logiciel d'alignement.

3.1.3.4 Appel de variants et annotations

Dans le domaine de la génétique humaine et dans de nombreuses autres applications, les données intéressantes sont les variations par rapport à la séquence de référence. Ces variations peuvent être de plusieurs natures : variations ponctuelles (un seul nucléotide affecté) et structurelles (plusieurs nucléotides). Cette étape se nomme "appel des variants" en français, elle est réalisée par des outils dédiés : des logiciels que l'on appelle *callers*. Il en existe plusieurs, qui possèdent chacun leurs forces et leurs faiblesses. Par exemple les *callers* les plus précis requièrent souvent un grand nombre de données pour être efficaces, ce qui peut être limitant. Ces outils fonctionnent globalement sur le même principe, ils vont détecter les changements entre le fichier *sam* ou *bam* du patient et le génome de référence. Le type de sortie est donc une liste de coordonnées génomiques associées à un changement, sous la forme par exemple *chr3 :101465C>A*, qui indique que la **101465**^{ème} base du chromosome **3**, qui est normalement une Cytosine, est changée en une Adénine. Il est encore impossible de connaître l'impact phénotypique de la mutation avec ce type de données, c'est pourquoi elles seront annotées afin d'augmenter le savoir associé à chacune des variations. Deux méthodes existent pour générer de l'information lors de l'annotation : soit l'information est pré-générée pour chaque variation possible et stockée dans une base de données, soit elle est calculée au moment de l'annotation. Chacune de ses approches possède ses avantages et ses inconvénients et elles sont très souvent utilisées conjointement. Les annotations dépendent grandement de l'objectif final

de l'étude. Par exemple dans le cadre du diagnostic moléculaire d'une DI, les annotations seront globalement de 6 types :

I La localisation. La position génomique n'est pas directement utilisée, en revanche, il est important de connaître les informations secondaires, c'est-à-dire déterminer si la variation est dans un gène, dans un exon ou sur un site d'épissage. Une variation dans un domaine intronique a moins de chance d'être pathogène qu'une variation située dans une partie codante.

II Des informations concernant le gène impacté. Il s'agit ici de connaître les informations à l'échelle du gène, notamment son éventuel mode de fonctionnement physiopathologique (plutôt dominant ou récessif, sa tolérance vis à vis des variations faux-sens ou non-sens ainsi qu'à l'haploinsuffisance), son expression (quels sont les tissus où le gène s'exprime) ou encore son implication dans une voie biologique particulière.

III L'impact sur la protéine. Ce point est en partie lié à la localisation exonique de la variation. L'objectif est ici d'estimer le potentiel impact sur la structure de la protéine en déterminant si un domaine protéique particulier est atteint, ainsi que les éventuels changements physico-chimiques.

IV Des données d'interprétation clinique. Ce sont les comptes-rendus des réflexions des précédents généticiens confrontés à cette variation. Il est donc important de savoir si cette variation a déjà été identifiée et si oui, comment a-t-elle été rendue au clinicien ou classée dans les bases de données (selon les règles définies par l'*American College of Human Genetics (ACMG)* (C. S. Richards et al., 2008)).

V Des fréquences alléliques. Il s'agit de connaitre la fréquence de cette variation dans des cohortes de patients définies (atteints de maladies cardiovasculaires, de diabètes,...) ou dans des populations contrôles, non malades. Les informations de fréquences alléliques sont capitales dans la recherche de maladies rares, mais sont moins intéressantes dans la plupart des autres domaines de la génétique.

VI Des scores de prédictions informatiques. Ce sont les prédictions de pathogénicité issus de différents outils, s'appuyant sur des critères tels que la conservation du nucléotide à travers différentes espèces, la conservation de l'acide aminé impacté, etc... Certains de ces outils sont détaillés dans les sections suivantes.

C'est sur la base de ces informations que le généticien biologiste peut rendre son diagnostic et dire laquelle ou lesquelles des variations causent la maladie génétique du patient.

3.2 La Déficience Intellectuelle

La Déficience Intellectuelle, souvent abrégée en DI, est un trouble défini par un déficit et un dysfonctionnement cognitif. On distingue deux types de DI en fonction de la gravité de l'atteinte, la DI légère, pour les quotients intellectuels compris entre 50 et 69, et la DI sévère, pour les quotients intellectuels inférieurs à 50. Une étude ancienne mais qui fait toujours foi (Roeleveld et Zielhuis, 1997) estimait la prévalence de la DI dans la population mondiale à 0.38 % pour la forme sévère et 3 % pour la forme légère. Ces taux sont définis en ne comptant que les patients ayant une DI non évolutive déclarée avant 18 ans. La validité de cette étude plus de 20 ans après sa publication indique une stabilité des prévalences. Une étude plus récente (Maulik et al., 2011) estime la prévalence de la DI à 1 % de la population mondiale, tous types et tous pays confondus.

Dans le processus de prise en charge des patients, il est important de s'intéresser à l'étiologie de la DI. Cela permet une bien meilleure prise en charge et potentiellement une amélioration de l'état du patient. Cela permet aussi d'évaluer le risque pour la famille, car dans la majorité des cas, la DI est d'origine génétique (Srour et Shevell, 2014). Des causes environnementales ont également été décrites, notamment des intoxications ou des infections pendant la grossesse (Mwaniki et al., 2012) ou encore la prématurité (Larroque et al., 2008). Enfin, des travaux

tendent à indiquer que des facteurs économiques et / ou psychosociaux pourraient également avoir un impact, même s'il est plus indirect et n'est pas incompatible avec une cause génétique sous-jacente (World Health Organization and others, 2011).

Environ un tiers des gènes humains sont exprimés dans le cerveau, ce qui explique le nombre important de gènes pouvant être impliqués dans la DI. Ce nombre est autour d'un millier (Wieczorek, 2018), en constante augmentation depuis quelques années au fur et à mesure des nouvelles découvertes. Ce grand nombre de gènes implique une grande diversité des modes de transmission impliqués, on peut donc trouver des cas dominants, récessifs, autosomiques ou liés au chromosome X, ainsi que des modes de transmissions non mendéliens, donc sans altération de la séquence nucléotidique.

Ce grand nombre de gènes oblige les généticiens à grouper les recherches en séquençant plusieurs dizaines, plusieurs centaines voire plusieurs milliers de gènes à la fois. C'est pour cela que l'analyse de l'exome supplante petit à petit les panels de gènes. Ces derniers ne sont plus utilisés que dans des cas spécifiques, où le descriptif de l'état clinique du patient permet d'orienter clairement la recherche. L'analyse de l'exome reste cependant à ce jour une technique de recherche, car malgré son utilisation de plus en plus répandue et de plus en plus banale, il ne fait pas encore parti de la liste ministérielle des actes innovants de la biologie médicale, ce qui pose des problèmes financiers. En effet, l'analyse d'un exome est "payé" au laboratoire par le ministère de la santé au même niveau qu'un panel de gènes, selon le standard "Références des actes Innovants Hors Nomenclature (RIHN)", ce qui correspond à un remboursement maximum de 2205,90€. Cette somme est un maximum car, dans les faits, il est souvent remboursé moins, ce qui ne couvre pas complètement les coûts.

Bien que ce retard ministériel soit un frein assez net, car les centres hospitaliers perdent de l'argent en produisant des analyses d'exomes, les bénéfices sont importants sur les versants médicaux et scientifiques pour un coût global abordable (de l'ordre de 600 euros de réactifs par patient plus une importante part de main d'œuvre et d'investissement dans le matériel de séquençage et d'analyse). Cette technique permet un pourcentage diagnostic assez élevé (de l'ordre de 30 à 50% selon les indications cliniques (Clark et al., 2018)) et permet également la découverte de nouveaux gènes impliqués dans la DI. De plus, pour de meilleurs taux de diagnostic, il est fréquent de réaliser l'analyse de l'exome en trio, c'est-à-dire de séquencer les ADN du cas index et de ses parents. Ces informations permettent de filtrer les variations de façon rapide, par exemple, si les parents ne sont pas atteints par la maladie et qu'il n'y a pas d'autres individus atteints dans la famille, on peut partir du principe que la variation causale est survenue de manière*de novo*, c'est-à-dire qu'elle n'est apparue que chez l'enfant. On peut donc éliminer toutes les variations également présentes chez les parents, ce qui permet très rapidement de limiter l'analyse à une dizaine de variations au lieu de plusieurs dizaines de milliers. Cependant, cette technique à un coût multiplié par trois, car trois analyses d'exomes doivent être réalisées. Les analyses des exomes parentaux ne sont actuellement pas codifiées selon la nomenclature RIHN dans ce cas-là, ce qui représente un coût très important pour le laboratoire. Malgré tous les intérêts que peuvent présenter cette technique du trio, son utilisation en France est *de facto* limitée par les finances ministérielles.

3.3 Les scores

Le système de score permet de rapidement comparer des données hétéroclites en les réduisant à une valeur, le plus souvent numérique, définie sur une échelle. Avant de nous attarder sur l'utilisation des scores dans notre domaine d'intérêt, nous ferons un bref rappel historique, puis nous verrons les différents types de scores ainsi que leurs limites. Nous verrons enfin l'intérêt de ce système pour prioriser les variations génétiques.

3.3.1 Historique

Le mot *score* en français vient de l'anglais *score*, qui dérive lui-même du vieux norrois *skor* et qui signifie encoche. L'utilisation du mot score en français est attesté dans un contexte sportif depuis 1896 (Centre National de Ressources Textuelles et Lexicales, 2019). Cela signifie que la première trace écrite que nous ayons utilisant le mot score dans ce sens date de 1896. Par extension, d'autres résultats ont été définis comme des scores, notamment les sondages (attesté depuis 1948), les suffrages (attesté depuis 1964), les tests psychologiques (attesté depuis 1948) ou tout simplement les résultats en général (attesté depuis 1954). En anglais, l'utilisation du mot *score* dans un domaine sportif est antérieure, la première trace datant de 1742.

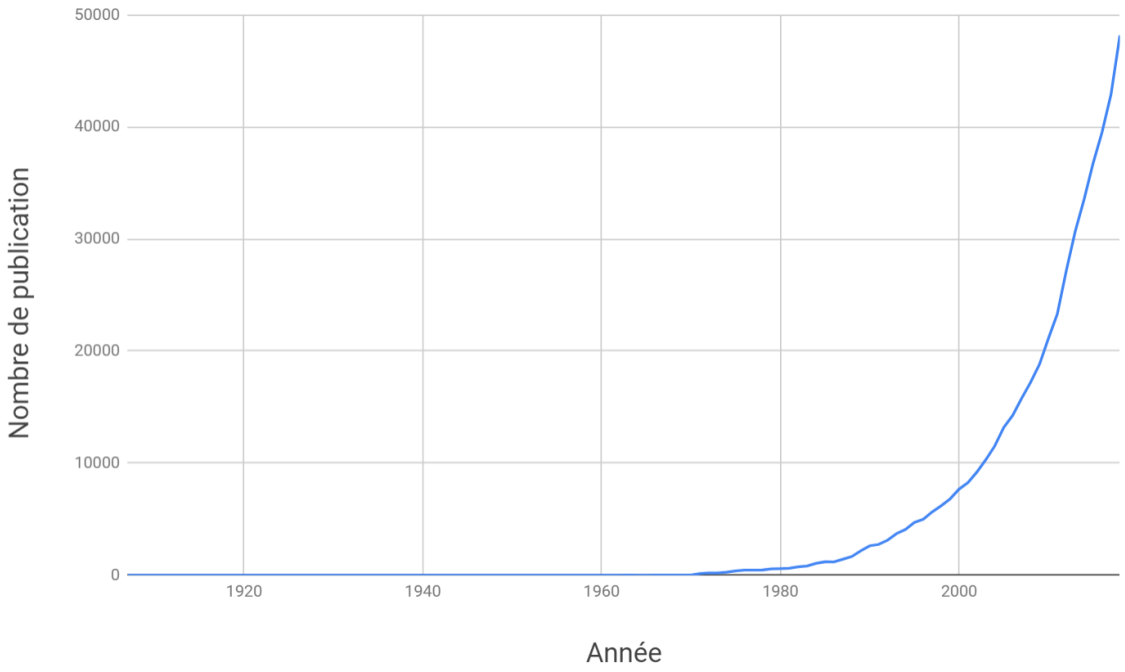


Figure 3.4 – Nombre de publications contenant la notion de score par année dans la base de données PubMed

Dans le domaine des sciences, l'utilisation du mot score n'a commencé à se développer que récemment elle est devenue exponentielle comme on peut le voir dans la figure 3.4. Cette figure représente le nombre de publications associées à un score sur la base de données PubMed de 1907 à 2018. On peut constater que l'évolution a commencé à partir des années 1980 et qu'elle est toujours en cours. Cette figure est uniquement illustrative, le nombre de publications ayant aussi augmenté, on ne peut en tirer de conclusions scientifiques, mais l'illustration est assez nette de la sur-représentation des scores dans les milieux biologiques et médicaux de nos jours.

3.3.2 Types de scores

Dans un contexte sportif, on peut distinguer plusieurs types de systèmes de scores. Le plus simple et probablement le plus fréquent est le score incrémental. À chaque fois qu'une action particulière se produit, le score est augmenté d'une certaine valeur. Trois paramètres principaux peuvent être distingués, qui permettent de trier ces systèmes de scores dans plusieurs sous-catégories :

— Le type d'incrémentation, qui peut être simple, avec une augmentation unité par unité (ex. : handball), ou plus complexe, avec des valeurs prédéfinies correspondant à des actions différentes (ex. : rugby).

— Le nombre de compteurs qui permet de différencier les compteurs uniques, présent dans la plupart des sports d'équipes, des compteurs multiples, que l'on retrouve plus souvent dans les sports de raquettes où les scores sont divisés en sets (ex. : badminton), eux-mêmes pouvant être sous-divisés (ex. : tennis).

— Le type d'échelle, qui peut être ouverte ou fermée. Une échelle ouverte est souvent associée à une limite de temps (basketball), alors qu'avec une échelle fermée, l'objectif est souvent d'atteindre la valeur la plus haute de l'échelle avant son adversaire (volleyball).

D'autres systèmes existent, notamment avec des déterminations de scores par jugement, comme dans la plupart des disciplines artistiques (gymnastique, patinage artistique). Dans ce cas, le score est défini par une échelle, le plus souvent de 0 à 10, et des juges situent la performance des athlètes sur cette échelle, ce qui permet des comparaisons plus objectives des performances. Il existe également un système de score par conversion, utilisé notamment dans le décathlon, ou les performances dans chaque épreuve, originellement mesurées dans des grandeurs physiques différentes (temps, distance), sont converties en une unité abstraite identique permettant leur addition.

Dans un contexte scientifique, deux types d'utilisations des scores existent. Le premier est leur utilisation pour quantifier des concepts abstraits, comme la similarité, la dispersion ou la confiance. Ce type d'approche est cependant très minoritaire et restreinte à certains domaines, comme par exemple cela peut être le cas dans le domaine des mathématiques appliquées aux statistiques.

La seconde et principale utilisation des systèmes de scores est une transformation de données qualitatives en données pseudo-quantitatives. C'est un procédé très fréquent en sciences humaines ou en biologie, en particulier en médecine, où il est particulièrement répandu. Cette utilisation a deux avantages principaux : le premier est une simplification statistique, car il est beaucoup plus facile d'effectuer des statistiques sur des données numériques que sur des données qualitatives. Le second avantage est une meilleure compréhension du problème et une analyse plus rapide de la situation. Par exemple, le score de *CHA2DS2-VASC* en médecine ("prononcé Chad-Vasc"), qui permet de rapidement estimer les risques d'accident vasculaire cérébral selon 8 critères. Ici, un score de 0 signifie un patient sans risques particuliers alors qu'un score de 2 à 9 marque une surexposition à un risque d'accident vasculaire cérébral. Il est ainsi beaucoup plus simple et parlant de dire que le patient a un score *CHA2DS2-VASC* à 6 que de lister les 8 critères. D'un point de vue statistique, c'est également beaucoup plus simple car 8 critères séparés sont rassemblés en une seule valeur référence.

3.3.3 Limites du score

Les systèmes de scores ont prouvé leur utilité dans des domaines variés, avec des applications diverses. Ils souffrent cependant de limites, qui doivent être gardées à l'esprit.

Si l'un des principaux avantages des scores est la simplification, la première limite est le pendant de cet avantage, avec une possible perte d'informations. Pour reprendre l'exemple précédent du score *CHA2DS2-VASC*, derrière un même résultat de score peuvent se trouver des profils très différents. Ainsi, une femme de plus de 75 ans aura le même score (3) qu'un patient jeune avec des complications classiques liées à l'obésité comme par exemple un diabète et des plaques d'athéromes.

Un autre exemple de cette simplification est le score de Glasgow, utilisé pour déterminer le degré de profondeur d'un coma en se basant sur des réponses motrices et verbales, ainsi que sur l'ouverture des yeux. Ce score donne les mêmes résultats pour un patient en état de mort cérébrale que pour un patient avec un "simple" coma éthylique.

Une seconde limite se trouve à la génération des scores, c'est l'effet boîte noire, avec une génération de scores si complexe qu'elle en devient parfois incompréhensible. Cette limite est peu fréquente dans le domaine de la médecine, car les scores y sont souvent simplistes pour être calculés rapidement de tête, mais on peut la retrouver par exemple dans le système

bancaire, où des scores de risques sont attribués à chaque emprunteur et à chaque projet. Ces scores sont générés par des algorithmes très complexes, prenant en compte de très nombreux facteurs, ce qui donne parfois des résultats étranges pour des cerveaux humains. Cette limite est importante, en particulier avec le développement croissant de nombreux scores générés avec des techniques de *machine learning* et d'algorithmes de plus en plus complexes.

Enfin, la dernière limite se trouve liée à la précédente, car elle est surtout dépendante de la génération du score. Il s'agit de l'introduction de biais dans le système de calcul. Cela concerne aussi les scores générés par ordinateur *via* un auto-apprentissage, car les données peuvent facilement être biaisées par des *corpus* non-représentatifs ou mal annotés, faussant les algorithmes. Un bon exemple de ce problème est le cas du score *Combined Annotation Dependent Depletion* (CADD) développé dans la section 3.4.2.1.

Cependant, cette introduction de biais peut aussi poser problème sur des scores beaucoup plus simples. On peut citer le cas du score *HAS BLED*, qui permet de déterminer la probabilité de saignement majeur chez des patients traités par anticoagulants. Ce score prend en compte l'âge du patient, comme le fait le score *CHA2DS2-VASC*, ce qui peut entraîner des situations ubuesques. Ainsi, un score *CHA2DS2-VASC* élevé indique un fort risque de caillot, il est donc recommandé aux médecins de traiter leurs patients avec des médicaments anticoagulants si le score est supérieur à 2. En revanche, un score *HAS BLED* élevé implique des risques élevés de saignements majeurs dans l'année chez des patients prenant déjà des anticoagulants. Dus à la présence de facteurs communs et d'imprécisions dans leurs définitions, ces scores peuvent entrer en conflit et conclure simultanément au besoin d'anticoagulants pour éviter un accident vasculaire cérébral et à un fort risque d'accident vasculaire cérébral en cas de prescription d'anticoagulants.

3.3.4 Application du score dans le domaine de la génétique

Le domaine de la médecine est globalement très friand de scores, comme on peut le voir sur les relevés de PubMed. La génétique ne fait pas exception, l'usage des scores y est assez développé, mais les cas d'utilisation ne sont pas forcément les mêmes que dans d'autres disciplines médicales. Sur le versant clinique, les scores peuvent être utilisés pour décrire les phénotypes de patients en tranchant des situations phénotypiques floues. Cette utilisation est relativement proche des autres utilisations médicales. Cependant, l'intérêt principal est la cotation des variants. On trouve plusieurs niveaux de scores pour cela, il y a bien sûr les différents scores de prédiction, dont certains seront développés dans les paragraphe 3.4.2.4 et 3.4.2.5. Ces scores se basent sur des algorithmes et / ou des grands *corpus* de variations pour déterminer la probabilité de pathogénicité d'une variation présente dans l'ADN du patient. Ils sont nombreux et diversifiés à cause du grand nombre de cas différents possibles. Une seconde utilisation des scores est la cotation d'une mutation selon les critères définis par l'*ACMG* (Sue Richards et al., 2015). L'idée est d'homogénéiser le classement des variations sur une échelle commune avec des critères communs. Cette échelle est composé de 5 classes : bénin, probablement bénin, signification inconnue, probablement pathogène, pathogène. Une série d'arguments précis est associée, par exemple : une fréquence allélique trop élevée est un argument fort pour un variant bénin, des études fonctionnelles montrant un effet délétère sont à l'inverse un argument fort pour une variation pathogène.

3.3.5 Comparaison du score par rapport à l'approche "filtres" pour prioriser les variations

Le principe de prioriser les variants avec un score global sert les mêmes objectifs que celui de les filtrer pour ne garder que les variants intéressants. Cependant, ces deux approches ont des principes assez différents, ce qui engendre des forces et faiblesses qui leurs sont propres. Les filtres sont extrêmement simples à mettre en place, ils sont très faciles à adapter par l'utilisateur même sans notion informatique, un simple tableur convient parfaitement. Ils

permettent de réduire considérablement les données afin d'être certain de ne pas traiter de données inintéressantes. Cependant, cet aspect est aussi responsable de leur faiblesse principale, plus le nombre de critères filtrés est important, plus le risque d'obtenir des variants qualifiés de faux-négatifs l'est également. Pour ce type d'application, cela peut devenir très contraignant, car l'analyse d'un patient où aucune variation pathogène n'est identifiable ne sera pas rendu rapidement. D'autres analyses seront plutôt effectuées en relâchant les filtres petit à petit, pour vérifier qu'un variant d'intérêt n'a pas été supprimé. Cette méthode est très efficace tant que les données sont parfaitement annotées et que les critères sont bien filtrables, mais peut être très complexe lorsqu'il n'y pas de variation pathogène dans le fichier de résultat. Certains tris peuvent également être complexes, notamment sur des données qualitatives, comme le type de variation par exemple (insertion, délétion, substitution, etc...).

Les scores n'éliminent pas de variants, ce qui compense la principale faiblesse des filtres, cependant, ils conservent toutes les données, ce qui peut amener une analyse fastidieuse. Ils sont également beaucoup plus complexes à développer et nécessitent forcément l'intervention d'un bio-informaticien ainsi qu'un consensus sur les échelles à mettre en place. Ils doivent également être adaptés aux données, ce qui rend leur maintenance difficile et leur portabilité quasiment-nulle. Enfin, ils doivent surtout être consensuels, ce qui est également compliqué dans un contexte de pratiques, d'habitudes et d'analyses très diversifiées. Leur utilisation dans les outils de la littérature n'est que très peu présente et se restreint à des cas d'utilisation précis. À l'inverse, les processus de filtration étant faciles à créer, de nombreux outils les intègrent directement.

3.3.6 Type de score biologiques

Dans le domaine de la biologie, en particulier pour la génétique, deux grands types de scores existent. Il s'agit des scores calculés et des scores générés par des intelligences artificielles.

3.3.6.1 Les scores calculés

Nous appellerons scores calculés les scores bio-informatiques qui sont générés par des algorithmes de calcul. Le plus simple et le plus ancien d'entre eux est le score *Grantham* (Grantham, 1974) qui se calcule à partir d'une matrice. Ce score n'est pas vraiment bio-informatique, sa mise en place date de 1974, mais il est désormais calculé pour toutes les variations faux-sens. D'autres bons exemples de ce type de scores sont les scores de conservations, qui sont calculés en fonction de la conservation du nucléotide ou de l'acide aminé parmi plusieurs génomes d'espèces différentes ou plusieurs protéines / séquences fonctionnellement proches. La dernière grande catégorie de scores calculés sont les scores à nombres magiques, qui sont définis empiriquement par des utilisateurs ou des développeurs. C'est par exemple le cas du score *VaRank* (Geoffroy et al., 2015) qui associe un score arbitraire en fonction du type de variation et d'autres facteurs annexes. L'objectif de ces scores est de permettre une comparaison rapide de plusieurs facteurs et de les traduire dans une seule valeur numérique plus facilement comparable. Leur utilisation est très souvent conditionnée par la justesse des critères et des seuils empiriques définis par les auteurs aux yeux des utilisateurs.

3.3.6.2 Les scores générés *via* des intelligences artificielles

Les techniques de *machine learning* ont révolutionné le champ des scores bio-informatiques. Si les premiers scores étaient tous des scores calculés à partir de données de références ou empiriques, ce nouveau type de scores s'est désormais répandu et intégré dans le paysage de l'annotation bio-informatique. Le principe est de constituer des bases de données contenant de très nombreuses variations (plusieurs millions) et de disposer d'une liste de variations pathogènes. Les algorithmes de *deep learning* vont alors établir leurs propres critères pour déterminer les valeurs qui définissent une variation pathogène et ils vont en déduire des règles.

L'application de ces règles à de nouvelles variations permet d'évaluer le possible caractère pathogène.

Ces scores ont une efficacité remarquable, mais souffrent de plusieurs limites majeures. La première est probablement la difficulté que représente leur mise au point : des compétences en informatiques ainsi que des infrastructures conséquentes sont nécessaires. Ils sont peu évolutifs, il est nécessaire de relancer l'ensemble du mécanisme d'apprentissage pour générer un nouveau score si de nouvelles données sont à prendre en compte. Les données sont d'ailleurs l'un des points noirs, car elles peuvent amener des biais si elles ne sont pas rigoureusement choisies. Par exemple, si les données catégorisées "saines" sont en réalité incertaines, la génération de règles peut en devenir extrêmement complexe, en empêchant l'intelligence artificielle de trouver des critères pertinents, ce qui va nuire à l'utilisation du score par la suite en diminuant son efficacité. Enfin, le dernier point négatif est l'effet "boîte noire", car personne ne sait exactement pourquoi une variation est annoncée pathogène, ce point empêche également les généticiens de tenir compte des limites de ces scores lors de l'analyse. En effet, les valeurs prédites par les scores sont toujours nuancées par les généticiens, en fonction de l'adéquation entre les spécificités du score et la variation notamment. Comme les critères de décision du score ne sont pas connus, il ne peut être nuancé facilement par les généticiens. À cause de ces limites, ces scores de prédiction sont souvent utilisés comme arguments supplémentaires pour appuyer une variation pathogène, mais ne sont pas utilisés comme critère premier. Ils peuvent cependant permettre de faire ressortir des variations intéressantes.

3.4 Tâches du généticien

L'objectif du généticien est d'extraire de la liste complète des variations celle(s) qui est / sont pathogènes. Il est tout à fait possible qu'il n'y ait pas de variations pathogènes dans les données, cela peut être le cas si la pathologie est causée par un mécanisme non génétique, ou tout simplement si elle est située hors des régions couvertes par le séquençage. La tâche consiste à analyser chacune des variations à l'aide de ses connaissances et expériences personnelles, ainsi que de l'appui de documents bibliographiques, et d'en déduire la pathogénicité potentielle. Comme la liste de variations est très conséquente, environ 60 000 entrées, le généticien ne les analyse pas toutes mais les priorise en fonction de leur pathogénicité probable. Pour cela il les filtre en éliminant les variations probablement bénignes.

Pour déterminer la pathogénicité, le généticien se base sur les annotations de la variation comme décrit plus haut dans ce chapitre (3.1.3.4). Plus les annotations sont complètes, précises, fiables et à jour, plus la tâche du généticien est facilitée. Si quelques outils commencent à utiliser des scores pour prioriser les variations plutôt que de les filtrer, notamment QueryOR (Bertoldi et al., 2017), cette approche reste encore très minoritaire dans la littérature. Le détail de ces outils est développé au début du chapitre IV. Selon les laboratoires, des préfiltres peuvent être effectués par les bio-informaticiens pour simplifier l'échange de fichiers. Cela ne change pas la tâche, les seuils restant définis par les généticiens.

Il existe une variabilité très importante des méthodes pour accomplir cette tâche. Selon les généticiens, les filtres ne seront pas effectués en fonction des mêmes critères. C'est pour cette raison que les annotations sont aussi nombreuses, avec en particulier des notions similaires qui sont présentent plusieurs fois émanant de sources différentes, afin de recouper les données et d'avoir une redondance de l'information. C'est également à cause de cette variabilité que le résultat de l'analyse d'un exome est généralement déterminé par différents professionnels (généticiens biologistes et généticiens cliniciens), qui mettent en commun leurs résultats et leurs connaissances pour définir le diagnostic. Cette confrontation des données clinico-biologiques permet de décider collégialement s'il faut poursuivre l'étude d'une variation donnée dans la famille du patient par exemple, ou dans le cas de patients sans variant candidat, de décider de la poursuite des analyses à effectuer.

3.4.1 Outils d'annotation

Les logiciels d'annotation, aussi appelés annotateurs, sont des outils qui prennent en entrée un ou plusieurs fichiers *Variant Call Format* (VCF), et à l'aide de bases de données de références sélectionnées, qui vont fournir en sortie sous la forme de fichiers tabulaires des variations annotées, c'est-à-dire enrichies en informations. Il en existe plusieurs, mais deux sont des références internationales incontestables, *ANNOVAR* (K. Wang, M. Li et Hakonarson, 2010) et *Variant Effect Predictor* (VEP) (McLaren et al., 2016). Il existe d'autres annotateurs, par exemple *Genotator* (Wall et al., 2010) ou *Artemis* (Rutherford et al., 2000), ainsi que *VaRank*(Geoffroy et al., 2015), c'est ce dernier que nous avons choisi d'ajouter à *ANNOVAR* et VEP dans notre processus d'analyse de nos données.

3.4.1.1 *ANNOVAR*

ANNOVAR a été publié en 2010 par Kai Wang (K. Wang, M. Li et Hakonarson, 2010). C'est un outil en ligne de commandes open source, codé en *Perl*. Depuis sa publication, une version web a été implémenté, *wANNOVAR*[1], qui est simplement la version standard sur un serveur web, mais qui permet aux utilisateurs ne maîtrisant pas les lignes de commandes de le faire fonctionner. Plusieurs utilisations sont possibles, notamment des approches par gène, par région génomique, ou par variation. La première approche va surtout se baser sur des bases de données de références à l'échelle du gène, comme *Ensembl*[2] ou *RefSeq*[3]. Le but est de visualiser l'impact d'une variation sur la protéine produite. L'approche par région est centrée sur les remaniements larges, comme les *Copy Number Variation* (CNV). Les annotations vont concerner les régions conservées chez différents organismes modèles, ainsi que sur des bases de données spécialisées dans les régions intergéniques et les modifications structurales. Enfin, l'approche par filtre est la plus proche de ce qui est effectué dans le domaine des maladies rares. Elle combine des informations de fréquences dans la population générale selon plusieurs sources, des logiciels prédicteurs de pathogénicité ainsi que des bases de données spécialisées dans la description de variations pathogènes.

Deux commandes sont nécessaires quelle que soit l'approche. La première permet de préparer le fichier VCF d'entrée et de le convertir en un format spécial pour *ANNOVAR*, la seconde accomplit l'étape d'annotation à proprement parler. Le téléchargement des bases de données est réalisé *via* des arguments de commandes spécifiques. On dénombre plus d'une centaine de bases de données disponibles en téléchargement, mais certaines sont très anciennes et plusieurs versions de la base sont souvent présentes. Par exemple on trouve les versions de *dbSNP128* à *dbSNP150*. La version actuelle de *dbSNP* étant la 153, disponible depuis aout 2019, alors que la version la plus à jour d'*ANNOVAR* date de février 2017.

Ce système de mise à jour est une des limites majeures d'*ANNOVAR*, car cela nuit à la réactivité bibliographique des utilisateurs. Cette limite vient en plus des limites classiques des annotateurs, notamment l'effet couloir, qui réduit les possibilités de l'utilisateur aux seules bases de données sélectionnées par les développeurs. Sur ce point, *ANNOVAR* est plus performant que la plupart de ses concurrents, car son catalogue de bases de données est vaste et permet plusieurs approches d'analyses, ce qui est très pertinent. Enfin, les limites techniques, telles que l'utilisation en ligne de commandes, repoussante pour un généticien, ou le téléchargement de grande quantité de données, sont amoindries par la mise en place du logiciel en tant que service web. Cette version en ligne d'*ANNOVAR* ne règle cependant pas tous les problèmes, car c'est une solution qui apporte aussi certains inconvénients notamment en terme de disponibilité et de ressources réseaux, mais qui offre tout de même une certaine facilité d'utilisation bienvenue.

1. `http://wannovar.wglab.org/`
2. `https://www.ensembl.org/index.html`
3. `https://www.ncbi.nlm.nih.gov/refseq/`

3.4.1.2 *VaRank*

VaRank a été publié en 2015 par Geoffroy et collaborateurs (Geoffroy et al., 2015). C'est un outil en ligne de commandes, codé en *Tcl/Tk* destiné à l'annotation et à la priorisation de variants. L'annotation n'est pas directement effectuée par *VaRank* mais sous-traitée à deux annotateurs, *SnpEff* (Cingolani et al., 2012) ou *Alamut Batch* (Biosoftware, 2014). *VaRank* prend une liste de fichiers VCF en entrée, effectue l'annotation *via* au moins l'un des deux moteurs, ajoute des concepts et produit des fichiers structurés en sortie. Deux innovations sont importantes : le code barre *VaRank* ainsi que son score.

Le code barre est une visualisation des allèles de tous les fichiers d'entrées. Contrairement à d'autres annotateurs, *VaRank* est utilisé plutôt sur un groupe d'exomes provenant de plusieurs patient plutôt qu'individuellement. Cette dernière utilisation n'est pas impossible, mais prive *VaRank* de l'un de ses meilleurs atouts, son code barre. En effet, *VaRank* produit un code barre contenant autant de chiffres que de patients. Chaque chiffre correspond à l'haplotype du énième patient, 0 si l'allèle n'est pas présent, 1 si l'allèle est présent à l'état hétérozygote et enfin 2 si l'allèle est présent à l'état homozygote. *VaRank* produit également ces informations sous la forme de colonnes, avec pour chaque variation le nombre d'allèles, la fréquence, le nombre d'homozygotes et d'hétérozygotes parmi les patients analysés. Il est de plus possible de grouper les patients par famille, ce qui est très utile dans le cas d'analyses en trio, fréquentes pour l'analyse des exomes. Cette fonctionnalité est très intéressante et permet l'analyse rapide de trio ainsi que la mise en place de filtres puissants. *VaRank* souffre de quelques limites, notamment le manque de discernement sur les chromosomes sexuels, conduisant à l'interprétation des hémizygotes comme des homozygotes, ce qui peut être gênant dans certains cas, car cela augmente artificiellement la fréquence allélique et peut mener à des confusions.

VaRank présente enfin un score particulier, détaillé dans la section 3.4.2.3. Globalement, c'est donc un excellent annotateur, car il produit plus que de l'annotation, il présente également des informations plus structurées que les sorties standards d'annotateurs. Le passage à l'échelle du génome n'est cependant pas vraiment envisageable, les calculs devenant très longs (plusieurs jours), associés à des valeurs trop grandes pour que *Tcl* puisse les gérer, ce qui se traduit concrètement par un abandon de l'annotation avant les résultats. *VaRank* reste cependant un annotateur très compétitif sur les exomes et les panels de gènes.

3.4.1.3 *Variant Effect Predictor*

Variant Effect Predictor, plus souvent appelé VEP, est un annotateur développé par le projet *Ensembl* (McLaren et al., 2016). C'est un logiciel en ligne de commandes développé en *Perl*. VEP est une référence des annotateurs, il fonctionne sur une base de *packages* pouvant être ajoutés ou non par l'utilisateur, ce qui permet de diversifier les entrées, selon un principe un peu similaire à celui d'*ANNOVAR*. Ce fonctionnement en *packages* permet aussi de diviser les données de références dont le téléchargement est nécessaire, les utilisateurs n'ayant à télécharger que les données des *packages* qu'ils utilisent.

Une version web existe, permettant l'annotation de petits fichiers, issus de panels réduits par exemple. Cette version vise un public de généticiens ou de scientifiques ayant occasionnellement besoin d'annotations, mais n'a pas pour vocation d'être utilisée en routine diagnostique, surtout pour des données volumineuses. VEP est un annotateur très rapide dû à sa bonne gestion des ressources informatiques, permettant de paralléliser l'analyse, ce qui permet d'envisager des annotations à l'échelle du génome sans temps de calculs trop long.

3.4.2 Exemples de scores représentatifs

Les scores génétiques sont nombreux. Dans cette partie nous en détaillerons un petit échantillon représentatif permettant d'apercevoir leur diversité.

3.4.2.1 CADD

Le score CADD a été mis au point en 2014 par Martin Kircher (Kircher et al., 2014). Il découle de l'observation du manque de critères pris en compte lors de l'analyse génétique d'un variant, les généticiens se focalisant sur peu d'entre eux. Le CADD, pour *Combined Annotation Dependent Depletion* est une méthode pour prendre en compte automatiquement différentes annotations et de les convertir en un seul score, il permet donc une simplification et une variation des sources et des types d'informations. Le score CADD a été développé *via* des techniques de *machine learning* en entraînant le modèle sur 14,7 millions de variations générées par ordinateur.

C'est un score pré calculé pour toutes les variations possibles sur le génome, c'est-à-dire environ 8.6 milliards de variations. Il est plutôt efficace et est considéré comme une référence encore aujourd'hui. Il est d'ailleurs fréquemment mis à jour. Cependant, des publications récentes (Mather et al., 2016) soulignent les limites de ce score, qui de par son *corpus* de pseudo-variations, mésestime la pathogénicité de certains types de variations à cause de l'inadéquation entre les données générées et les données réelles.

3.4.2.2 *DOMINO*

DOMINO est un outil de prédiction de dominance des gènes basé sur du *machine learning* et publié en 2017 par Quinodoz (Quinodoz et al., 2017). L'objectif de ce score est de déterminer automatiquement la probabilité qu'un gène soit de transmission autosomique dominante. Il se place donc à l'échelle du gène plutôt que du variant, le score étant associé au gène. Il se base sur un grand nombre de facteurs, notamment la conservation du gène parmi différentes espèces, l'expression des gènes, les interactions protéiques et les structures. *DOMINO* a été entraîné sur 985 gènes dont les modes de transmission étaient bien connus et documentés.

Il a ensuite été testé sur des nouveaux gènes au fur et à mesure que les connaissances se développaient, avec pour résultats des prédictions plutôt satisfaisantes. Il a de plus été mis au point particulièrement sur des gènes impliqués dans l'épilepsie et la DI, il est donc adapté à ce type de recherche. C'est un bon score pour prédire le mode de transmission de gènes peu documentés, il permet d'évaluer rapidement l'intérêt que peut avoir un variant dans un gène candidat.

3.4.2.3 *VaRank*

Une autre fonctionnalité intéressante de *VaRank* est son score, s'échelonnant de 0 à 110. Ce score est un bon exemple de score "magic number" comme défini précédemment, les valeurs étant fixées arbitrairement. Il n'en reste pas moins utile pour trier rapidement les variations selon leur potentiel délétère. Ici encore, quelques limites sont présentes, notamment la sous-estimation des variants d'épissage, particulièrement marquée si l'annotation n'est faite qu'avec *SnpEff*. Néanmoins, ce score présente l'intérêt de rendre le type de mutation triable simplement, puisque les valeurs sont numériques et non plus textuelle, ce qui est très utile pour les généticiens.

3.4.2.4 *GERP++*

Le score *GERP++* (Davydov et al., 2010) est la suite d'un premier score, *Genomic Evolutionary Rate Profiling* (GERP) (Cooper, Stone et al., 2005). C'est un score de conservation qui utilise une approche "*bottom-up*" contrairement aux autres scores classiques de conservation, tel que *PhastCons*, qui utilise plutôt une approche basée sur des modèles de Markov cachés, "*Hidden Markov Model*", pour estimer les contraintes de conservation pesant sur les nucléotides. L'approche de *PhastCons* peut mener à des mésestimations des régions conservées, une limite que ne présente pas l'approche *bottom-up* de *GERP++*. Le principe est

d'estimer la contrainte de conservation sur chacun des nucléotides, puis de diminuer au fur et à mesure la résolution pour chercher des régions conservées de plus en plus grandes.

Si GERP utilisait déjà cette approche, *GERP++* est désormais la référence, car il est beaucoup plus rapide et statistiquement plus robuste. C'est l'un des scores de conservation les plus complets à la disposition des généticiens. Il permet d'estimer rapidement la pression de sélection sur un nucléotide : plus cette pression est élevée, plus une variation sur ce nucléotide est susceptible d'être dommageable puisque les nucléotides conservés sont ceux qui sont probablement les plus utiles dans l'information génétique. Il est à noter qu'une variation sur un nucléotide peu conservé n'est pas pour autant automatiquement bénigne.

3.4.2.5 *SIFT*

SIFT pour *Sorting Intolerant From Tolerant* est un programme qui prédit la pathogénicité probable d'une substitution d'acide aminé présenté par Pauline Ng et Steven Henikoff en 2003 (Ng et Henikoff, 2003). Il se concentre sur les variations d'un seul nucléotide qui cause des variations faux-sens, c'est-à-dire le remplacement d'un acide aminé par un autre dans la séquence de la protéine. Le taux de faux-positif de SIFT est élevé, mais il permet cependant de donner un argument supplémentaire pour des variations qui ne sont pas faciles à interpréter. Les variations faux-sens peuvent en effet avoir des effets très délétères ou aucun effet, même sur des gènes très importants, en fonction de leur localisation dans la structure tridimensionnelle de la protéine et de leur éventuel impact sur le changement conformationnel de cette même protéine.

Pour effectuer ses prédictions, l'algorithme de SIFT se base surtout sur les propriétés physico-chimiques des acides aminés et utilise des données de conservation avec une approche intelligente. Par exemple, si une position de la protéine ne comporte que des isoleucines et que cette isoleucine est remplacée, il est probable que ce changement ait des effets délétères. En revanche, si à cette position on trouve des isoleucines, des valines et des leucines, la plupart des prédicteurs considèrent que l'acide aminé n'est pas critique. SIFT prend en compte le type d'acide aminé, par exemple l'isoleucine, la leucine et la valine sont tous les trois hydrophobes, SIFT prédit donc qu'un changement de cet acide aminé par un autre qui n'est pas hydrophobe peut avoir des effets délétères. Les données de références sont les séquences protéiques proches, c'est-à-dire les protéines remplissant les mêmes fonctions chez l'organisme ou chez des espèces proches. La fiabilité des prédictions de SIFT est donc meilleure si les bases de données disposent de nombreuses séquences homologues pour référence et comparaison. Malgré cette limite, l'algorithme de SIFT reste plus performant qu'une simple matrice Grantham ou qu'un score de conservation brut même avec peu de séquences homologues.

3.4.2.6 *PolyPhen-2*

PolyPhen est un outil permettant de prédire l'impact d'une substitution d'acide aminé, il remplit exactement les mêmes fonctions que SIFT, les deux logiciels sont donc très souvent associés. La version actuelle de *Polyphen*, *PolyPhen-2* a été publiée en 2010 par Ivan Adzhubei et al (I. A. Adzhubei et al., 2010)(I. Adzhubei, Jordan et Sunyaev, 2013). Son utilisation est très similaire à celle de SIFT, il est utilisé en tant qu'argument supplémentaire pour évaluer le caractère pathogène d'une substitution d'acide aminé. Son fonctionnement est également relativement proche, il se base sur des données phylogénétiques de plusieurs protéines pour estimer l'impact de la substitution. L'algorithme est un petit peu plus complexe, il contient des notions de *machine learning* et utilise également les données de structure tridimensionnelle lorsqu'elles sont connues pour évaluer l'effet de la substitution.

Ces deux outils, SIFT et *PolyPhen-2* sont très proches que ce soit d'un point de vue algorithmique, même si la nouvelle version de *PolyPhen* est désormais plus complexe, ou d'un point de vue de son utilisation. Ils reflètent bien la multitude d'outils présents dans le domaine de la génétique et de la bio-informatique, puisqu'aucun des deux n'a réussi à

supplanter l'autre. Ces deux outils sont quasiment systématiquement employés côte à côte et très peu de généticiens les utilisent comme base de filtre ou de tri, ils servent d'arguments pour estimer précisément la pathogénicité d'un variant, mais n'emportent pas une décision à eux seuls.

3.4.3 Principales bases de données utilisées en génétique

Comme les scores, les bases de données utilisées en génétique sont nombreuses et variées. Dans cette section nous en détaillerons quatre qui sont représentatives de différentes approches.

3.4.3.1 *ClinVar*

Dans le domaine de la génétique médicale, *ClinVar* est une référence très importante, publiée en 2015 par Melissa Landrum (Landrum et al., 2015). C'est une base de données contenant des interprétations de variants par des généticiens. Les généticiens ne réalisent pas de réel travail d'interprétation pour tous les variants, ils se concentrent sur les quelques-uns qui subsistent après les filtres. Pour ces variations là, ils peuvent les saisir *via* une interface web très détaillée et indiquer l'interprétation clinique qu'ils en ont fait. On trouve donc dans *ClinVar* une liste de variants associés à des avis de généticiens correspondant aux classes de l'*ACMG*. On peut également y trouver le nombre de généticiens ayant fait cette interprétation, les éventuels conflits ainsi que d'autres informations cliniques.

C'est une ressource très précieuse pour les généticiens, car si elle n'élimine pas le besoin d'interprétation, elle permet de prioriser rapidement, puisqu'un variant déjà rendu pathogène par plusieurs généticiens est très probablement intéressant. Ce qui fait la force de *ClinVar* fait aussi sa faiblesse, les entrées sont réalisées par des utilisateurs. Les données ne sont donc pas toujours d'une fiabilité absolue, bien qu'une étape de relecture par une tierce personne (personnel du NCBI) permette de limiter les erreurs. Pour contrebalancer ceci, les conflits d'interprétations sont clairement spécifiés, le nombre de "votes" pour chaque interprétation est précisé, ainsi que les arguments des différents partis. Cette base ne remplace donc pas le travail d'interprétation des variants et ne peut être utilisée pour filtrer les variations, mais est un critère de tri très efficace et très utilisé par les généticiens pour gagner en efficience dans leurs analyses.

3.4.3.2 OMIM

OMIM pour *Online Mendeleian Inheritance in Man* est le pendant informatique de *Mendeleian Inheritance in Man* (MIM), publié initialement en 1966 (Victor Almon McKusick et al., 1966,Victor A McKusick, 2014). La version en ligne est quant à elle publiée en 2005 par Ada Hamosh (Hamosh et al., 2005). L'évolution est donc l'accès à des mises à jour très régulières sans attendre la publication d'un recueil, ainsi qu'un système de requêtes beaucoup plus simple qu'une fastidieuse recherche dans un recueil imprimé. Les entrées MIM peuvent définir un gène ou une maladie. Elles sont constituées d'un identifiant semi structuré de 6 chiffres qui décrit partiellement le type d'informations qu'il identifie. À chacune des entrées est associé un descriptif, les références bibliographiques correspondantes, une description des phénotypes associés, ainsi que des liens vers d'autres entrées OMIM pour poursuivre la recherche.

OMIM est un outil primordial dans la génétique médicale, les entrées y sont complètes et à jour et servent de référence aux généticiens. Cependant, même si le nombre d'entrées est exponentiel, seule une fraction est décrite sur OMIM. C'est un avantage et une limite, les généticiens s'en servent pour trier les variations en fonction des gènes, pour se concentrer sur les gènes répertoriés dans OMIM. Ce tri ne se base pas sur des critères biologiques, mais plutôt sur un principe d'interprétation : sans informations disponibles sur le gène et le variant, il est impossible de l'interpréter. Les généticiens commencent donc par étudier les variations dans les gènes sur lesquels ils disposent des informations OMIM, car ils peuvent les interpréter. Ce

filtre impliquant la base de données OMIM permet également de réduire de façon importante les variations (d'environ 80%). Cependant cette approche n'a pas de sens d'un point de vue recherche, car elle ne permet pas de découvrir de nouveaux gènes. C'est un compromis permettant d'augmenter grandement l'efficience au prix d'une légère baisse de l'efficacité et de la découverte de nouveaux gènes.

3.4.3.3 *dbSNP*

dbSNP est comme son nom l'indique une base de données contenant des *Single Nucleotide Polymorphism* (SNP), publiée en 2001 par S. Sherry (Sherry et al., 2001). On parle maintenant de *Single Nucleotide Variation* (SNV) pour désigner les SNP, afin de s'affranchir des notion de fréquence allélique. L'idée de *dbSNP* est de cataloguer les variations à l'échelle du génome. Les SNV étant des variations très communes, la grande majorité n'a aucun impact phénotypique, mais quelques-uns peuvent avoir des effets pathogènes importants. L'objectif de *dbSNP* est de catégoriser tous les SNV du génome et de les associer avec d'autres données, notamment associés aux gènes, aux protéines, à de la pharmaco-génétique, des données de biologie de l'évolution et de biologie des populations. Les variations sont associées à un identifiant unique nommé *rsID* qui permet de simplifier la recherche et la discussion autour des variations génétiques. Les variations concernent principalement les humains, mais des versions de *dbSNP* dédiées aux organismes modèles existent, par exemple pour la souris.

La base de données est hébergée par le *National Center for Biotechnology Information* (NCBI), un acteur majeur de la biologie en général qui héberge une collection de ressources importantes. Cet hébergement permet de lier facilement les SNV à d'autres ressources disponibles également sur le site NCBI, notamment *PubMed*, *ClinVar* ou *GenBank*. C'est une base de données importante pour l'annotation, car elle permet l'agrégation de données non seulement à l'échelle du gène mais également à l'échelle de la variation. De nouvelles versions sont publiées régulièrement pour être utilisées hors ligne par des annotateurs. La version actuelle est *dbSNP153* depuis août 2019.

3.4.3.4 SysID

Systems biology approaches to Intellectual Disability (SysID) est une base de données de gènes impliqués dans la DI publiée par Korinna Kochinke en 2016 (Kochinke et al., 2016a,Kochinke et al., 2016b). C'est une base de données de petite taille, qui contient seulement quelques centaines de gènes, mais qui sont tous suivis par une équipe de généticiens et qui s'appuie sur la littérature scientifique. Cette base de données a pour avantage principal d'être mise à jour très régulièrement par des experts du domaine, les données sont donc en principe très fiables. Les descriptions des gènes comprennent des liens vers d'autres ressources, notamment OMIM ou *Gene Ontology* (GO), mais aussi des descriptifs cliniques des patients. Cette base de données est donc doublement importante pour les généticiens, d'une part elle permet de mieux comprendre globalement la DI, en montrant les voies métaboliques concernées par exemple. Cet aspect permet de cibler de nouveaux gènes candidats par leur proximité avec des gènes connus. D'autre part, elle est très utile pour un contexte diagnostic, les informations y étant en l'état actuel des connaissances, aussi fiables que possible. Cela permet donc aux généticiens d'être très rapidement à jour sur les nouveaux gènes publiés et d'avoir accès à des synthèses des dernières publications scientifiques liés à la DI.

Cette base illustre bien la dynamique du domaine, avec des fréquences de mises à jour très importantes. Il est en revanche plus complexe d'intégrer ces données dans un *pipeline* figé, puisque le fichier de référence doit être régulièrement actualisé. Cette base est donc un autre exemple de la dépendance des généticiens vis-à-vis des bio-informaticiens, car soit ils doivent se contenter de l'interface web, donc rechercher les gènes un par un, soit inclure SysID dans le *pipeline* d'annotation, donc dépendre du bio-informaticien pour chaque mise à jour.

Comme nous l'avons vu, les bases de données génétiques sont variées dans les formats, dans

les objectifs, dans les approches et dans les concepts stockés. Ces bases de données ne sont pas utilisées de façon isolée, mais toujours conjointement. Cette utilisation permet de bénéficier des forces des différentes bases, tout en limitant leurs faiblesses. Cela pose cependant deux problèmes, l'interopérabilité des données, mais surtout cela démultiplie les problématiques de mises à jour, car chaque base de données doit être maintenue.

3.5 Manipulation de base de données

Une base de données est une structure informatique permettant de stocker et d'organiser des données et d'y accéder *via* des requêtes. Les données sont classiquement structurées pour faciliter ces étapes de requêtes et pouvoir restituer l'information le plus rapidement possible. L'ajout d'informations dans la base de données se fait classiquement petit à petit, en ajoutant des entrées. Il peut arriver que l'on ait besoin de fusionner des bases de données. En cas de différence de structure et / ou de données hétérogènes, cette tâche peut vite devenir complexe. Les ontologies permettent à la fois de mieux anticiper ce type de besoins et d'apporter des solutions pour la manipulation de bases de données complexes.

Les annotateurs actuels utilisent les bases de données séparément, mais pouvoir fusionner plusieurs bases de données pourrait être un gain de temps pour le développeur et pour l'utilisateur. Cette fusion serait un premier pas vers un annotateur plus complet, qui pourrait permettre d'incorporer à façon des bases de données d'intérêt pour l'utilisateur dans sa banque personnelle de bases références utilisées pendant le processus d'annotation.

3.5.1 Ontologies et définitions

Les ontologies sont définies initialement par Gruber (Gruber, 1993) comme "la spécification explicite d'une conceptualisation". En informatique, les ontologies servent à modéliser les connaissances et à les structurer de façon explicite et consensuelle. Une autre définition plus récente, plus complète et certainement plus claire est celle fournie par Pierra (Guy Pierra, 2003) :"*Représentation formelle, explicite, référençable et consensuelle de l'ensemble des concepts partagés d'un domaine sous forme de classes, de propriétés et de relations qui les lient*". Cette définition permet de poser les caractéristiques capitales d'une ontologie ainsi que ses composants.

Le formalisme signifie que les différents concepts seront décrits dans un langage référence. Plusieurs langages de définitions et modèles existent, on peut citer rapidement le modèle PLIB (Pierra et al., 1998), ainsi que les langages *Web Ontology Language* (OWL) (M. K. Smith, 2004) et *Ressource Description Framework Schema* (RDFS) (Brickley, Guha et McBride, 2014). L'aspect explicite signifie qu'aucune information supplémentaire n'est nécessaire pour comprendre l'ontologie et qu'elle est vraie quel que soit le contexte. Le référencement se fait simplement *via* des identifiants uniques, dépendant du formalisme adopté. Ce système de références permet d'éviter des quiproquos ou des ambiguïtés. Enfin, le consensus est l'aspect principal d'une ontologie, elle n'a aucune valeur sans ce point. Tous les utilisateurs doivent être d'accord sur la validité de l'ontologie. Ce consensus permet ensuite de s'assurer que tout le monde parle des mêmes concepts derrière les mêmes mots, ce qui permet notamment une interopérabilité des bases de données en supprimant les conflits sémantiques.

Concernant ses composants, une ontologie est composée de concepts liés entre eux par des relations. Les concepts sont complétés par des propriétés permettant de décrire leurs différentes caractéristiques. Concepts et propriétés sont des termes génériques, pour désigner un concept en particulier, on parlera d'une instance. Il peut y avoir d'autres composants pour inférer du savoir, notamment des axiomes qui sont des règles toujours vraies dans le domaine décrit par l'ontologie. Si aux débuts des ontologies informatiques, leurs premiers emplois étaient dans les domaines de l'ingénierie des connaissances, les ontologies sont maintenant utilisées dans des domaines très variés, notamment la médecine, l'aéronautique, la sémantique

ou encore l'intelligence artificielle. L'exemple probablement le plus connu dans le domaine de la génétique est *Gene Ontology* (GO) (Ashburner et al., 2000) dont le but est de gérer, enrichir et structurer le vocabulaire des processus biochimiques et génétique. Il s'agit, comme la plupart des ontologies biologiques, d'une ontologie très simple, avec peu de types de relations (Aranguren, 2005). L'objectif ici n'est pas d'inférer du savoir pour découvrir de nouvelles règles, mais simplement de structurer les connaissances et le vocabulaire employé pour les décrire.

Il n'existe pas d'ontologie complète du domaine de la génétique pour plusieurs raisons, les deux principales étant son étendue et son instabilité. En effet, comme le décrit Kevin Royer (Royer, 2015) le développement d'une ontologie globale nécessite un domaine clairement défini et fini, ainsi que la capacité des acteurs à aboutir à un consensus global. Aucune de ces deux hypothèses n'est valide dans le domaine de la génétique où les bornes sont encore en mouvement et les débats sont toujours en cours sur de nombreux sujets. Le processus de création étant long et fastidieux, il n'est d'aucune utilité de l'entamer pour une ontologie qui sera obsolète dès sa naissance.

Pour bénéficier des ontologies dans les domaines associés à la génétique, il faut diminuer l'échelle, comme on peut le voir dans GO, qui se concentre sur le rôle des gènes et des produits géniques ou la *Human Phenotype Ontology* (HPO) (Köhler, Doelken et al., 2013)(Köhler, Carmody et al., 2018) qui se concentre sur les phénotypes. On parle alors d'ontologies locales, qui ont de nombreux avantages, notamment au niveau de la maintenance et de la création.

3.5.2 Création d'une ontologie

Initialement, la création d'ontologies suivait l'idée de représenter un domaine complet et défini. Le processus de création suit donc une création pas à pas. La plus connue de ces approches est décrite par Noy (Noy, McGuinness et al., 2001) qui comporte 7 étapes :

1 La détermination du domaine et de l'étendue de l'ontologie

2 La réutilisation si possible d'ontologies existantes

3 L'énumération des termes importants de l'ontologie

4 La définition des classes et leurs hiérarchisations

5 La définition des propriétés des classes et des relations

6 La caractérisation des propriétés, le type de données, les valeurs possibles,...

7 Le peuplement de l'ontologie, c'est-à-dire la création des instances.

Cette approche a depuis été retravaillée et réduite à 6 étapes par De Nicola et al (De Nicola et Missikoff, 2016) :

1 Création de la terminologie, la liste des mots utilisés dans ce domaine

2 Création du glossaire, les définitions des termes de la terminologie

3 Création de la taxonomie, les relations entre les termes du glossaire

4 Création des propriétés des termes du glossaire

5 Création des hiérarchies, mise en place des relations d'héritage

6 Finalisation de l'ontologie, formalisation

Ces deux méthodologies sont faites pour ne partir de rien ou presque, même si une ontologie existante peut être réutilisée. Elles nécessitent également l'intervention d'experts du domaine et d'experts en ontologie, même si pour la seconde méthodologie, les experts en ontologie n'interviennent qu'à la dernière étape. Dans cette méthode, le but est de créer une ontologie conséquente en une seule étape. Depuis, d'autres philosophies se sont mises en place, avec notamment des créations itératives comme suggérées par Menolli (Menolli et al., 2013) ou modulaires, comme résumées dans la thèse de Kevin Royer (Royer, 2015). Cependant, même

ces approches plus modernes nécessitent l'intervention poussée de spécialistes de l'ontologie. Un excellent résumé des approches peut être trouvé dans l'article de Simperl et collaborateurs (Simperl et Luczak-Rösch, 2014).

Dans un souci de clarté, le mot **expert** sera dans la suite de ce travail réservé aux experts métiers, les experts des domaines d'applications des ontologies développées. Pour désigner les experts en ontologie, on utilisera le néologisme "ontologiste".

3.5.3 *Mapping* d'ontologies

L'une des tâches dans lesquelles sont utilisées les ontologies est le *mapping*. Ce procédé consiste à modifier la structure d'une source de données pour la faire correspondre à une autre. Le but est souvent de regrouper des bases de données contenant des concepts similaires dans une seule base de données. Les deux principales utilisations du *mapping* d'ontologies sont très bien décrites par Wache et collaborateurs (Wache et al., 2001).

La première est la jointure de deux ontologies. La solution la plus simple est de définir des relations entre les concepts des deux ontologies *via* des agents d'interfaces, plus ou moins contrôlés par les utilisateurs. C'est une approche très adaptable, mais qui peut rapidement produire des résultats aberrants en cas de mauvais appariement de concepts. Des guides peuvent être ajoutés pour restreindre la liberté de l'utilisateur mais aussi pour augmenter la confiance dans les résultats. Une autre approche, plus complexe, est de placer les deux ontologies sous une troisième plus grande.

La seconde utilisation est de faire un *mapping* d'une ontologie contre une ou plusieurs sources de données. Cette opération est principalement effectuée dans le but d'enrichir une ontologie avec le contenu d'une source de données, mais elle peut parfois être réalisée pour restructurer une source de données existante dans une forme plus adaptée. Comme pour le *mapping* de deux ontologies, il existe de nombreuses approches possibles. Parmi les plus communes, on trouve l'utilisation de proximités structurelles entre la source de données et l'ontologie, qui n'est bien entendu possible que si de telles proximités structurelles existent. Dans le domaine du Web sémantique, on trouve une approche de méta-annotation, qui consiste à ajouter des informations sémantiques aux sources de données. Enfin, on peut également citer l'enrichissement de structure. Le principe est d'utiliser une source de données pour agrandir l'ontologie, en profitant d'une partie commune. Cette approche est la base de l'*ontology learning*, décrit dans la section 3.5.5.

3.5.4 Outils de *mapping*

Comme nous l'avons vu, il existe de très nombreuses approches différentes du *mapping* d'ontologies. Ce grand nombre d'approches induit un grand nombre d'outils et de langages possibles. On peut distinguer trois principales stratégies :

— La génération semi-automatique d'une ontologie à partir d'un schéma relationnel d'une base de données. C'est une approche notamment utilisée par Stojanovic (L. Stojanovic, N. Stojanovic et Volz, 2002). Des *mappings* sont ensuite établis entre la base de données et les concepts de l'ontologie. Comme l'ontologie est déduite de la base de données, les deux structures sont très proches et les *mappings* sont simples à mettre en place. Cependant, cela ne permet pas de réutiliser une ontologie déjà existante.

— Une option intéressante est d'utiliser la sagesse collective, ici représentée par le savoir et l'expérience d'un grand nombre d'utilisateurs. Cette approche est utilisée par Handschuh (Handschuh, Staab et Volz, 2003) et Ermilov (Ermilov, Auer et Stadler, 2013) notamment. Elle n'est cependant pas toujours pertinente car elle requiert un très grand nombre d'utilisateurs et un problème facilement divisible et relativement simple.

— Enfin, la troisième approche principale est celle proposée par Barrasa pour l'outil *R2O* (Barrasa, Corcho et Gomez-pérez, 2004) qui consiste à partir d'une base de données

et d'une ontologie déjà existantes et d'essayer de faire correspondre les données de la base vers l'ontologie. Cela suppose que l'ontologie soit adaptée, c'est-à-dire qu'elle recouvre entièrement le domaine de la base de données.

Cette dernière approche a beaucoup évolué depuis la publication de Barrasa en 2004, des standards existent maintenant, c'est le cas de *DirectMapping* (Arenas et al., 2012) et de *R2RML* (Das, sundara et Cyganiak, 2012), tous les deux validés par le W3C (*World Wide Web Consortium*) qui édite les standards du Web. *DirectMapping* permet de se passer de l'intervention de l'utilisateur, la structure de la sortie se base sur le schéma relationnel de la base de données d'entrée. Á l'inverse, *R2RML* permet de définir les correspondances entre chaque élément de la base de données vers leur équivalent dans l'ontologie. L'opération pouvant être très longue dans le cas de bases de données conséquentes, des algorithmes d'automatisation et des éditeurs graphiques ont été développés pour simplifier le travail de l'utilisateur, par exemple *Ultrawrap Mapper* (Sequeda et Miranker, 2015).

3.5.5 *Ontology learning*

L'*ontology learning* est le processus de construction, d'enrichissement et d'adaptation d'une ontologie de façon automatique ou semi automatique (Maedche et Staab, 2004). Petasis défini trois approches (Petasis et al., 2011) pour cela :

— En intégrant plusieurs ontologies existantes qui couvrent des domaines proches ou joints pour en créer une nouvelle plus complète, que ce soit en les fusionnant en une seule ou plus communément en définissant des ponts entre elles permettant leur entre-utilisation, ou encore en effectuant un *mapping* comme décrit précédemment.

— En spécialisant une ontologie de haut niveau pour l'adapter à un sous-domaine spécifique.

— En construisant une ontologie depuis le départ ou en augmentant une ontologie à partir d'informations extraites de *corpus* de documents du domaine.

Dans le domaine qui nous intéresse, les approches se centrent sur ce dernier point, c'est-à-dire la création d'une ontologie à partir de documents du domaine et sa population à partir des instances trouvées dans ces documents. Une très grande partie de la bibliographie se concentre sur la première partie de cette démarche, c'est-à-dire l'extraction de savoir à partir de textes ou d'autres éléments multimédias non structurés. C'est une étape importante et souvent indispensable, puisque l'*ontology learning* est principalement utilisée dans le domaine du Web sémantique, où l'enjeu principal est effectivement d'extraire du savoir de pages web non structurées. Ce n'est cependant pas la seule étape du processus. D'après Buitelaar (Buitelaar, Cimiano et Magnini, 2005), le processus peut être découpé en 6 étapes comme montré figure 3.5.

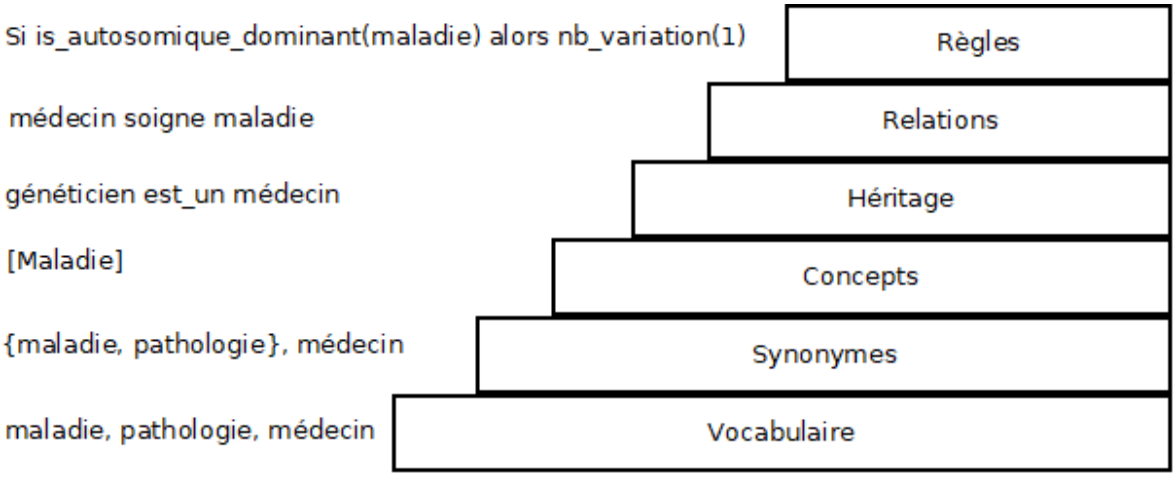


Figure 3.5 – Les six étages de l'ontoloy learning

Le premier étage est celui des termes. L'extraction des termes, qui sont des instances, est une tâche complexe, qui se base souvent sur des algorithmes de *machine learning* et / ou d'analyse du langage. Le second étage est celui des synonymes, le but est de déterminer lesquelles des instances préalablement extraites sont équivalentes. Cela peut consister aussi en la recherche de traduction dans le cas d'application multi-langues. Cette phase est souvent réalisée par le même outil que celui utilisé pour l'extraction des termes en se basant sur des dictionnaires ou d'autres ressources sémantiques. La troisième étape est d'identifier les concepts, qui sont des ensembles d'instances équivalentes. On compte ici sur des outils de création de *clusters* pour les approches automatiques, ou sur l'aide du ou des utilisateurs pour les approches semi-automatiques. Le quatrième étage est la hiérarchisation des concepts, afin de savoir quels concepts précisent quels autres. Les algorithmes de *machine learning* sont ici aussi très utiles, permettant notamment d'extraire du texte des structures sémantiques décrivant la hiérarchisation. La recherche de relations, le cinquième étage, est basée sur le même principe, la hiérarchisation étant aussi une forme de relation, il est logique que les mêmes outils effectuent ces deux tâches. Enfin, la dernière phase est la détermination des règles de l'ontologie, c'est une étape très délicate, assez peu étudiée par rapport aux autres étages. Les approches utilisées se reposent encore une fois sur la détection de composants sémantiques.

On peut trouver le terme *ontology enrichment* à la place de celui d'*ontology learning*. L'*ontology enrichment* est en fait un sous-genre de l'*ontology learning*, la différence étant qu'une ontologie est déjà présente dès le départ et que l'on va l'enrichir avec des nouveaux concepts, règles et relations. Toutes les étapes de l'*ontology learning* sont donc reproduites à l'identique.

3.5.6 *Ontology population*

L'*ontology population* est le remplissage d'une ontologie existante avec des instances de concepts et de relations. Contrairement à l'*ontology enrichment*, la structure de l'ontologie ne change pas, les concepts et relations restent les mêmes. Seules les deux premières phases de l'*ontology learning* sont utilisées ici : les instances des concepts et des relations sont extraites des *corpus* et insérées dans l'ontologie. La phase de gestion des conflits sémantiques, notamment des synonymies, n'existe pas directement, car les différents synonymes sont décrits dans l'ontologie. En revanche, d'autres problèmes se posent, notamment la gestion des instances multiples, c'est-à-dire les cas où au moins deux instances ne décrivent qu'un seul objet physique et ne devrait donc correspondre qu'à une seule instance dans l'ontologie. Ces problèmes sont souvent réglés *via* des intelligences artificielles et des algorithmes de *machine learning*, mais certains systèmes utilisent également des règles de *mapping* particulières qui permettent de réutiliser les instances existantes, en interrogeant le système pour chaque nouvelle instance (Buitelaar, Cimiano, Racioppa et al., 2006). Ce sont des systèmes plus coûteux en temps, mais leur fiabilité est meilleure.

3.5.7 Rôle du *mapping* en biologie

Pour simplifier les problèmes d'interopérabilité des données, il serait intéressant de pouvoir fusionner plusieurs bases de données. Cette fusion permet aussi de maximiser l'information et de bénéficier des avantages des différentes bases. Comme nous l'avons déjà décrit, les bases de données biologiques sont diversifiées que ce soit sur le fond mais aussi sur la forme. En revanche, certains concepts sont identifiants et peuvent jouer le rôle de clés car ils sont partagés par plusieurs bases de données. Chaque base possède un système d'identifiants en place, il peut être interne et exclusif à cette base, mais il est plus généralement basé sur une des notations de références, comme par exemple l'identifiant OMIM, *ensembl*, GO ou *entrez* pour les gènes, le *rsID* pour les variations, ou encore le code HPO ou OMIM pour les phénotypes. Les identifiants sont donc variés, mais il est possible de trouver des correspondances et donc

de trouver des liens entre les différentes bases de données. Il s'agit d'ailleurs du principe des annotateurs, qui vont faire des requêtes sur différentes bases de données en se basant sur quelques concepts seulement, on appelle ces concepts initiaux la graine.

Dans le cas d'un annotateur, il est parfaitement raisonnable de requérir cette graine, puisque cela constitue la base de travail. Les graines contiennent la position génomique ainsi que le chromosome et au moins l'altération présente. Elles peuvent contenir d'autres données comme l'allèle de référence, un nom de gène, etc... mais la graine minimum peut ne contenir que les trois informations primaires : chromosome, position, altération.

Les fichiers annotés en sortie des annotateurs sont en fait des sous-échantillons de *mapping*, puisqu'ils peuvent être vus comme une base de données. Ils sont parcellaires car seules les variations présentent dans l'entrée sont annotées, mais il est tout à fait possible de produire le même type de résultats pour l'ensemble des positions génomiques.

D'un point de vue utilisateur, on trouve souvent un enchaînement de requêtes. Une requête **A** va retourner un premier résultat **R1**, qui va être utilisé comme clé pour une requête **B** afin d'obtenir le résultat **R2**. Ce schéma est encore plus fréquent si l'annotation n'est pas complète, ou si elle est inadaptée aux besoins des utilisateurs. Ceux-ci passent alors par une succession d'interfaces web de différentes bases de données pour obtenir les informations manquantes.

3.6 Problématiques d'Interaction Homme Machine

Les problématiques d'Interaction Homme Machine sont une part importante de ces travaux. Nous allons ici commencer par les contextualiser, puis nous verrons deux méthodes pouvant être liées à notre thématique, la programmation par l'utilisateur final et la programmation par l'exemple.

3.6.1 Généralités

De nos jours, une immense majorité des corps de métiers font appel à l'informatique. Le besoin en solutions logicielles est toujours grandissant, mais l'offre peine parfois à suivre. Pour les informaticiens et les développeurs, les phases de compréhension du domaine cible et d'adaptation de solutions logicielles précises pour les utilisateurs sont très coûteuses en temps et sont complexes car elles demandent de véritablement se plonger dans le domaine cible. Si les utilisateurs finaux ont sans conteste la maîtrise requise du domaine, ils manquent très souvent des compétences informatiques nécessaires. Plusieurs domaines de recherches étudient les différentes approches pour permettre aux utilisateurs finaux de développer ou d'adapter des outils informatiques à leurs besoins, de façon à ce que le processus soit le moins coûteux pour tout le monde.

Ces problématiques illustrent parfaitement le domaine de l'Interaction Homme Machine (IHM), ou Human Computer Interaction (HCI) en anglais. Ce domaine de recherche étudie la conception et les usages des technologies, principalement informatiques. La zone principale d'intérêt est l'interface entre l'utilisateur et la machine. C'est un champ de recherche transversal, empruntant à l'informatique, aux sciences humaines, notamment la psychologie, les sciences cognitives et la linguistique, ainsi qu'à l'électronique, le design et l'ingénierie. C'est un domaine de recherche actif, le terme apparait pour la première fois dans la littérature scientifique en 1976 (Carlisle, 1976) et sera ensuite popularisé par Stuart Card dans un premier article en 1980 (Card, Moran et Newell, 1980) puis dans un livre référence en 1983 (Card, 1983). Une définition générique du domaine est fournie par la respectée "Association for Computing Machinery (ACM)" : "*L'IHM est une discipline étudiant la conception, l'évaluation et l'implémentation des systèmes informatiques interactifs destinés à un usage humain ainsi que l'étude des phénomènes majeurs autour de ces systèmes.*" (traduit depuis Hewett et al., 1992)

Ce champ de recherche est assez vaste, il suffit de parcourir les différents articles publiés

dans les conférences génériques d'IHM pour s'en persuader. Ainsi, on peut lister de façon non exhaustive quelques domaines de recherche actifs :

— L'étude des modes d'interactions, par exemple les commandes vocales ou gestuelles (T.-H. Lee et H.-J. Lee, 2018 W.-j. Hou et al., 2018)

— L'étude des modes d'interactions d'objets connectés, par exemple pour les montres connectées (Akita, Tanaka et Sagawa, 2018)

— L'étude des interfaces, pour présenter au mieux les informations (Duruisseau et al., 2018 Gutbell, Pandikow et Kuijper, 2018)

— Le développement de nouveaux objets, par exemple des outils chirurgicaux (Koeda et al., 2018)

— Le comportement des utilisateurs en fonction des interfaces (Bodrunova et Yakunin, 2018)

— Le développement de moyens d'interactions adaptés aux handicap (Oliveira Camenar, Nascimento et Almeida, 2018 Meyer et al., 2018)

— L'étude des techniques de réalité virtuelle (Ihemedu-Steinke et al., 2018)

Les champs d'études sont donc extrêmement variés, d'autant plus que les champs d'applications le sont aussi. Parmi ces champs d'études, la recherche de solutions pour permettre à un utilisateur final de créer ou d'adapter un outil informatique à ses besoins s'appelle le développement par l'utilisateur final, plus fréquemment appelé End-User Development et abrégé EUD.

3.6.2 Programmation par l'utilisateur final

La programmation par l'utilisateur final, plus souvent appelée *End-User Programming* (EUP) est considérée comme une sous branche de l'EUD (Burnett et Scaffidi, 2014). C'est un ensemble d'approches et de techniques qui permettent à l'utilisateur final de créer son propre programme soit en élaborant des langage spéciaux adaptés, soit en développant des interfaces pour des langages de programmation classiques. Ces techniques comprennent par exemple la programmation par démonstration ou la programmation visuelle.

La différence avec le développement par l'utilisateur final est dans le logiciel fini. L'EUP vise à permettre à un utilisateur de créer un logiciel qui remplit ses besoins spécifiques. L'EUD a une vision plus large et prend en compte tout le cycle de vie d'un logiciel, cela permet donc à un utilisateur de créer un logiciel plus complet, adaptable et robuste. Cela ne comprend pas uniquement les activités de programmation, mais cela intègre aussi le design plus global de l'outil.

3.6.3 Programmation par démonstration et programmation par l'exemple

La programmation par démonstration et la programmation par l'exemple sont des techniques de la programmation par l'utilisateur final, permettant aux utilisateurs de créer des programmes sans compétences informatiques particulières. Le principe est le même que celui de la manipulation directe. L'utilisateur ne travaille pas à un niveau abstrait de programmation, mais interagit directement avec les données qui l'intéressent et montre au logiciel les actions à effectuer sur les données ou les objets en les réalisant lui même une première fois. Ce système permet d'automatiser ensuite les actions de l'utilisateur, ce qui peut permettre de gagner en efficience de façon très importante. Le problème d'une telle approche est situé autour de la validité de l'exemple, puisqu'il faut pouvoir générer des variables abstraites pour généraliser les actions. Cela demande une notation particulière pour pouvoir exprimer les actions à effectuer sur les données. Prenons un exemple simpliste pour illustrer ce propos : une liste de 100 dates. L'utilisateur veut souligner en rouge toutes celles antérieures au 22 novembre 2010. Il a besoin d'exprimer cette date charnière ($DateCharnière = 22-11-2010$), la condition (*si antérieure*

à DateCharnière) et l'action à effectuer (*souligner en rouge*) si la condition est respectée. Deux approches existent pour résoudre ces problèmes.

La première correspond à la programmation par démonstration, où un niveau d'abstraction permet de fournir une notation et une généralisation. L'utilisateur montre à une personnification de cette couche abstraite, qui peut être représentée comme un petit robot par exemple (Kahn, 1996), les tâches à accomplir en utilisant des notations abstraites correspondants aux ordres pour le robot. Chacune des manipulations abstraites sur les données est associée à une action concrète dans l'environnement virtuel du robot. Les variables sont également représentées dans ce même environnement virtuel. Cela demande donc à l'utilisateur un effort d'apprentissage pour maîtriser la notation abstraite permettant de contrôler le robot et son environnement virtuel. Cela demande également un effort de développement important pour réussir à faire correspondre toutes les abstractions à des métaphores concrètes dans le monde virtuel. Il est des domaines où une telle tâche est quasiment impossible, par exemple, si le domaine comprend trop de concepts et de tâches, ou si ces concepts et tâches changent souvent, cela entraîne une mise à jour pour créer une nouvelle métaphore à chaque fois.

L'autre approche est la programmation par l'exemple. Ici, l'utilisateur fourni un certain nombre d'exemples du comportement souhaité par le programme et le système en déduit des règles qui permettront de traiter des situations qui n'ont pas encore été rencontrées. Cette approche permet d'éviter complètement le besoin d'apprentissage d'une nouvelle notation abstraite par l'utilisateur, en revanche elle peut aussi limiter la précision des règles, ou demander un très grand nombre d'entraînements avant d'être efficace. Dans notre exemple simpliste précédent, si parmi les exemples de l'utilisateur, les dates $20-11-2010$ et $21-11-2010$ n'apparaissent pas, le système ne peut savoir comment les traiter. De plus, dans la plupart des cas, des exemples négatifs sont nécessaires pour apprendre au système ce qu'il ne doit pas faire. Cela peut être perçu comme une perte de temps par l'utilisateur de faire volontairement des exemples. Pour limiter cet aspect, cette technique est souvent mise en place en plus des tâches classiques de l'utilisateur et n'intervient que pour proposer une généralisation lorsqu'elle en a trouvé une. Dans le domaine de la bio-informatique, une application de la programmation sur exemple a été faite par Catherine Letondal (Letondal, 2001), mais dans un objectif d'apprentissage de la programmation aux biologistes.

3.7 Conclusion sur la bibliographie

Ce résumé bibliographique permet tout d'abord de constater les particularités du domaine de la génétique médicale et de la recherche ayant pour thématique les maladies génétiques rares. Ces particularités se retrouvent dans les outils du domaine, au niveau des bases de données mais aussi dans la diversité des méthodes d'élaboration et d'utilisation de score. Cependant, ces outils ne sont pas à destination des généticiens, mais plus souvent des bio-informaticiens à cause des compétences en informatique avancées qui sont nécessaires. Ce décalage est un frein au travail déjà complexe de l'analyse et de l'interprétation des variants de l'ADN.

Si une ontologie globale du domaine pourrait être la solution, dans les faits, elle est impossible à construire à cause de l'étendue du domaine et de son activité. Une ontologie plus réduite est un bon candidat pour assister les généticiens dans leurs travaux, cependant les différentes méthodes de créations des ontologies ne sont pas centrées sur les experts domaines, dans notre cas ce sont les généticiens, mais plus souvent sur les ontologistes au vu des compétences informatiques nécessaires à la création d'une ontologie.

Enfin, nous avons constaté que les techniques de l'EUP permettent de faire créer des solutions logicielles à des utilisateurs inexpérimentés en développement ou adaptation d'outils informatiques. Ces techniques n'ont à notre connaissance pas été utilisées à ce jour dans le but d'assister l'utilisateur dans sa tâche pour créer des ontologies.

Compte tenu de ces différentes observations, notre solution consistera à utiliser les techniques de la programmation par l'utilisateur final (EUP) pour assister l'expert généticien

dans sa tâche d'analyse et d'interprétation des variations génétiques. Deux contributions sont présentées dans les deux chapitres suivants. La première, intitulée "Contribution base de données" concerne la possibilité pour un expert généticien de construire son référentiel d'annotation à partir des bases de données qu'il utilise. La seconde, intitulée "Contribution score", propose une solution de construction de score par l'expert généticien, et évalue sa pertinence en situation d'utilisation.

Chapitre 4

Contribution base de données

4.1 Introduction et rappel

Devant le problème de l'annotation, les généticiens sont très dépendants des bio-informaticiens. Cette dépendance limite considérablement leur liberté d'expérimenter et d'explorer de nouvelles ressources. En effet, les annotateurs sont des outils en ligne de commandes, alors que les solutions qui existent pour annoter *via* des interfaces graphiques ne permettent pas d'utiliser n'importe quelles bases de données et présentent une liste de sources disponibles réduite.

Les annotateurs fonctionnent en agrégeant des bases de données *via* des jeux d'identifiants communs. Les bases de données utilisées ne possèdent pas de structures ou de terminologies unifiées ; l'assemblage de ces bases de données s'apparentent alors à la création d'une ontologie *ad hoc*. Les différentes bases utilisées par l'annotateur sont requêtées une à une avec l'appui d'un petit jeu de concepts essentiels (coordonnées génomiques, la variation nucléotidique et sa référence), que l'on appelle les concepts graines. La création d'un assemblage de base de données permet donc de mettre en place tous les éléments nécessaires à une annotation au travers d'une interface graphique simple. La difficulté de l'assemblage de bases de données n'est pas une problématique nouvelle, il s'agit même d'un domaine de recherche actif. En revanche, ces problématiques ne sont que très rarement abordées du point de vue de l'utilisateur novice en informatique, ce qui est le cas dans le domaine de la génétique.

La création d'un annotateur est une tâche de développement longue et complexe. Nous allons ici concentrer nos efforts sur l'étude de la faisabilité de l'approche et non sur la réalisation d'un annotateur effectif. Ce développement pourra intervenir dans un second temps, mais ne relève pas d'un travail de recherche. La problématique de recherche de cette partie peut donc se résumer à l'étude de la faisabilité de la construction d'une ontologie par programmation basée sur l'exemple, appliquée au domaine de la génétique.

En effet, comme nous l'avons expliqué dans le chapitre précédent, les ontologies peuvent être utilisées pour regrouper des sources de données hétérogènes. Dans ce contexte d'utilisation, elles permettent de regrouper les savoirs de plusieurs bases dans une seule structure, ce qui facilite grandement le travail d'annotation. Notre approche vise à augmenter la liberté du généticien en lui permettant d'inclure les bases de données qu'il utilise au processus d'annotation. Notre hypothèse est que la création semi-automatique d'une ontologie est à la portée des généticiens, ce qui permettrait de diminuer leur dépendance vis-à-vis des spécialistes bio-informaticiens pour leurs activités de diagnostic et de recherche.

4.2 État de l'art de la création d'ontologies

La création d'ontologies est désormais un aspect important dans le domaine des sciences. Nous avons abordé les théories et méthodes générales dans le chapitre précédent. Nous allons ici détailler des approches de création d'ontologies utilisables dans le contexte de notre problématique.

4.2.1 Contraintes spécifiques de la création automatique d'ontologies

La plupart des approches actuelles utilisent des notions de *machine learning* (apprentissage automatique en français) ou d'intelligence artificielle pour limiter voire supprimer le rôle de l'expert du domaine. Cette démarche est très cohérente avec le champ du Web sémantique, où les données sont des *corpus* de pages web, avec une structuration faible et où les experts du domaine sont quasiment inaccessibles, ce qui représente l'une des plus grosses limites.

Les contraintes sont différentes dans notre cas, car nous avons un accès privilégié aux experts du domaine puisque ce sont nos utilisateurs cibles. Elles ne sont pas particulièrement liées au domaine de la génétique, mais plutôt à cette proximité de l'expert et cette volonté de l'impliquer le plus possible pour avoir la meilleure fiabilité possible. Ce sont des contraintes fortes que l'on peut retrouver dans d'autres domaines médicaux et plus globalement, dans les approches centrées sur l'utilisateur. On peut résumer les contraintes en trois points principaux :

— Nous avons accès aux experts du domaine, mais ces derniers sont dans la grande majorité des cas novices en informatique.

— Lors de l'utilisation du système, les experts sont seuls, ils n'ont pas accès à des spécialistes en ontologie ou en base de données pour les aider.

— Les experts doivent avoir une totale confiance dans les résultats, cela signifie qu'ils doivent pouvoir décortiquer les éventuelles étapes automatisées.

4.2.2 Généralités sur la création d'ontologies

Avec la popularité grandissante des ontologies, leur création est devenue un problème très étudié et un sujet de recherche à part entière. Les principales méthodologies sont décrites dans la section 3.5.2. Le problème auquel sont confrontées ces méthodologies est l'interdisciplinarité, le besoin d'experts du domaine et de consensus, ainsi que le besoin d'ingénieurs spécialisés dans la création d'ontologies.

Devant ces contraintes, de nombreux travaux ont été menés pour automatiser tout ou partie de la création des ontologies. Deux approches existent principalement, découlant de deux visions et de deux domaines principaux. La première approche vise à se passer de l'expert métier. C'est une vision très répandue dans les domaines du Web sémantique et de l'Internet des objets, où les experts ne sont pas accessibles facilement à cause de la variabilité des domaines et où l'ontologie a pour but de faciliter le travail des machines en posant des bases communes et en simplifiant considérablement les problèmes d'hétérogénéité et d'interopérabilité des données. Nous appellerons cette approche l'approche traditionnelle, car historiquement, elle fut le premier point de vue adopté.

Cependant, les bases de données sont désormais employées dans de nombreuses disciplines, les problèmes d'hétérogénéité et d'interopérabilité ont suivi, ce qui a entrainé le développement d'ontologies dans beaucoup de domaines . La seconde approche utilisée dans des domaines aussi variés que le commerce, l'industrie ou la médecine vise à se passer de l'expert ontologiste. Cette approche est pertinente dans ces domaines où les experts ontologistes sont rares et difficiles d'accès, mais où les experts métiers sont les utilisateurs finaux et sont donc impliqués dans la création des ontologies. Nous appellerons cette approche l'approche centrée expert, puisqu'elle se focalise sur l'apport de l'expert du domaine.

4.2.3 Difficultés des approches traditionnelles

Les approches actuelles sont confrontées à plusieurs écueils à différents niveaux d'actions. La première de ces difficultés est la construction de l'interface et du mode d'interaction avec l'utilisateur. Comme les outils de la littérature scientifique visent des utilisateurs experts des bases de données et des ontologies, leur approche est souvent de se baser sur un format standard du domaine, comme le *SPARQL Protocol And Rdf Query Language* (SPARQL) (Lefrançois, Zimmermann et Bakerally, 2017).

Une seconde difficulté est l'extraction des relations à partir de *corpus* de données non structurées (Kim et al., 2002). En revanche, lorsque les données proviennent de documents semi-structurés, la dépendance au format est très importante, il est alors fréquent d'observer des erreurs, des malformations ou simplement des différences d'interprétation des standards qui peuvent rendre très compliquée l'interprétation par une machine (O'Connor, Halaschek-Wiener et Musen, 2010).

Les outils de *machine learning* sont très dépendants de la taille des *corpus* et de la confiance attribuée aux sources primaires. Ce sont des paramètres délicats à gérer, notamment dans les champs où les données ne sont pas nombreuses ou peu disponibles (Brewster et al., 2007). L'inclusion source à source peut aussi poser des problèmes de duplication, avec des entités provenant de deux sources différentes reconnues comme deux instances différentes alors qu'elles sont identiques (Dimou, Sande et al., 2013).

Enfin, la dernière difficulté majeure des approches actuelles est l'évaluation du résultat, c'est-à-dire de l'ontologie produite et de son adéquation avec le domaine. La solution reconnue comme la meilleure est une revue manuelle par un ou plusieurs experts du domaine pour valider ou non les liens qui existent entre les différentes entités. Cependant cette solution est très coûteuse, ce qui a conduit au développement d'algorithmes automatisant partiellement cette tâche (Drumond et Girardi, 2008).

4.2.4 Généralités sur les approches centrées expert

Il existe de nombreux outils de création d'ontologies, à commencer par le plus célèbre, Protégé (Noy, Crubézy et al., 2003), mais la plupart demande un haut niveau d'expertise et / ou d'entrainement dans le domaine de la création d'ontologies, ce qui rend leur utilisation impossible pour beaucoup d'applications. Cela n'empêche pas la création d'ontologies par des experts métiers *via* d'autres outils. L'un des plus célèbres exemples est la *Gene Ontology* (GO) (Ashburner et al., 2000). Bada (Bada et al., 2004) attribue ce succès à plusieurs raisons :

— La demande de la communauté, car GO répond à un réel besoin des généticiens et des chercheurs en génétique. La communauté s'est totalement impliquée dans le processus de son développement ce qui a permis une meilleure utilisation et un meilleur accueil que si elle n'avait été créée que par des ontologistes, puis imposée comme une référence. Les termes sont définis par les utilisateurs, ce qui augmente leur pouvoir consensuel.

— Des objectifs et une échelle claire. L'objectif de GO est de fournir un vocabulaire clair et non ambigu pour parler des gènes dans trois catégories : les composants cellulaires, la fonction moléculaire et les processus biologiques.

— La structure est simple, le fait qu'elle soit définie par les utilisateurs experts a également permis de maintenir une structure très simple, contenant seulement deux types de relations "*PartOf*" et "*IsA*", c'est-à-dire les deux les plus simples possibles. Les trois domaines d'intérêt sont également conservés dans des structures séparées.

— Enfin, l'ontologie n'est pas fixée, elle est en perpétuelle évolution et n'a pas pour prétention de décrire tout le domaine. Elle évolue avec les connaissances du domaine. C'est un aspect rarement présent dans les ontologies, car les domaines sont rarement aussi instables et le processus de curation nécessite l'implication de beaucoup d'experts, peu souvent présents en quantité et durée suffisantes lors du processus de création d'une ontologie.

Ce succès de GO a permis à la communauté scientifique travaillant autour de la création des ontologies de prendre conscience de l'intérêt d'une approche centrée expert pour des ontologies dont l'objectif est d'être utilisée quotidiennement par des experts et non par des machines. Cette prise de conscience a mené au développement de plusieurs outils dédiés développés ci-dessous.

4.2.5 Méthodes de création d'ontologies

Après avoir vu la théorie de la création d'ontologies dans l'état de l'art et les principales approches dans la section précédente, nous allons nous pencher plus particulièrement sur les outils permettant concrètement cette création.

4.2.5.1 *DILIGENT*

La méthode *DILIGENT* est publiée en 2005 par Vrandečić et Pinto (Pinto, Staab et Tempich, 2004)(Vrandečić et al., 2005). Le processus de création est ici itératif, basé sur la réunion d'experts du domaine, d'utilisateurs et d'experts ontologistes. C'est donc une approche collaborative à petite échelle, puisqu'elle se base sur des réunions régulières de plusieurs personnes. Le rôle des ontologistes est de mettre en place les changements dans la

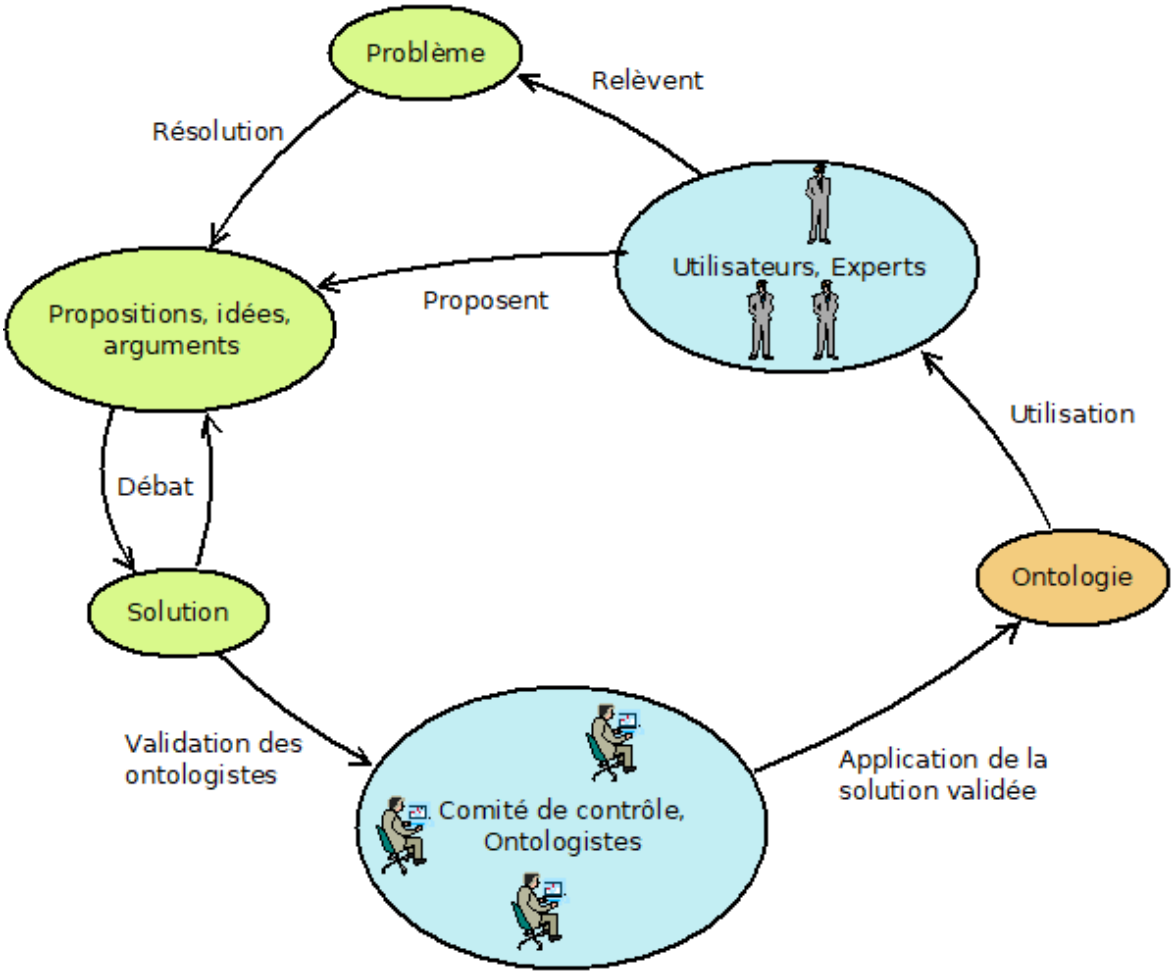


FIGURE 4.1 – Schéma du fonctionnement de la méthode *DILIGENT*

structure de l'ontologie et d'aider à structurer l'information. Celui des experts du domaine correspond donc à toutes les autres étapes de mise en place de l'ontologie, comme par exemple décrit par De Nicola (DE NICOLA et MISSIKOFF, 2016) ou présenté dans la figure 4.1.

La méthode *DILIGENT* attribue un rôle très important pour l'ontologiste, l'approche est globalement traditionnelle. L'idée est de définir une méthode applicable par les ontologistes dans tous les cas, mais cette méthode demande de nombreuses réunions qui sont elles centrées experts. Cette méthode est plutôt destinée à l'adaptation d'une ontologie déjà existante, qu'elle soit d'un domaine proche ou générique. L'adaptation est faite par un groupe d'experts qui se concentre sur les extensions ou les précisions de l'ontologie actuelle. Le rôle des ontologistes est essentiellement de valider la cohérence des changements.

Globalement, cette méthode fait partie des premières publiées à centrer une partie de la création sur les experts, elle reste cependant majoritairement sous le contrôle d'ontologistes, et n'est pas envisageable sans eux.

4.2.5.2 *HCOME*

HCOME, pour *Human Centered Ontology Engineering Methodology* est publiée par Kotis en 2006 (KOTIS et VOUROS, 2006). Comme *DILIGENT*, l'objectif est de fournir une méthode de création d'ontologies traditionnelles. *HCOME* est spécialisée dans le développement et l'évaluation d'ontologies non fixées, c'est-à-dire d'ontologies définies sur des domaines où les connaissances sont instables. Elle est composée de listes de tâches pour structurer le développement des ontologies *via* la création de petites parties, assimilables à des modules, qui seront ensuite assemblées les uns avec les autres.

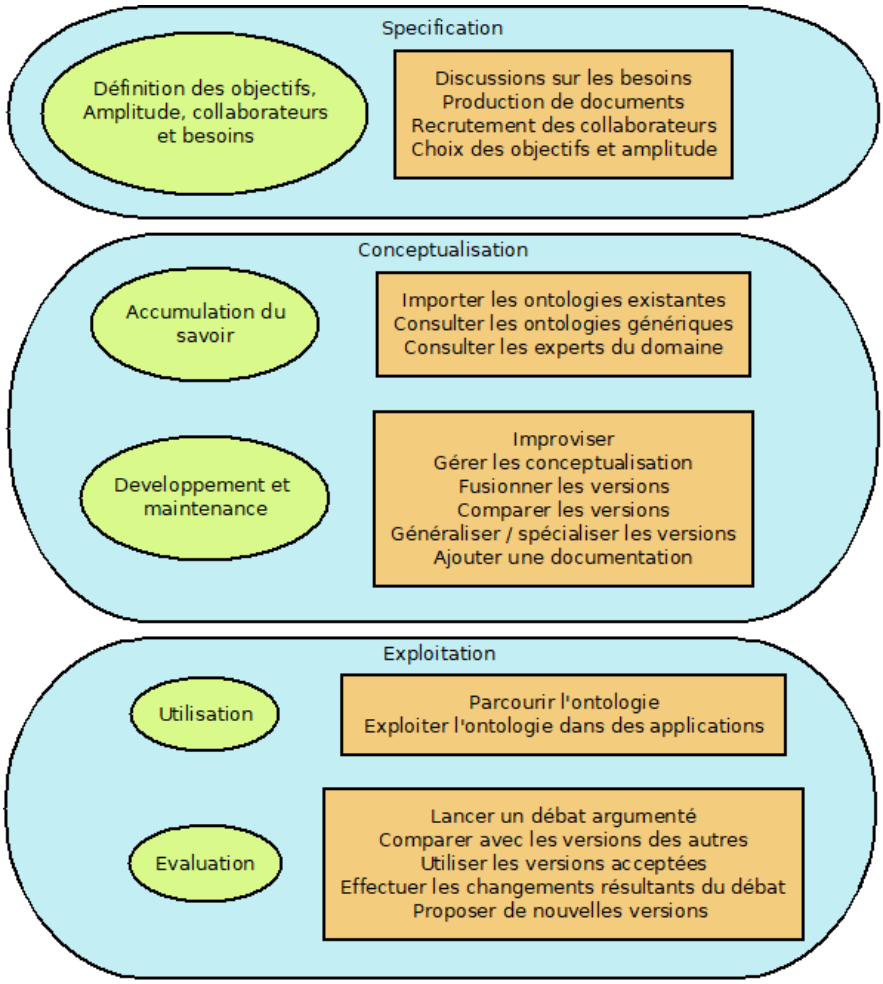


FIGURE 4.2 – Schéma du fonctionnement de la méthode *HCOME*

Le fonctionnement est distribué sur trois phases, la première est la phase de spécification qui permet de définir les objectifs, l'amplitude, les collaborateurs,... Vient ensuite une phase de conceptualisation, où le but est d'acquérir du savoir, puis de développer et de maintenir l'ontologie. Enfin, la dernière phase correspond à l'exploitation de l'ontologie, elle correspond à son usage et à son évaluation. La répartition des tâches dans les phases est présentée dans la figure 4.2.

Cette méthode souffre de plusieurs limites, notamment dues au manque d'implication des experts du domaine, ce qui peut mener à des erreurs et à des problèmes d'acceptation. Ce fait est particulièrement marqué si l'on regarde la liste des tâches, la seule où sont impliqués les experts est "*consultations des experts du domaine par discussions*" lors de la phase d'accumulation du savoir durant la conceptualisation. Ce sont des limites d'autant plus marquantes que les objectifs de la méthode sont particulièrement adaptés à une approche centrée expert, la maintenance et les mises à jour ne pouvant être effectuées sans eux. Ces limites sont probablement à l'origine de la très faible utilisation de cette méthode.

4.2.5.3 *UPON Lite*

La méthode *UPON Lite* (DE NICOLA et MISSIKOFF, 2016) est une méthode centrée expert. Elle dérive de la méthode *UPON* (DE NICOLA, MISSIKOFF et NAVIGLI, 2005)(DE NICOLA, MISSIKOFF et NAVIGLI, 2009), qui elle n'est pas du tout centrée expert. Cette méthode fournit une suite de 6 étapes pour la création d'ontologies, déjà décrite dans la section 3.5.2.

Pour rappel, le fonctionnement commence par la définition d'un lexique, ensuite précisé en glossaire. Les termes du glossaire sont alors précisés au niveau de leurs propriétés, de leurs

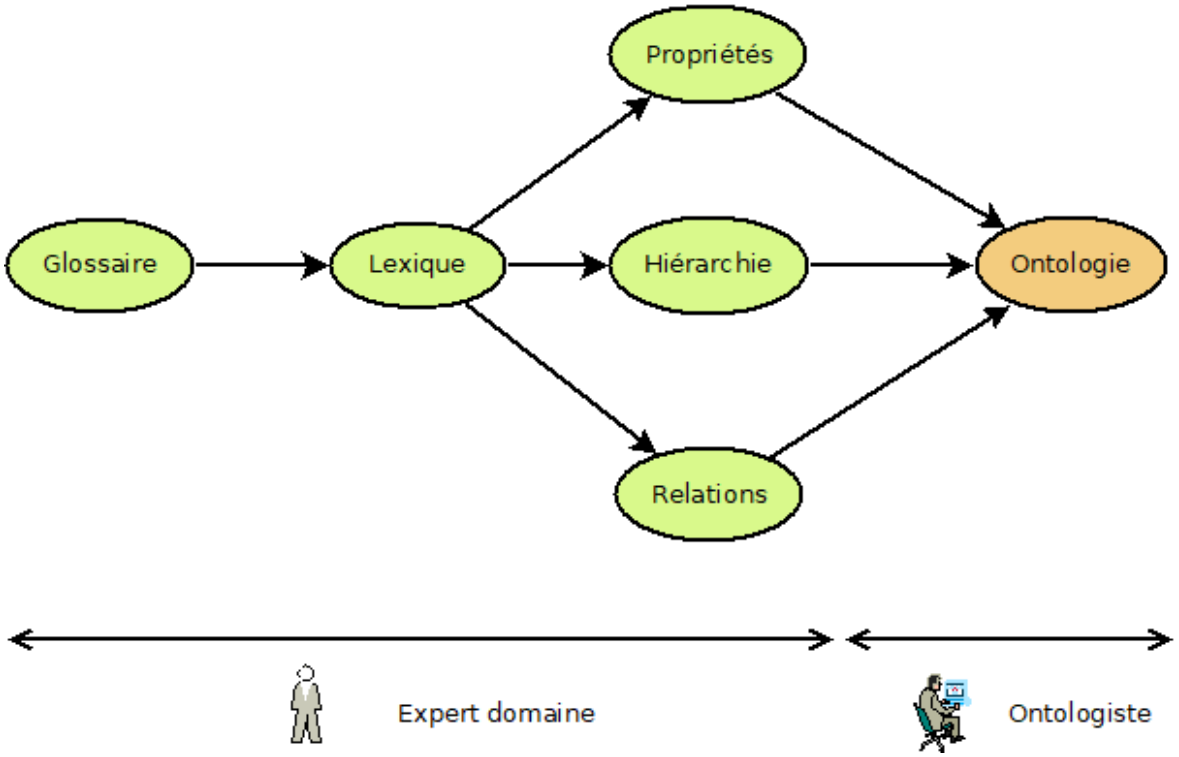


FIGURE 4.3 – Schéma de l'agencement des étapes de création d'une ontologie dans la méthode *UPON Lite*

hiérarchies et de leurs relations. Enfin, des ontologistes, à partir des résultats de ces cinq premières étapes formalisent l'ontologie. La clé est l'utilisation de tableurs collaboratifs, tels que *Google Spreadsheet*[1] ou *Framacalc*[2].

Deux limites sont souvent relevées sur cette approche. Premièrement, elle se base sur l'utilisation de tableurs, ce qui peut être une limite d'un point de vue ontologiste, puisque l'essence d'une ontologie est un graphe. En revanche, c'est un choix cohérent d'un point de vue Interaction Homme Machine, les tableurs étant des outils répandus et maîtrisés par beaucoup de monde, alors que peu de gens ont l'habitude de manipuler des graphes. La seconde limite est la nécessité d'intervention d'un ontologiste dans le processus, comme on peut le voir sur la figure 4.3. Cette limite est partagée par beaucoup d'outils, mais ici l'expert semble avoir un rôle très réduit, qui pourrait en effet être automatisé. Ces limites n'ont pas eu énormément d'impact, car cette méthode a été utilisée avec succès pour de nombreux projets dans des applications variées.

4.2.5.4 *ROBOT* et *Oort*

Ces deux outils viennent du consortium *Open Biomedical Ontologies* (OBO) (B. SMITH et al., 2007). Ce consortium vise à fournir des notations et des règles communes pour les ontologies dans le domaine biomédical. Plusieurs suites d'outils existent, notamment *OWLTools* et *Oort* (*OBO ontology release tool*). Ces deux suites d'outils devraient être remplacées par *ROBOT*, qui était dans un stade de développement précoce en 2015, mais qui n'a pas été publié depuis. Contrairement à *DILIGENT* ou *UPON Lite*, *Oort* et *ROBOT* sont des outils et non des méthodes de construction d'ontologies.

ROBOT est un outil en ligne de commande développé en Java et destiné au milieu biomédical (OVERTON et al., 2015). Il permet de fusionner des ontologies, d'en extraire les différences, d'inférer du savoir, de filtrer les propriétés ou encore d'extraire des modules d'une

1. `https://www.google.fr/intl/fr/sheets/about/`
2. `https://accueil.framacalc.org/fr/`

ontologie plus vaste. Ces commandes sont créés *via* l'*OWLAPI* (l'*Application Programming Interface* (API) du langage OWL). La principale limite est l'utilisation en ligne de commande, rendant son utilisation par un non-informaticien très improbable, ce qui va à l'encontre de la spécialisation dans le domaine biomédical.

D'un point de vue méthode, le consortium OBO en propose une qui reprend les principes qui ont fait la réussite de GO. Environ 200 ontologies sont construites sur ce modèle, dans des domaines biologiques variés et à différents stades de développement. L'objectif du consortium est de continuer à favoriser le développement des ontologies dans le domaine biomédical tout en évitant une prolifération de formats et d'approches rendant finalement plus complexe l'intégration et l'interopérabilité des données.

4.2.5.5 *NeOn toolkit*

NeOn toolkit (Haase et al., 2008) se base sur la méthode *NeOn* (Suárez-Figueroa, 2010)(Suárez-Figueroa, Gómez-Pérez et Fernández-López, 2012). Contrairement aux autres méthodes décrites, la méthode *NeOn* ne fournit pas de *pipelines* rigides, mais plutôt des guides visant à s'adapter à une plus grande diversité de scénarios. Cela permet de couvrir des cas comme la réutilisation d'une ontologie existante, son agrandissement, son reformatage, la création d'une ontologie par spécialisation, etc... Cette méthode s'adresse donc à des ontologistes et part du principe que les connaissances du domaine sont déjà présentes dans des ressources accessibles aux ontologistes.

L'une des ressources clés sont les *ODP*, pour "*ontologies design patterns*" ; ils sont de plusieurs types et permettent de diviser l'ontologie en mini-ontologies, proches de modules, correspondants aux différents cas possibles. Ces *ODPs* doivent ensuite simplement être adaptés pour correspondre aux concepts de l'ontologie cible. Les objectifs d'utilisation du projet *NeOn* se trouvent plutôt dans le domaine du Web sémantique, ce qui explique le manque de points de vue centrés expert du domaine.

NeOn toolkit est donc la suite d'outils permettant concrètement la gestion d'ontologies, comme peuvent l'être *ROBOT* ou *Oort*. L'objectif est de permettre à des chercheurs en ontologies ou à des clients possédant une bonne maitrise des ontologies de collaborer autour de leurs ontologies, de raisonner, etc... Cette suite de logiciels se veut ouverte, l'ajout de *plugin* étant laissé possible, ce qui augmente la flexibilité de l'ensemble. Ces *plugins* permettent de réaliser des *mappings*, des conversions de données textuelles en ontologies, de réparer des ontologies, de faire de l'*ontology population* ou de l'*ontology learning*.

Globalement, cette méthode accompagnée de sa suite d'outils est l'une des plus poussées, mais elle n'est pas du tout adaptée de par sa complexité et son abstraction, à l'usage que peuvent avoir des experts du domaine recherchant une ontologie. De plus, le développement de la suite d'outils semble arrêté, les dernières mises à jour datant de décembre 2011.

4.2.5.6 *Ontology Maturing*

Ontology Maturing est présentée par Braun en 2007 (Braun et al., 2007). C'est une approche centrée sur l'expert du domaine, ce qui permet une bonne acceptation des ontologies produites par cette méthode. Comme présenté dans la figure 4.4, le processus de maturation peut se découper en 4 phases :

— L'émergence des idées, qui correspond à la récupération des concepts.

— La consolidation, qui se fait au sein de la communauté d'experts du domaine. Le lexique généré lors de la première phase est réutilisé pour en extraire les concepts et aboutir à une réelle terminologie.

— La formalisation, qui permet d'ajouter les relations et la hiérarchisation, ainsi qu'un format d'ontologie.

— L'axiomatisation, qui correspond à la fin de la maturation. C'est la définition des axiomes qui gouverneront l'ontologie.

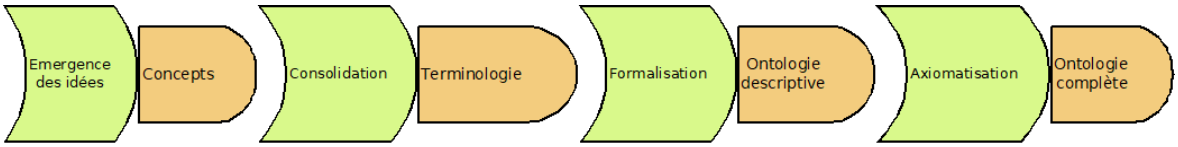


FIGURE 4.4 – Schéma des étapes de création d'une ontologie *via* la méthode *Ontology Maturing*

L'origine des concepts étant les experts du domaine eux-mêmes, les ontologies produites sont de bien meilleure qualité que celles produites par d'autres méthodes. Plusieurs cas d'utilisations réussies sont rapportés dans la littérature. Cependant, les ressources du domaine ne sont pas du tout utilisées, ce qui oblige les experts à fournir un travail très conséquent dans le cas de larges ontologies ou de processus de mises à jour / restructuration d'ontologies existantes. Ce point peut freiner l'utilisation de cette méthode pour des applications reposant sur de nombreux concepts ou instances.

4.2.5.7 Conclusions sur la bibliographie

La plupart de ces méthodes et outils sont des approches traditionnelles. Seules *UPON Lite* et *Ontology Maturing* sont complètement centrées sur l'expert du domaine, même si les guides de l'OBO consortium peuvent aussi être considérés centrés expert. Les approches traditionnelles ne sont clairement pas adaptées à l'appropriation d'ontologies par les communautés scientifiques non-informatiques, l'instabilité du domaine et son ampleur obligent à une mise à jour très fréquente, qui ne se fait que *via* des ontologistes. De plus, ces approches souffrent toutes de défauts d'acceptation lors de leurs tentatives de création d'ontologies pour des milieux non-informatiques, dus à la complexité de la mise en place, et aux erreurs liées aux manques d'implications des experts du domaine lors des processus de créations.

Sur ce point, les approches centrées expert sont beaucoup plus adaptées, mais elles sont également plus rares et souffrent aussi de limites. L'absence d'appui sur les ressources du domaine augmente considérablement la charge de travail des experts. Le besoin de validation et de formalisation par des ontologistes est également une limite importante. Si ce dernier point est moins conséquent pour des ontologies de grandes envergures, pour des ontologies locales, il apparait très problématique.

4.3 Approche de recherche

Nous avons précédemment constaté que les annotateurs permettent de regrouper différentes bases de données par le biais de requêtes enchainées. Si nous pouvons construire une base de données en regroupant les bases de données nécessaires à l'utilisateur, nous pouvons créer facilement un algorithme d'annotation. Cependant, les structures de ces bases n'étant pas connues à l'avance et les extractions d'informations étant des protocoles complexes, nous ne pouvons le faire automatiquement. Nous disposons cependant des experts métiers, puisque ce sont les utilisateurs ciblés. Nous pouvons donc nous appuyer sur leurs compétences pour trouver les concepts similaires dans différentes bases de données.

Les contraintes auxquelles se heurtent les outils du secteur ne sont pas les plus marquantes dans notre cas. Notre objectif est de faire créer une ontologie à un utilisateur expert et non de réaliser une ontologie à partir de données non structurées. Ce décalage simplifie le processus sur un grand nombre de points, car l'accès à l'expertise est souvent la base du problème pour les approches automatiques. Nous bénéficions également de la structure existante des exports des bases de données, garantissant des données bien organisées. Nous sommes en revanche confrontés à d'autres problèmes, notamment la capacité des utilisateurs à créer une ontologie sans notions informatiques.

Notre première hypothèse est qu'il existe de fortes similitudes entre le modèle mental de l'utilisateur et le modèle de données du domaine. Nous basons cette hypothèse sur nos

observations du travail des généticiens et leur capacité à enchainer les requêtes sur différentes bases de données présentes sur le Web. Nous faisons également l'hypothèse que si de tels liens existent, il est possible de trouver une méthode d'interaction permettant à l'utilisateur de les exprimer, puis de les traduire dans un langage informatique formel spécialisé permettant de créer une ontologie *ad hoc*.

Nous avons séparé notre approche en deux temps. La première phase est de tester le vocabulaire métier pour vérifier nos hypothèses. Ce test permet de valider la première de nos hypothèses, donc de valider les liens entre le modèle mental de l'utilisateur et les notions de bases de données formelles. La deuxième phase de notre approche sera ensuite de concevoir, de développer et de tester une interface capable de démontrer la deuxième hypothèse.

4.4 Détermination des liens entre les notions ontologiques et le vocabulaire métier

Pour valider notre première hypothèse, l'existence de liens entre le modèle mental de l'utilisateur expert et le formalisme informatique du domaine, nous avons mis en place une liste des notions nécessaires à la construction d'une ontologie. En nous appuyant sur cette liste, nous avons créé un test de détermination du vocabulaire métier afin de vérifier si ces notions sont exprimées par les biologistes. En cas de validation de l'hypothèse, le test devra également permettre de déterminer et caractériser ces liens. Nous pourrons alors en déduire des solutions adaptées pour la phase suivante, la réalisation d'un prototype.

4.4.1 Notions ontologiques nécessaires

Pour fusionner plusieurs bases de données, il faut maîtriser plusieurs notions importantes concernant les bases de données. Ces notions doivent être fournies par l'utilisateur ou déterminées à partir des données quand cela est possible.

— Les concepts ou entités. Ce sont les briques élémentaires du système, ils doivent être connus afin de créer les tables adaptées. Ils correspondent à des catégories concrètes ou abstraites, clairement définies, comme par exemple un gène ou une variation génétique. Il existe très souvent de nombreux synonymes pour désigner les concepts et plus rarement des homonymes.

— Les attributs des concepts. Ce sont les propriétés de chaque concept, qui le définissent. Ce sont concrètement les données présentes dans les bases de données, par exemple, le nom du gène ou la position de la variation. Ici aussi, il peut y avoir des ambiguïtés sémantiques.

— Les types des attributs. Le typage des attributs explicite le format dans lesquels ceux-ci sont décrits. Par exemple le nom du gène est une chaîne de caractères alors que la position génomique est un nombre entier. Ces deux types suffisent pour gérer la majorité des cas mais certains formats spécifiques peuvent améliorer les performances du système. Cette notion peut être déterminée à partir des données sans l'intervention de l'utilisateur.

— Les attributs identifiants, aussi appelés clés primaires. Ce sont les attributs qui permettent d'identifier un concept, ils sont donc uniques et présents pour chaque instance. Ce sont des informations essentielles. Il peut y avoir plusieurs attributs identifiants pour un seul concept, ce qui est très fréquemment le cas avec les gènes par exemple. Lorsque la combinaison de plusieurs attributs est utilisée comme clé, il s'agit d'une clé secondaire. Pour les variations génétiques, une clé primaire peut être l'identifiant RS, ou rsID, mais on peut aussi utiliser la combinaison "chromosome, position, altération" comme clé secondaire. Les informations graines doivent être des clés.

— Les relations. Cette notion est l'une des bases du système, ce sont les liens entre les concepts qu'elles permettent de caractériser. Par exemple, un codon-stop est un type de variation, ou bien encore, une maladie présente plusieurs phénotypes.

— Les cardinalités. Cette notion vient pondérer les relations en indiquant les quantités de chaque concept impliquées dans la relation. Par exemple une variation est dans un seul gène, mais un gène contient aucune, une ou plusieurs variations.

4.4.2 Test du vocabulaire métier

Pour effectuer ce test, nous avons sélectionné trois bases de données de références, OMIM, *dbSNP* et HPO. Ces trois bases sont centrées sur 3 concepts différents, les gènes pour OMIM, les variations génétiques pour *dbSNP* et les phénotypes pour HPO. Ces bases sont formatées sous la forme de fichiers plats exportés d'une base relationnelle traditionnelle. Les figures 4.5, 4.6 et 4.7 présentent les premières lignes de ces exports. On peut voir la duplication d'informations typique des exports de bases de données relationnelles, par exemples les 7 premières lignes de la figure 4.5 décrivent le même gène. Les concepts peuvent être présents dans plusieurs bases de données. On retrouve des phénotypes dans la base OMIM par exemple et des gènes dans *dbSNP* et dans HPO. Ces trois bases vont nous servir de support pour le test. Nous avons préalablement extrait les modèles entités-associations de ces trois bases de données. Ces modèles sont présentés dans les figures 4.8, 4.9 et 4.10.

```
#Format: diseaseId<tab>gene-symbol<tab>gene-id(entrez)<tab>HPO-ID<tab>HPO-term-name
OMIM:608980     FREM1   158326  HP:0000414      Bulbous nose
OMIM:608980     FREM1   158326  HP:0000077      Abnormality of the kidney
OMIM:608980     FREM1   158326  HP:0000322      Short philtrum
OMIM:608980     FREM1   158326  HP:0011803      Bifid nose
OMIM:608980     FREM1   158326  HP:0000143      Rectovaginal fistula
OMIM:608980     FREM1   158326  HP:0001545      Anteriorly placed anus
OMIM:608980     FREM1   158326  HP:0000007      Autosomal recessive inheritance
OMIM:611528     JUP     3728    HP:0004756      Ventricular tachycardia
OMIM:611528     JUP     3728    HP:0000006      Autosomal dominant inheritance
```

FIGURE 4.5 – Les dix premières lignes de l'export de l'ontologie HPO

#chr	pos(1-coor)		ref	alt	rs_dbSNP141	genename	Ensembl_geneid
1	69428	T	G	rs140739101	OR4F5	ENSG00000186092	
1	69453	G	C	rs2854682	OR4F5	ENSG00000186092	
1	69453	G	T	rs2854682	OR4F5	ENSG00000186092	
1	69476	T	C	rs148502021	OR4F5	ENSG00000186092	
1	69496	G	A	rs150690004	OR4F5	ENSG00000186092	
1	69511	A	G	rs75062661	OR4F5	ENSG00000186092	
1	69536	C	T	rs200013390	OR4F5	ENSG00000186092	
1	69569	T	C	rs2531267;rs372127752	OR4F5	ENSG00000186092	
1	69590	T	A	rs141776804	OR4F5	ENSG00000186092	
1	69610	C	T	rs376022826	OR4F5	ENSG00000186092	
1	69761	A	T	rs200505207	OR4F5	ENSG00000186092	
1	721415	G	A	rs368097292	AL669831.1	ENSG00000197049	
1	721649	A	C	rs371519651	AL669831.1	ENSG00000197049	
1	721757	T	A	rs189147642	AL669831.1	ENSG00000197049	
1	861329	A	G	rs371217242	SAMD11	ENSG00000187634	
1	861349	C	T	rs200686669	SAMD11	ENSG00000187634	
1	861357	C	G	rs370046315	SAMD11	ENSG00000187634	
1	865544	C	T	rs144245409	SAMD11	ENSG00000187634	

FIGURE 4.6 – Les dix premières lignes de l'export de *dbSNP*

4.4.3 Démarche expérimentale

Dans un premier temps, nous souhaitons vérifier que les généticiens sont bien capables d'extraire les concepts et les relations des bases de données. Nous évaluons également si les autres notions de bases de données sont précisées spontanément. Nous observons enfin comment ces notions sont décrites d'un point de vue sémantique. Quel est le vocabulaire

17,20-lyase deficiency, isolated, 202110 (3)	CYP17A1, CYP17, P450C17	609300	10q24.32
17-alpha-hydroxylase/17,20-lyase deficiency, 202110 (3)	CYP17A1, CYP17, P450C17	609300	10q24.32
17-beta-hydroxysteroid dehydrogenase X deficiency, 300438 (3)	HSD17B10, HADH2, ERAB, MRXS10	300256	Xp11.22
2-aminoadipic 2-oxoadipic aciduria, 204750 (3)	DHTKD1, KIAA1630, AMOXAD, CMT2Q	614984	10p14
2-methylbutyrylglycinuria, 610006 (3)	ACADSB, SBCAD	600301	10q26.13
3-M syndrome 1, 273750 (3)	CUL7, 3M1	609577	6p21.1
3-M syndrome 2, 612921 (3)	OBSL1, KIAA0657, 3M2	610991	2q35
3-M syndrome 3, 614205 (3)	CCDC8, 3M3	614145	19q13.32
3-Methylcrotonyl-CoA carboxylase 1 deficiency, 210200 (3)	MCCC1, MCCA	609010	3q27.1
3-Methylcrotonyl-CoA carboxylase 2 deficiency, 210210 (3)	MCCC2, MCCB	609014	5q13.2

FIGURE 4.7 – Les dix premières lignes de l'export d'OMIM

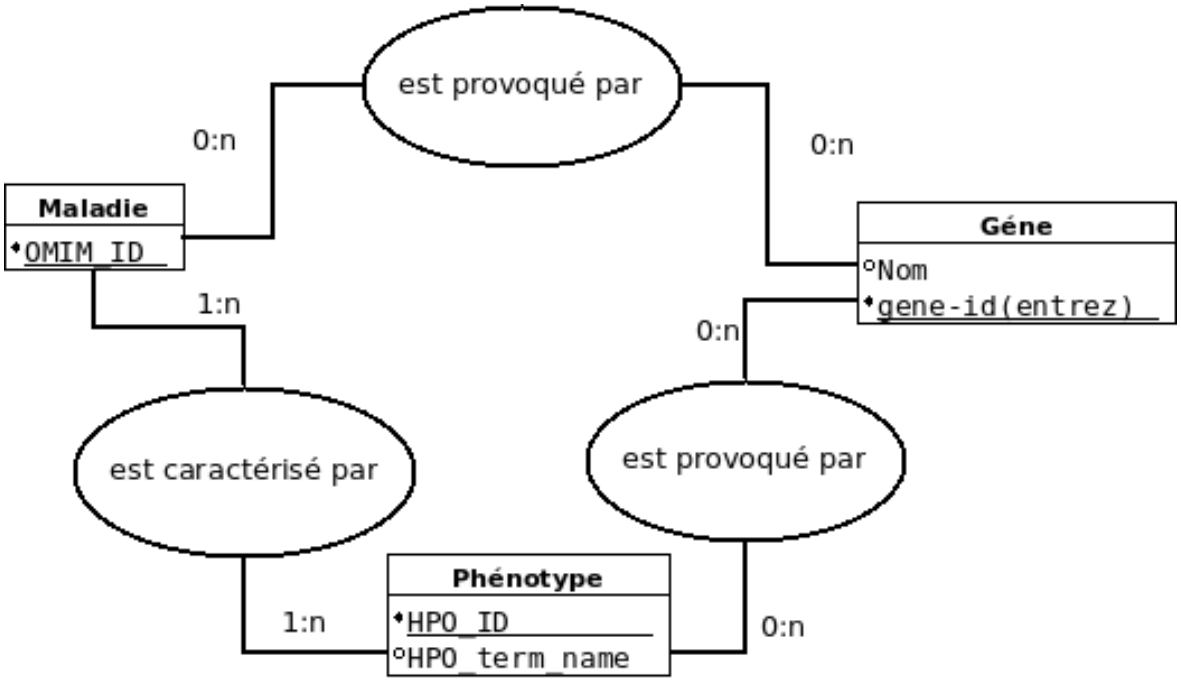


FIGURE 4.8 – Le schéma entités-association de l'export de l'ontologie HPO

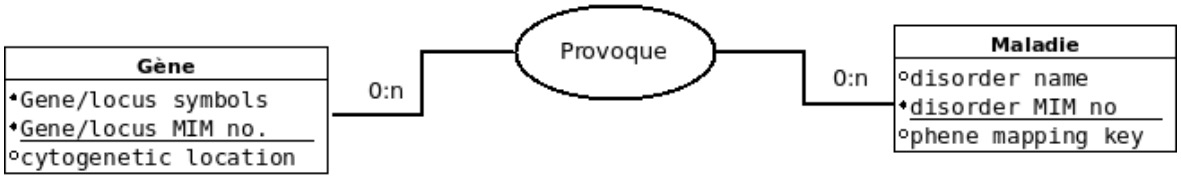


FIGURE 4.9 – Le schéma entités-association de l'export de la *morbidmap* d'OMIM

employé ? Quelles sont les structures grammaticales ? Est-ce que des homonymes ou des synonymes sont présents ?

Notre test consiste à faire décrire à des utilisateurs les exports des bases de données. Les exports sont présentés dans des tableurs et l'utilisateur doit répondre à deux questions ouvertes : "Que représente la base de données ?" et "Á quoi correspondent les différentes colonnes ?". Ces questions sont posées sur les trois bases de données pour chaque testeur. Les conversations sont enregistrées afin de s'assurer de ne rien manquer. Outre les réponses aux questions posées, le testeur est incité à verbaliser ses pensées et ses doutes, mais l'opérateur ne le guide pas. Cette liberté peut influencer grandement le contenu des conversations, puisque rien n'oblige le testeur à parler des cardinalités ou des types de données par exemple. Cette approche permet en revanche de noter ce qui est exprimé spontanément par l'utilisateur, ce qu'il considère comme important et naturel.

L'une de nos difficultés récurrentes est d'obtenir des testeurs naïfs mais cependant experts du domaine. Pour ce test, nous avions 5 participants avec des profils assez différents :

— Un généticien moléculaire, praticien hospitalier mais ayant une formation en biologie

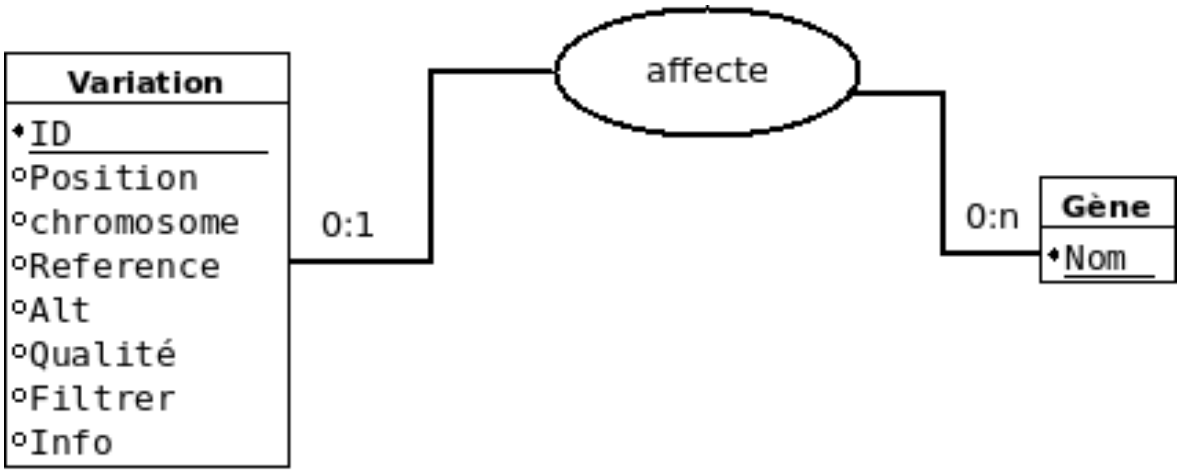


FIGURE 4.10 – Le schéma entités-association de la base de données *dbSNP*

fondamentale et appliquée. Il utilise quotidiennement les bases de données génétiques afin de rendre des résultats d'analyses, notamment les bases de données utilisées pour ce test.

— Un généticien clinicien avec un passé d'études biologiques fondamentales réalisées en amont de ses études médicales. Il utilise très fréquemment les bases de données génétiques, mais plutôt celles centrées sur les maladies et les phénotypes comme peuvent l'être OMIM et HPO.

— Une ingénieure (PhD) en génétique moléculaire, issue d'un cursus scientifique fondamental. Elle utilise beaucoup les bases de données biologiques, mais n'a pas de rapport préférentiel avec les trois bases de données utilisées au cours du test.

— Une cytogénéticienne, issue d'un cursus d'études de pharmacie. Elle n'utilise que très rarement les bases de données génétiques.

— Une biologiste moléculaire, issue d'un cursus d'études de pharmacie également. Elle utilise principalement des bases de données simples créées au sein du laboratoire, listant par exemple tous ses patients.

Au cours de cette expérience, nous souhaitons vérifier deux points majeurs :

— la capacité des généticiens à isoler un concept de leur domaine à partir de données. Plus globalement, sont-ils capables d'y rattacher les différentes notions de bases de données capitales, les attributs ainsi que les différentes relations ?

— la façon dont ils expriment ces notions, nous cherchons ici à mettre en place une méthode d'interaction leur permettant de les exprimer dans une structure informatique, nous devons donc étudier la façon dont ils les expriment naturellement : le vocabulaire est il spécifique ou existe-t-il des synonymes ou des homonymes ? Quelles sont les notions précisées spontanément ?

Enfin, l'objectif étant d'avoir des réponses les plus naturelles possibles, nous avons fait le choix d'enregistrer les conversations pour ce test, afin de limiter les intermédiaires pouvant réduire ou diriger les réponses des généticiens comme cela aurait pu être le cas avec un questionnaire par exemple.

4.4.4 Résultats

À partir des enregistrements des conversations nous avons pu vérifier notre première hypothèse et constater que les généticiens utilisent bien les notions de bases de données. On peut le constater sur des structures sémantiques ou du vocabulaire qui a un sens pour une base de données. Ces résultats sont présentés dans le tableau 4.1. Nous avons listé le vocabulaire et les structures syntaxiques afin d'étudier leurs utilisations.

Mots utilisés	Type de relation	Totaux
associé à	*0 :n*	5
avoir plusieurs	*0 :n*	5
corréler à	*0 :n*	2
en fonction de	*0 :n*	2
en rapport avec	*0 :n*	1
peut avoir	*0 :n*	1
répertorie	*0 :n*	2
associé à un ensemble	*1 :n*	1
avoir plusieurs	*1 :n*	1
associé à	*Attribut*	3
avec	*Attribut*	3
avoir plusieurs	*Attribut*	1
correspond à	*Attribut*	5
est composé de	*Attribut*	1
identifie	*Attribut*	1
lié à	*Attribut*	1
référence	*Attribut*	2
sur lequel	*Attribut*	1

TABLEAU 4.1 – Les différentes structures sémantiques utilisées pendant le test.

— Les concepts sont de façon générale facilement isolés. Cette tâche devient en revanche plus complexe lorsque les concepts ne sont pas directement dans le champ d'expertise du testeur. On obtient par exemple les citations suivantes : "*c'est le numéro OMIM de la maladie, donc cela représente la maladie*" ou "*c'est le numéro HPO du phénotype*"

— Les attributs sont bien répartis entre les concepts si ceux-ci ont été correctement identifiés. La notion d'identifiant est très présente et très souvent citée, puisque chaque concept a au moins un identifiant par base de données. Les participants les repèrent sans aucune difficulté, même pour des concepts qu'ils ne maitrisent pas :"*ce doit être l'identifiant de la maladie je suppose*".

— Les types des attributs n'étaient pas demandés aux participants. Ils ne sont jamais précisés spontanément, mais comme nous l'avons décrit plus haut, ce n'est pas un point bloquant car ils peuvent être déduits des données ou demandés spécifiquement.

— Concernant les relations, nous avons relevé 21 descriptifs différents pour parler de trois types de relation *0 :n*, *1 :n*, et *Attribut* qui est une relation d'héritage. Les trois descriptifs revenant le plus souvent sont "*avoir plusieurs*" et "*associé à*" qui décrivent une relation *0 :n*, ainsi que "*correspond à*" pour décrire une relation entre un concept et son attribut. Les testeurs utilisent donc de nombreux synonymes, c'est-à-dire plusieurs mots pour décrire une relation. Un seul utilisateur peut utiliser jusqu'à 4 synonymes pour un seul type de relation. Plusieurs homonymies, c'est-à-dire l'utilisation du même mot pour décrire plusieurs types de relations, ont également été relevées. Elles sont liées à la présence de plusieurs participants, car à la différence des synonymes, chaque participant est resté constant, en n'utilisant un mot que pour un type de relation. Les deux homonymies relevées sont "*associé à* " et "*avoir plusieurs*". Ces deux termes sont utilisés pour décrire les trois types de relations présentes, mais sont cependant majoritairement utilisés pour décrire des relations "*0 :n*".

— Enfin, les cardinalités, au même titre que les types, n'étaient pas demandées. Elles ne sont pas toujours décelables lors des conversations. Lorsqu'elles sont précisées, cela passe par l'utilisation de verbes modaux que sont "*peut*" et "*doit*", ainsi que par l'utilisation de déterminants particuliers comme "*plusieurs*". Cela construit des structures telles que

"*une maladie doit avoir un ou plusieurs phénotypes*".

4.4.5 Remarques additionnelles

En plus des remarques liées à l'emploi du vocabulaire, d'autres remarques ont pu être faites concernant le comportement et les raisonnements explicités par les participants. Nous avons ainsi pu constater que certains attributs étaient problématiques, notamment l'identifiant OMIM. Il y a deux types d'identifiants OMIM, basés sur le même format, c'est-à-dire un nombre à 6 chiffres, mais qui peuvent correspondre à un gène ou une maladie. Certains utilisateurs, ceux qui maitrisaient cette base de données, ont indiqué le problème et ont correctement identifié les attributs en recherchant un exemple sur l'interface web de la base OMIM. Parmi les trois autres utilisateurs, un seul a fait une erreur, mais il est probable que l'absence d'erreur des deux autres ne soit que fortuite.

Globalement, les exemples apparaissent d'une importance capitale, tous les participants retournent à des exemples concrets et se basent d'abord sur les données avant de regarder l'entête descriptive. Ce comportement peut engendrer des situations étranges, où l'utilisateur va rechercher sur internet une information présente sur la première ligne du tableur. Dans tous les cas, le fait de montrer les données apparait d'une importance capitale pour la compréhension de l'utilisateur

La provenance des données semble également un paramètre important à prendre en compte. Les concepts sont très fortement corrélés à la source d'où ils proviennent traditionnellement dans l'esprit des utilisateurs. Cette source n'est pas forcément la source officielle, mais peut être celle où l'utilisateur a l'habitude de rencontrer le concept ou un attribut particulier de ce concept. Les sources références sont donc différentes en fonction des utilisateurs et des spécialités.

L'un des utilisateurs avait un domaine de compétence trop extérieur aux trois bases de données tests, ce qui a rendu le processus très compliqué. Peu de concepts et de relations ont pu être identifiés, ce qui indique la nécessité d'une expertise dans le domaine pour en extraire les informations et donc pour en construire l'ontologie. Même si cela paraît évident globalement, il est intéressant de noter qu'au sein d'un même laboratoire, les spécialités peuvent être si différentes que deux utilisateurs ne manipulent pas du tout les mêmes concepts au quotidien. Cela rappelle la grande spécialisation du domaine et le grand besoin d'adaptabilité des solutions.

Enfin, la durée des enregistrements est en moyenne de 30 minutes, ce qui est assez constant d'un participant à l'autre. Le temps utile, c'est-à-dire le temps passé à réellement décrire les bases de données se situe autour de 10 minutes. Le reste du temps est surtout consacré aux vérifications sur internet et aux réflexions des participants.

4.5 Discussion

Les résultats de l'expérience nous permettent de répondre aux questions initiales concernant la capacité des généticiens à expliciter des concepts de leur domaine. Comme nous l'avons observé et décrit dans la partie "Résultats" (4.4.4), ils n'ont aucune difficulté à isoler les concepts et à leur assigner des attributs. Les relations sont exprimées par un vocabulaire plus varié, pouvant rendre plus délicate la mise en place d'une méthode d'interaction.

Les remarques additionnelles nous ont également permis de déduire des lignes de conduite pour la mise en place d'un prototype, telles que la très grande dépendance aux exemples, la provenance des données ou des cas particuliers importants à prendre en compte.

Les résultats de cette expérience préliminaire ne sont pas contraires à nos hypothèses de départ et permettent d'envisager une méthode de construction semi-automatique d'ontologies par les utilisateurs experts, en s'appuyant sur les données du domaine.

4.6 Prototype COPUNG

Le test de vocabulaire métier prouve qu'il est possible de trouver une correspondance entre les mots utilisés par les généticiens et les notions d'ontologies nécessaires à la construction du système. Nous avons choisi une approche de construction d'ontologies *via* une agrégation incrémentale de bases de données choisies par l'utilisateur. Notre approche est de nous appuyer sur l'expertise de l'utilisateur pour indiquer les notions primordiales de bases de données (quels sont les concepts, les attributs, les relations, *etc*...). Cette section présente la construction du prototype COPUNG, pour *Constructeur d'Ontologies Pour Utilisateur Novice Généticien*.

4.6.1 Principe de COPUNG

Avant de présenter la mise en place concrète du prototype COPUNG, il est important de rappeler le contexte d'utilisation prévu, ainsi que les objectifs que nous nous sommes fixés. Ces rappels permettent de déduire un fonctionnement théoriquement efficace permettant d'accomplir ces objectifs dans le cadre de son utilisation.

4.6.1.1 Rappel du contexte et principe

Le principe de ce prototype est de faire construire à un utilisateur une ontologie par l'ajout successif d'exports de bases de données relationnelles. Ce principe peut être complété avec plusieurs notions précisant le contexte d'utilisation :

— Les exports proviennent de bases de données relationnelles, ils sont structurés sous la forme de fichiers plats tels que des *.csv* ou des *.tsv*. Ce postulat correspond à nos observations du domaine, les bases de données génétiques étant disponibles au téléchargement sous ce type de format dans la très grande majorité des cas.

— Les concepts présents dans ces exports se recoupent d'une base de données à une autre. Ce point est un postulat important. Il est impossible de créer une ontologie à partir de deux exports de bases de données n'ayant aucun concept commun. Cependant, ce postulat est basé sur nos observations du domaine au cours desquelles nous avons pu constaté la présence quasi-systématique de concepts centraux, tels que sont les gènes ou les maladies.

— L'utilisateur est un expert du domaine, il maîtrise parfaitement les concepts présents dans les bases de données. Cependant, il n'a pas de compétences informatiques particulières, il est notamment novice dans la structuration de bases de données ou d'ontologies.

— L'utilisateur a cependant un modèle mental pouvant servir de base à une ontologie, comme nous l'avons évalué dans la section 4.4.4.

À partir de ce principe et de ce contexte nous pouvons définir les objectifs que doit atteindre le prototype.

4.6.1.2 Objectifs de COPUNG

Le prototype doit remplir plusieurs objectifs :

— Il doit permettre l'ouverture et la visualisation des fichiers d'exports. Cela suppose la possibilité pour l'utilisateur de définir des paramètres de *parsing* adaptés à chaque export de base de données, notamment le type de séparateur ou la présence d'un entête par exemple.

— Il doit permettre à l'utilisateur d'indiquer les concepts présents dans le fichier actif. La capacité de l'utilisateur à effectuer cette tâche a été vérifiée dans la section 4.4.4, cependant l'interface doit permettre à l'utilisateur d'accomplir cet objectif en minimisant la charge mentale.

— Les concepts extraits doivent pouvoir être ajoutés à l'ontologie issue des bases de données précédentes (s'il y en a une).

— Enfin, l'ensemble du processus ne doit requérir que des compétences dans le domaine cible, dans notre exemple la génétique, et non des compétences en informatique.

La définition de ces objectifs, en lien avec le contexte d'utilisation décrit ci-dessus, nous permet d'élaborer le fonctionnement du prototype.

4.6.1.3 Fonctionnement souhaité

Le fonctionnement général du prototype est assimilable à une boucle iterative. À chaque tour de la boucle, un nouvel export de base de données est ajouté à l'ontologie. Pour chaque nouvel export, la première étape est de trouver les paramètres de *parsing* adaptés, afin de pouvoir traiter son contenu. Cette phase doit permettre à l'utilisateur d'adapter les paramètres à toutes les situations, en jouant sur plusieurs champs de paramètrage. La seconde étape est l'extraction des concepts et de leurs attributs. Cette phase repose grandement sur les compétences de l'utilisateur, à la différence des approches bibliographiques utilisant très majoritairement de l'intelligence artificielle à cette étape. L'appui sur les compétences de l'utilisateur n'est pas une méthode transposable à toutes les situations, mais est très adapté dans notre cas où les concepts sont complexes et les utilisateurs disponibles. Enfin, la troisième étape est de lier les concepts extraits à l'ontologie en établissant des relations. Ce fonctionnement général se répète pour chaque nouvelle base de données ajoutée. Un schéma récapitulatif du fonctionnement d'un tour de boucle est présenté dans la figure 4.11.

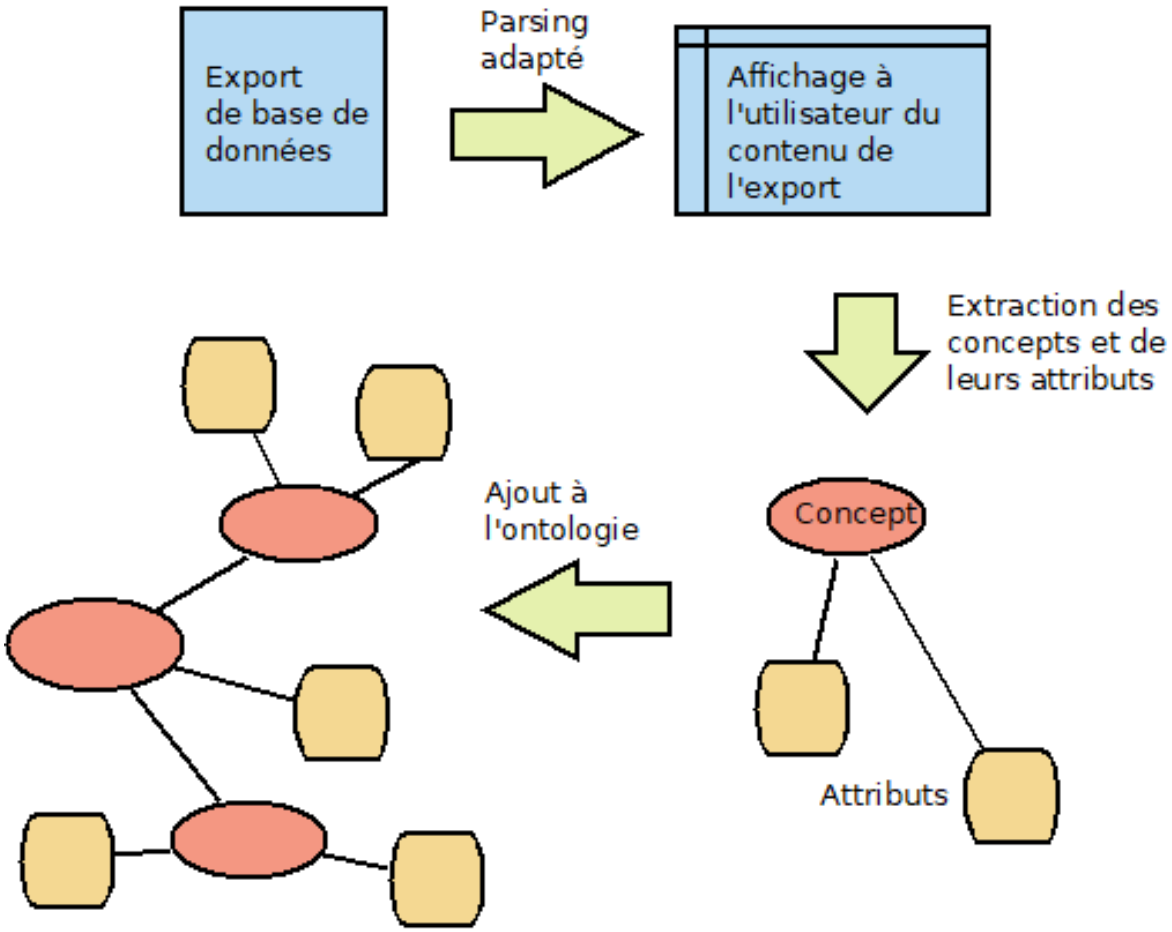


FIGURE 4.11 – Schéma récapitulatif de l'ajout d'un export de base de données à l'ontologie

Plusieurs points issus de la bibliographie et de notre précédente expérience permettent de guider la conception du prototype. Les étapes de détermination des paramètres de *parsing* et

d'extraction des concepts doivent se faire avec les données disponibles à la vue de l'utilisateur. Nous avons en effet constaté dans la section 4.4.4 que lors du processus d'identification des concepts, les utilisateurs s'appuyaient principalement sur les données visualisées. On se trouve ici typiquement dans le domaine de la programmation basée sur l'exemple.

Cette même étape d'extraction doit également se faire *via* une interface guidant l'utilisateur sur les notions essentielles à extraire, en particulier les attributs identifiants, qui peuvent servir de clé primaire. En effet, si nous avons constaté que les utilisateurs étaient capables de préciser la plupart de ces paramètres, ils ne sont pas spontanément précisés, il nous faut donc les demander explicitement.

Enfin, pour construire l'interface permettant de lier les concepts à l'ontologie, il nous parait important de nous appuyer sur les résultats décrits dans la section 4.4.5, où nous avons constaté que les utilisateurs exprimaient les relations entre les concepts par des phrases contenant des verbes modaux. Cette interface peut être construite en mimant cette syntaxe et en reprenant le vocabulaire exprimé dans la section 4.4.4. Ce système nous garantit, tout comme le précédent, que les différentes notions essentielles seront renseignées tout en préservant une structure facilement compréhensible par les utilisateurs.

4.6.2 Développement

Nous avons mis au point et développé un prototype d'interface servant de traducteur entre le généticien et son ontologie, que nous avons confronté à nos utilisateurs. Ce prototype se nomme COPUNG. Le prototype est développé en PHP et Javascript.

Comme nous l'avons constaté, les utilisateurs se rattachent fortement aux données visualisées. Nous avons donc fait en sorte qu'elles soient visibles tout au long du processus. Celui-ci se compose de 4 étapes :

— L'import et le *parsing* des données

— La création des concepts présents, avec leurs attributs

— La création des relations entre les concepts, avec leurs cardinalités

— La visualisation de l'ontologie créée

4.6.2.1 Import des données

Ces différentes étapes sont effectuées au travers d'une interface web. Les technologies utilisées sont donc les technologies web classiques PHP, *Javascript*, HTML, CSS et MySQL. La première étape permet à l'utilisateur de charger un export de base de données et de le visualiser directement. Les données sont présentées dans un tableau sur la moitié inférieure de l'écran. Il peut ainsi tester en direct un jeu de paramètres de *parsing* qui lui permettent d'afficher correctement les données et donc de les traiter de manière adéquate par la suite. L'utilisateur dispose d'un bouton pour sélectionner un export de base de données intitulé "*Choix de fichier*", d'un compteur pour indiquer le nombre éventuel de lignes à sauter, d'une liste déroulante avec les séparateurs et enfin d'une *radiobox* indiquant la présence ou l'absence d'un entête décrivant les données. Deux captures d'écran illustratives sont présentées dans les figure 4.12 et 4.13. La première présente l'interface de saisie, la seconde présente l'intégralité de la page, y compris l'affichage des données lorsque les paramètres sont renseignés. Chacune des saisies est associée à une aide, disponible en passant le pointeur de la souris sur le point d'interrogation. Enfin, deux boutons sont présents, le premier, "*Tester les paramètres*", permet de tester le jeu de paramètres, c'est-à-dire mettre à jour l'affichage des données en prenant en compte les nouveaux paramètres de *parsing*. Le second bouton, "*Valider les paramètres*" permet de valider ces paramètres d'affichage et de passer à l'étape suivante.

FIGURE 4.12 – L'interface d'import des données dans le prototype COPUNG, avant le test des paramètres.

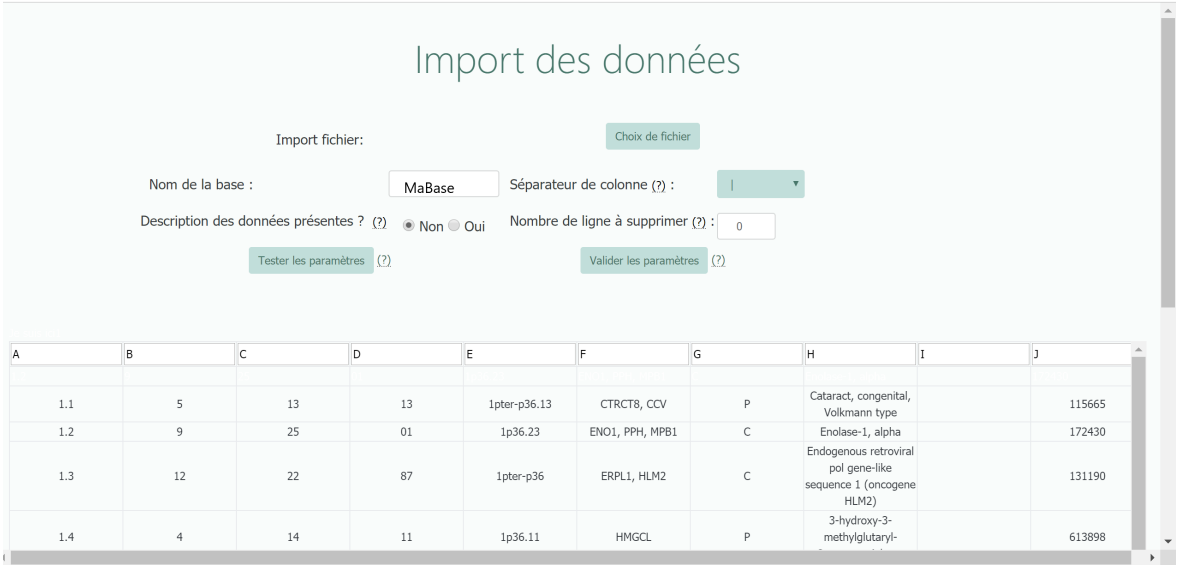


FIGURE 4.13 – L'interface d'import des données dans le prototype COPUNG avec l'affichage des données.

4.6.2.2 Création des concepts

Lorsque l'affichage des données est clair, l'utilisateur passe à la saisie des concepts et de leurs attributs. Les données sont toujours présentées sur la moitié inférieure de l'écran. Cette phase se déroule en deux temps, l'utilisateur commence par choisir un concept existant dans l'ontologie ou un nouveau concept à partir d'une liste déroulante. Un exemple de l'interface complète est présentée figure 4.14. La partie dédiée aux saisies est présentée figure 4.15. Une fois le concept choisi, l'utilisateur peut le renommer ou lui attribuer un synonyme. Il peut ensuite visualiser les attributs du concept en cliquant sur le bouton "*Choisir ce concept*". Cette première étape joue le rôle de vérification et permet d'éviter la double saisie d'un concept. Enfin, une vue de l'interface lorsque plusieurs concepts sont saisis est présentée figure 4.16

Une fois le concept choisi, l'utilisateur visualise la liste des attributs associés à ce concept sous la forme d'un tableau de quatre colonnes. La première colonne, intitulée "*Actions*" permet de supprimer l'attribut, c'est un bouton illustré par une croix rouge. La seconde indique le nom de l'attribut ou un champ texte à remplir s'il est en cours de création. La troisième colonne contient une *checkbox* pour indiquer si cet attribut est un identifiant ou non. Au moins un identifiant doit être sélectionné pour le concept. Enfin, la dernière colonne contient un champ texte permettant de rentrer le nom de la colonne correspondant aux attributs.

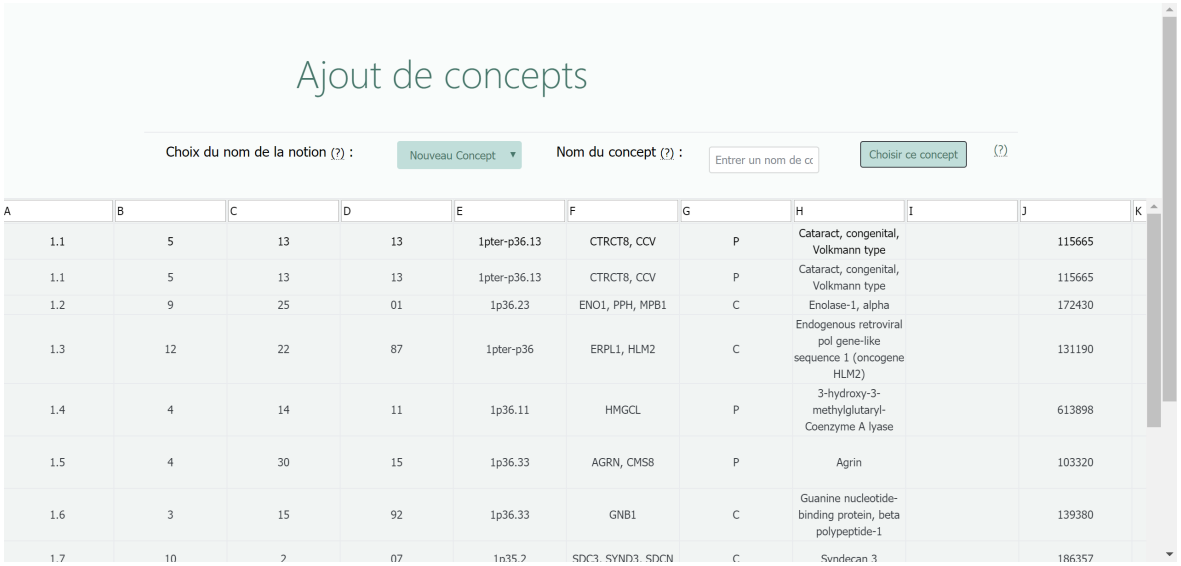


FIGURE 4.14 – L'interface d'ajout des concepts dans le prototype COPUNG

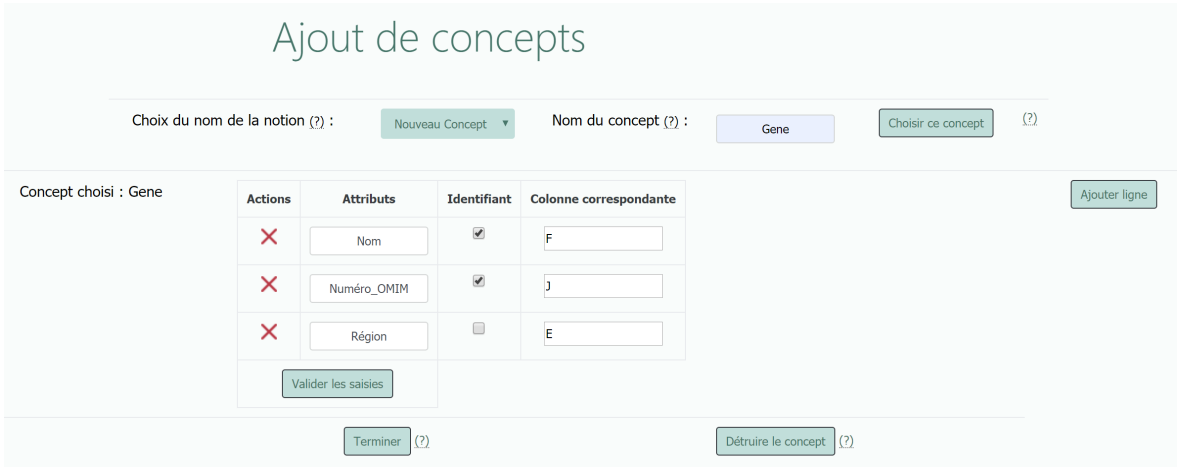


FIGURE 4.15 – L'interface de création d'un nouveau concept dans le prototype COPUNG

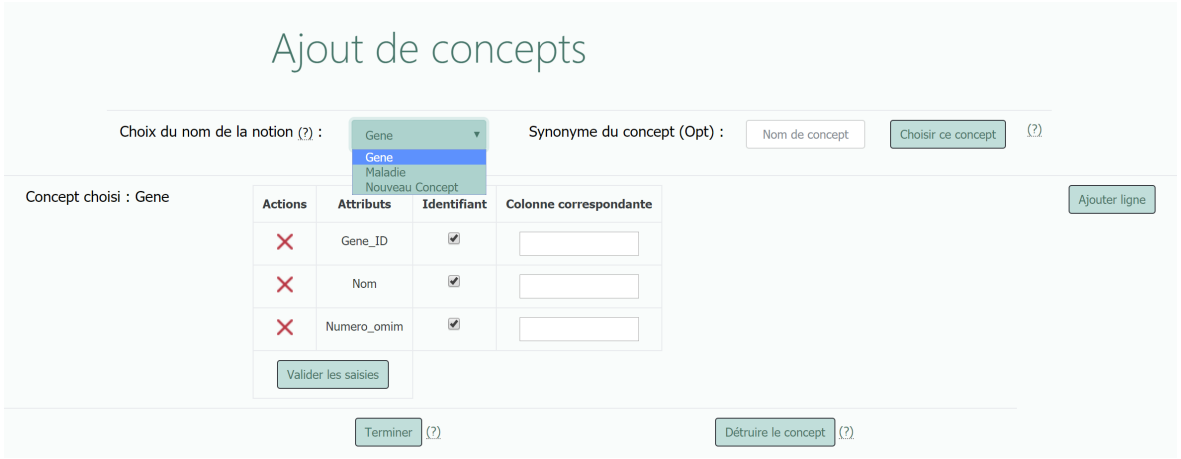


FIGURE 4.16 – L'interface de création des concepts dans le prototype COPUNG

Un bouton "*Ajouter ligne*" permet d'ajouter des lignes pour ajouter des attributs, un second bouton "Détruire le concept" permet de détruire complètement le concept et ses attributs. Enfin, le bouton "*Valider les saisies*" permet d'enregistrer les saisies dans l'ontologie, et celui

"*Terminer*" permet de passer à l'étape suivante.

Les concepts sont stockés dans une ontologie, permettant de les définir en les associant à un identifiant unique, ce qui permet de gérer les synonymes et de stocker leurs différentes propriétés ainsi que leurs attributs. Un exemple de ce stockage est proposé figure 4.17. Chaque entrée est associée à un numéro unique incluant un mot-clé correspond au type de classe : concept, attribut ou relation, ainsi qu'une suite de chiffres unique. Les concepts ont pour propriétés un nom et des synonymes éventuels. Les attributs sont définis comme des sous-classes de concepts. Ils sont donc associés à un concept particulier *via* son identifiant. Leurs propriétés sont plus nombreuses, elles contiennent le nom, les éventuels synonymes, les bases de données d'origines ainsi que la capacité identifiante de l'attribut.

```
Class Declaration concept180525120248035500
concept180525120248035500 DTA:name Gene
Class Declaration attribut180525120248037300
attribut180525120248037300 DTA:name Gene_ID
attribut180525120248037300 SubClassOf concept180525120248035500
Class Declaration attribut180525120248039700
attribut180525120248039700 DTA:name Nom
attribut180525120248039700 SubClassOf concept180525120248035500
attribut180525120248039700 DTA:origin numero_omim
attribut180525120248039700 DTA:identifiant on
```

FIGURE 4.17 – Le concept de gène tel qu'il est sauvegardé dans l'ontologie

4.6.2.3 Ajout des relations

Lorsque les concepts sont saisis, l'utilisateur peut définir des relations entre eux. La syntaxe est très proche de celle utilisée par les participants au test du vocabulaire métier, en s'appuyant notamment sur l'usage de listes déroulantes afin de canaliser l'utilisateur et sur l'emploi de verbes modaux et de structures sémantiques adaptées.

La saisie se fait *via* cinq champs et une validation présentés dans la figure 4.18. Le premier champ est le nom du premier concept. La saisie se fait par une liste déroulante listant tous les concepts de l'ontologie. Le second est le type de la relation, permettant de définir une partie de la cardinalité. L'utilisateur a le choix entre deux verbes semi-auxiliaires "*peut*" et "*doit*" là aussi présentés sous la forme d'une liste déroulante. Le troisième champ est libre, il s'agit du nom de la relation. Le quatrième champ permet de préciser l'autre partie de la cardinalité. L'utilisateur a cette fois le choix entre "*un ou plusieurs*" ou "*un seul*". Enfin, le dernier champ est le second concept de la relation, présenté sous la forme d'une liste des concepts présents dans l'ontologie, de la même façon que le premier concept. À partir de cette étape, les données

FIGURE 4.18 – L'interface de création des relations dans le prototype COPUNG

ne sont plus affichées. En effet, les relations à saisir doivent permettre d'ajouter les nouveaux

concepts à l'ontologie existante, les données du dernier export ne peuvent donc pas servir d'appui pour cette tâche. À la place du tableau de données se trouve donc le tableau de relations présentant toutes les relations déjà définies dans un tableau à 6 colonnes. Les cinq premières correspondent aux saisies décrites précédemment, la dernière est une croix rouge permettant de supprimer la relation. Un exemple de ce tableau contenant une relation est présenté dans la figure 4.19. Les relations sont stockées dans l'ontologie au même endroit

FIGURE 4.19 – L'interface de création des relations dans le prototype COPUNG avec une visualisation de relation

que les concepts. Leur définition se fait *via* le même mécanisme de déclaration de classe, l'identifiant est composé du mot-clé "Relation" ainsi que d'une suite de chiffres unique. Les relations n'ont pas d'attributs mais 5 propriétés permettant de stocker les différentes saisies de l'utilisateur. Les propriétés CAlpha et COmega correspondent aux identifiants des concepts impliqués dans la relation, le type et le nombre correspondent aux cardinalités. Ils sont stockés sous la forme entrée par l'utilisateur. Enfin, une dernière propriété permet d'enregistrer le nom donné par l'utilisateur à la relation comme présenté dans la figure 4.20.

```
Relation Declaration relation180525120516155200
relation180525120516155200 OAP:CAlpha concept180525120248035500
relation180525120516155200 OAP:COmega concept180525120305425800
relation180525120516155200 OAP:type peut
relation180525120516155200 OAP:number un_plusieur
relation180525120516155200 OAP:name Affecte
```

FIGURE 4.20 – La sauvegarde d'une relation dans l'ontologie du prototype COPUNG

4.6.2.4 Visualisation

L'étape de visualisation permet de résumer simplement les concepts et les relations présents dans l'ontologie. Dans le prototype testé, elle n'est pas disponible pour les utilisateurs, son interface est donc très simpliste. Il s'agit d'une liste des concepts et de leurs attributs. Les liens entre les concepts sont indiqués par la présence de l'identifiant du concept lié parmi leurs attributs, comme le serait une clé étrangère.

4.6.3 Objectif et protocole du test du prototype

L'objectif principal du test du prototype COPUNG est d'évaluer la faisabilité de notre approche. Si le test est un succès, même partiel, nous pourrons conclure que l'approche est possible. En revanche si le test est un échec, nous ne pourrons conclure, car cela pourra provenir d'une approche irréalisable ou tout simplement d'une solution d'interface inadaptée. Dans les deux cas, l'objectif est d'évaluer les différentes étapes et d'identifier les points de blocages éventuels.

Nous avons mis en place un protocole de test que nous avons soumis à 4 utilisateurs. Par rapport au test initial concernant le vocabulaire métier, nous avons éliminé l'utilisateur dont le domaine de compétence ne correspondait pas à notre exemple et remplacé un second utilisateur par un nouveau participant pour des raisons logistiques. Ce nouvel utilisateur est interne en génétique clinique et a une bonne connaissance des bases de données génétiques. Les trois autres utilisateurs ont donc passé les deux tests. Afin d'éviter que le premier test n'influe sur le second, nous n'avons pas discuté des résultats du premier test avec les participants et nous avons attendu deux mois entre les deux tests.

Chaque testeur est placé devant le prototype et les données des trois mêmes bases de données tests. Le scénario lui est expliqué : il doit regrouper les données de trois bases de données afin de se créer un module d'annotation personnalisé. Nous laissons ensuite les utilisateurs procéder et observons leurs actions. L'utilisateur est ici aussi encouragé à verbaliser ses pensées, il est enregistré et chronométré.

Les résultats seront évalués sur deux séries de critères, l'une objective, l'autre subjective. La première est obtenue en comparant les résultats de l'utilisateur au schéma global présent figure 4.21. Combien de concepts sont correctement créés ? Combien de relations sont correctement décrites ? Combien d'attributs sont bien répartis ? Nous complèterons cette série de critères en comparant les temps passés par les utilisateurs. Nous pourrons ainsi quantifier objectivement la réussite de nos utilisateurs et la mettre en relation avec le temps nécessaire à la création de l'ontologie.

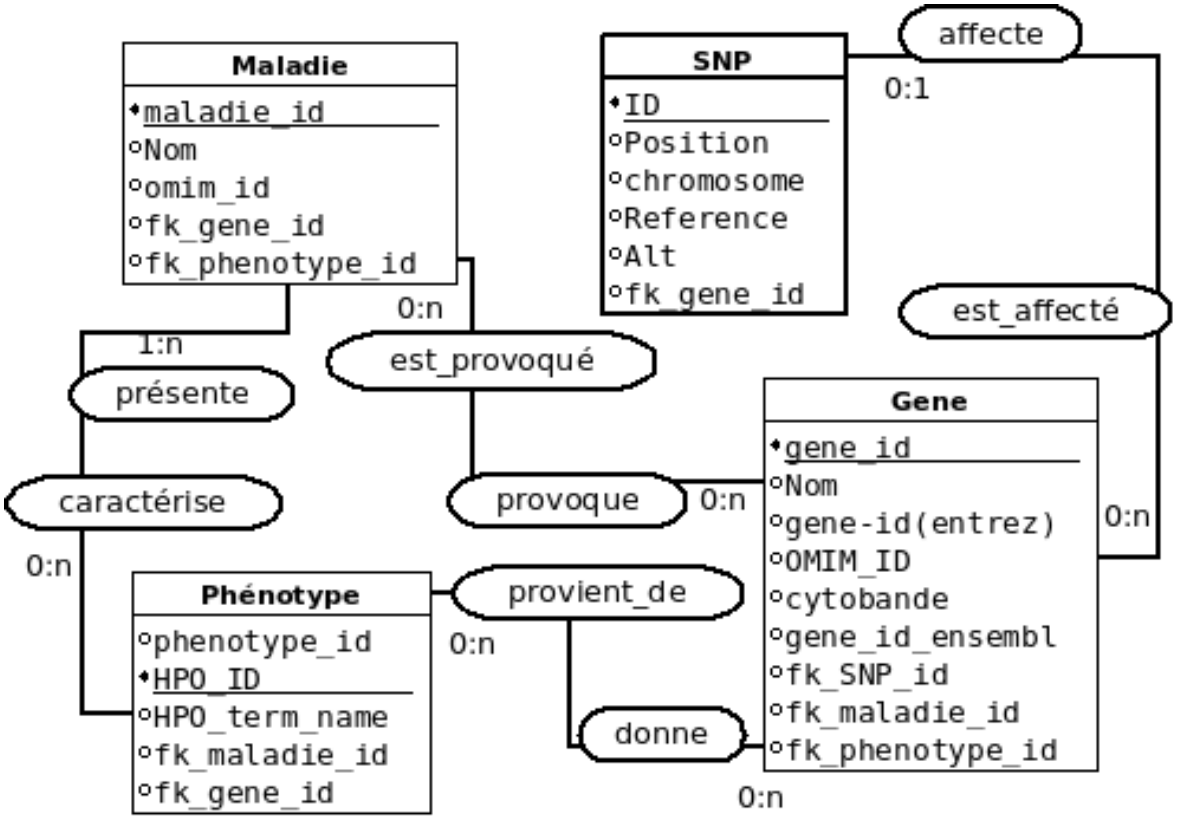


FIGURE 4.21 – Le schéma entités-association global des trois bases de données OMIM, HPO et *dbSNP*

Concernant la série subjective, elle est constituée de réponses de l'utilisateur à nos questions après le test. Nous leur demandons de répondre en plaçant leurs réponses sur une échelle allant de 1, pas du tout, à 5, parfaitement. Les questions sont au nombre de trois :

— Est-ce que le scénario parait crédible ?

— Est-ce que les tâches paraissent compliquées ?

— Est-ce que l'interface était claire ?

Enfin, le prototype est repris étape par étape, en demandant ce que l'utilisateur a compris de chaque étape, afin de compléter ses commentaires enregistrés. Nous lui demandons également ce qui, selon lui, pourrait améliorer l'interface.

4.6.4 Résultats

Comme on peut le voir dans la figure 4.21, il y avait 4 concepts à identifier, Maladie, SNV, Gène et Phénotype. Quatre relations liaient ces concepts. Ce schéma est basé sur les relations effectivement présentes dans les données, c'est-à-dire les correspondances concrètes dans ces trois exports. Par exemple, il y a une relation entre les gènes et les phénotypes car les deux concepts sont présents dans l'export provenant de la base de données HPO et sont liés par un système d'identification. En revanche, il n'y a pas de relation entre les variations nucléotidiques et les phénotypes car aucun fichier ne permet de lier ces deux concepts. Pour autant, il n'est pas insensé de définir une telle relation dans la pratique courante.

4.6.4.1 Ontologie de l'utilisateur *F*

L'ontologie produite par l'utilisateur F est présentée figure 4.22. On peut observer que beaucoup trop de concepts sont présents, 9 au lieu des 4 attendus. Les relations sont donc logiquement augmentées, au nombre de 11. On peut cependant noter que plusieurs relations ont des cardinalités en *1 :1*, ce qui signifie qu'elles ne devraient pas exister. On peut retravailler le schéma de cet utilisateur en simplifiant ses relations pour obtenir la figure 4.23. Ce schéma est beaucoup plus proche de l'objectif, les 4 concepts sont identifiés : Maladie et Gène sont identiques, Symptômes correspond au Phénotype et Variation nucléotidique correspond au SNV. On note toutefois 2 concepts supplémentaires, la cytobande et le transcrit. Ces deux concepts annexes étaient considérés comme des attributs ou absents dans les données, mais peuvent effectivement être considérés comme des concepts à part entière dans une vue plus globale du domaine. Nous reviendrons sur ce concept de transcrit dans la partie discussion. Les relations sont décrites sauf celle entre les gènes et les variations phénotypique. À la place,

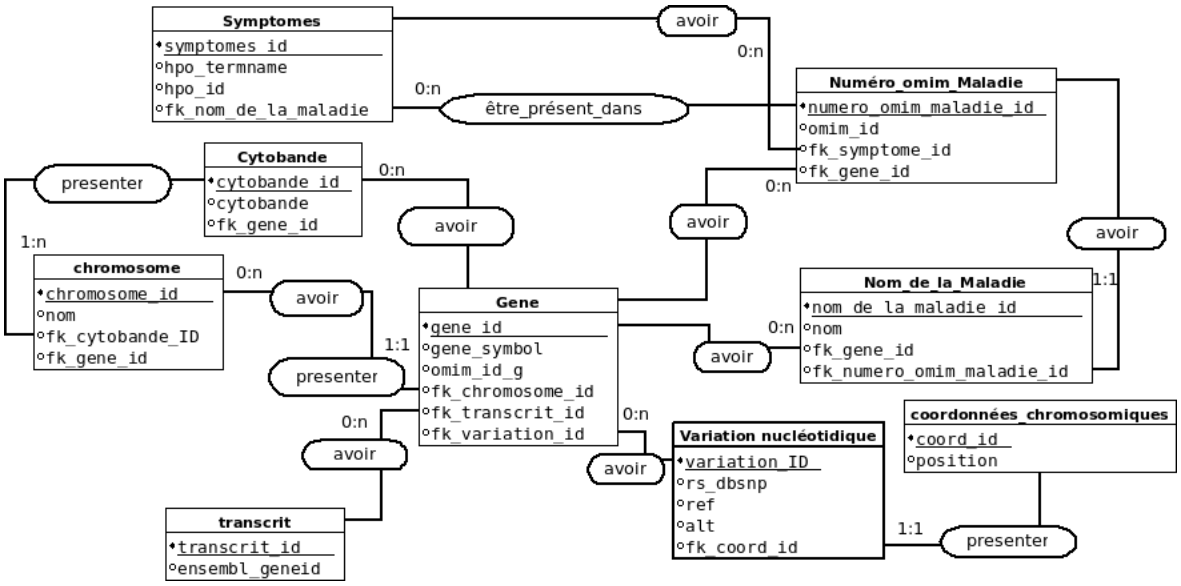


FIGURE 4.22 – Le schéma de l'ontologie produite par l'utilisateur F

l'utilisateur a créé une relation directe entre maladie et phénotype. Ici aussi, cette relation n'était clairement pas présente dans les données, mais peut se justifier sur une vue globale du domaine. Les cardinalités saisies sont bonnes, mais comme les relations n'ont pas été retournées, elle ne sont pas toutes présentes.

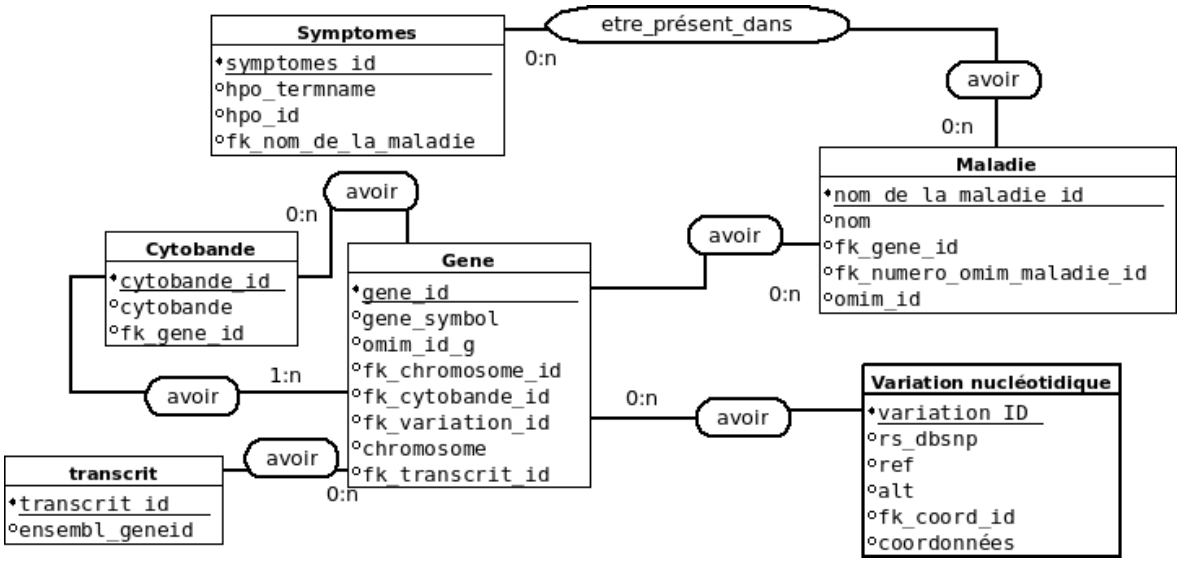


FIGURE 4.23 – Le schéma retravaillé de l'ontologie de l'utilisateur *F*

4.6.4.2 Ontologie de l'utilisateur *X*

L'ontologie produite par l'utilisateur *X* est présentée figure 4.24. Les quatre concepts sont correctement et clairement identifiés. En revanche, une relation est manquante, celle entre les gènes et la maladie. Cette relation est remplacée par une relation entre les SNV et la maladie. Cette relation n'est pas présente dans les données, elle peut s'argumenter dans certains cas sur l'ensemble du domaine, mais n'est clairement pas la vision majoritaire. Les cardinalités saisies sont justes sur les trois relations correctement indiquées. Concernant les attributs, deux

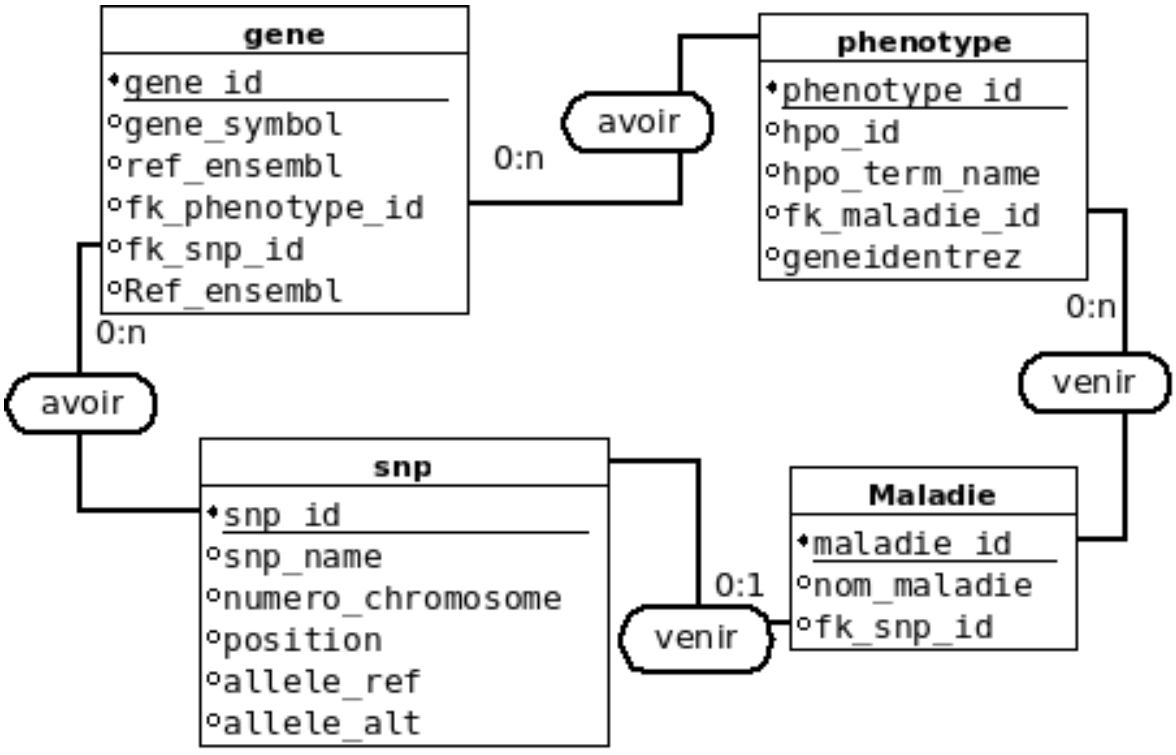


FIGURE 4.24 – Le schéma de l'ontologie produite par l'utilisateur *X*

ont été omis, l'identifiant OMIM de la maladie, confondu avec celui du gène et la cytobande, un des attributs non essentiels du gène. Les autres sont correctement répartis.

4.6.4.3 Ontologie de l'utilisateur *S*

L'ontologie produite par l'utilisateur *S* est présentée figure 4.25. Cette ontologie se distingue des autres par l'absence d'un concept et d'une relation. L'utilisateur a délibérément écarté le concept de variation nucléotidique. Nous reviendrons sur ce point dans la section discussion. Les trois autres concepts sont en revanche correctement isolés. Les relations sont au nombre de 2, dont une retournée. Il manque logiquement la relation entre la variation nucléotidique et le gène, ainsi que celle entre le gène et le phénotype. Les cardinalités sont justes, bien que l'une d'entre elles soit élargie de *1 :n* à *0 :n*. Concernant les attributs, des absences sont

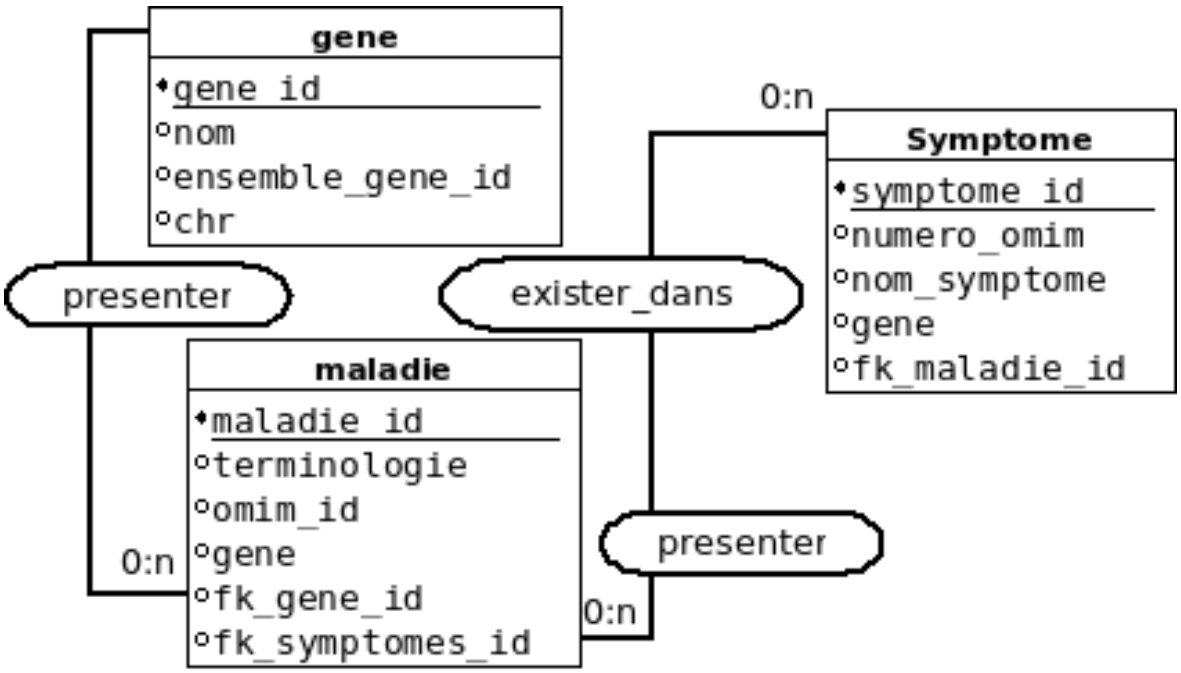


FIGURE 4.25 – Le schéma de l'ontologie produite par l'utilisateur *S*

notables, notamment dans le concept de Gène, réduit à son identifiant *ensembl*, son nom et son chromosome. Ces absences sont également volontaires et seront discutées plus en détails dans la section 4.6.4.6. On peut également noter la présence de clés étrangères avec l'identifiant des Gènes déjà présent dans le concept de Symptôme. Ce point sera également abordé dans la section 4.6.4.6.

4.6.4.4 Ontologie de l'utilisateur *G*

Enfin, l'ontologie produite par l'utilisateur *G* est le résultat le plus proche des données au niveau des concepts, présentée figure 4.26. En effet les 4 concepts sont correctement isolés, et tous les attributs sont justes, sans oublis. En revanche, il y a plusieurs erreurs au niveau des relations, avec le manque de la relation Gène - Variation nucléotidique ainsi que l'ajout de 2 relations entre les Variations nucléotidiques avec les Maladies et les Phénotypes.

4.6.4.5 Résumé des résultats

Sur les 4 concepts à isoler, 2 utilisateurs seulement les ont correctement identifiés. Cependant, l'absence d'un concept chez l'utilisateur *S.* s'explique par le fait qu'il n'est pas rentré dans le scénario proposé et a préféré construire le système qui lui paraissait le plus adapté à sa pratique, en négligeant sciemment une partie des données. L'utilisateur *F* a quant à lui isolé les concepts, mais il les a divisé en sous-concepts.

Il y avait également 4 relations présentes dans les données à trouver, entre Gène et SNV, Gène et Phénotype, Gène et Maladie et enfin Maladie et Phénotype. Aucun utilisateur n'a

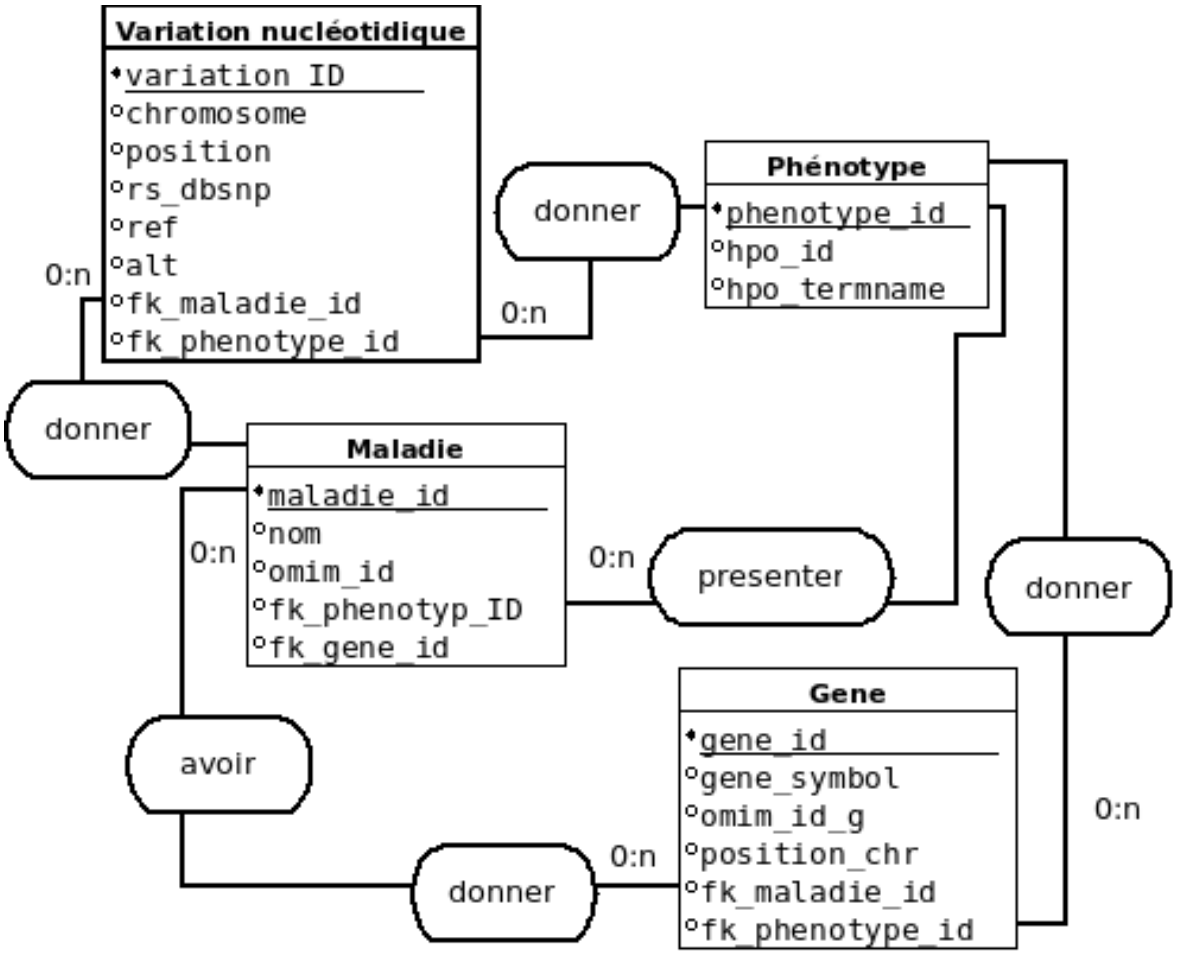


FIGURE 4.26 – Le schéma de l'ontologie produite par l'utilisateur G

trouvé les 4 relations correctes, tous n'en ont décrit que 3, sauf l'utilisateur S, qui n'en a trouvé que 2 puisque l'un des concepts était manquant. Les trois autres utilisateurs ont cependant créé des relations supplémentaires : G et X relient directement les Variations nucléotidiques aux Maladies, G relie également les Variations nucléotidiques aux Phénotypes, enfin, l'utilisateur F. ayant décrit beaucoup trop de concepts, les a liés par des relations supplémentaires.

Concernant les cardinalités, il faut préciser que les relations étaient demandées orientées. L'utilisateur pouvait ainsi les saisir deux fois en les retournant, ce qui amène à 8 le nombre de cardinalités totales potentielles. Peu de relations ont été retournées ce qui limite donc le nombre de cardinalités saisies, ainsi, les utilisateurs G et F ont saisi 4 cardinalités attendues et les utilisateurs X et S 3. Elles sont toutes bonnes, sauf la relation *1 :n* entre une Maladie et un Phénotype (une maladie a au moins un phénotype), a été élargie en lien *0 :n*. On peut aussi noter chez l'utilisateur F un grand nombre de relations *1 :1* expliquant la multiplication des concepts.

4.6.4.6 Remarques

En plus de ces résultats obtenus par comparaison avec le schéma global, quelques points particuliers ont attiré notre attention dans les résultats et le comportement des utilisateurs pendant les tests.

Plusieurs synonymes ont été utilisés au niveau des concepts, notamment les utilisations de Phénotypes et Symptômes, ainsi que de Variations nucléotidiques et SNV. On ne retrouve pas cette utilisation de synonymes au niveau des attributs car les utilisateurs ont globalement repris ceux employés dans les bases de données, allant parfois jusqu'à les copier-coller directement.

Dans le schéma de l'utilisateur S, on peut remarquer l'apparition de clés étrangères qui

ont été saisies comme étant des attributs de concepts, donc avant la création des relations. On peut probablement attribuer cette anticipation à l'expérience en création de bases de données au travers du logiciel Access [3] que possède cet utilisateur. Globalement il est celui qui est le moins entré dans le scénario, il a construit son scénario personnel en négligeant les données qui ne l'intéressaient pas et a commencé à construire sa base de données comme il sait le faire sous *Access*.

Enfin, l'utilisation de l'entête, présent dans deux des trois bases, est très intéressante. Nous avons vu précédemment qu'ils les employaient pour nommer les attributs, mais nous avons aussi pu constater qu'ils ne cherchaient pas vraiment à les interpréter au travers de deux exemples. Dans les données était présente une colonne stockant les identifiants *ensembl* des gènes. Ces identifiants sont sous la forme "*ENSG000001860952*", or les généticiens manipulent beaucoup plus fréquemment des identifiants de transcrits, provenant également d'*ensembl*, sous la forme "*ENST00000534990*". Trois des quatre utilisateurs ont associé à tort cette colonne à un numéro de transcrit, allant même pour l'utilisateur *F* jusqu'à créer un concept de transcrit complètement absent des données. Le second exemple est l'identifiant OMIM, qui sous le même format d'une suite de 6 chiffres peut décrire un gène ou une maladie. Aucun utilisateur n'a attribué ces attributs de manière incorrecte, mais deux d'entre eux les ont vérifiés en préférant consulter l'interface web d'OMIM plutôt qu'en faisant confiance à l'entête.

En plus des résultats concrets du test produits par les utilisateurs, nous leur avions demandé des résultats plus subjectifs, en leur posant trois questions. Les réponses sont présentées dans le tableau 4.2. On peut constater que le scénario parait probable, sauf pour l'utilisateur *S* qui a préféré créer son propre scénario. Les tâches paraissent faisables, mais l'interface n'apparait pas optimale, manquant en particulier d'exemples prédéfinis selon les utilisateurs. Les testeurs ont mis entre 7 et 14 minutes par base de données, avec une moyenne de 9 minutes 30. Les incorporations les plus longues sont celles des deuxièmes bases, car c'est à cette étape qu'il y a le plus de relations à saisir. Cependant ils estiment tous pouvoir aller beaucoup plus vite en maitrisant le logiciel.

Enfin, lors de la saisie des relations, plusieurs utilisateurs ont demandé s'il était pertinent de retourner les relations. Il leur a été répondu qu'ils pouvaient faire comme ils l'entendaient. Ainsi, certaines relations jugées importantes ont été retournées. Un utilisateur s'est également questionné sur la transitivité des relations et a choisi ensuite de ne pas rajouter les relations supplémentaires en partant du principe qu'on pourrait les déduire.

Question	F	G	S	X	Total /20
Plausibilité du scénario	5	5	3	5	18
Simplicité des tâches	3	3	4	3	13
Clarté de l'interface	5	3	2	3	13

TABLEAU 4.2 – Synthèse des résultats aux questions posées à chaque utilisateur après le test.

4.6.5 Discussion

Comme nous avons pu le constater, les utilisateurs ont globalement réussi à fournir un résultat implémentable à partir des données et ce malgré l'absence totale d'aide. Les assemblages produits n'étaient pas optimaux, mais ne comportent pas d'erreurs majeures empêchant leur utilisation pour construire un amas de données. En revanche, le test a permis de mettre en évidence des situations délicates qui peuvent être améliorées, que ce soit en guidant mieux l'utilisateur ou en extrapolant à partir des données.

C'est le cas par exemple des cardinalités, les relations *1 :1* peuvent être détectées et les notions concernées peuvent être proposées regroupées à l'utilisateur. À l'inverse, une notion initialement incluse dans une autre peut en être sortie à la lumière de nouvelles données afin

3. `https://products.office.com/fr-fr/access`

de créer une entité indépendante. Ici aussi, il est possible de détecter automatiquement cette situation, mais l'aide de l'utilisateur est nécessaire pour y apporter une solution. En revanche, il est impossible de détecter la création de faux concepts, comme l'a été celle du transcrit pour l'utilisateur *F*. On peut cependant espérer que l'utilisateur se rende compte tout seul du problème, puisque lorsque la remarque a été faite aux utilisateurs ayant saisi un transcrit, tous ont admis l'erreur instantanément. Le problème peut alors être réglé avec une simple suppression de concept et un ajout d'attribut dans un autre.

Des oublis de concepts peuvent être détectés si toutes les colonnes d'une base de données ne sont pas chargées, mais ces oublis peuvent être volontaires, comme avec le cas de l'utilisateur *S*. Les oublis de relations peuvent également être détectés, la présence de deux concepts dans une seule base de données indique une relation entre ces deux concepts. Les relations peuvent également être automatiquement réfléchies, les cardinalités manquantes peuvent être demandées à l'utilisateur. Enfin, les cardinalités peuvent être vérifiées dans les données. Il ne sera pas possible de trouver une réponse certaine dans tous les cas, par exemple de contredire une relation *0 :n* saisie à la place d'une *1 :n*, par contre, il est facile de vérifier une relation *0 :1*.

4.7 Conclusion

Nous avons prouvé qu'il était possible de faire reconstruire une ontologie à un utilisateur novice à partir des données sous deux conditions, qu'il soit expert du domaine et qu'il puisse s'appuyer sur des exports de bases de données issus de bases relationnelles. Ces résultats confortent l'utilisation de la programmation sur exemple, qui correspond tout à fait aux pratiques naturelles des utilisateurs, qui n'hésitent pas à vérifier leurs hypothèses en observant les données. Nous avons listé les problèmes rencontrés par les utilisateurs et proposé des solutions afin de faciliter leurs tâches et résoudre les différents problèmes rencontrés au cours du processus. Ce travail a fait l'objet d'une publication lors de la conférence *Human Computer Interaction International*, qui s'est tenue à Las Vegas en juillet 2018 (Riché-Piotaix et al., 2018).

Un travail d'ingénierie informatique est désormais nécessaire pour construire un logiciel complet et robuste basée sur cette approche, le prototype décrit dans cet ouvrage n'ayant pas d'autres objectifs que de permettre le test de l'approche.

Chapitre 5

Contribution score

5.1 Introduction et rappel

Nous avons constaté que les tâches d'analyse des données d'exomes sont complexes pour les généticiens. Elles requièrent une grande maîtrise du domaine ainsi que l'appui de méthodes informatiques. Les généticiens utilisent principalement des filtres successifs pour épurer la liste de variations génétiques et concentrer leurs efforts sur celles qui sont vraisemblablement les plus pathogènes. Cette approche par filtre est risquée, puisqu'elle est une source importante de faux-négatifs. Elle est également peu efficiente, l'analyse devant souvent être répétée plusieurs fois en élargissant les filtres pour identifier les variants candidats.

Nous avons observé une grande dépendance des généticiens vis-à-vis des bio-informaticiens sur ces questions, car de la qualité de l'annotation dépend beaucoup l'analyse finale des données. Dans le chapitre précédent, nous avons posé les bases d'un système permettant aux généticiens de réaliser eux-même cette annotation en utilisant les bases de données de référence qu'ils souhaitent. Nous avons démontré la viabilité de cette approche sur ce point particulier.

Nous pensons qu'il est possible d'améliorer les pratiques des généticiens sur les parties d'analyses des données afin qu'ils gagnent en efficacité et en efficience. Ces améliorations doivent cependant rester cohérentes avec l'ensemble de notre démarche. Cela implique que les données analysées soient diverses dans leurs contenus, puisque les généticiens travaillent sur des fichiers aux annotations variées, complexes et qui diffèrent d'un laboratoire à l'autre. De plus, cette problématique va encore s'accentuer avec la réalisation de nos travaux précédents, permettant de réaliser des annotations personnalisées. Notre méthode d'analyse ne doit donc pas être dépendante des données. Bien entendu, notre approche ne doit pas non plus entraîner une dépendance vis-à-vis du bio-informaticien.

Nous pouvons donc formuler la problématique comme ceci : "*Est-il possible d'améliorer l'efficacité et l'efficience du processus d'analyse de données génétiques d'exomes par les généticiens sans créer de dépendance ni aux données ni aux bio-informaticiens ?*"

Dans ce chapitre, nous présentons tout d'abord les différents outils de la littérature scientifique actuelle pouvant répondre au moins partiellement à la problématique. Nous synthétiserons et analyserons les réponses de ces outils, puis nous présenterons un prototype permettant de tester deux hypothèses distinctes :

Hypothèse 1 Est il possible d'améliorer l'efficacité de l'analyse de données d'exome ?

Hypothèse 2 Est il possible d'améliorer l'efficience de l'analyse de données d'exome ?

Nous présenterons les résultats obtenus lors des tests de ces deux hypothèses et les discuterons.

5.2 État de l'art spécifique

Nous avons abordé dans le chapitre "État de l'art" quelques outils et bases de données génétiques. Nous allons ici nous concentrer sur les outils d'assistance à l'analyse et à l'interprétation à destination des généticiens.

5.2.1 QueryOR

Le premier outil d'aide à l'analyse des variations génétiques se nomme *QueryOR*. Nous allons commencer par exposer un cas d'utilisation permettant de décrire le processus, puis nous verrons les points forts et les points faibles de l'approche choisie par les développeurs de *QueryOR*.

5.2.1.1 Présentation

QueryOR est un outil web d'annotation et de priorisation publié en 2017 par Bertoldi et collaborateurs (Bertoldi et al., 2017). Son fonctionnement est assez simple, ciblant un public d'utilisateurs non formés à l'informatique. Il permet d'annoter les fichiers VCF de l'utilisateur et de trier les variants en fonction des critères choisis par celui-ci.

L'utilisateur commence par créer un compte sur la plateforme, puis un projet dans lequel il charge des données sous la forme de fichier VCF. Les données qui sont groupées dans un projet sont réunies, ce n'est donc pas un format adapté pour une manipulation de séquençage d'une série de patients, mais qui vise plutôt l'analyse d'un exome d'un cas index ou d'un trio. En effet, il est généralement peu pertinent de rassembler les variations de plusieurs patients dans une seule analyse, cela augmente le bruit de fond et rend encore plus délicate la tâche d'analyse. *QueryOR* annote ensuite le fichier, comme on peut le voir dans la figure 5.1. Le temps d'exécution est de quelques minutes à quelques dizaines de minutes, ce qui est logique puisque l'annotation est effectuée par *ANNOVAR*, un annotateur rapide.

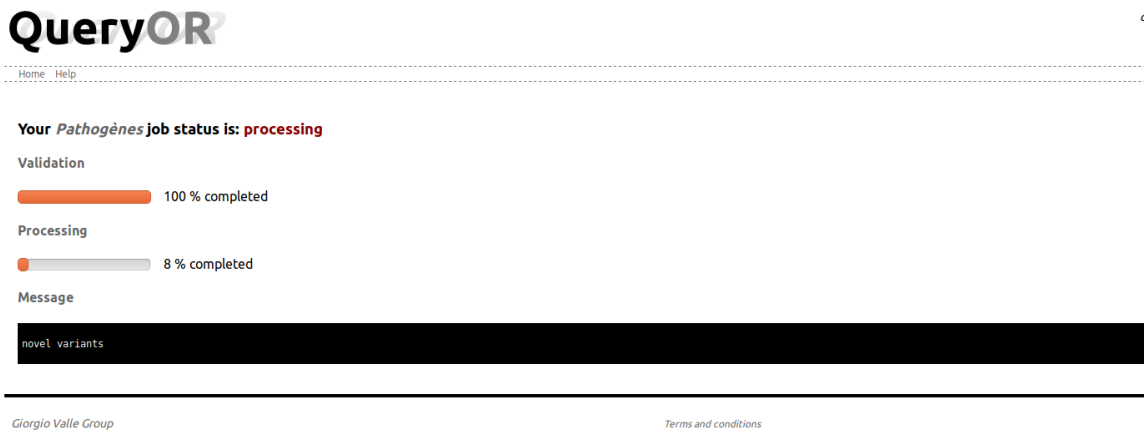


FIGURE 5.1 – Capture d'écran de l'interface d'annotation de *QueryOR*

Lorsque l'annotation est effectuée, l'utilisateur peut définir des règles en choisissant l'un des 70 critères d'annotation, puis en y associant un ensemble de valeurs possibles à une valeur de score. Si la variation se situe dans cet ensemble de valeurs, son score total sera augmenté du nombre de points correspondants à sa valeur de score. Par exemple, dans la figure 5.2, si la variation est considérée pathogène ou probablement pathogène par *ClinVar*, son score sera augmenté de 5 points. L'utilisateur peut définir un ensemble de règles qui s'appliqueront simultanément. Lorsque les règles sont définies, l'interface suivante présente un tableau récapitulatif des différentes variations et de leurs scores correspondants, comme on peut le voir figure 5.3.

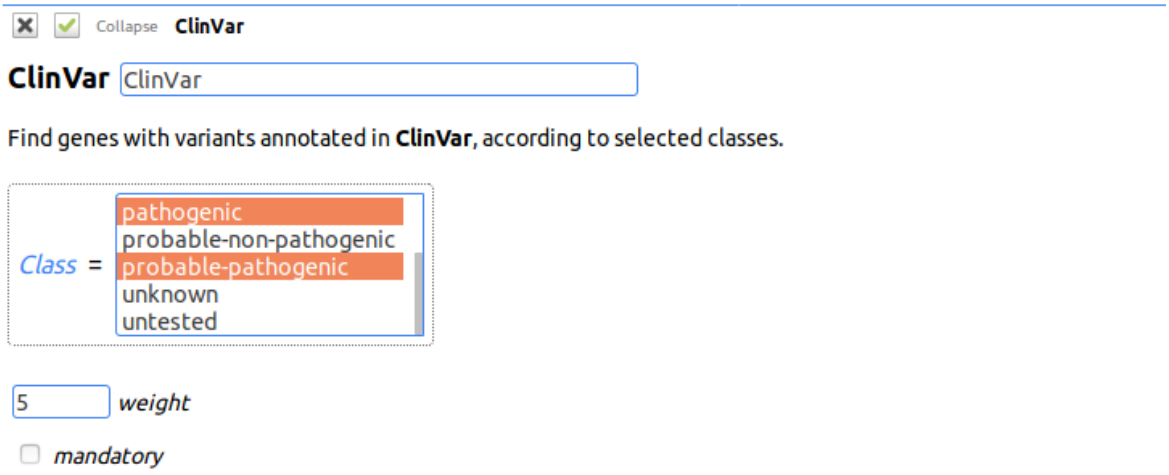
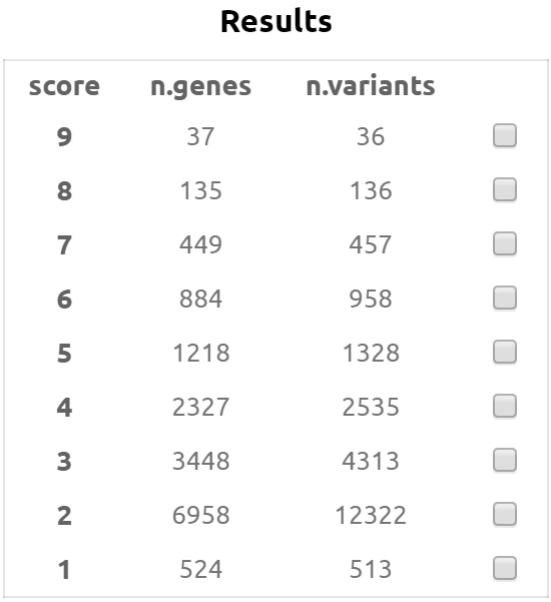


FIGURE 5.2 – Capture d'écran d'une règle élaborée sous *QueryOR*

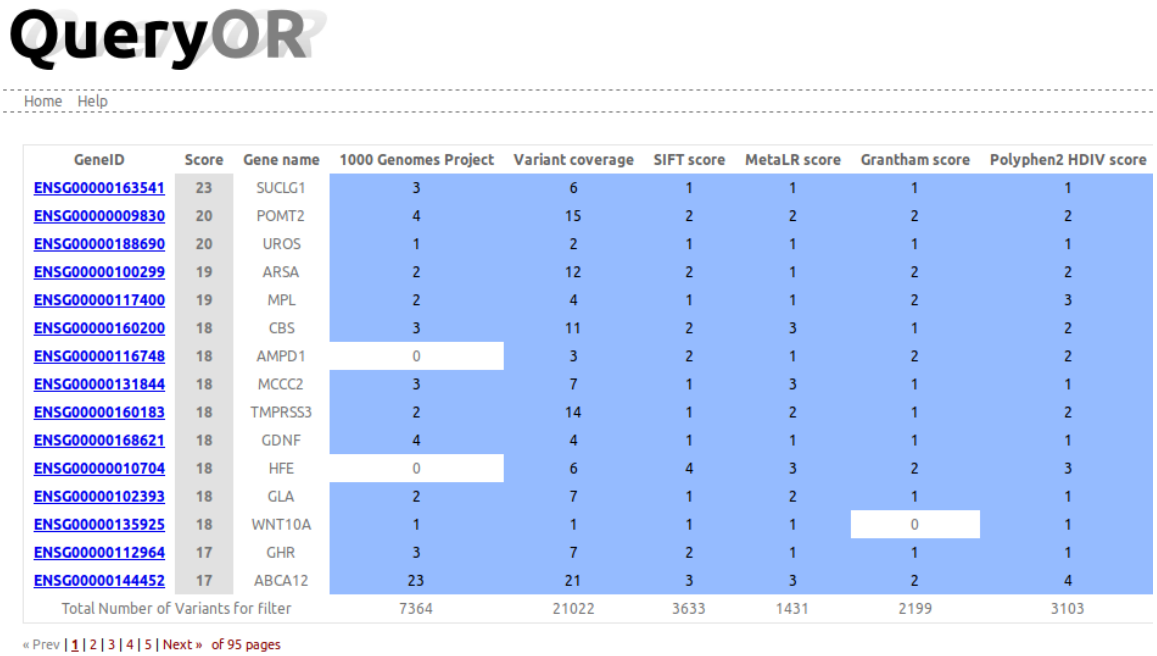
Enfin, dans cette interface, l'utilisateur peut choisir les valeurs de score qu'il considère



Results

score	n.genes	n.variants	
9	37	36	☐
8	135	136	☐
7	449	457	☐
6	884	958	☐
5	1218	1328	☐
4	2327	2535	☐
3	3448	4313	☐
2	6958	12322	☐
1	524	513	☐

FIGURE 5.3 – Capture d'écran de la présentation des résultats groupés de *QueryOR*



QueryOR

Home Help

GeneID	Score	Gene name	1000 Genomes Project	Variant coverage	SIFT score	MetaLR score	Grantham score	Polyphen2 HDIV score
ENSG00000163541	23	SUCLG1	3	6	1	1	1	1
ENSG00000009830	20	POMT2	4	15	2	2	2	2
ENSG00000188690	20	UROS	1	2	1	1	1	1
ENSG00000100299	19	ARSA	2	12	2	1	2	2
ENSG00000117400	19	MPL	2	4	1	1	2	3
ENSG00000160200	18	CBS	3	11	2	3	1	2
ENSG00000116748	18	AMPD1	0	3	2	1	2	2
ENSG00000131844	18	MCCC2	3	7	1	3	1	1
ENSG00000160183	18	TMPRSS3	2	14	1	2	1	2
ENSG00000168621	18	GDNF	4	4	1	1	1	1
ENSG00000010704	18	HFE	0	6	4	3	2	3
ENSG00000102393	18	GLA	2	7	1	2	1	1
ENSG00000135925	18	WNT10A	1	1	1	1	0	1
ENSG00000112964	17	GHR	3	7	2	1	1	1
ENSG00000144452	17	ABCA12	23	21	3	3	2	4
Total Number of Variants for filter			7364	21022	3633	1431	2199	3103

« Prev | 1 | 2 | 3 | 4 | 5 | Next » of 95 pages

FIGURE 5.4 – Capture d'écran de la présentation des résultats détaillés de *QueryOR*

intéressantes et en afficher la liste plus détaillée (figure 5.4). Cette interface est dédiée à l'exploration des résultats et permet de visualiser les règles qui ont fonctionné sur chacun des variants, ainsi que d'ouvrir vers une interface d'exploration à l'échelle du gène. Cette dernière est très utile dans le cas d'analyse en trio, puisqu'elle permet de visualiser simplement la répartition des variations chez les parents et les enfants, comme présenté dans la figure 5.5. Dans cette figure, la variation 1 est hétérozygote chez l'enfant, héritée de la mère, alors que la seconde variation est homozygote, héritée des deux parents.

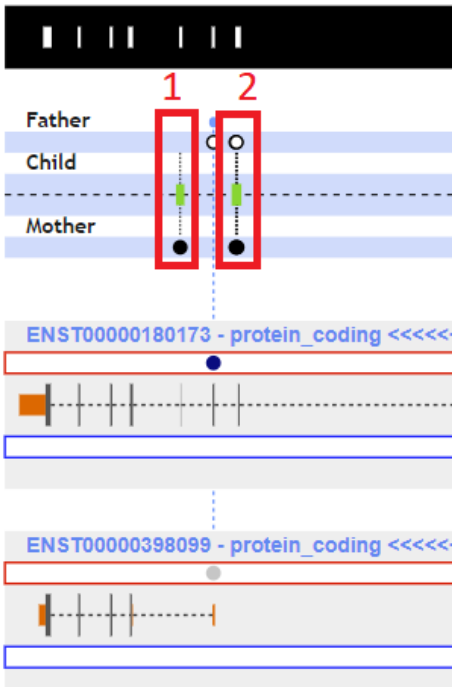


FIGURE 5.5 – Capture d'écran de la présentation des résultats par gène de *QueryOR* ; les cadres 1 et 2 correspondent aux variations ; les exons sont représentés par les bandes blanches au-dessus des variations et détaillés par transcrit en-dessous.

5.2.1.2 Points forts

QueryOR présente plusieurs points forts et innovations qui en font un très bon outil, adapté à sa problématique.

- Il est utilisable par tous et ne nécessite pas de posséder des notions avancées en informatique.
- Il ne requiert pas d'installation, il est indépendant du système d'exploitation ; il est donc multiplateformes.
- L'approche de *scoring* est justifiée et permet d'éviter les limites importantes apportées par l'utilisation de filtres.
- Les opérateurs sont adaptés au type de données du critère.
- Il est possible de choisir plusieurs valeurs pour élaborer un score comme nous l'avons vu dans la figure 5.2.

Après l'analyse des contraintes du domaine, son utilisation sans compétences particulières en informatique apparait comme un point primordial et il est très bien pris en compte dans *QueryOR*. Les différentes tâches sont accessibles à un utilisateur non formé en informatique, mais il requiert en revanche de posséder de très bonnes notions dans le domaine de la génétique des maladies rares. En effet, bien que des ensembles de requêtes prédéfinies permettent de cibler facilement les variants rares pathogènes par exemple, la personnalisation et l'adaptation des règles n'est pas destinée à un débutant en génétique.

Les étapes de requêtes s'effectuent avec différents opérateurs en fonction du type de données. Dans le cas de données numériques, il s'agit d'un système de seuil, où l'utilisateur entre une valeur limite ainsi qu'un opérateur (supérieur, inférieur, supérieur ou égal, inférieur ou égal, égal). Si les données sont qualitatives, *QueryOR* propose les différentes valeurs possibles et l'utilisateur en choisit une ou plusieurs de façon intuitive.

L'avantage de la structure sous forme de plateforme web est la facilité d'usage et de mise en place pour l'utilisateur. Le seul pré-requis est un navigateur internet, ce qui peut

être considéré acquis. De plus, l'utilisation d'un navigateur web permet à l'utilisateur de s'affranchir d'éventuelles mises à jour et de la maintenance.

L'approche en *scoring* plutôt qu'en filtration adoptée par *QueryOR* est très pertinente. Comme le soulignent les auteurs, cela permet d'éviter d'éliminer des variants s'ils sont en dessous d'un des multiples seuils de filtres définis par l'utilisateur. Cette approche offre de très bons résultats, en particulier sur les données à petite échelle, tel que les panels de gènes.

5.2.1.3 Points faibles

QueryOR présentent des limites, consenties ou non par les auteurs. Une première source de limites est l'utilisation d'un outil en tant que plateforme web. Cela pose plusieurs problèmes conséquents :

— Les temps de chargement des données volumineuses : avec une bonne connexion l'import d'un fichier VCF d'exome demande une dizaine de secondes, ce qui est acceptable. Cependant, le passage à l'échelle du génome est inenvisageable de cette façon, les chargements seraient de plusieurs Go dans ce cas.

— Le chargement de données sensibles : les données génétiques sont considérées comme des données personnelles très sensibles (Niemiec et Howard, 2016), il peut donc être légalement impossible pour certains laboratoires de charger des données génétiques sur un serveur qui n'est pas hébergé en France, comme le montre la *CNIL* (*Commission Nationale de l'Informatique et des Libertés*) dans un ouvrage dédié aux données génétiques (Commission nationale de l'informatique et des libertés, 2017).

— Les temps de traitement : l'utilisation d'un serveur externe permet de faire baisser le coût de l'analyse des données. En effet, il nécessite une infrastructure moindre puisque l'utilisateur n'emploie plus sa propre machine de calculs pour effectuer les étapes d'annotations. Cependant, cela signifie aussi qu'il faut accepter des temps de calcul bien supérieurs, car la charge de travail du serveur dépend du nombre d'utilisateurs connectés, ainsi, l'annotation d'un exome prend en moyenne une trentaine de minutes, ce qui représente environ le double de temps par rapport à une annotation sur un serveur local.

— La confiance dans le service : le fait que l'ensemble des processus s'effectuent en ligne demande de faire confiance aux auteurs et au service pour rester actif. Si le serveur cesse ou si l'équipe décide d'en arrêter la maintenance, il devient impossible d'utiliser le service. Ce n'est pas le cas avec des logiciels distribués qui peuvent toujours être utilisés localement, même si le logiciel n'est plus maintenu.

— L'absence de main-mise sur l'annotation, que ce soit de façon qualitative, puisqu'il n'est pas possible d'ajouter ses bases de données, mais aussi sur le processus en lui-même. Nos différents tests ont mis en évidence que des fichiers VCF pouvaient engendrer des erreurs d'annotation sans que la raison ne soit expliquée. L'erreur disparait généralement à la seconde analyse, mais certains fichiers ne sont pas acceptés par *QueryOR* sans aucune raison apparente. Nous avons également constaté la disparition de variations, présentes dans le VCF chargé mais pas dans le compte-rendu final sans que *QueryOR* n'indique d'éventuelles erreurs.

En plus de ces limites liées principalement à l'architecture, *QueryOR* présente d'autres faiblesses structurales. En effet, comme dit ci-dessus, les auteurs ont souhaité un outil flexible et l'ont conçu ainsi. Ils ont cependant développé un outil flexible du point de vue du développeur et non de l'utilisateur. Ainsi, il est facile pour le développeur d'ajouter des nouvelles composantes, que ce soit des nouvelles données ou des nouvelles fonctionnalités. Cette flexibilité est louable, mais elle n'est pas perceptible pour l'utilisateur, car une mise à jour de l'outil par les développeurs est requise pour prendre en compte de nouvelles données.

Dans le même registre, *QueryOR* présente actuellement 70 critères sur lesquels peuvent s'effectuer un *scoring*. Au moment de la publication de l'outil, il s'agit de l'annotation la plus

complète parmi les outils d'assistance à l'analyse, cependant, cela reste limité quantitativement et surtout qualitativement. En effet, quantitativement, s'il est très rare que 70 critères soient utilisés simultanément, le problème réside dans le champ de critères possibles, qui est imposé et qui ne permet pas de prendre en compte des critères provenant de bases de données spécifiques de pathologies ou de bases de données locales.

Cinq limites de conception sont présentes :

1 L'absence de bornes lors de la présentation des valeurs rend difficile la détermination de seuil lorsque le critère n'est pas parfaitement maitrisé.

2 L'absence de combinaison de critères empêche l'élaboration de règles plus complexes correspondant aux filtres combinés utilisés par les généticiens dans la méthode manuelle, c'est-à-dire en s'appuyant sur un tableur.

3 L'absence d'un opérateur *non*, ou "*ne contient pas*", pour des règles permettant par exemple d'augmenter les scores des variations avec un identifiant de type *rsID*, donc déjà connues dans la base de données *dbSNP*. Ce filtre est souvent utilisé par les généticiens, les variations qu'il permet d'extraire étant très rares.

4 L'impossibilité de traiter plusieurs fois un critère en y attribuant des scores différents pour des valeurs différentes, afin de pouvoir augmenter fortement une valeur "pathogène" tout en augmentant également mais de façon moins marquée une valeur "probablement pathogène" par exemple.

5 L'impossibilité de baisser un score, pour diminuer le rang de variants mal couverts ou notés "bénin" par exemple. La saisie de valeur négative est laissée possible, mais cause une erreur dans le traitement des données et n'est donc pas utilisable.

Ces cinq limites diminuent grandement l'utilisation et l'efficacité de ce logiciel. Concernant l'utilisation, la première limite peut entraîner la définition de seuils complètement inadaptés par méconnaissance des données. Par exemple, l'utilisateur a beau connaître le principe du score CADD, il ne connaît pas forcément l'échelle des valeurs et par conséquent, il peut être difficile de déterminer une valeur seuil pertinente. C'est le cas pour beaucoup de scores de prédiction de pathogénicité qui ont des valeurs s'échelonnant de 0 à 1, ou de 0 à 100, parfois de façon croissante, parfois de façon décroissante. De plus, les données n'étant pas présentées, l'utilisateur ne peut pas facilement vérifier l'échelle utilisée.

Concernant l'efficacité, les quatre limites suivantes s'additionnent pour créer un système de score manquant de nuances. Globalement, le système proposé par *QueryOR* ne fonctionne que dans un sens, il permet à l'utilisateur d'attribuer un haut score aux variants satisfaisant le plus de critères. Comme nous l'avons vu, le système ne permet pas de définir des règles négatives, que ce soit *via* un opérateur dédié ou un score négatif. L'utilisateur ne peut pas non plus combiner les critères ou utiliser plusieurs fois le même critère. Ces points sont pourtant des pratiques courantes dans la méthode d'analyse manuelle que nous avons observée.

Enfin, malgré une interface globalement bien pensée et bien construite, deux points sont limitants : l'absence de tests dynamiques et l'impossibilité de visualiser les variations complètement. Seules les modifications de poids de critères sont possibles à la page de résultats, et leurs modifications entrainent de toute façon un rechargement de la requête. La phase de tâtonnement nécessaire à la mise en place d'un jeu de règles et de sélections adaptés est donc complexe. De plus, il n'est pas possible de visualiser facilement les variants annotés pour en déduire des critères pertinents. La seule façon de voir l'ensemble des champs est de télécharger un export de données en cochant tous les champs. Cependant, même cette méthode reste insatisfaisante, car en plus d'être très fastidieuse, seuls les variants répondant à la requête sont présents, et ils sont dupliqués autant de fois que de transcrits, ce qui rend très difficilement lisible le tableau de résultats, comme on peut le constater dans la figure 5.6.

Ensembl ID	Gene Symbol	Transcript	Chrom	Position	Ref All	Alt All	Type	Ref Coverage	Alt Coverage	
ENSG00000137501	SYTL2	ENST00000316356	11	85445365	G	C	snp	71	82	153 rs74718633
ENSG00000137501	SYTL2	ENST00000389960	11	85445365	G	C	snp	71	82	153 rs74718633
ENSG00000137501	SYTL2	ENST00000438197	11	85445365	G	C	snp	71	82	153 rs74718633
ENSG00000137501	SYTL2	ENST00000524452	11	85445365	G	C	snp	71	82	153 rs74718633
ENSG00000137501	SYTL2	ENST00000527523	11	85445365	G	C	snp	71	82	153 rs74718633
ENSG00000137501	SYTL2	ENST00000528231	11	85445365	G	C	snp	71	82	153 rs74718633
ENSG00000138162	TACC2	ENST00000260733	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000334433	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000358010	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000360561	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000368997	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000368999	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000369000	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000369001	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000369004	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000369005	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000440764	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000453444	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000496913	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000505639	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000508411	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000513429	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000514539	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000515273	10	123976242	C	T	snp	92	70	162 rs34944231
ENSG00000138162	TACC2	ENST00000515603	10	123976242	C	T	snp	92	70	162 rs34944231

FIGURE 5.6 – Capture d'écran de la présentation des résultats téléchargés depuis *QueryOR*

5.2.1.4 Conclusion sur *QueryOR*

QueryOR est à ce jour le meilleur outil de la bibliographie pour aider à la filtration des variants. Il est innovant sur le système d'interface et simple d'utilisation. Cependant, sa conception n'est pas du tout optimisée pour une analyse de variant à l'échelle de l'exome, encore moins à l'échelle du génome. Ces difficultés sont liées à des contraintes structurelles (plateforme web), mais aussi à une flexibilité construite uniquement d'un point de vue développement ainsi qu'à un très net manque de nuance dans l'approche du score.

QueryOR est donc un bon outil, qui apporte un début de réponse sur la problématique de l'indépendance du généticien sur un plan bioinformatique. Cependant, les contraintes d'utilisation sont nombreuses et l'outil ne semble pas adapté à notre contexte de l'analyse d'exomes de patients atteints de maladies rares. Le développement est trop central pour répondre pleinement à la problématique et la mise à l'échelle est complexe au vu des résultats d'utilisation avec des exomes. De plus, la publication ne présente pas de cas illustrant son utilisation.

5.2.2 *Varaft*

Le deuxième outil que nous allons étudier est *Varaft*. Sur le même modèle que *QueryOR*, nous allons commencer par un exemple concret permettant de comprendre le fonctionnement de l'outil, puis nous détaillerons ses points forts et ses points faibles.

5.2.2.1 Présentation

Varaft est un logiciel développé en 2018, il est le plus récent des outils de ce type. Il est développé en Java par Jean-Pierre Desvignes et collaborateurs (DESVIGNES et al., 2018). Il utilise une approche de filtration sur 58 critères.

Le logiciel étant exécuté en local, l'utilisateur doit commencer par l'installer. Cette étape est difficile sur le système *Windows*, mais réalisable sans difficulté majeure sur une distribution Linux comme Ubuntu. L'utilisateur doit ensuite s'inscrire sur deux sites d'autres projets développés par les auteurs, *UMD-predictor* (SALGADO et al., 2016) et *Human Splicing Finder* (DESMET et al., 2009) et importer les paramètres de connexion dans *Varaft*. Il faut ensuite télécharger les bases de données de références pour l'annotation, ce qui représente une cinquantaine de Gigaoctets. Ce sont les mêmes bases de données que celles utilisées par *ANNOVAR*, puisque *VARAFT* utilise ce moteur d'annotation. D'autres bases de données annexes sont également à télécharger, notamment *dbSNP* qui représente à elle seule une dizaine de Gigaoctets supplémentaires.

L'utilisateur peut alors annoter ses premiers VCF. Il crée un projet, importe des VCF, sélectionne les annotations qu'il souhaite et exécute le processus. Ce processus dure quelques minutes, le temps qu'*ANNOVAR* annote le fichier. L'utilisateur enchaine ensuite la partie filtration, ce module présente les différents critères de filtres avec les différentes options dans un format plus ergonomique qu'un tableur, comme on peut le voir dans la figure 5.7. En dessous de ces filtres sont présentés les variations, comme montré dans la figure 5.8. La mise à jour s'effectue en temps réel, les variants affichés étant les variants conservés au fur et à mesure que s'appliquent les différents filtres. Une dernière étape de visualisation détaillée permet d'accéder à des ressources extérieures si elles sont disponibles, notamment *IGV* (*Integrative Genome Viewer*, un *genome browser* permettant de visualiser les fichiers *bam*).

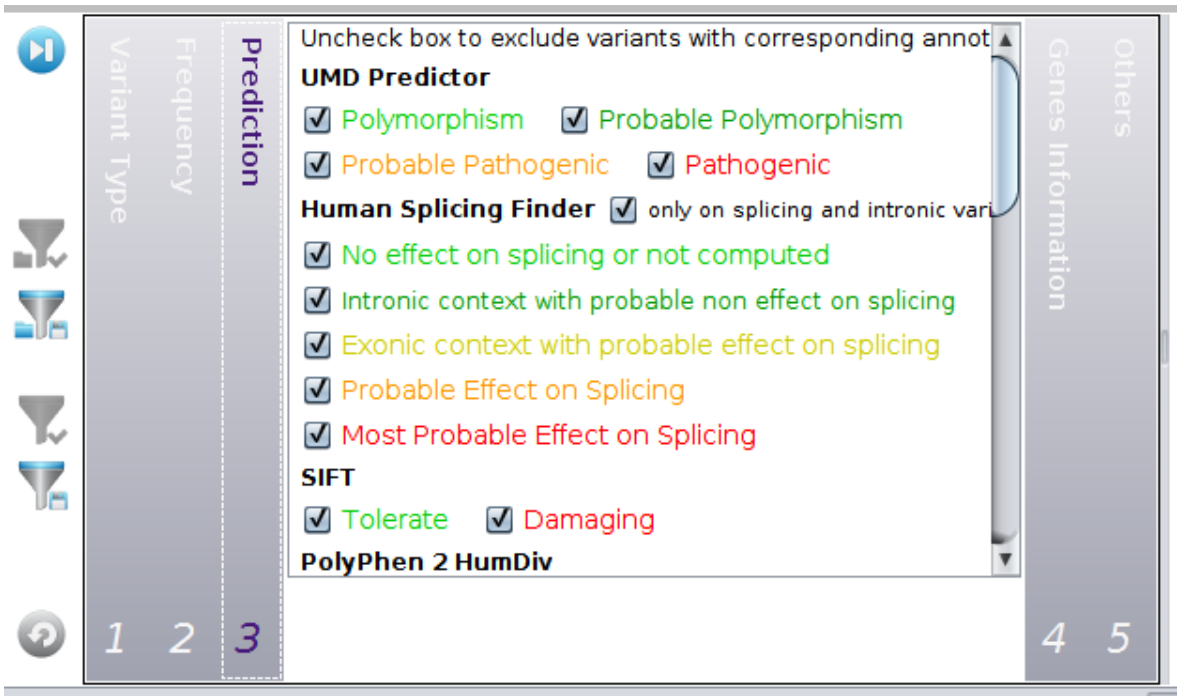


FIGURE 5.7 – Capture d'écran de la présentation des filtres dans *Varaft*

Chr	Start	End	Ref	Alt	Genotype	Depth	Frequency	SNV Score	Func.refgene	Gene.refgene	ExonicFunc.refgene	AAChange.refgene	1000g2015au
1	138 593	138 593	G	T	het	-1	0.5	100	ncRNA_exonic	LOC729737	.	.	0.0451278
1	949 608	949 608	G	A	het	-1	0.5	100	exonic	ISG15	nonsynonymous SNV	ISG15:NM_005101...	0.338858
1	1 026 801	1 026 801	T	A	het	-1	0.5	100	intronic	C1orf159	.	.	0.626597
1	1 139 216	1 139 216	G	A	het	-1	0.5	100	exonic	TNFRSF18	nonsynonymous SNV	TNFRSF18:NM_148...	0.00159744
1	1 167 796	1 167 796	C	T	het	-1	0.5	100	exonic	B3GALT6	synonymous SNV	B3GALT6:NM_080...	0.161542
1	1 238 492	1 238 492	G	A	het	-1	0.5	100	intronic	ACAP3	.	.	0.161142
1	1 243 896	1 243 896	C	T	het	-1	0.5	100	upstream	ACAP3;PUSL1	.	dist=64	0.310503
1	1 263 451	1 263 451	C	T	het	-1	0.5	100	UTR3	CPTP	.	NM_001029885:c....	0.0433307
1	1 270 054	1 270 054	G	A	het	-1	0.5	100	UTR3	TAS1R3	.	NM_152228:c.*21...	0.00139776

FIGURE 5.8 – Capture d'écran de la présentation des variants dans *Varaft*

5.2.2.2 Points forts

L'un des principaux points fort de *Varaft* est le fait qu'il soit distribué. Les auteurs ont choisi cette option pour s'affranchir de problèmes légaux au niveau de l'hébergement de données génétiques confidentielles. Ce constat est pertinent, il peut même être soutenu par plusieurs autres arguments comme notamment l'entretien d'un serveur, les temps de chargement et de traitement des données ou des problèmes de mises à jours. Ce constat légal a orienté les auteurs vers un logiciel développé en Java, donc théoriquement portable sur tous les systèmes d'exploitation, et s'exécutant uniquement sur la machine de l'utilisateur, ce qui parait une stratégie adaptée et permet notamment de toujours garantir l'accès au service. Certains cas de la bibliographie que nous aborderons dans les sections suivantes prouvent que

ce point est important.

Les autres points forts de *Varaft* sont liés à la très bonne connaissance du domaine par ses auteurs, qui ont inclus dans leur programme la gestion des variants structuraux (les CNV), ainsi que la gestion des analyses complexes, en particulier les structures d'analyses en trio et la recherche de variants liés aux maladies de transmission récessive. Ce sont des problématiques classiques de la recherche de maladies rares et leur présence contribue à spécialiser le logiciel.

5.2.2.3 Points faibles

L'installation de *Varaft* est relativement simple, bien que très longue, sur un système Linux. En revanche,nous n'avons pas réussi à finaliser cette installation malgré plusieurs tentatives sur 5 machines de configuration différentes utilisant une système d'exploitation Windows (XP, 7 familial, 7 professionnel, 10). Ces problèmes d'installation sont un frein majeur, car le public visé n'a pas le temps de consacrer plusieurs heures à l'installation d'un outil. De plus, la plupart des généticiens n'ont pas les compétences pour installer des logiciels autrement que *via* le classique installateur *Windows* qui est en général le seul système qu'ils utilisent. En plus de cette limite prépondérante, de nombreux autres problèmes d'approches sont présents :

— Téléchargement de bases de données volumineuses

— Approche de priorisation en filtration

— Approche centrale et réinstallation du logiciel à chaque mise à jour des outils

— Approche centrée maladies génétiques rares, mais l'annotation reste incomplète, puisque des bases importantes du domaine des maladies rares sont manquantes comme denovo-db (Turner et al., 2016), SysID ou intervar (Q. Li et K. Wang, 2017).

Le premier point est lié à l'installation et à la présence d'un annotateur dans *Varaft*. Cette possibilité d'annoter *via* une *GUI*(*Graphical User Interface*, une interface graphique) est un atout, mais qui a un coût important. Elle est fréquemment adoptée sur des outils en ligne, mais beaucoup plus rarement sur des outils distribués car il faut télécharger plusieurs centaines de Go de données pour obtenir l'ensemble des bases disponibles dans *Varaft*, la qualité de l'annotation restant de plus relativement réduite par rapport à une annotation faite par un bioinformaticien, puisque des bases importantes sont manquantes et qu'il est impossible d'adapter l'annotation à la problématique médicale précise. Si philosophiquement c'est une approche intéressante, dans les faits elle est assez mal mise en place et ne satisfait pas vraiment les besoins de l'utilisateur, en étant à la fois coûteuse en temps et en ressources, et peu efficace. Enfin, il est impossible de télécharger ces bases de données *via* une connexion internet utilisant un *proxy*, ce qui est souvent le cas pour les réseaux hospitaliers ou universitaires. La solution proposée par les auteurs pour contourner ce problème, c'est-à-dire désactiver temporairement le *proxy*, s'adapte mal au contexte hospitalier également.

L'approche de priorisation en filtration est ici étonnante, car la bibliographie a déjà permis de montrer la pertinence d'une approche en *scoring*, notamment avec *QueryOR*. De plus, les critères étant ici imposés par l'étape d'annotation, ils s'avèrent assez peu nombreux par rapport aux outils similaires.

L'approche *via* un outil non web est pertinente pour éviter les problèmes légaux liés à l'hébergement de données en ligne, mais sert aussi à gagner en indépendance pour l'utilisateur, qui n'est pas soumis aux évolutions obligatoires du service. Ce second avantage est ici réduit, car les ajouts de bases de données ne peuvent se faire que *via* une réinstallation d'une nouvelle version prenant en compte plus de données. Cela demande donc une mise à jour centrale de tous les installateurs et un nouveau téléchargement pour les utilisateurs.

Enfin, *Varaft* est clairement développé pour les maladies génétiques rares, ce qui est à la fois un atout et une faiblesse, car son champ d'action est de ce fait très restreint. Il prend en charge des formats d'analyses actuels, comme l'analyse en trio par exemple, ainsi que les différentes approches prenant en compte le mode de transmission supposé de la maladie

(dominant, récessif), mais interdit par la même occasion son utilisation pour d'autres contextes, génétiques ou non.

5.2.2.4 Conclusion sur *Varaft*

Varaft est l'outil le plus récent, mais certainement pas le plus abouti de la bibliographie. Si certains de ses aspects sont innovants et intéressants, notamment la prise en charge de l'analyse en trio ainsi que la mise en évidence des CNV, il souffre de nombreux problèmes de conception qui limitent très fortement sa portée et son utilisation routinière. Il pâtit également d'un manque de flexibilité lié à une spécialisation importante sur la génétique des maladies rares, mais aussi à un système verrouillé empêchant par exemple l'utilisation de certaines bases de données pourtant dédiées à ce même domaine de la génétique des maladies rares. L'approche est très linéaire, le logiciel cloisonne l'utilisateur et le force à utiliser un système prédéfini et imposé par les auteurs. Cette approche parait peu adaptée à un domaine en constante évolution où les pratiques et méthodes d'analyses varient de façon conséquente d'un utilisateur à l'autre.

Enfin plusieurs limites techniques et conceptuelles ternissent l'interface graphique plutôt intuitive et adaptée aux biologistes, notamment une mise en place très complexe, demandant beaucoup de temps, de ressources et surtout de compétences en informatique.

5.2.3 *VCF-Miner*

Le troisième outil bibliographique permettant l'aide à l'analyse de variations génétique est *VCF-Miner*. Nous allons commencer par décrire son fonctionnement, puis nous verrons ses points forts et ses points faibles.

5.2.3.1 Présentation

VCF-Miner est une plateforme web de priorisation de variants à partir de fichiers VCF. Il a été développé en 2014 par Steven Hart et publié en 2015 (S. N. Hart et al., 2015). Le code source est téléchargeable pour être utilisé dans un réseau local, ce qui permet d'éviter les principaux problèmes des plateformes web tout en profitant des avantages de développement. Les premières versions se mettaient en place avec un installateur *Windows*. Désormais, l'installation est beaucoup plus complexe, demandant notamment une bonne maitrise de *Github* et surtout de *Docker* ; l'installation est réalisée en ligne de commande *via* ces deux outils et n'est pas accessible sans compétences en informatique. Cet outil ne semble plus maintenu, les dernières mises à jour remontant à juin 2017.

5.2.3.2 Points forts

VCF-Miner fait parti des outils les plus vieux, pourtant il présente des atouts très intéressants. Le premier est le choix de l'approche, la plateforme web à installer localement qui permet un bon compromis, puisque l'outil ne dépend pas d'un serveur extérieur une fois qu'il est installé. Cela permet en outre un développement d'interface simplifié *via* les technologies du Web qui sont dédiées à la création d'interfaces graphiques. Ce choix permet également de supprimer le second inconvénient principal du service web, les contraintes de téléchargement.

Il est important de noter que l'annotation des variants n'est pas incluse, dans un souci de liberté pour l'utilisateur. C'est un choix très intéressant, qui est sensé dans son argumentation, puisque les auteurs justifient ce choix par leur volonté de proposer un outil utilisable dans des pratiques variées et permettant de prendre en charge des annotations personnalisées par les utilisateurs, donc adaptées à leur pratique spécifique. C'est une approche très logique et il est étrange de constater que des outils plus récents n'aient pas repris ce schéma argumenté et cohérent.

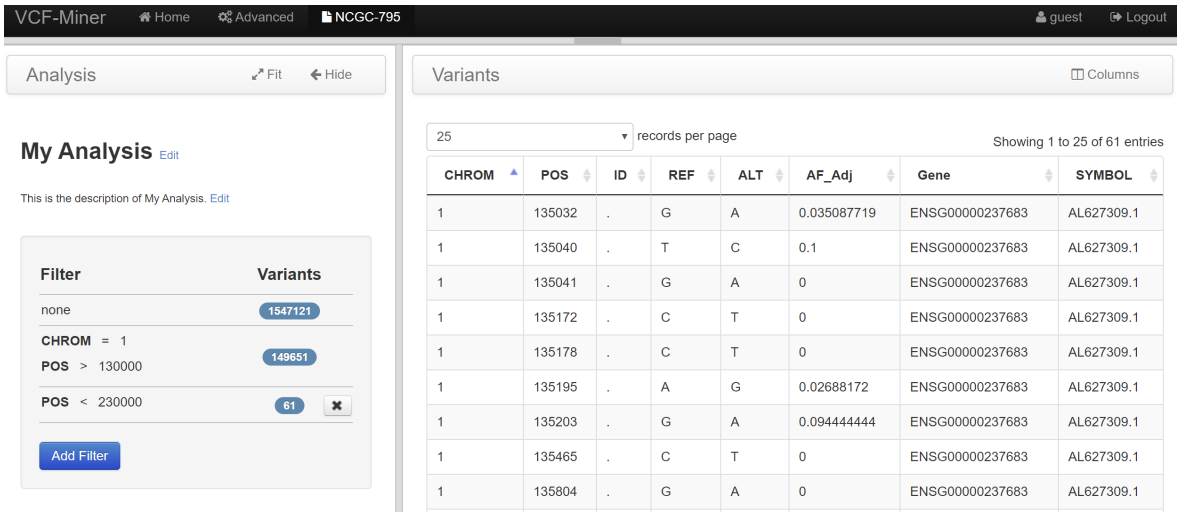


FIGURE 5.9 – Capture d'écran de la présentation de *VCF-miner*

Enfin, l'interface présenté dans la figure 5.9 est adaptée au travail des généticiens. Les données y sont constamment visibles, le récapitulatif des filtres actif est également présent en permanence.

5.2.3.3 Points faibles

Concernant les points faibles, l'un des plus important est la contrainte sur le format, avec l'utilisation d'un fichier VCF strict. Si l'annotation n'est pas contrainte à l'étape précédente, elle l'est très fortement ici, la plupart des annotateurs produisant des résultats qui ne sont pas sous un format VCF. Cette limite n'est pas incohérente dans le cadre de l'utilisation prévue du logiciel, plutôt développé au départ pour l'oncologie biologique, puisque l'annotation est un sujet moins critique dans cette discipline. En revanche, dans le cadre de maladies génétiques rares, où une annotation la plus exhaustive possible est absolument nécessaire, cette contrainte de format diminue drastiquement la liberté d'annotation.

Si son approche de plateforme web locale est intéressante, elle est cependant complexe à mettre en place, demandant une bonne maitrise de *Docker* et de *mongoDB*, ce qui demande donc l'assistance d'un informaticien ou d'un bio-informaticien.

Enfin, le dernier point faible est la priorisation *via* des filtres successifs. Ce point est ici aussi cohérent avec une utilisation en oncologie biologique, mais est vite limitant pour la recherche de variations rares.

5.2.3.4 Conclusion

VCF-miner est l'un des premiers outils à interface graphique à destination des généticiens visant à simplifier l'analyse des variations. Son développement axé sur l'oncologie biologique a induit un certain nombre de postulats contestables, conduisant au développement de fonctionnalités inadaptées à la recherche de variations rares. C'est le cas notamment des contraintes de format et de l'approche de tri en filtration. Cependant, le choix de l'architecture et la réflexion argumentée des auteurs sur le besoin d'une annotation libre sont très pertinents.

Cet outil est toujours une référence, il est systématiquement utilisé en comparaison lorsque de nouveaux outils sont publiés, parfois en écartant un petit peu trop rapidement ses deux points forts principaux que sont son architecture et sa philosophie de l'annotation.

5.2.4 BierApp

Comme *VCF-Miner*, *BierApp* est un outil d'aide à l'analyse parmi les plus anciens. Après une présentation de son fonctionnement, nous verrons plus en détail ses points forts et ses

points faibles.

5.2.4.1 Présentation

BierApp a été publié en 2014 par Aleman et collaborateur (Alemán et al., 2014). C'est donc un des premiers logiciels visant à aider les biologistes à filtrer les variants. Il prend la forme d'une plateforme web permettant le chargement de données VCF, leur annotation et leur filtre sur les critères préalablement annotés. L'annotation est réalisée en utilisant plusieurs ressources extérieures de références comme SIFT, *Polyphen*, *Ensembl* ou encore *ClinVar*. Il n'est malheureusement plus accessible à ce jour (mars 2019), les remarques à son sujet se baseront donc exclusivement sur la publication associée.

5.2.4.2 Points forts

BierApp est l'un des premiers logiciels à avoir proposé des éléments de réponses à notre problématique, l'assistance à l'analyse de variations rares. Ce côté précurseur est son principal point fort, car les autres logiciels ayant été développés depuis se sont beaucoup appuyés sur cette mise en place. Comparativement à l'ensemble de la bibliographie actuelle, il est résulte de fait peu de points forts. On peut néanmoins citer 2 aspects importants :

— L'infrastructure web choisie par les développeurs lui permet d'être simple d'utilisation. Comme nous l'avons vu avec d'autres exemples, l'un des principaux intérêts d'une structure web est la facilité de mise en place pour l'utilisateur. Globalement, l'interface semble bien construite, avec un circuit clair pour l'utilisateur. Les ressources extérieures utilisées sont des ressources de références indiscutables.

— Le second aspect est une prise en charge assez fine des notions de ségrégation familiale, avec des filtres dédiés à cet aspect. Ces fonctionnalités témoignent d'un grande spécialisation de l'équipe de développement et d'une grande interaction avec des généticiens et des biologistes.

5.2.4.3 Point faibles

BierApp est un outil en ligne, l'une des principales faiblesses de ce choix de structure est la possible disparition du service, de façon temporaire ou définitive. Cet outil en est un bon exemple, car ce logiciel n'est plus accessible. L'adresse étant encore active, l'inaccessibilité du service peut venir d'un problème de navigateur, la façon dont les technologie web sont prises en charge par les navigateurs ayant évolué. Les auteurs recommandaient dans la publication l'utilisation du navigateur web Chrome version 14. La version actuelle de Chrome étant la $72^{ème}$.

Un autre point faible est l'approche par filtration, inadaptée à une telle quantité de variants. Ce point faible très répandu est en plus corrélé ici à une très faible diversité des critères de filtres. Ces limites risquent d'entraîner un très grand nombre de variants candidats pour les généticiens, les obligeant ainsi à augmenter les seuils, et donc à potentiellement manquer des variations importantes.

Enfin les temps d'exécution semblaient assez importants et la quantité d'information ajoutée semblait limitée, mais ce sont probablement des points qui tiennent plus de l'âge du logiciel que d'une mauvaise structure. En date de la publication, les auteurs prévoyaient une migration de leur structure vers *MongoDB* pour améliorer les temps de retours des requêtes.

5.2.4.4 Conclusion

L'outil BierApp fut l'un des précurseurs des outils de filtrations de variants facilement accessibles aux utilisateurs finaux. Il a servi de référence et de point de comparaison pour les autres outils et est encore cité. Cependant il reste très incomplet pour répondre aux besoins

actuels, que ce soit à cause de limitations techniques mais aussi conceptuelles. Il n'est plus maintenu et n'est plus disponible.

5.2.5 GenDomus

GenDomus est le cinquième et dernier outil de la littérature permettant d'assister les généticiens dans leurs travaux d'analyses.

5.2.5.1 Présentation

GenDomus est un outil développé par Iniguez-Jarrin et publié en 2017 (Iñiguez-Jarrìn et al., 2017). La publication de présentation a été suivie d'une seconde (Iniguez Jarrin et al., 2017) dont l'objectif est de fournir des lignes de conduite à l'attention des développeurs d'applications visant à aider à l'analyse de données génomiques. Il est intéressant de noter que les 4 auteurs de ces 2 publications sont tous informaticiens et travaillent sur des problématiques d'Interaction Homme Machine, mais qu'aucun n'est bioinformaticien, ni qu'aucun biologiste ou généticien n'est cité. Les deux publications ont été publiées dans une conférence d'informatique (*ENASE : Evaluation of Novel Approaches to Software Engineering*). Ces précisions sont importantes pour contextualiser les remarques faites sur cet outil. Enfin, il est important de noter que l'outil n'est pas en ligne et qu'il ne l'a probablement jamais été, car aucune adresse URL n'est fournie. L'analyse est donc basée exclusivement sur les 2 publications associées à l'outil.

5.2.5.2 Points forts

De par ses auteurs et son contexte, *GenDomus* est probablement l'outil le plus réfléchi de la bibliographie en terme d'expérience utilisateur, les deux publications s'attardent particulièrement sur ces aspects là. La structure choisie est celle d'une plateforme web. Les auteurs justifient ce choix en arguant d'un fort besoin de travail collaboratif, ils ont donc à raison choisi une structure qui le permet. Ils ont également utilisé leurs compétences dans le développement d'outils, ainsi la plateforme se veut "*responsive*", c'est-à-dire adaptée à tous les supports (téléphone, tablette, ordinateur, télévision connectée).

Il est difficile de traiter la partie biologique du processus, c'est-à-dire les bases de données de références utilisées, le mécanisme de l'annotation ou encore la mise en place du protocole de curation car ils ne sont pas discutés dans ces publications. Comme la plateforme n'est pas non plus en ligne, les informations sur ces bases de données sont rares. Il est fait mention de *ClinVar*, *dbSNP* et HPO, ce qui ne représente pas une grosse quantité d'informations. On peut cependant noter que les exemples pris sur les captures d'écran sont issus de thématiques oncologiques, or, nous avons déjà mentionné que le besoin d'annotation est plus faible dans ce domaine, les objectifs et les nombres de gènes analysés n'étant pas les mêmes.

Enfin, les recommandations définies dans la seconde publication sont plutôt pertinentes, mais cela ne concerne pas vraiment le logiciel en lui-même. Ces recommandations seront discutées dans la section éponyme (paragraphe 5.2.6).

5.2.5.3 Points faibles

Les points faibles de *GenDomus* découlent de la méconnaissance évidente du domaine de la génétique par ses auteurs. La pratique n'a pas été bien analysée, ce qui a engendré des postulats complètement inadaptés au réel travail de filtration et de curation des généticiens. L'un des exemples les plus frappants est le scénario d'utilisation présenté : trois généticiens travaillent dans un espace collaboratif équipé de 3 tableaux connectés, de tablettes et d'ordinateurs. À notre connaissance, aucun service de génétique en France ne dispose de ce type de matériel, et s'il est très fréquent que plusieurs généticiens se penchent sur le même dossier, il est préférable

de le faire de façon séparée puis de regrouper les données, afin de minimiser le risque de manquer une variation pertinente.

Le manque d'informations sur la partie *back-end*, la partie moteur du logiciel, est ici un point faible important. Le mécanisme d'annotation n'est pas décrit. L'absence de citation d'un outil extérieur suggère qu'un *pipeline* a été mis en place à façon par les développeurs, mais cette information n'est pas explicitement précisée. Les ressources citées comme sources de savoir sont également très limitées en nombre. Enfin, la partie filtration est inexistante. Les seuls filtres dont il est question sont les filtres de qualités du VCF, qui sont simplement basés sur le score *Phred* de la variation.

En plus de ne pas donner d'informations sur les méthodes utilisées pour le traitement des résultats biologiques, les publications ne mentionnent ni exemple ni portée d'utilisation. Il semble qu'il soit plutôt conçu pour des échantillons de gènes très réduits et très bien connus, mais cela n'est pas précisé explicitement.

5.2.5.4 Conclusion

GenDomus est un outil très peu cité en général, il est ne l'est pas du tout par les communautés bioinformatiques et génétiques. Son manque de lien avec le domaine, de sa conception à sa publication lui porte gravement préjudice, car il n'a pas eu l'exposition et les retours adaptés. Il n'est plus en ligne et si sa carrière a probablement été très brève, il n'en reste pas moins un outil intéressant sur sa conception, que ce soit par les leçons tirées des erreurs de ce logiciel, mais aussi et surtout pour son orientation de développement dédié à l'utilisateur final ainsi que pour ses mécanismes d'interactions avec l'utilisateur. Ainsi, les lignes de conduite proposées par les auteurs sont malgré tout intéressantes.

5.2.6 Recommandations

Les auteurs de *GenDomus* se sont appuyés sur leur expérience pour publier une liste de 10 recommandations pour créer des outils bioinformatiques efficaces à l'attention d'un public d'utilisateurs non informaticiens. Le manque d'expérience bioinformatique et les erreurs de *GenDomus* incitent à garder une pensée très critique vis à vis de ces recommandations, mais leur solide expérience en création logicielle est un argument en faveur de leur légitimité.

Recommandation 1 Tableaux de données interactifs. Dans cette recommandation, les auteurs suggèrent de fournir des moyens interactifs qui permettent de présenter les données et de les filtrer simultanément. C'est une réflexion d'autant plus intéressante qu'ils la comparent à des outils en ligne de commande. En effet les outils en ligne de commande ne sont pas adaptés pour un public non informaticien. Globalement, l'approche en visualisation est très pertinente, mais la règle se réduit à une approche en filtration. Elle est cependant très adaptée avec cette approche, puisque le fait de définir des filtres sans visualiser les données est une des limites importantes de beaucoup d'outils de la littérature, notamment *Varaft* et *VCF-Miner*. On peut extrapoler cette règle et souligner l'importance de pouvoir visionner les données lorsque l'on agit dessus.

Recommandation 2 Visualisation Parallèle. Cette ligne recommande la possibilité de visualiser en parallèle les données à comparer. Elle tombe sous le sens, l'effort mental étant bien moindre lorsque deux objets à comparer sont sous nos yeux plutôt que lorsque l'on doit mémoriser les caractéristiques des deux objets pour les comparer l'un à l'autre. Cette règle se définit plutôt sur des outils de comparaison de cascades de réactions biologiques, mais peut être extrapolée. Globalement, moins l'utilisateur doit mémoriser de choses, meilleure est l'interface. Il est à noter que ces deux premiers points sont totalement en phase avec les approches de programmations sur et avec exemple, qui constituent la base de l'approche *End-User Programming*.

Recommandation 3 Opération sur les échantillons. Les auteurs recommandent de prévoir des opérations sur les données permettant d'extraire une partie des résultats, pour

par exemple les fusionner à d'autres échantillons afin d'obtenir de nouveaux jeux de données. C'est une opération qui peut être pertinente dans le cas d'analyse familiale par exemple, mais qui peut aussi amener à des risques importants en identito-vigilance. Ces opérations sont très souvent effectuées en amont par les généticiens, sans fusionner les données mais plutôt en annotant les variants présents chez les parents par exemple. De cette façon les risques de mélanges d'échantillons sont quasi nuls. Cette étape est réalisée au niveau de l'annotation, donc dans le cas d'une annotation par l'utilisateur, elle est effectivement à prévoir, mais doit être bien plus encadrée que ne le suggèrent les auteurs, pour limiter les risques de manipulations erronées.

Recommandation 4 Recherche dans la littérature. Les auteurs suggèrent une amélioration du processus de recherche dans la littérature scientifique en se basant sur un historique de l'utilisateur. La définition de cette ligne de conduite est plutôt floue et n'est pas non plus mise en place dans *GenDomus*. Elle ne correspond pas vraiment à la réalité, car les auteurs partent du principe que les recherches de publications des généticiens ne retournent pas les résultats attendus. Dans les faits, les généticiens ont une excellente maitrise des outils tels que pubmed [1], les mots clés utilisés sont très souvent les gènes, donc il y a très peu de risque de manquer un résultat bibliographique intéressant.

Recommandation 5 Documentation des résultats. Les auteurs recommandent de permettre à l'utilisateur d'enregistrer ses résultats lors de ses recherches bibliographiques et de l'analyse de données génétiques. Cette recommandation est également floue, les auteurs suggèrent de garder l'origine du savoir, à la manière d'un gestionnaire de référence tels Mendeley (X. Li et Thelwall, 2012) ou Zotero (Ahmed et Al Dhubaib, 2011). L'idée est apparemment de se souvenir du raisonnement si la même variation était rencontrée dans un autre échantillon. C'est une réflexion pertinente, mais qui existe déjà dans tous les laboratoires, puisque c'est exactement l'objectif des bases de données locales, qui associent à chaque variant déjà mis en évidence la pathogénicité qui lui a été attribuée, afin qu'en cas de nouvelle rencontre, l'analyse soit beaucoup plus rapide.

Recommandation 6 Assister l'utilisateur dans les interfaces. L'idée est de guider l'utilisateur dans l'exploration de ses données *via* une interface en enregistrant les actions précédentes de l'utilisateur mais aussi d'autres utilisateurs experts. Cette recommandation montre le manque de connaissances du domaine des auteurs, puisque les généticiens ne naviguent jamais parmi l'ensemble de leurs données, qui sont bien évidemment trop conséquentes pour une telle approche. L'idée d'enregistrer et de partager les démarches est en revanche logique et pertinente, elle est déjà mise en place par la plupart des outils, au moins pour l'enregistrement des filtres.

Recommandation 7 Gestion de l'échelle. Le système doit supporter une augmentation de la taille des données. Dans cette recommandation, les utilisateurs entendent surtout l'augmentation du nombre d'utilisateurs, qui n'est un problème que si l'application est basée sur l'utilisation d'un serveur central. Cela tombe sous le sens que l'infrastructure doit être adaptée régulièrement au nombre d'utilisateurs connectés. Dans le domaine de la génétique, il est plus pertinent de se pencher sur la gestion de différentes tailles d'analyses, la plus commune étant l'exome, il est donc important de pouvoir gérer cette échelle de façon rapide, mais il faut aussi anticiper des échelles supérieures, comme par exemple les analyses de génomes complets.

Recommandation 8 Disponibilité. Le système doit être disponible. C'est une recommandation basée sur une architecture plateforme web, elle est plutôt évidente, d'autant plus avec les exemples de *BierApp*, *VCF-Miner*, ou encore *GenDomus*, puisqu'il semble que les auteurs n'aient pas suivi leurs propres recommandations.

Recommandation 9 Traitement transparent. Les étapes de chargements ou d'actions doivent être transparentes pour les utilisateurs. Cette ligne est étrange, car si l'objectif est louable

1. `https://www.ncbi.nlm.nih.gov/pubmed/`

et logique, elle est complètement illusoire. Aucun outil ne peut effectuer des opérations complexes sur des données aussi volumineuses de façon transparente pour l'utilisateur. On peut cependant extrapoler et supposer que l'objectif est de permettre à l'utilisateur d'avoir instantanément au moins un aperçu de ses actions sur les données. Avec cette extrapolation, cette ligne devient à la fois pertinente et réalisable.

Recommandation 10 Limiter les temps de réponses des requêtes. Pour cette dernière règle, le contexte est également une plateforme web. Il parait en effet important de réduire ce temps de requête. On peut faire une analogie avec le temps de calcul lorsque le contexte est une application locale. La solution suggérée par les auteurs semble adaptée, avec l'utilisation de retour partiel, calculé sur un petit échantillon en attendant que l'ensemble du calcul soit terminé. En ce sens, cette ligne rejoint la précédente.

Globalement, ces règles sont donc relativement pertinentes, elles n'apportent pas beaucoup de nouveautés du fait d'un certain manque de connaissances sur les pratiques du domaine, mais permettent de poser un cadre. Une application qui ne respecte aucune de ces règles sera très clairement inadaptée à une utilisation par des généticiens.

5.3 Synthèse de la bibliographie

Ces cinq outils nous ont permis d'appréhender différents aspects importants de la création d'une méthode d'analyse. Nous allons commencer par nous pencher sur la conception, permettant de déterminer les interactions avec l'utilisateur, ainsi que les règles génériques de mise en place. Nous verrons ensuite trois axes importants de ces outils, le type de déploiement, la gestion de l'annotation et la méthode de priorisation choisie.

5.3.1 Règles de conception

Comme nous l'avons vu, les règles définies par *GenDomus* sont plutôt pertinentes, mais peu spécifiques au domaine. Afin de clarifier le propos et de supprimer les ambiguïtés, nous proposons un nouvel ensemble de règles, adaptées à toutes les architectures et plus cohérentes avec le domaine de la bioinformatique et de la génétique :

règle 1 Les données doivent être visibles lorsque la tâche y fait référence.

règle 2 L'utilisateur doit avoir un aperçu du résultat de ses actions en temps réel.

règle 3 En cas de comparaison, les données à comparer doivent être présentées côte à côte.

règle 4 Les démarches peuvent être enregistrées.

règle 5 Les démarches peuvent être ré-exécutées rapidement.

règle 6 Les démarches peuvent être partagées.

règle 7 Le système doit rester utilisable à l'échelle du génome.

règle 8 Le système doit fonctionner sur le plus de configurations possibles.

règle 9 Le système doit être facile à mettre en place et à tester, même pour des utilisateurs qui ne sont pas informaticiens.

règle 10 Le système doit rester accessible et doit pouvoir être maintenu.

Les règles 1, 2 et 3 permettent d'évaluer rapidement l'interface utilisateur, elles se basent sur les règles de *GenDomus* élargies pour correspondre à des situations plus variées. Les règles 4, 5 et 6 se concentrent sur l'aspect automatisation, ce sont des points évidents pour un logiciel permettant de faire de l'automatisation, mais ces points restent importants à contrôler étant donné leur importance sur l'utilisation de ces systèmes. Enfin, les 4 dernières règles permettent d'évaluer la robustesse du système et sa capacité d'adaptation. Ici aussi, les règles découlent des règles de *GenDomus* élargies pour être adaptées au plus grand nombre de situations

Règle	*VARAFT*	*GenDomus*	*BierApp*	*VCF-Miner*	*QueryOR*
R1	✓	✗	✓	✗	✗
R2	✓	✓	✓	✗	✗
R3	✓	✓	✓	✓	✓
R4	✓	✓	✓	✓	✓
R5	✓	✓	✓	✓	✓
R6	✓	✓	✓	✗	✗
R7	✗	✗	✗	✗	✗
R8	✗	✗	✗	✓	✓
R9	✗	✓	✓	✗	✓
R10	✓	✗	✗	✗	✓

TABLEAU 5.1 – Récapitulatif des principaux outils de la littérature et leur adéquation aux règles de conception

possibles. En nous basant sur ce jeu de règles, nous pouvons évaluer plus objectivement les différents outils de la littérature, comme présenté dans le tableau 5.1.

Cependant, ces règles ne permettent d'évaluer ni l'efficacité ni l'efficience des outils, mais permettent simplement de déterminer à quel point l'outil est adapté au domaine et de comparer les points forts et faibles des différents outils plus facilement. On peut remarquer dans le tableau 5.1 que certaines règles sont respectées par tous les outils, c'est le cas des règles 3, 4 et 5. La règle 3 est un petit peu particulière, car les comparaisons ne sont pas toujours identiques, mais globalement, lors de comparaisons, les données sont présentées groupées. Pour les règles 4 et 5, ce sont des fonctionnalités de base de logiciels d'automatisations, il aurait été très étonnant de voir des systèmes ne pas valider ces règles. On peut remarquer que la règle numéro 6, qui concerne le partage de données sauvegardées, n'est validé que par 3 logiciels, qui ne sont par ailleurs pas les plus performants en terme de priorisation de variations. Cet exemple permet de rappeler que le respect de ces règles ne permet pas d'affirmer que le logiciel soit efficace, ni que l'absence de respect de certaines règles soit synonyme de performances diminuées.

5.3.2 Déploiement web

Le premier axe important différenciant les outils de la littérature est le choix de l'architecture. Dans leur mise en place, ces différents outils ont choisi deux options. *Varaft* se présente comme un logiciel distribué, alors que *QueryOR*, *VCF-miner*, *GenDomus* et *BierApp* ont choisi la forme d'un service web. Nous avons pu constater les problèmes liés à ces approches en testant les différents logiciels.

La sélection de la structure impacte plusieurs points :

— la phase d'installation

— la portabilité

— la dépendance à la qualité des réseaux

— la confidentialité des données

— la pérennité du service

Concernant la phase d'installation, elle est évidemment beaucoup plus simple dans le cas d'une plateforme web, car il suffit à l'utilisateur de se connecter au site. Avec une structure de logiciel distribué, il faut commencer par une phase d'installation du logiciel et de ses dépendances. L'effet sur la portabilité est alors évident, car cela sous-entend pour le développeur de prendre en compte une grande diversité de systèmes et de configurations, pour le logiciel et ses dépendances. Le problème est bien moindre avec la plateforme web, où il faut éventuellement prévoir des problèmes mineurs de compatibilité de navigateurs.

Concernant le rapport au réseau et à Internet, les deux approches ont leurs avantages et leurs inconvénients. L'approche web demande de charger de grandes quantités de données (tous les fichiers à étudier) ce qui peut être long et fastidieux pour certaines connexions internet, car cela dépend de la bande passante du réseau. De son côté, l'approche logiciel distribué demande le téléchargement de grandes quantités de données, car il est nécessaire d'installer les bases de données de références. Concernant la confidentialité des données, le problème est légal. Les données de génétiques sont des données sensibles et leur envoi sur des serveurs peut poser des problèmes en fonction des lois en place et des pays. En France, il faut que le serveur d'accueil soit agréé hébergeur de données de santé pour permettre aux Centre Hospitalo-Universitaire (CHU) d'y charger des données non anonymisées. L'approche locale n'est pas concernée par ces problèmes. Enfin, concernant la pérennité du service, l'approche locale parait beaucoup plus stable, car une fois le logiciel installé, les risques de ne plus pouvoir l'utiliser sont bien mieux contrôlés que dans le cas d'une plateforme web, soumise à beaucoup plus de contraintes extérieures.

Les créateurs de *Varaft* justifient leur choix d'architecture par le seul argument légal. En effet, en France notamment, l'envoi de données génétiques, donc sensibles, sur des serveurs peut poser des problèmes de confidentialité. Les arguments décrits par les créateurs des logiciels ayant choisi une approche web tournent autour de la facilité de développement et de mise en place, ce choix leur permet de développer rapidement une interface utilisable par des novices en informatique. Ces arguments sont transparents pour l'utilisateur, car les étapes de développement ne le concernent pas. Les créateurs de *BierApp* soulignent simplement l'inadéquation d'un logiciel en ligne de commande, mais ne semblent pas envisager la possibilité d'un logiciel distribué avec une interface graphique.

Cependant, l'implémentation de *Varaft* démontre bien les inconvénients de l'approche en logiciel "distribué" par ses difficultés d'installation et son système de téléchargement de bases qui est relativement lourd et peu aisé. En revanche, *VCF-miner*, *BierApp* et *GenDomus* montrent eux les inconvénients des services web, car ils ne sont plus accessibles aujourd'hui.

L'argument légal est clairement important dans notre cas. Le chargement de données génétiques sur des serveur distants est déjà très encadré et même s'il est possible que ces lois évoluent, le constat des développeurs de *Varaft* est pertinent.

5.3.3 Inclusion de l'annotation

Le deuxième axe divisant les approches bibliographiques est l'inclusion ou non de l'annotation dans le système. Deux approches sont possibles : compter sur l'utilisateur pour qu'il effectue lui même l'annotation ou l'inclure dans le processus de tri des variants. Le choix de l'approche est important, car il indique le placement du logiciel dans le *pipeline*. De la même façon que pour le choix de la structure, chacune des options possède des avantages et des inconvénients. Plusieurs points sont concernés par ce choix :

— la liberté de l'utilisateur

— la simplicité d'utilisation

— l'impact sur la structure

L'annotation incluse dans le processus réduit drastiquement la liberté de l'utilisateur, car les seules sources de données disponibles sont celles comprises dans l'annotateur utilisé. Ce sont souvent des sources de références, mais cela empêche l'utilisateur d'utiliser ses sources de données personnelles. Concernant l'utilisation, c'est en revanche plus simple pour l'utilisateur si l'annotation est dans le processus, car il n'a plus à s'en occuper. Il gagne également en indépendance vis à vis du bioinformaticien sur cette étape. Ne pas inclure l'annotation oblige l'utilisateur à annoter lui même, ou à travailler avec un bioinformaticien sur cette étape. Enfin, le choix de l'approche a un gros impact sur la structure précédemment discutée, web ou locale. En effet, l'annotation est une étape demandant une quantité importante de données références

pour en garantir la qualité. Exclure l'annotation du processus permet de s'affranchir des étapes de téléchargements et de simplifier considérablement l'annotation.

Varaft, *GenDomus*, *QueryOR* et *BierApp* ont fait le choix d'inclure l'annotation. Ce choix peut paraitre logique pour les plateformes web, mais semble beaucoup moins cohérent pour *Varaft*, étant donné les conséquences en terme de téléchargement. Les auteurs de ces publications ne justifient pas leur choix d'inclure l'annotation, ce qui est étrange étant donné que tous citent *VCF-miner*, qui en plus d'avoir choisi l'approche opposée, soulignait déjà cette limite majeure :"*Ils contraignent l'utilisateur à utiliser les annotations qu'ils fournissent. [...] Cela présente une limite importante, en particulier lorsque l'on considère l'intérêt croissant pour l'incorporation d'annotations locales ou collectées par l'utilisateur dans le processus de filtration*".

Concernant les annotateurs utilisés, *Varaft* et *QueryOR* utilisent *ANNOVAR* augmenté de quelques scripts, *BierApp* utilise des scripts spécifiques ; les auteurs de *GenDomus* n'ont pas précisé comment ils accomplissaient la tâche d'annotation. Que ce soit *QueryOR*, *BierApp* ou *Varaft*, les auteurs justifient à chaque fois le nombre de critères disponibles dans leur annotation pour indiquer la qualité de l'annotation. Ces annotations, même si elles sont relativement adaptées, *QueryOR* présente 70 critères par exemple, ne remplacent pas une annotation complète dédiée par un bioinformaticien. À titre de comparaison, l'annotation mise en place pour l'analyse des données de cette étude comporte 123 critères provenant de 3 annotateurs différents ainsi que de plusieurs sources de données indépendantes, spécifiques ou locales. Les annotations sont de plus en constante évolution, les premières versions de ce même *pipeline* d'annotation présentaient environ 65 critères.

5.3.4 Méthode de priorisation

Enfin, le troisième axe définissant les outils bibliographique est la méthode de priorisation. La plupart des outils de la littérature, à l'exception de *QueryOR*, se basent sur une approche en filtration. C'est une approche qui est une suite logique de ce que réalisent les généticiens avec les tableurs, qui consiste en une succession de filtres masquant tous les variants dépassant les seuils définis par le généticien. Ce processus permet de réduire assez rapidement la longue liste de variations en une liste beaucoup plus facile à appréhender pour l'utilisateur. La taille de la liste finale dépend beaucoup de la qualité de l'annotation, mais varie aussi en fonction des habitudes du biologiste. Le plus souvent, les généticiens appliquent une succession de filtres de plus en plus fins. L'idée est de commencer par réduire drastiquement la liste de variations possibles, en appliquant des seuils très stricts. Les variations de cette première étape sont les variations les plus probablement pathogènes. Si cette étape ne donne aucun résultat positif, les généticiens recommencent le processus en étant un petit peu moins strict sur les seuils, pour capturer des variations dont la signification clinique est plus difficile à évaluer. Ce relâchement des filtres à pour effet d'augmenter le nombre de variations présentes dans la liste finale, ce qui nuit à la lisibilité. Il devient très rapidement impossible d'analyser toutes les variations de la liste.

Comme on peut le constater dans la figure 5.10, les concepts sur lesquels s'appliquent les filtres sont finalement assez peu nombreux, mais chacun peut provenir de plusieurs bases de données présentant chacune de petites différences aussi bien sur le fond que sur la forme. Les premiers tris sont réalisés sur les fréquences de l'allèle dans la population générale. Dans le cas de l'étude des maladies génétiques rares, il faut garder à l'esprit que les variations que l'on recherche sont également très rares, il est donc très improbable de les retrouver dans des bases de données de populations supposées "saines", c'est-à-dire une population contrôle, en bonne santé. On recherche donc les variations les plus rares possibles, ou absentes de ces bases de données. Des références de ce type de base de données sont ExAC (Exome Aggregation Consortium)(Karczewski, Weisburd et al., 2016), 1000genomes (Thousand Genomes Project Consortium et al., 2015), GnomAD (Karczewski, Francioli et al., 2019) ou encore Ensembl (Zerbino et al., 2017). Ces bases sont souvent utilisées combinées, mais les

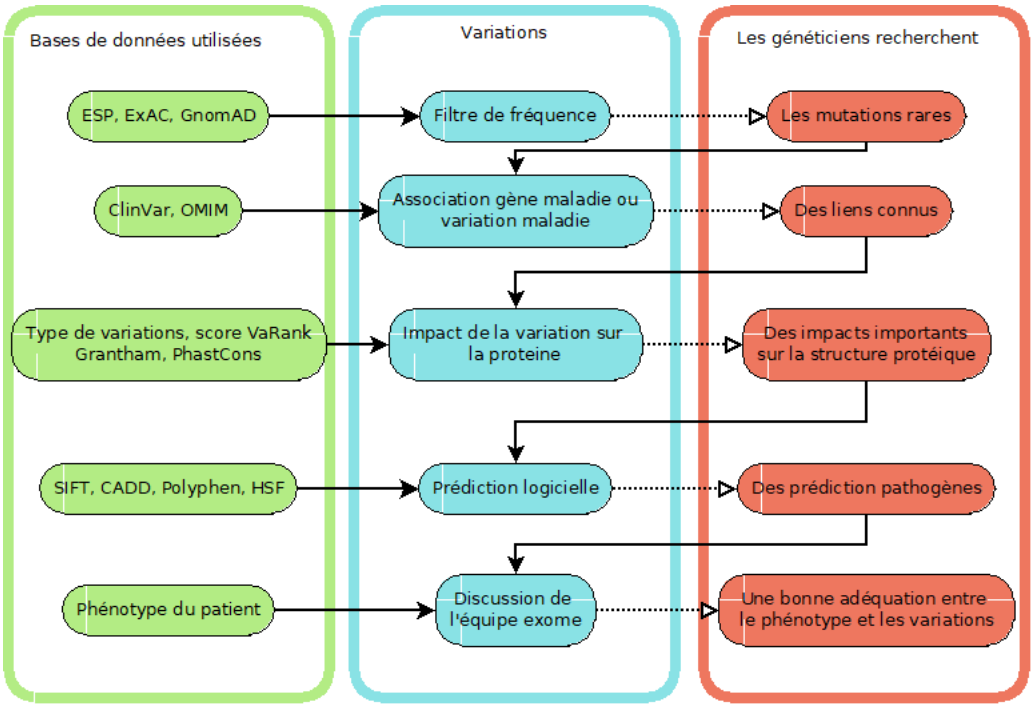


FIGURE 5.10 – Récapitulatif du processus de filtration

généticiens n'attribuent pas forcément de la valeur à l'information en fonction de chacune des bases, certaines pouvant parfois être polluées par des données d'individus malades, augmentant le risque d'y trouver des allèles pathogènes risquant de biaiser les résultats. C'est également à ce stade que peuvent être utilisées des bases de données de fréquence locale, basées sur les patients dont l'ADN a déjà été analysé par le laboratoire. Ces bases sont très utiles, car même si elles contiennent souvent une quantité de données moindre, la proximité géographique permet un plus fort pouvoir discriminant. La contrepartie est le risque de rencontrer plusieurs fois le même allèle pathogène, risque qui doit être pris en compte, par exemple avec la mise en place d'une classification des variations pathogènes rencontrées.

Le second filtre est la recherche de lien entre le gène affecté par la variation et la maladie présentée par le patient. Les principales bases de données de références à cette étape sont OMIM (HAMOSH et al., 2005) et HPO (KÖHLER, DOELKEN et al., 2013) mais aussi toutes les bases de données spécifiques de la maladie recherchée. En effet, ces bases offre souvent des informations plus fines et plus à jour que les bases génériques. Dans le cadre de la DI, des références utiles sont *PanelApp* (ENGLAND, 2018) et SysID (KOCHINKE et al., 2016b) qui sont mises à jour de façon très régulière. Les gènes connus pour provoquer des pathologies semblables au phénotype du patient sont plus intéressants et sont donc plus finement analysés.

L'étape de filtre suivante est l'évaluation du potentiel délétère de la variation : plus la variation impacte la structure tridimensionnelle de la protéine, plus le potentiel pathogène de la variation est élevé. Pour cela, il faut analyser le type de variation, les codons stop ainsi que les décalages du cadre de lecture ayant assez souvent les plus gros potentiels. Pour les variations faux-sens, beaucoup plus fréquentes, plusieurs outils aident à évaluer l'impact du changement de l'acide aminé de référence par un autre. On peut citer les bases de données de conservation, qui évaluent ce changement en se basant sur le degré de conservation du nucléotide ou de l'acide aminé dans les gènes orthologues. Les ressources les plus utilisées sont *PhastCons* (SIEPEL et al., 2005) et *PhyloP* (POLLARD et al., 2010). On peut aussi évaluer la différence entre l'acide aminé normal et celui causé par la variation en utilisant la matrice *Grantham* (GRANTHAM, 1974), qui bien que très ancienne, sert toujours de référence aujourd'hui. Le principe est d'associer à chaque changement un score qui reflète l'impact du changement d'acide aminé.

Enfin, de très nombreux outils de prédiction *in silico* sont utilisés, les plus classiques étant SIFT (Ng et Henikoff, 2003), CADD (Kircher et al., 2014), DANN (Quang, Chen et Xie, 2014), *Polyphen-2* (I. Adzhubei, Jordan et Sunyaev, 2013) ou encore *Mutation taster* (Schwarz et al., 2014). Ces outils ainsi que l'ensemble du processus est, pour rappel, détaillé dans la section 3.4.2

5.3.5 Conclusion sur la bibliographie

Comme on peut le voir dans le tableau récapitulatif des approches présenté dans le tableau 5.2, il y a assez peu de diversité dans les approches choisies par les différents outils de la bibliographie. Il est également très intéressant de constater que les approches les plus courantes ne sont pas celles qui présentent les meilleurs arguments. Le choix de la structure web est compréhensible dans certains cas comme nous l'avons dit dans la section 5.3.2 ; celui d'inclure l'annotation en ne se basant que sur *ANNOVAR* est déjà plus discutable, mais surtout, l'approche filtre par rapport à une approche en *scoring* représente clairement une limite pour l'utilisateur.

Outil	Structure	Annotation	Ranking
VARAFT	Local	Incluse (ANNOVAR)	Filtration
GenDomus	web	Incluse (inconnue)	Filtration
BierApp	web	Incluse (Scripts)	Filtration
VCF-Miner	web	Non incluse	Filtration
QueryOR	web	Incluse (ANNOVAR)	Score

Tableau 5.2 – Tableau récapitulatif des approches utilisées par les différents outils

En conclusion, les outils actuels présentent tous au moins une limite majeure empêchant ou restreignant leur utilisation, leur efficacité ainsi que leur portée. L'outil le plus avancé est clairement *QueryOR*, de par son approche en *scoring*, encore incomplète mais qui offre déjà une bien plus grande finesse que les approches traditionnelles en filtration. Á partir de ces constats, nous pouvons maintenant choisir l'approche la plus pertinente possible pour développer notre prototype.

5.4 Approche choisie

Avant de mettre concrètement en place ce prototype, il est important de détailler son fonctionnement théorique et de justifier ces choix. Une fois ces points discutés, il nous faut ensuite choisir les solutions techniques les plus adaptées à ce fonctionnement théorique.

5.4.1 Objectifs du prototype

La description simple du principe du prototype est la mise en place d'un système de priorisation des variations identifiées par séquençage NGS programmable par l'utilisateur. Ce principe nécessite cependant des explications plus approfondies, puisque des objectifs nombreux sont en réalité présents. Nous pouvons préalablement rappeler notre contexte d'utilisation : un utilisateur seul, expert de la génétique mais sans compétences informatiques, avec en sa possession des fichiers contenant des variations nucléotidiques, qui sont des tableaux de taille importante contenant une variation par ligne et une annotation par colonne. L'objectif de cet utilisateur est de prioriser les variations contenues dans ces fichiers selon ses propres critères de façon fiable, efficace et efficiente. À partir de cette situation de départ, nous pouvons déterminer plusieurs objectifs :

— Le premier objectif est la mise en place d'un système de priorisation efficace. Les deux possibilités sont d'utiliser un système de filtres successifs pour éliminer les variations au

fur et à mesure ou de mettre en place un système de *scoring* associé à chaque variation, permettant de condenser tous les critères d'annotations en une seule valeur triable. Nous avons constaté dans la littérature scientifique les fortes limites de cette approche en filtres successifs, ainsi que le système intéressant de *scoring* proposé par *QueryOR*. En nous appuyant sur ces deux points, nous avons orienté notre choix sur une approche basée sur un système de score. L'approche en filtration n'est pas fondamentalement incompatible avec la problématique ni avec nos hypothèses, mais les limites de performances de ces systèmes en terme d'efficacité nous paraissent trop importantes.

— Le deuxième objectif principal est la possibilité de l'utilisation du logiciel sans posséder de compétences informatiques particulières. C'est sur cet aspect capital que s'articulent les contraintes relevant du domaine de l'IHM et en particulier de l'EUP.

— Le troisième objectif concerne quant à lui l'inclusion ou non de l'annotation au spectre du logiciel. La première possibilité est d'inclure cette étape. Cependant, nous avons constaté les limites de cette approche, qui réduit l'envergure des analyses de l'utilisateur et le contraint à l'utilisation de sources, certes référentes en la matière, mais génériques et potentiellement inadaptées aux analyses spécifiques à une pathologie ou un domaine d'intérêt restreint. La seconde possibilité est de laisser cette tâche à la charge de l'utilisateur ou du bio-informaticien. Nous pouvons également faire le lien avec le chapitre IV, traitant de cette problématique et posant les bases d'une annotation libre par et pour l'utilisateur. Dans ce contexte, il parait plus pertinent de ne pas inclure l'annotation et de la laisser à la charge du bio-informaticien ou du généticien assisté d'un annotateur adapté. L'incorporation de l'annotation dans le processus nous parait incompatible avec notre problématique initiale et en contradiction avec notre vision du domaine.

— Le prototype doit permettre une capitalisation du temps investi et sa réutilisation. Ce critère peut être séparé en deux points : la réutilisation de la méthode sur de nouvelles données d'analyse d'exomes de nos patients de façon simple et l'automatisation du processus afin de pouvoir prioriser des données en grandes quantités.

— L'utilisateur doit également pouvoir programmer finement les détails de sa priorisation. Cet objectif comprend la gestion du poids relatif de chacun de ses critères dans le résultat final, puisque tous n'ont pas la même importance biologique. Il faut pour cela pouvoir utiliser des pondérations négatives et positives, afin de faire remonter les variations intéressantes tout en écartant les variations inintéressantes. Enfin, les pondérations multiples, c'est-à-dire le fait de pouvoir prioriser de façon adaptée et simultanée différentes valeurs pour un même critère d'annotation, sont également une nécessité.

— Le prototype doit présenter une certaine robustesse, permettant notamment de supporter des critères d'annotations vides ou manquants ainsi que de supporter la variabilité inhérente à la manipulation de données biologiques.

Ce rappel des objectifs nous permet à présent d'effectuer des choix sur le fonctionnement du prototype.

5.4.2 Organisation de la phase de création

La première phase de fonctionnement du prototype est la mise en place de la priorisation. Les objectifs de cette phase sont de permettre à l'utilisateur de programmer la méthode de priorisation qui corresponde à ses besoins tout en limitant la charge mentale pour ce dernier. Nous avons choisi une interface en deux parties, l'une destinée à la création de la méthode, l'autre présentant les données de l'utilisateur priorisées par la méthode en cours de développement. Ce choix s'appuie sur la littérature ainsi que sur nos travaux précédents, notamment ceux décrit dans la section 4.4.5, qui ont montré l'importance capitale de la visualisation des données au cours des réflexions des généticiens.

Nous avons conçu le déroulement de cette étape comme une boucle itérative dans laquelle l'utilisateur perfectionne sa méthode de priorisation jusqu'à ce qu'il en soit satisfait. Cette phase est présentée dans le pseudocode 5.1. La mise au point de la méthode se fait par un système de conditions créées par l'utilisateur. Nous avons choisi une méthode d'interaction qui a fait ses preuves auprès de ce public notamment dans la section 4.6.2.3 du chapitre précédent. Les conditions et leurs conséquences sont ainsi présentées sous la forme de phrases en langage naturel où les différents mots-clés peuvent être choisis par l'utilisateur dans les ensembles à sa disposition. Ce système de phrases permet à l'utilisateur d'exprimer d'une façon naturelle sa volonté avec une grande liberté de termes, tout en garantissant une structure contrainte permettant un traitement informatique robuste et rapide.

Listing 5.1 – Pseudocode décrivant la phase de création d'un ensemble de règles

```
while ( rules_list.status not perfect )
{
        add_or_modify_rules(User, rules_list)
        results.update()
        analyse(User, results)
        if(is_perfect(results))
        {
                rules_list.status = perfect
        }
}
```

L'application des règles est effectuée en temps réel, les modifications de l'utilisateur se répercutent donc directement sur les données qui lui sont présentées. Il peut alors continuer de mettre au point sa méthode en ajoutant ou modifiant des règles, ou alors estimer son ensemble de règles suffisamment efficace et quitter la boucle de création.

Une fois l'ensemble de règles créé par l'utilisateur, il n'est pas utile de repasser par les étapes de visualisation et d'ouverture de fichier pour chaque patient. L'objectif de cette phase est plutôt de pourvoir traiter une série de données de patients le plus rapidement possible et le plus simplement possible pour l'utilisateur.

Cette phase ne demande pas de tâche de programmation particulière à l'utilisateur, puisqu'il ne s'agit que de l'application d'un ou des ensembles de règles qu'il a développé sur un ou plusieurs fichiers de variants structurés de manière identique. L'utilisateur a pour seule tâche la sélection des fichiers correspondants sur son disque dur.

5.4.3 Choix des technologies

En nous appuyant sur cette analyse des outils et des approches actuelles de la littérature scientifique, nous pouvons préciser les moyens à utiliser pour répondre à la problématique. Nous avons développé un prototype appelé *GenSCor*, pour "***Gen**etic **Sc**ore **Creat**or*. Nous avons choisi de développer cet outil sous la forme d'un script local indépendant. Nous avons vu que le choix d'un positionnement en tant que service web ne correspond pas à notre problématique, puisqu'il crée une nouvelle dépendance à un service extérieur qui peut s'arrêter à tout moment.

GenScor est développé en python, il utilise la bibliothèque graphique native de python, Tkinter, pour l'interface graphique. Cette bibliothèque étant fournie lors de l'installation de python, cela signifie que le seul pré-requis pour l'utilisateur est de disposer d'une machine avec python3 installé. Python3 est compatible avec les environnements Linux, Mac et windows, *GenScor* l'est donc aussi et a été testé avec succès sur ces trois plateformes.

5.5 *GenSCor*

Après avoir vu les limites des outils de la bibliographie et le fonctionnement théorique de *GenSCor*, nous pouvons décrire plus en détail ses fonctionnalités. Le prototype, la documentation et un exemple de données sont disponibles à cette adresse : `https://github.com/qRp/GenSCor`.

5.5.1 Principe et définitions

Pour commencer, après avoir décrit le fonctionnement général du prototype, nous allons détailler des points importants du fonctionnement du prototype, ce qui nous permet de définir certains termes propres à *GenSCor*.

5.5.1.1 Généralités

Lors de la première utilisation, le généticien commence par charger les données qu'il souhaite analyser, sous la forme d'un fichier plat annoté classique, contenant une variation par ligne et une annotation par colonne. Ce fichier est visualisé sur la moitié basse de l'écran ou dans une fenêtre à part selon le choix de l'utilisateur. Il peut ensuite créer autant de règles qu'il le souhaite. Le tableau est mis à jour à chaque action de l'utilisateur, montrant les variations associées à leurs scores afin que les valeurs de scores et le tri effectué représentent l'ensemble de règles actuellement défini. Cet ensemble est appelé *ruleset*.

Avant de détailler le fonctionnement de l'outil, trois termes importants sont à expliciter, les notions de règles, de *ruleset* et de score. Ces notions existent dans le langage courant, mais il est important de comprendre à quoi elles correspondent dans le contexte de *GenSCor*.

5.5.1.2 Les règles

Les règles sont les briques élémentaires du système. Elles peuvent prendre plusieurs formes, les règles complexes seront détaillées dans la partie 5.5.2.2. Les règles fonctionnent selon le principe d'instruction conditionnelle, c'est-à-dire qu'une action n'est effectuée qu'en fonction de l'évaluation d'une condition booléenne. Une règle simple est composée de 6 points, comme illustré dans la figure 5.11. Trois de ces points servent à la définition de la condition : la colonne, l'opérateur et la référence. Ils permettent de construire la condition booléenne à évaluer sous la forme ($colonne.opérateur.référence$), par exemple ($score\_grantham > 80$). Les trois autres points servent à définir l'action à effectuer si la condition est vraie, ce sont le coefficient, la direction et la valeur de score.

— Un coefficient – C'est un nombre entier de 0 à 5 permettant de nuancer l'importance de la règle. L'utilisateur le choisit dans une liste déroulante de 6 possibilités.

— Une colonne – C'est le nom d'une colonne du tableau de données. L'utilisateur choisit la colonne parmi une liste déroulante présentant toutes les colonnes du tableau.

— Un opérateur – C'est l'opérateur de la condition. Huit choix sont disponibles : égal à, différent de, supérieur à, inférieur à, supérieur ou égal à, inférieur ou égal à, contient, ne contient pas. L'utilisateur choisit parmi une liste déroulante présentant les 8 possibilités.

— Une référence – C'est la valeur utilisée lors de la comparaison. Les valeurs de la colonne choisie sont présentées à l'utilisateur dans une combobox, permettant donc de choisir une valeur présente dans les données ou de saisir librement une valeur.

— Une direction – C'est le sens dans lequel faire varier le score, positif ou négatif. L'utilisateur choisit parmi une liste de 2 possibilités : "Augmenter" ou "Diminuer".

— Une valeur de score – C'est la valeur à ajouter ou enlever au score total de la variation. L'utilisateur peut y entrer n'importe quel nombre au format *float*.

FIGURE 5.11 – Capture d'écran d'un exemple de règle

Les points sont présentés enchaînés sous la forme d'une phrase en langage naturel grâce à l'ajout de mots de liaisons entre les zones de choix de l'utilisateur. Comme on peut le voir dans la figure 5.11, un exemple de condition peut être "**1** If **Grantham> 80** then **increase** score by **50**". Dans cet exemple, le **1** correspond au coefficient, ici, il n'y a pas de multiplicateur particulier sur cette règle. La colonne est *Grantham*, qui correspond à l'entête de la colonne contenant les valeurs du score Grantham. L'opérateur est > qui permet de faire des comparaisons numériques. La référence est **80**, qui peut être choisie dans la liste de valeurs possibles ou saisie par l'utilisateur. Le sens choisi est **increase** et la valeur **50**. Cette règle a donc pour effet d'augmenter le score de chaque variation présentant un score Grantham strictement supérieur à 80 de 50 unités.

5.5.1.3 Le *ruleset*

Le *ruleset* est un ensemble de règles, il correspond à l'ensemble des règles définies pour être actives simultanément. L'utilisateur peut disposer de plusieurs *rulesets*, pour les adapter à une technique, une pathologie ou une approche d'analyse différentes. Dans un *ruleset*, toutes les règles sont dans une structure de conditions, que l'on pourrait représenter en informatique par une suite de blocs ***IF*** à la suite, mais non enchâssés les uns dans les autres. Ainsi, la validation ou le rejet d'une condition n'influe pas sur la règle suivante. Des types de règles plus complexes existent pour surpasser ce comportement simpliste pouvant être limitant, elles sont exposées dans la partie 5.5.2.2.

Les *rulesets* peuvent être sauvegardés et rechargés. Sous leur forme sauvegardée, ce sont des fichiers textes structurés de petites tailles (de l'ordre du ko) et non compressés. Ils peuvent donc être facilement échangés. Une fois que le *ruleset* est créé, il peut être appliqué automatiquement sur des fichiers dont la structure est la même que ceux ayant servi à sa mise au point. Un *ruleset* peut contenir autant de règles que l'utilisateur le juge nécessaire. Ils sont créés par défaut avec uniquement 3 règles, mais les règles peuvent être facilement ajoutées à partir d'un menu présenté figure 5.12. C'est également ici que se fait l'ajout des règles complexes qui seront détaillées dans la section 5.5.2.2.

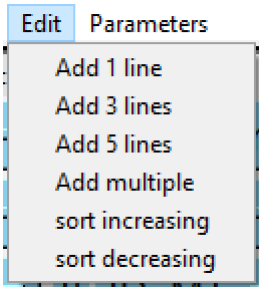


FIGURE 5.12 – Capture d'écran du menu d'ajout des règles

5.5.1.4 Le score

Chaque règle du *ruleset* associe une condition à un gain ou à une perte en points. Les points sont ici l'unité abstraite du score, ne représentant rien. Le score d'une variation est l'ensemble des points gagnés ou perdus en additionnant l'ensemble des valeurs des règles dont les conditions sont respectées par la variation. Le score est donc spécifique à la variation et au *ruleset* utilisé. Il sert à quantifier le degré d'intérêt que peut avoir la variation au regard de

ce *ruleset*. À ce titre, il sert de critère de tri et permet de prioriser les variations de la plus intéressante à la moins intéressante.

Si le score est principalement conçu pour servir de critère de tri, il peut aussi être utilisé en filtre avec une notion de seuil. Cette approche a les mêmes avantages que l'approche en filtres successifs classiques, mais n'en a pas les inconvénients car, grâce à l'intermédiaire du score, le risque d'éliminer un variant intéressant mais en dessous d'un seuil est très faible. Cette utilisation demande cependant une bonne maitrise du *ruleset* par l'utilisateur, car le seuil étant propre à chaque *ruleset*, sa détermination peut être complexe. L'une des méthodes de détermination de seuil est présentée dans la section 5.7.4.2. Le pseudocode 5.2 présente le principe de l'algorithme pour les règles simples. Il est important de préciser que les différents exemples présentés en pseudocode ont uniquement un but illustratif, ils ne reflètent en rien l'algorithme utilisé pour *GenSCor* qui est plus complexe. Il s'agit ici d'exemples naïfs afin d'illustrer le principe des différents algorithmes.

Listing 5.2 – Pseudocode de l'algorithme d'évaluation des règles simples

```
for variation in exome :
  for rule in rule_list :
    if ( rule[column] rule[operator] rule[ref]) :
      score_var += rule[direction]*rule[value]
sort exome by score_var
print exome
```

5.5.2 Fonctionnalités

Après avoir défini ce vocabulaire spécifique de *GenSCor*, nous pouvons détailler ses principales fonctionnalités.

5.5.2.1 Les options et le *parsing*

Le *parsing* en informatique est le fait de lire le contenu d'un fichier, généralement du texte ou un fichier semi-structuré et de le charger en mémoire sous une forme plus utile. Cette action correspond plus ou moins à l'interprétation du contenu du fichier. Dans le cas de fichier ayant une structure, c'est-à-dire des fichiers qui ne sont pas simplement du texte, il faut connaître la structure pour pouvoir réaliser cette interprétation. Dans la très grande majorité des cas, ce sont des formats standards comme le *XML* ou le *Json*. Dans notre cas, les fichiers sont des tableaux de données, les lignes représentant les variations et les colonnes les annotations. Cependant, cette structure peut varier légèrement en fonction des laboratoires et des outils d'annotations. Par exemple les séparateurs des colonnes peuvent être des points-virgules ou des tabulations, il peut y avoir des lignes de commentaires avant le tableau à proprement parler, ou encore certains fichiers ont un entête et d'autres non.

Afin de pouvoir interpréter ces différents fichiers sans le recours à des connaissances informatiques, *GenSCor* propose une série d'options (voir figure 5.13) permettant de définir un séparateur de colonne, un nombre de lignes à éliminer ainsi que la présence d'un entête. Ces options sont dans un menu qui permet également de gérer la langue utilisée dans l'interface ainsi que la disposition des fenêtres. Le français et l'anglais sont gérés, mais l'ajout d'autres langues est possible de façon simple. Les options sont stockées dans un fichier caché dans le répertoire de travail de l'utilisateur.

5.5.2.2 Les niveaux de règles

L'utilisation de règles simples, tel qu'évoqué ci-dessus, ne permet pas d'obtenir un degré d'expression suffisant. Il a donc été proposé d'ajouter des règles complexes, en associant plusieurs conditions à une seule action sur le score, ce qui correspond à des ***SI*** imbriqués, ou

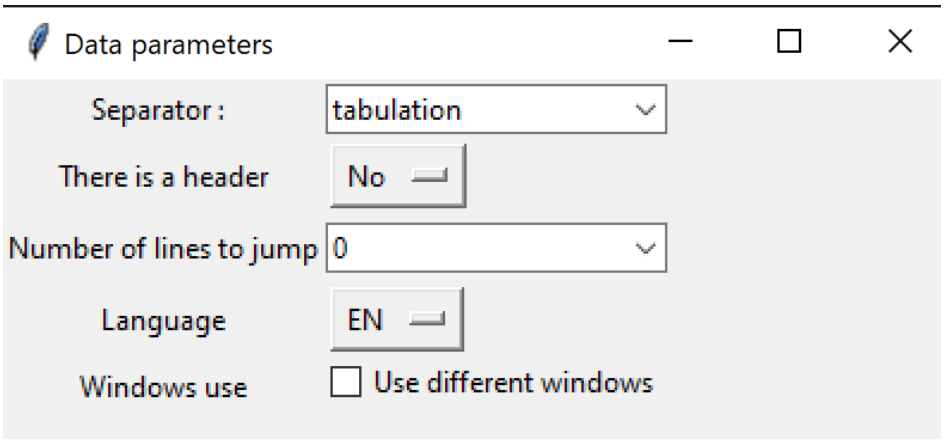


FIGURE 5.13 – Capture d'écran de la fenêtre des options de *GenSCor*

à un opérateur ***ET*** sur les conditions. Pour créer une telle règle, l'utilisateur peut utiliser le menu éditer, et sélectionner "Ajouter une règle multiple". Il peut ajouter autant de conditions différentes qu'il le souhaite, elles devront toutes être vraies pour que l'action soit effectuée sur le score. Une capture d'écran illustrative est présentée figure 5.14 et un algorithme simplifié dans le pseudocode 5.3.

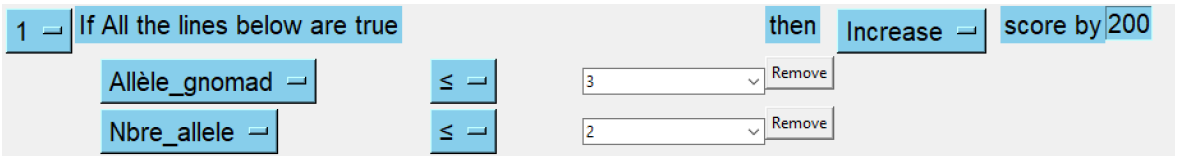


FIGURE 5.14 – Capture d'écran d'une règle ***ET***

Listing 5.3 – Pseudocode de l'algorithme d'évaluation des règles complexes ***ET***

```
for variation in exome :
  for rule in rule_list :
    if ( rule[kind] == "AND" ) :
      condition=True
      for i in range(len(rule[col])) :
        if not ( rule[col][i] rule[ope][i] rule[ref][i]) :
          condition=False
          break
      if (condition):
       score_var += rule[dir]*rule[val]
sort exome by score_var
print exome
```

Pour éviter de dupliquer ces blocs ***ET*** volumineux, un second niveau a été ajouté, permettant de définir des structures ***OU*** à l'intérieur de la structure ***ET***. Cela permet de prendre facilement en charge des différences sur les façons d'écrire des colonnes, par exemple de considérer à la fois les "**.**" pour les valeurs manquantes et les "**0**" pour les valeurs nulles. Une capture d'écran illustrative est présentée figure 5.15 et un algorithme simplifié dans le pseudocode 5.4. On peut observer dans la capture d'écran des boutons de plusieurs types :

Bouton "Add condition" Ce bouton permet d'ajouter une condition au bloc ***ET*** actif. La règle serait alors composée de 3 conditions simples et d'un bloc ***OU*** lui-même composé de deux conditions.

Bouton "Add OR condition" Ce bouton permet d'ajouter une condition au bloc ***OU***. La

règle serait alors composée de 2 conditions simples et d'un bloc ***OU*** lui-même composé de 3 conditions.

Bouton "Add new OR group" Ce bouton permet d'ajouter un nouveau bloc ***OU*** au bloc ***ET*** déjà présent. La règle serait alors composée de 2 conditions simples et de 2 blocs ***OU***, eux-mêmes composés de 2 conditions chacun. Les blocs ***OU*** sont distingués par un numéro, on peut d'ailleurs constater sur la figure 5.15 la nomenclature, le bloc présenté étant intitulé "OR bloc 1".

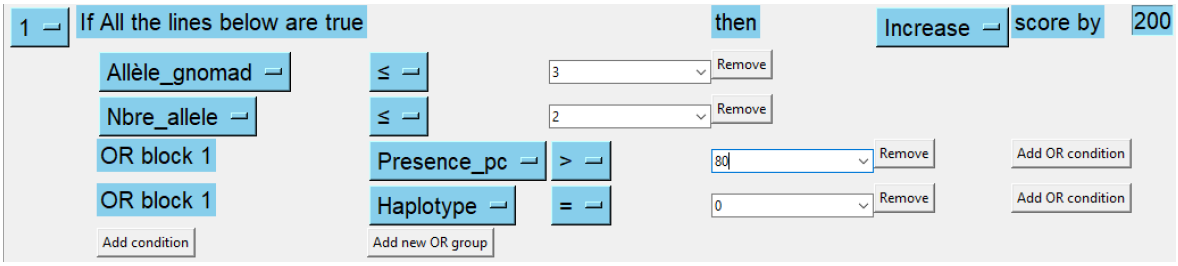


FIGURE 5.15 – Capture d'écran d'une règle ***OU***

Listing 5.4 – Pseudocode de l'algorithme d'évaluation des règles complexes ***OU***

```
for variation in exome :
  for rule in rule_list :
    if ( rule[typ] == "OR" ) :
      condition=False
      for i in range(len(rule[col])) :
        if ( rule[col][i] rule[ope][i] rule[ref][i]) :
          condition=True
          break
      if (condition):
        score_var += rule[dir]*rule[val]
sort exome by score_var
print exome
```

5.5.2.3 La gestion des données et des *rulesets*

Comme nous l'avons présenté, les données sont chargées et visualisées dans *GenSCor* lors de l'étape de création des règles. Il est aussi possible de charger et visualiser les données en utilisant un *ruleset* déjà défini. Il est pertinent pour les généticiens de tester leurs règles sur plusieurs patients afin de s'assurer une portabilité et une robustesse de leurs *rulesets*. L'interface présentant les données est une simple visualisation, elle ne permet pas d'interagir avec les données. Pour cela l'utilisateur peut exporter les données aux formats *tsv*, l'export contient exactement les mêmes données qu'en entrée mais ajoute une colonne supplémentaire contenant le score associé à chaque variation. Les données sont exportées triées de manière croissante ou décroissante.

```
1;Allèle_gnomad/Nbre_allele/;≤/≤//;3/2//;Augmenter;200
1;Allèle_gnomad/Nbre_allele/;contient/≤//;./2//;Augmenter;200
1;Couverture/;≤//;10//;Diminuer;300
1;Presence_pc/;≤//;30//;Diminuer;300
1;CLINSIG/;contient//;athogenic//;Augmenter;100
```

FIGURE 5.16 – Capture d'écran du format d'une sauvegarde de *ruleset*

Les *rulesets* peuvent bien entendu être sauvegardés sous la forme de fichiers textes structurés, comme présenté dans la figure 5.16. Chaque règle est stockée sur une ligne, les différents champs des règles sont séparés par des points-virgules et des barres obliques servent de séparateurs pour les règles complexes ***ET*** et ***OU***. Ce sont donc des fichiers très légers facilement échangeables par mail, ce qui permet une entraide à la création et une diversité des points de vues lors de l'analyse, même en cas d'éloignement géographique des généticiens. La gestion des données comme celles des *rulesets* se fait *via* le menu *File* présenté figure 5.17.

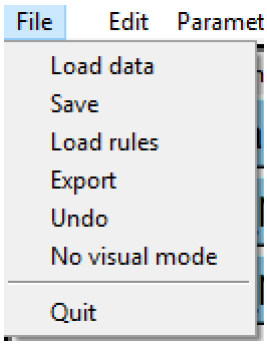


FIGURE 5.17 – Capture d'écran du menu *File* de *GenSCor*

5.5.2.4 Mode non visuel

Le mode non visuel est ainsi appelé car les données ne sont pas visualisées. Il permet d'appliquer un ou plusieurs *rulesets* sur un ou plusieurs fichiers de données. Un fichier export est produit pour chacun des fichiers de données fourni en entrée, préfixé d'une colonne par *ruleset* sélectionnés. Ce mode correspond à l'automatisation du processus, il est plus rapide car aucun retour visuel n'est produit et permet donc de traiter plus rapidement les données.

L'objectif de ce mode est d'être utilisé en routine, lorsque le ou les scores sont bien établis pour une analyse, afin qu'ils puissent être appliqués sur une série complète de plusieurs analyses de patients. L'utilisateur peut ainsi continuer à visualiser les données dans un tableur et bénéficier de ses fonctionnalités, tout en ayant les variations déjà triées selon ses propres règles.

5.6 Méthodologie d'évaluation

Dans cette section, nous détaillons la méthodologie que nous avons appliquée pour évaluer l'utilisabilité de *GenSCor*. Dans un premier temps, nous rappelons les objectifs que nous nous sommes fixés. Dans un deuxième temps, nous détaillons le protocole que nous avons suivi.

5.6.1 Rappel des objectifs

GenSCor doit permettre au généticien de mettre en place un système de priorisation personnel basé sur un système de score. Il doit pouvoir être utilisé par des experts généticiens sans l'aide d'un bio-informaticien et sans compétences informatiques. Il doit permettre au généticien de construire aisément et de réutiliser un ensemble de règles de priorisation personnelles sans le contraindre à utiliser des annotations ou un format de fichier particulier. Enfin, *GenSCor* doit supporter les différentes approches de recherches pouvant être déployées par les généticiens et ne doit pas les contraindre à une méthode particulière.

5.6.2 Participants

Il est difficile d'évaluer une approche comme celle de *GenSCor* sur le plan quantitatif : les experts généticiens sont rarement nombreux sur un même site et ils ont peu de temps à

consacrer à des études d'utilisabilité, même si le logiciel proposé peut leur apporter une aide précieuse. Nous avons donc opté pour une étude qualitative, l'observation des utilisateurs en étude longitudinale sur une cohorte significative étant prévue pour les perspectives de ce travail, une fois l'application validée et déployée.

Nous avons pu obtenir la participation de 4 experts, aux profils variés : un biologiste généticien, que nous appellerons *F.*, une interne biologiste, *L.*, un généticien clinicien, *G.* et enfin un interne clinicien, *T.*. Les deux premiers testeurs, *F* et *G*, disposent d'une plus grande expérience dans l'analyse d'exomes, qui est en place depuis plus de deux ans au sein du laboratoire. Les deux internes, *T* et *L*, disposent de bases théoriques solides, mais d'une expérience pratique plus limitée dans l'analyse de données d'exomes.

La méthode préalablement utilisée par les participants et avec laquelle ils sont familiers est une approche en filtres successifs réalisés dans un tableur. Cette méthode est décrite en détail dans la section 2.1.2.

5.6.3 Données utilisées

Nous avons utilisé pour évaluer *GenSCor* une cohorte rétrospective d'analyses d'exomes provenant de 129 patients répartis sur 14 séries. Ces données proviennent majoritairement de séquençages réalisés localement, mais certaines ont été séquencées par une entreprise privée sous-traitante, tandis que d'autres ont été séquençées dans d'autres laboratoires dans le cadre de projets inter-régionaux. Parmi ces données d'exomes, 29 variations pathogènes ont été relevées par des méthodes d'analyses traditionnelles, c'est-à-dire sans l'utilisation de *GenSCor*. Les données sont annotées par le *pipeline* d'annotation mis en place en préambule de ce travail, contenant plus de 120 critères différents.

5.6.4 Protocole expérimental

On peut séparer le protocole en deux volets, l'organisation des tests, décrivant la mise en place des tests auprès des utilisateurs, et l'organisation des analyses, qui correspond à la façon dont sont analysés les résultats obtenus.

5.6.4.1 Organisation des tests

La conception du prototype s'est déroulée de manière itérative, les différentes étapes du prototype étant régulièrement testées et discutées auprès d'un expert, l'utilisateur *F.*. Lorsque le développement du prototype a été considéré comme complet, il a été fourni aux 4 utilisateurs pour la phase de test. Leur tâche était de développer une approche de priorisation à l'aide de *GenSCor* permettant d'extraire des variations pathogènes d'analyses d'exomes. Cette tâche n'est pas nouvelle pour eux, puisqu'ils l'effectuent déjà en s'appuyant sur l'approche manuelle. Les utilisateurs n'ont pas de contraintes sur la façon de programmer leur règles, la seule contrainte est de sauvegarder leur *rulesets* régulièrement avec des noms explicites et un numéro de version.

Les tests s'effectuent sur les machines des utilisateurs sur lesquels ils sont également chargés de la mise en place du prototype. Ils disposent de la documentation et d'un support technique en cas de comportement imprévu du prototype. L'objectif est de laisser les utilisateurs s'approprier l'outil afin qu'ils puissent créer leurs *rulesets* avec le moins d'interférences possibles. La mise à disposition n'est pas contrainte dans la durée, ce qui permet de voir l'évolution de l'usage du prototype sur le long terme, qui peut par exemple mener à une utilisation sur de nouvelles tâche ou à un arrêt de l'utilisation.

Les *rulesets* sont récupérés au bout de plusieurs mois d'utilisation autonome, afin de garantir une certaine maturité des données. Ils sont ensuite appliqués simultanément sur les données à l'aide de *GenSCor*

5.6.4.2 Organisation des analyses

L'analyse des résultats se fait à 3 niveaux. Le premier niveau est l'analyse du contenu des *rulesets* des utilisateurs. Il s'agit d'analyser de façon qualitative les productions des utilisateurs et d'en extraire des informations sur leur usage de *GenSCor*. Nous cherchons par cette analyse à vérifier l'usage des différentes fonctionnalités et à comparer les approches choisies par les utilisateurs.

Dans un second temps, nous nous intéressons à l'efficacité de *GenSCor*, c'est-à-dire sa capacité à prioriser correctement les variations pathogènes dans notre contexte. Pour cela nous utilisons le *corpus* de variations pathogènes déjà identifiées et nous analysons les rangs auxquels les approches développées par nos utilisateurs les placent. Nous ne pouvons utiliser le taux de diagnostic comme mesure réelle de l'efficacité, car ce taux varie en fonction de très nombreux paramètres que nous ne maitrisons pas. Nous avons fait le choix d'une étude sur des données rétrospectives pour contrôler une partie de cette variabilité inhérente aux données biologiques et pour comparer *GenSCor* à la méthode manuelle. Cette approche nous permet de vérifier que *GenSCor* est au moins aussi efficace que la méthode manuelle.

Enfin, le dernier point d'étude est l'efficience. Dans ce contexte, elle est corrélée à deux aspects : le temps passé par l'utilisateur pour analyser un fichier et la charge mentale qu'une telle analyse représente. Il est complexe de comparer les temps pris par chaque méthode pour plusieurs raisons. La première est la différence de fonctionnement : les temps d'analyses de l'approche traditionnelle sont simples à calculer, les bornes sont claires, l'analyse commençant à l'ouverture du fichier et se terminant lorsque la variation pathogène en est extraite. Concernant l'évaluation du temps passé avec le prototype, la tâche apparait plus complexe : faut-il compter le temps de mise en place du *ruleset*, qui peut être long mais qui est rentabilisé sur un grand nombre d'analyses ? Faut-il compter le temps machine ? De plus, un généticien ne peut pas être chronométré deux fois sur le même patient, car lorsque le travail d'analyse est déjà effectué, il connait la variation pathogène, la deuxième analyse serait donc complètement biaisée. Il est de même complexe de comparer les temps de plusieurs généticiens pour un même patient, les méthodes d'analyse étant très différentes, les durées le sont également. Enfin, d'un patient à l'autre, la variabilité est également très importante.

Ainsi, nous avons choisi de ne pas étudier le temps en comparaison de méthode pour le moment, mais de nous concentrer sur l'analyse du temps machine ainsi que sur le temps d'élaboration d'un score. Nous avons également recueilli les estimations des différents testeurs en indication du ressenti. La variabilité de ces valeurs montre les différentes stratégies, ainsi, certains utilisateurs avancent des temps d'analyse complète de plusieurs heures avec la méthode manuelle et d'environ une heure avec *GenSCor* alors qu'un utilisateur compte seulement une vingtaine de minute par patient avec *GenSCor*, environ une heure sans. Ces différences même dans l'estimation montre la difficulté d'obtenir des valeurs fiables de temps d'analyse en quantité suffisante pour pouvoir réaliser une analyse statistique.

5.7 Résultats

Les tests ont permis de produire deux types de résultats, des résultats directs, correspondant aux différents *rulesets* créés par les utilisateurs, et des résultats indirects, correspondants à l'application des *rulesets* à des données réelles. Dans une première partie nous allons analyser les jeux de règles créés par les utilisateurs, puis dans les deux parties suivantes, les résultats de ces *rulesets* sur l'efficacité et l'efficience de la recherche de variations pathogènes dans l'exome en conditions réelles.

5.7.1 Confrontation du prototype aux règles issues de Gendomus améliorées

Dans un premier temps, nous pouvons vérifier que le prototype correspond bien aux règles de bonnes pratiques définies par GenDomus et que nous avons adaptées au domaine. Ce premier point nous permet de comparer *GenSCor* aux autres outils de la bibliographie et d'envisager en fonction de sa conception, la marge d'erreur liée au prototype.

règle 1 *Les données doivent être visibles lorsque la tâche y fait référence.* Les données sont continuellement visibles dans le prototype et servent de références.

règle 2 *L'utilisateur doit avoir un aperçu du résultat de ses actions en temps réel.* Les mises à jour du score sont effectuées en temps réel, le tri se met à jour à la volonté de l'utilisateur, pour ne pas changer les données affichées en continu et ainsi casser l'effet référence des données, c'est-à-dire obliger l'utilisateur à réfléchir à nouveau pour identifier les différents critères.

règle 3 *En cas de comparaison, les données à comparer doivent être présentées côte à côte.* Les données étant présentées groupées, cette règle peut être validée, même si, comme pour les outils de la littérature, il n'y a pas de comparaisons franches.

règle 4 *Les démarches peuvent être enregistrées.* Les *rulesets* sont enregistrables et exportables.

règle 5 *Les démarches peuvent être reexécutées rapidement.* Les *rulesets* peuvent être chargés très rapidement. Le mode non visuel permet d'accélérer encore le processus et de complètement automatiser cette partie de l'analyse.

règle 6 *Les démarches peuvent être partagées.* Les fichiers de sauvegarde de *rulesets* sont très légers et peuvent être partagés très facilement par mail.

règle 7 *Le système doit rester utilisable à l'échelle du génome.* Les *rulesets* sont applicables à l'échelle du génome dans des temps certes longs, mais possibles, de l'ordre d'une trentaine de minutes pour un fichier contenant 5 575 000 variations.

règle 8 *Le système doit fonctionner sur le plus de configurations possibles.* Le système a été testé sur Ubuntu 14, 16 et 18, sur Mac OS ainsi que sur les environnements Windows 7, 8 et 10 sans problème.

règle 9 *Le système doit être facile à mettre en place et à tester.* Le système ne nécessite que python3, présent par défaut sur bon nombre de machines. Il s'exécute d'un simple double-clic, les différents utilisateurs n'ont pas eu besoin d'assistance pour sa mise en place.

règle 10 *Le système doit rester accessible.* Le système est indépendant d'Internet, donc même sans réseau et avec un arrêt total du développement, les versions fonctionnant actuellement peuvent perdurer.

GenSCor valide toutes les règles de développement fixées, ce qui permet une certaine confiance dans la capacité du prototype à représenter l'approche dans les tests qui vont suivre. Cela permet aussi de se démarquer des outils de la littérature, puisque qu'aucun ne remplissait toutes les règles, les meilleures performances étant atteintes par *Varaft* et *BierApp* qui validaient tous les deux 7 des 10 règles.

5.7.2 Analyse qualitative de la construction des *rulesets*

Avant d'étudier les performances effectives des *rulesets* créés par nos utilisateurs, nous pouvons faire une première analyse qualitative en observant leurs contenus.

5.7.2.1 Nombre de règles

Le premier constat est que les utilisateurs ont choisi des nombres différents de *rulesets*. L'utilisateur *T.* n'en a fait qu'un seul composé de 48 règles. L'utilisateur *L.* a choisi de répartir

ses règles en 3 fichiers, chacun dédié à un mode de transmission de l'allèle, dominant, récessif ou lié à l'X. Ses fichiers comptent respectivement 28, 20 et 26 règles. Les deux derniers utilisateurs ont tous les deux choisis de séparer leurs règles en 4 fichiers, mais avec des méthodes de répartitions différentes. L'utilisateur *F.* a aussi utilisé le mode d'expression de l'allèle, mais il y a ajouté le degré d'information disponible sur le gène en se servant de la base de données OMIM pour moduler l'information. Les 4 *rulesets* sont centrés sur les variations liées à l'X, sur les variations récessives affectant un gène OMIM, sur les variations dominantes sur les gènes OMIM et enfin sur les variations dominantes dans d'autres gènes. Le nombre de règles est assez élevé, respectivement 36, 38, 46 et 41 règles. Enfin, l'utilisateur *G.* a choisi de séparer les règles selon une logique prenant en compte la transmission des gènes de façon plus fine. Trois *rulesets* sont ainsi classiquement dédiés aux variations liées à l'X, dominantes et récessives, le dernier recherche des variants *de novo*, c'est-à-dire qui ne sont pas hérités des parents. Ces *rulesets* contiennent respectivement 37, 41, 41 et 41 règles. Même si le nombre de règles est proche, elles portent sur des critères différents.

5.7.2.2 Nombre de critères

Les critères sont les colonnes sur lesquelles s'appliquent les conditions. Plus de 120 colonnes différentes sont disponibles dans les fichiers annotés, mais elles ne sont pas toutes destinées à être utilisées comme bases de seuil. Par exemple, la position génomique ou protéique ne présente pas d'intérêt pour filtrer un variant, mais elle apporte une information importante dans l'étude détaillée de cette variation. Si certains critères sont utilisés par les 4 testeurs, il existe une disparité importante correspondant à nos observations sur les pratiques variées des généticiens dans le processus d'analyse *via* la méthode traditionnelle. Les résultats sont résumés dans la figure 5.18. Le nombre de critères correspond aux nombres de colonnes différentes utilisées par chacun des utilisateurs (en prenant en compte tous leurs *rulesets*). Le nombre total de critères est de 55, cela ne correspond pas à la somme arithmétique mais au nombre total de colonnes différentes utilisées par les 4 testeurs. On constate ainsi une disparité relativement importante des critères utilisés.

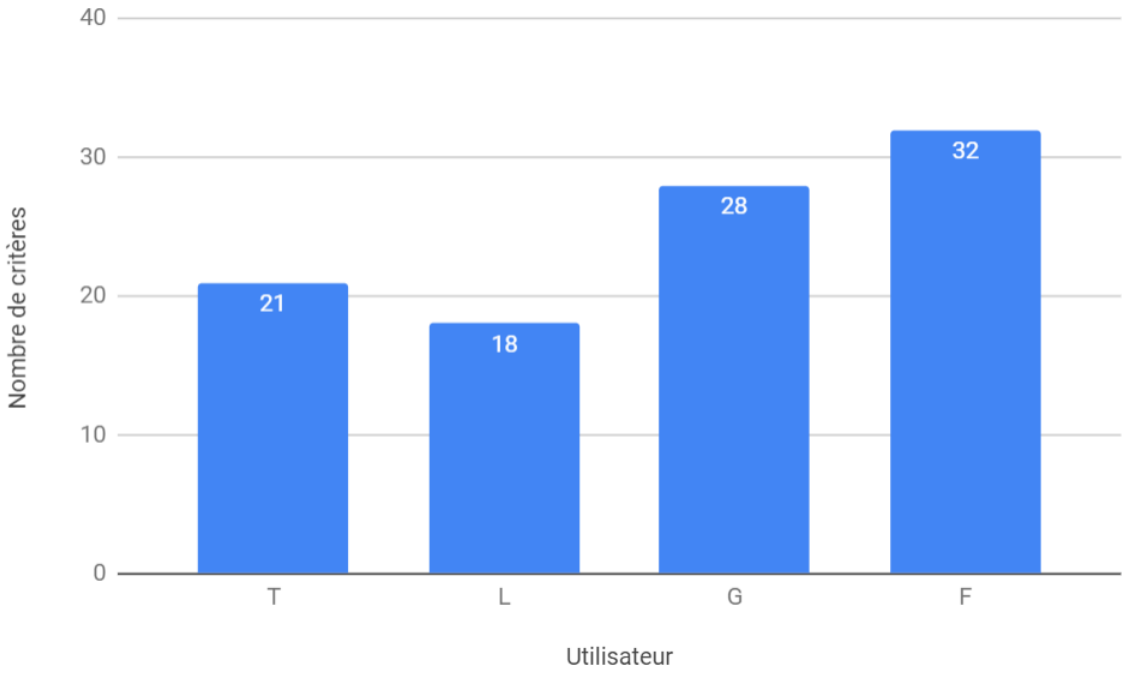


FIGURE 5.18 – Nombre de critères utilisés pour chaque utilisateur

Ces données confortent nos observations initiales où il est nécessaire d'obtenir un fichier

annoté le plus complet possible, puisqu'un très grand nombre de critères sont utilisés ici.

5.7.2.3 Partage des critères

Certains critères sont utilisés par plusieurs testeurs, comme présenté dans la figure 5.19. Parmi ces critères, 9 sont partagés par les 4 utilisateurs. Il s'agit de données standards du domaine ou très importantes dans le processus de tri. On y trouve des fréquences alléliques (celles de *gnomAD* et celles propres au laboratoire), des données de qualité, la profondeur de séquençage et le pourcentage de présence, les effets de la variation (représentés par la colonne dédiée ainsi que par la colonne contenant le score de *VaRank*), et enfin, le résultat de la base de données *ClinVar*. On trouve ensuite 7 critères partagés par 3 utilisateurs, et 9 partagés par 2. Nous pouvons constater que moins de la moitié des critères (25) sont partagés par au moins deux utilisateurs.

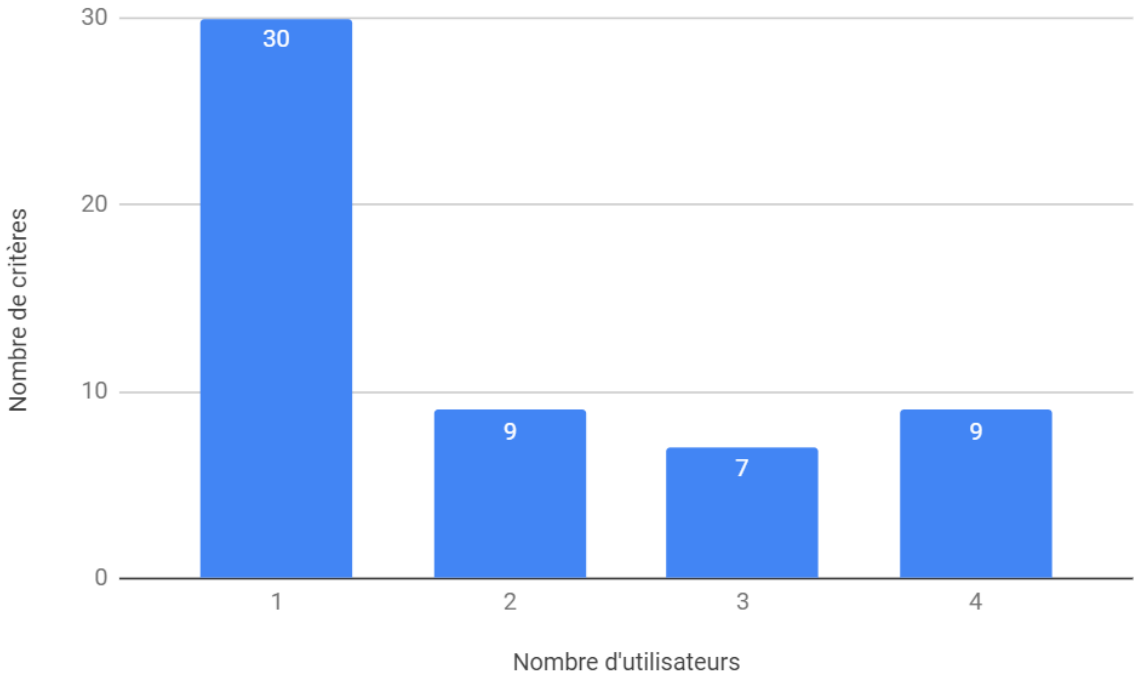


FIGURE 5.19 – Nombre de critères partagés en fonction du nombre d'utilisateurs

Ces données correspondent aux observations du domaine que nous avions faites, qui montraient une grande disparité des pratiques d'un généticien à l'autre. De plus, ces disparités apparaissent déjà importantes alors que l'on se trouve dans un cas d'utilisation au sein du même laboratoire et portant sur la même analyse. On peut raisonnablement émettre l'hypothèse que les disparités seraient encore plus importantes en variant le type d'analyse et en ouvrant le prototype à d'autres laboratoires.

5.7.2.4 Comparaison de l'annotation

À l'aide de cette liste des critères concrètement utilisés par les utilisateurs, on peut maintenant comparer l'annotation effectuée préalablement avec celles proposées par *QueryOR* et *Varaft*, les deux outils bibliographiques proposant les annotations les plus complètes. Les résultats sont présentés dans le tableau 5.3.

Parmi les critères partagés par 4 utilisateurs, la mauvaise gestion des bases de données personnelles des utilisateurs est la cause de la différence de performance entre *Varaft* et *QueryOR*. Seul le score *VaRank* est absent chez les deux outils. On peut constater sur ces points une relativement bonne performance de *QueryOR* en ce qui concerne la gestion des

Nombre d'utilisateurs	Nombre de Critères partagés	Nombre de critères présent dans *Varaft*	Nombre de critères présent dans *QueryOR*	Nombre de critères présent dans *Varaft* et *QueryOR*	Nombre de critères présent ni dans *Varaft* ni dans *QueryOR*
4	9	3	7	2	1
3	7	0	1	0	5
2	9	2	3	2	6
1	30	16	15	12	13

Tableau 5.3 – Tableau comparatif des annotations

critères les plus utilisés. Au niveau des critères partagés par 2 ou trois utilisateurs, il manque les informations *Domino*, *Intervar*, *denovo-db*, ou encore certaines données de *gnomAD*. *Varaft* présente une bonne couverture de score de prédictions, mais ce sont des critères peu employés par les utilisateurs. Les critères manquant à ce niveau sont des annotations provenant principalement des bases de données *PanelApp* et *SysID*. En conclusion, ces informations viennent conforter notre hypothèse qu'il est préférable de ne pas inclure l'annotation si l'utilisateur ne peut pas y inclure les concepts qu'il utilise dans ses protocoles d'analyse.

5.7.2.5 Utilisation des conditions

En plus du nombre de règles, il est intéressant de regarder le nombre de conditions, ainsi que la répartition des règles simples ou complexes. Les résultats sont récapitulés dans le tableau 5.4. Le nombre de conditions maximum est de 230 pour l'utilisateur *F.* alors que son nombre de règles n'est pas le plus élevé. Cette différence est due au plus grand nombre de règles complexes utilisées. De plus, cet utilisateur a souvent multiplié les conditions pour les règles complexes, pouvant aller jusqu'à 5 conditions pour une seule règle. Le nombre moyen de conditions par règle est de 3.12 pour l'utilisateur *F.* alors qu'il n'est que de 2.31 pour l'utilisateur *G.*

Utilisateur	*L.*	*T.*	*F.*	*G.*
Nombre de règles	74	48	162	170
Nombre de règles moyen par *ruleset* et écart-type	24,66 (4,16)	48	40,5 (4,20)	42,5 (4,57)
Nombre de conditions	80	48	230	195
Nombre de conditions moyen par *ruleset* et écart-type	26,66 (4,16)	48	57,5 (12,50)	48,75 (3,74)
Nombre de ***ET***	6	0	32	24
Nombre de ***OU***	0	0	0	1

Tableau 5.4 – Tableau comparatif de l'utilisation des conditions simples et complexes

Nous pouvons également constater l'absence de l'utilisation de règles avec l'opérateur ***OU***. Cette absence peut venir de son inutilité dans de tels cas d'analyse, ou d'une mise en place imparfaite dans le prototype empêchant son utilisation optimale par les testeurs. Les règles complexes avec l'opérateur ***ET*** sont en revanche utilisées en grande quantité par deux des 4 utilisateurs, on peut donc supposer que leur utilisation résulte d'un choix de méthode d'analyse. On peut également remarquer que le nombre moyen de conditions par *ruleset* est relativement stable pour 3 des 4 utilisateurs alors qu'il est environ 2 fois moindre pour le dernier.

5.7.2.6 Type de règles

L'un des derniers points de comparaison simple est le nombre de règles augmentant ou diminuant le score. Nous pouvons également observer les modifications moyennes, que nous pouvons diviser entre les pondérations positives et négatives en associant le sens de la modification à sa valeur, ces résultats sont présentés dans le tableau 5.5.

Utilisateur	*L.*	*T.*	*F.*	*G.*
Pondération positive	34	25	109	87
Valeur de pondération positive moyenne et écart-type	113,53 (55,56)	61,4 (61,07)	23,94 (19,69)	65,05 (68,29)
Nombre de pondération négative	41	23	50	69
Valeur de pondération négative moyenne et écart-type	-248,29 (177,18)	-120 (92,49)	-71,4 (28,71)	-165,94 (138,24)
Ratio nombre	0,83	1,09	2,18	1,26
Modification moyenne	-84,27	89,47	-6,04	-34,56

Tableau 5.5 – Tableau comparatif des approches de pondération des utilisateurs

On peut constater ici aussi des différences importantes au niveau des approches utilisées, notamment en observant les ratios entre le nombre de pondérations positives et négatives. Si trois utilisateurs ont des ratios relativement proches de 1, l'utilisateur *F.* a choisi une approche favorisant la pondération positive. Les valeurs des pondérations sont également beaucoup plus faibles que pour les autres utilisateurs. Les valeurs moyennes des pondérations varient beaucoup d'un utilisateur à l'autre, mais les rapports sont relativement stables, les valeurs négatives étant 2 à 3 fois plus importantes que les valeurs positives.

Enfin, il est intéressant de constater un usage particulier chez l'utilisateur *T.* : comme la plupart des utilisateurs, la très grande majorité des valeurs de scores qu'il ajoute ou soustrait est un nombre rond, finissant par 0. Il a cependant ajouté une seule règle ajoutant 55 points en cas de variation homozygote. Cette règle lui permet de déterminer en regardant le dernier chiffre du score si le variant est homozygote ou pas. Ainsi, ces variations sont plus mises en lumières, sans toutefois modifier l'échelle complète du score, ce qui risquerait de ne faire remonter que les variations homozygotes.

5.7.2.7 Utilisation sur la cohorte de données d'analyses d'exomes

La confrontation des *rulesets* aux données permet d'obtenir les scores moyen, médian, maximum et minimum de chaque *ruleset* sur notre corpus de données. Les résultats sont présentés dans le tableau 5.6.

Les différences d'échelles et d'approches que nous avons constatées en comparant les *rulesets* de nos différents utilisateurs se retrouvent lors de leur confrontation aux données. On peut ainsi observer des écarts de l'ordre de 400 % à 500 % entre les valeurs des scores de l'utilisateur *L.* et ceux de l'utilisateur *F.*. On peut de nouveau constater que les valeurs des pondérations négatives sont plus importantes que celles des positives. On observe ce décalage à la fois sur les scores moyens et médians, tous négatifs, ainsi qu'en comparant les valeurs absolues des maximums et minimums, où les valeurs minimums sont plus importantes.

Ruleset	Score moyen	Score médian	Score maximum	Score minimum
LD	-579	-550	700	-2250
LX	-1118	-1100	500	-1800
LR	-681	-600	950	-2200
T	-692	-750	450	-1310
FDR	-276	-250	285	-650
FDO	-233	-190	320	-630
FRO	-152	-200	345	-490
FX	-219	-130	235	-630
GAR	-579	-550	700	-2250
GX	-579	-550	700	-2250
GAD	-579	-550	700	-2250
GDN	-579	-550	700	-2250

TABLEAU 5.6 – Tableau comparatif des échelles des *rulesets* sur des données d'exomes. LD : *L.* Dominant, LX : *L.* sur le chromosome X, LR : *L.* récessif, T : *T.* ; FDR : *F.* dominant sur des gènes "recherche", FDO : *F.* dominant sur des gènes OMIM, FRO : *F.* récessif sur des gènes OMIM, FX : *F.* sur le chromosome X, GAR : *G.* autosomique récessif, GX : *G.* sur le chromosome X, GAD : *G.* autosomique dominant, GDN : *G. De novo.*

Les *rulesets* sont nommés en fonction du nom donné par l'utilisateur, préfixé de la lettre le désignant. Les utilisateurs n'ayant pas choisi les mêmes logiques de séparations de leurs *rulesets*, la nomenclature est différente d'un utilisateur à l'autre. On peut néanmoins noter des similarités, la lettre *X* correspond à des *rulesets* dédiés aux variations sur le chromosomes *X* (GX, FX, LX) pour tous les utilisateurs. Les scores dominants correspondent aux *D*, comme LD, FDO (Dominant OMIM), FDR (Dominant Recherche), GAD (Autosomique Dominant) et GDN (De novo). Enfin, les derniers correspondent à une recherche de variations récessive, LR, FRO (Récessif OMIM) ou enfin GAR (Autosomique récessif), bien que la lettre R soit également utilisé pour désigner des gènes "Recherche", c'est-à-dire des gènes qui ne sont pas inclus dans la base de données OMIM, donc où le savoir disponible est plus restreint.

5.7.2.8 Profil des *rulesets*

En plus de la comparaison directe aux données, on peut tracer pour chaque ensemble de *rulesets* la répartition des différents scores. Pour cela on compte le nombre d'occurrences de chaque valeur de score possible parmi la cohorte d'exomes, soit plus de 8,8 millions de variations. Les données sont groupées par intervalle de 100 pour les ensemble de trois utilisateurs et par intervalle de 20 pour l'utilisateur *F.*, pour correspondre à l'échelle des *rulesets*. Les profils sont présentés dans les figures 5.20, 5.21, 5.22 et 5.23.

Ces profils nous apportent plusieurs informations. Ils permettent dans un premier temps de vérifier que l'approche choisie par tous les utilisateurs est de pénaliser les variations qui ne sont pas intéressantes. On le constate très bien à la distribution des données, qui se situent très majoritairement dans des plages de valeurs négatives. On constate aussi que les données ne sont pas réparties selon une loi normale, mais qu'il y a des pics correspondant à des types de variations. C'est assez flagrant lorsque l'on analyse les *rulesets* spécialisés dans le chromosome X chez les utilisateurs *F.* et *L.*, où deux pics très importants correspondent en fait à toutes les variations autosomiques qui sont alors très fortement pénalisées.

La répartition nous permet aussi d'observer les deux phases des *rulesets*. La première phase est de pénaliser grandement toutes les variations dont l'aspect bénin est certain. La seconde phase vient ensuite nuancer les variations potentiellement intéressantes restantes, en effectuant une priorisation plus détaillée. Ces observations correspondent à ce que nous avons constaté dans le contenu des *rulesets*, c'est-à-dire de nombreuses conditions augmentant le score, mais avec de plus faibles valeurs que les conditions qui le diminuent.

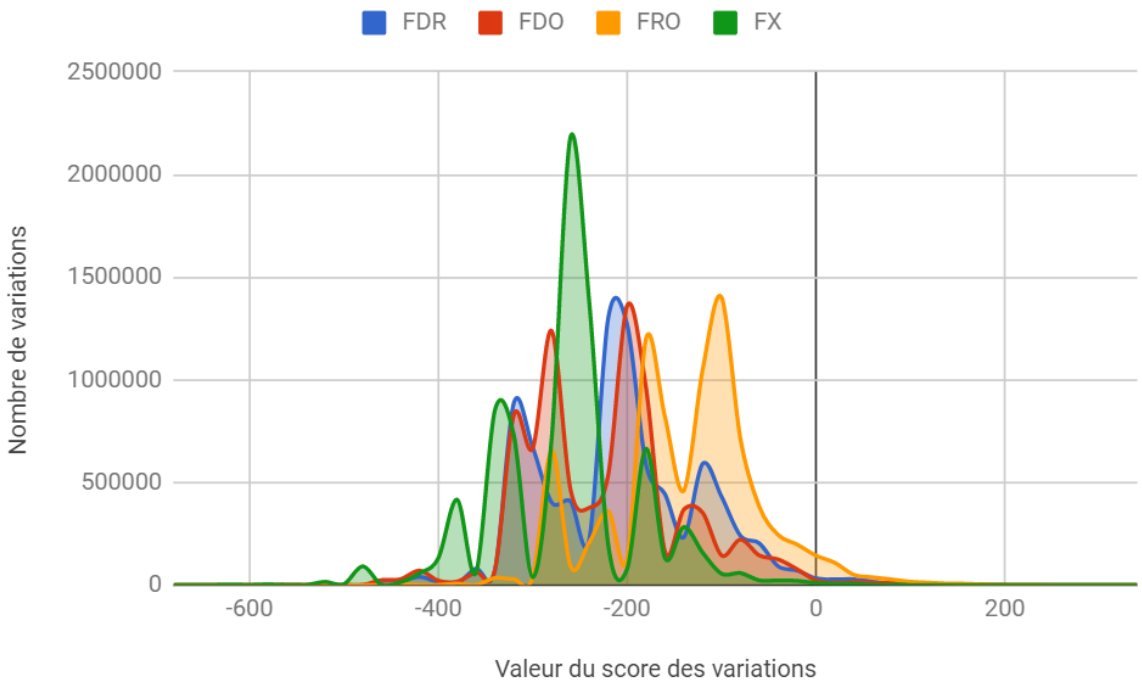


FIGURE 5.20 – Profil des *rulesets* de l'utilisateur *F.* FDR : *F.* dominant sur des gènes recherches, FDO : *F.* dominant sur des gènes OMIM, FRO : *F.* récessif sur des gènes OMIM, FX : *F.* sur le chromosome X.

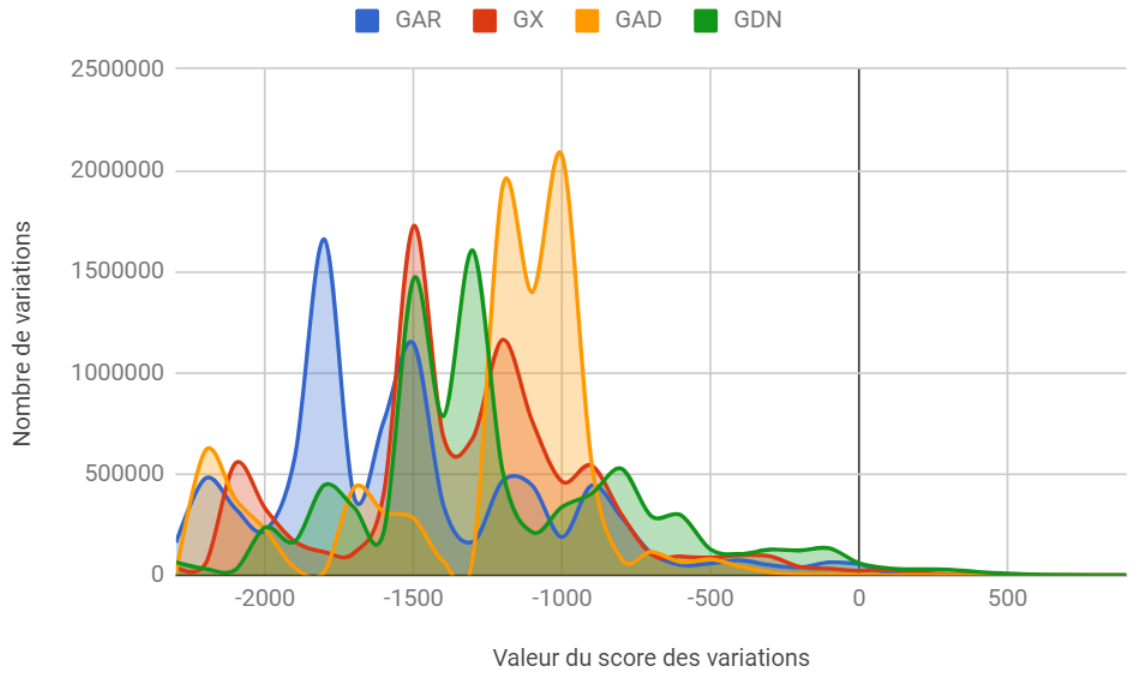


FIGURE 5.21 – Profil des *rulesets* de l'utilisateur *G.* GAR : *G.* autosomique récessif, GX : *G.* sur le chromosome X, GAD : *G.* autosomique dominant, GDN : *G. De novo.*

Enfin, ces profils permettent de visualiser la fraction réellement intéressante des variants issus d'une analyse d'exome et de commencer à mesurer le pouvoir filtrant que peuvent représenter ces *rulesets*. Ces mesures seront calculées plus en détail dans la partie traitant de l'impact de l'approche sur l'efficience et la charge mentale lors de l'analyse d'un exome.

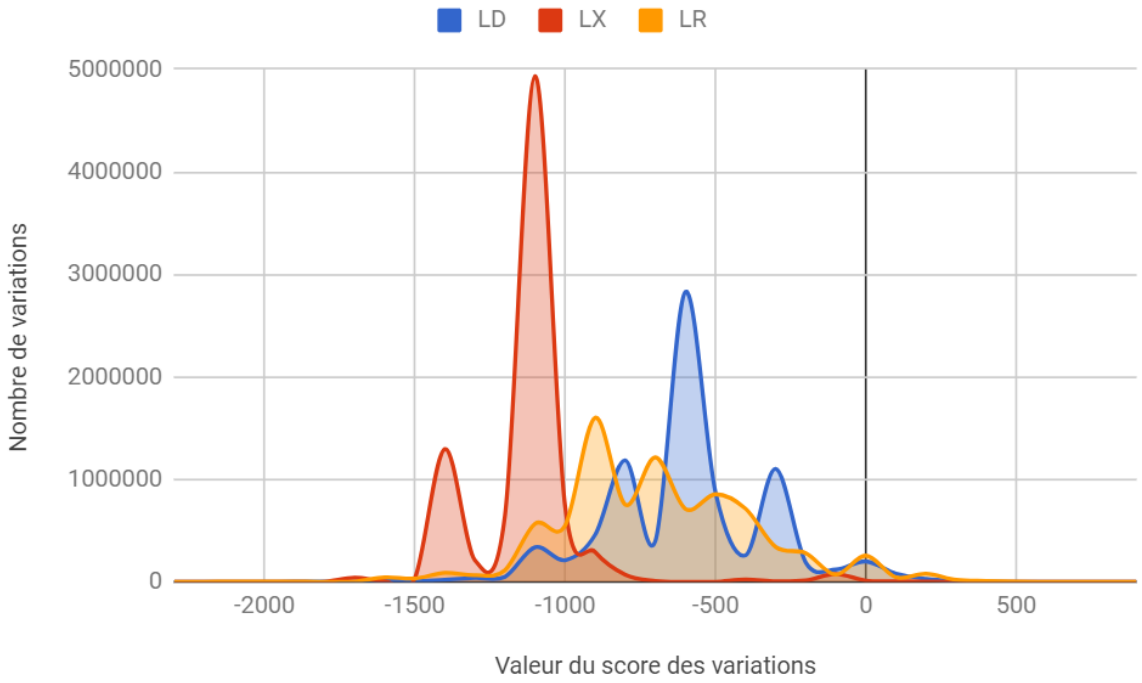


FIGURE 5.22 – Profil des *rulesets* de l'utilisateur *L.* LD : *L.* Dominant, LX : *L.* sur le chromosome X, LR : *L.* récessif

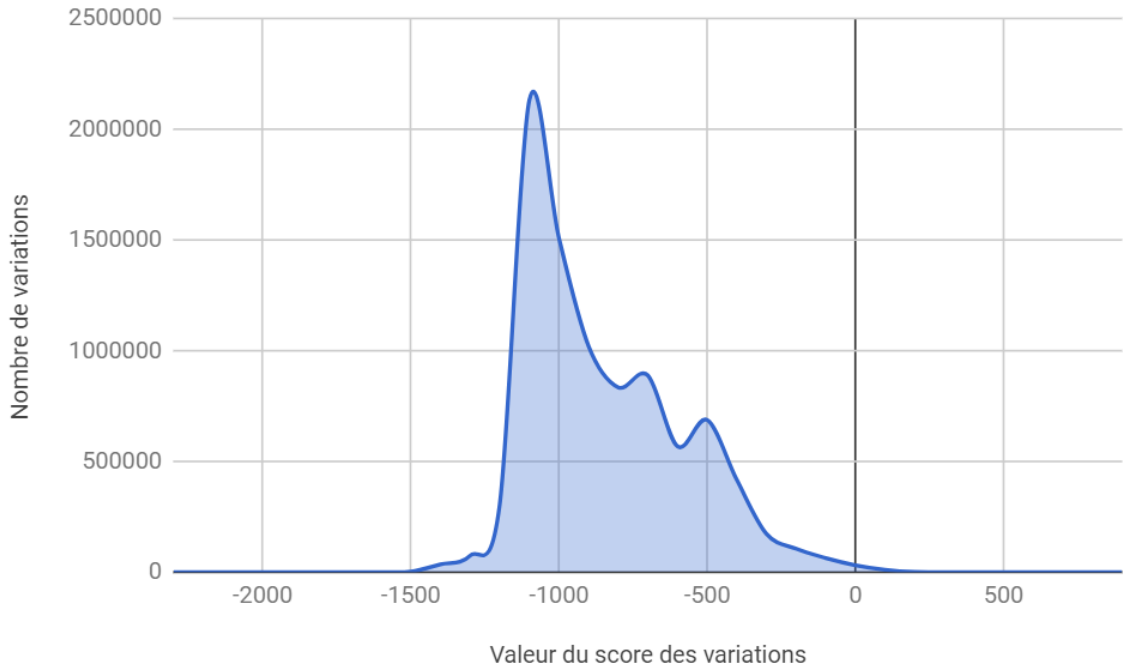


FIGURE 5.23 – Profil du *ruleset* de l'utilisateur *T.*

5.7.2.9 Conclusion

Ces premières observations permettent de constater la grande disparité des approches entre les différents utilisateurs. Cette disparité correspond à nos hypothèses, elle rend cependant très complexe la comparaison des scores les uns par rapport aux autres, ce qui est d'autant plus marqué lorsque que le nombre d'échantillons est réduit. La première confrontation des *rulesets* aux données dans la partie 5.7.2.7 permet de constater que ces différences d'approches se

retrouvent aussi sur les valeurs associées à chaque variation. Il est en revanche intéressant de noter que notre système permet à chacune de ces approches d'obtenir des résultats pertinents quelles que soient leurs différences. Cela démontre une certaine robustesse tant au niveau du prototype que de la méthode en elle-même.

Ces constatations permettent aussi de cerner les approches choisies par les utilisateurs. En effet, en dépit de l'évidente disparité, on peut noter des points communs, avec notamment 9 critères utilisés par tout le monde. Par ailleurs, l'ensemble des utilisateurs ont allié des pondérations positives et négatives, pour filtrer et prioriser simultanément. En effet, tous les utilisateurs utilisent pleinement les deux axes, reléguant les variations non souhaitées à des rangs plus lointains et rapprochant les variations intéressantes du haut de la liste. Cette approche est nuancée chez certains utilisateurs, par exemple, l'utilisateur *F.* qui a une approche orientée priorisation, utilisant plus de pondérations positives ce qui corrèle avec l'utilisation d'un plus grand nombre de règles et de critères pour capter un maximum d'indices de pathogénicité. À L'inverse, l'utilisateur *L.* a choisi une stratégie plus portée sur le filtre, avec principalement des pondérations négatives fortes pour éliminer le maximum de variations, tout en se concentrant sur des critères connus et maitrisés pour les pondérations positives.

L'analyse des *rulesets* des utilisateurs montre la diversité d'approche employée par les généticiens dans leur processus d'analyse. Cependant, pour vérifier à la fois les performances de notre approche, ainsi que celles des utilisateurs, il faut étudier les résultats concrets des *rulesets* en conditions réelles en analysant leur efficacité dans la priorisation des variations pathogènes.

5.7.3 Efficacité

Dans notre contexte, l'efficacité de *GenSCor* se mesure en vérifiant sa capacité à retrouver des variations pathogènes connues, précédemment identifiées.

5.7.3.1 Résultats

Nous pouvons analyser les résultats sous différents aspects. Nous allons commencer par identifier les meilleurs rangs, mais aussi les rangs moyens, la proportion de variations parmi les trois premiers rangs, enfin, nous pouvons comparer les résultats des différents utilisateurs entre eux et comparer l'outil à la bibliographie. La liste des variations pathogènes ainsi que les rangs associés des utilisateurs sont présentés dans le tableau récapitulatif 5.7.

5.7.3.1.1 Analyse des rangs extrêmes L'analyse des meilleurs rangs sur l'ensemble des variations pathogènes est simple, les meilleurs rangs sont identiques chez tous les utilisateurs, puisqu'ils trouvent au moins une fois la variation pathogène au premier rang. En revanche, il est intéressant de s'attarder sur les cas plus complexes, notamment les plus mauvais rangs de chaque ensemble de *rulesets*, puisqu'ils donnent une indication sur la façon dont sont manquées certaines variations. On n'analyse ici que les performances d'un seul *ruleset* par utilisateur, celui correspondant au mode de transmission de la variation.

On peut ainsi constater que les variations pathogènes les moins priorisées par les *rulesets* sont très éloignées dans le classement, jusqu'à la $1688^{ème}$ position. Les différents utilisateurs ont cependant des résultats assez contrastés, puisque ces variations manquées restent dans les 100 premières pour un utilisateur, alors qu'elles ne sont pas dans les 1 000 premières pour un autre. Enfin, il est très intéressant de noter que 3 des 4 *rulesets* présentent leur moins bonne performance sur la même variation. Ce plus mauvais rang n'est donc pas révélateur de la performance globale du score. Lorsque l'on prend du recul, en observant les 3 variations les moins bien priorisées pour chaque ensemble de *rulesets*, comme présenté dans le tableau 5.8, on peut constater que la dernière variation est clairement à part. Les rangs correspondant aux mêmes variations sont colorés dans la même couleur, l'absence de couleur indiquant une

Patient	Gène	Type	Variation	Rang *L.*	Rang *G.*	Rang *F.*	Rang *T.*
27A	DYNC1H1	Faux sens	c.3188T>C *p.Met1063Thr*	3	1	1	62
27B	MECP2	Décalage CL	c.842delG *p.Gly281AlafsTer20*	1	1	1	1
27C	HDAC8	Faux sens	c.473C>A *p.Ala158Asp*	3	1	1	8
29A	CACNA1A	Faux sens	c.1762C>T *p.Arg588Cys*	3	1	1	18
30A	FRMPD4	V. d'épissage	c.3965-2A>C	1	1	2	2
32A	AUTS2	Non sens	c.901C>T *p.Gln301Ter*	1	1	1	3
32B	CTCF	V. d'épissage	c.782-2A>G	3	28	2	1
33A	TAF8	V. d'épissage	c.903-1G>A	7	1688	330	166
34A	DLG4	Non sens	c.1978C>T *p.Arg660Ter*	1	1	1	4
34A	QARS	Faux sens	c.134G>T *p.Gly45Val*	1	9	1	2
34A	QARS	Faux sens	c.727T>G *p.Tyr243Asp*	73	7	12	27
35A	CHRDL1	Non sens	c.652C>T *p.Gln218Ter*	1	1	1	1
35B	BCL11A	Décalage CL	c.599_602delAAAG *p.Glu200AlafsTer7*	2	55	1	4
36A	CACNA1F	Décalage CL	c.3360_3361delCA *p.Ile1121HisfsTer31*	1	1	1	3
36B	KIF11	V. d'épissage	c.2548-1G>A	1	41	1	1
36C	EP300	Décalage CL	c.5571_5578delACCAACTG *p.Gly1860AlafsTer20*	3	1	8	15
37A	PURA	Décalage CL	c.648delC *p.Glu217SerfsTer8*	2	1	9	13
37B	HDAC8	Faux sens	c.908G>A *p.Gly303Glu*	1	1	1	8
38A	F8	Faux sens	c.2150G>T *p.Arg717Leu*	1	1	1	26
38B	PTPN11	Faux sens	c.853T>C *p.Phe285Leu*	1	1	1	1
38C	KIF11	V. d'épissage	c.2548-1G>A	1	23	1	2
38D	EP300	Décalage CL	c.5571_5578delACCAACTG *p.Gly1860AlafsTer20*	1	1	6	6
38E	SETD2	Décalage CL	c.4043delA *p.His1348LeufsTer25*	7	1	10	4
38F	CDK13	Faux sens	c.2525A>G *p.Asn842Ser*	1	1	2	1
38G	AUTS2	Non sens	c.376C>T *p.Arg126Ter*	1	1	1	1
39A	CHD3	Non sens	c.645C>A *p.Tyr215Ter*	2	1	1	15
39B	SETD1	Faux sens	c.4876C>T *p.Arg1626Trp*	3	1	1	50
40A	NAA10	Faux sens	c.206A>C *p.His69Pro*	1	1	1	6
41A	SIN3A	Non sens	c.1675C>T *p.Arg559Ter*	1	1	1	1

TABLEAU 5.7 – Tableau récapitulatif des variations pathogènes associées aux rangs obtenus par les utilisateurs. "Décalage CL" signifie décalage du cadre de lecture, "V. d'épissage" désigne une altération d'un site d'épissage.

variation unique dans le tableau. On peut y voir notamment que les 3 variations les moins bien priorisées sont les mêmes pour deux utilisateurs.

Rang	*T.*	*G.*	*L.*	*F.*
Antépénultième rang	50	41	7	10
Pénultième rang	62	55	7	12
Dernier rang	166	1688	73	330

TABLEAU 5.8 – Tableau résumant les 3 derniers rangs de chaque *rulesets*, les couleurs identiques désignent des variations identiques

On constate que les rangs d'une variation pour les différents utilisateurs sont liés, puisque ce sont les trois mêmes variations qui occupent les trois plus mauvais rangs des utilisateurs *F.* et *L* (correspondants aux variations bleue, verte et orange dans le tableau 5.8. Les utilisateurs *T.* et *G.* n'ont en commun qu'une seule de ces trois variations. D'une manière globale, on peut constater que certaines variations sont plus délicates à faire ressortir que d'autres, ce qui est cohérent vis-à-vis du mécanisme d'annotation. Cependant, les différentes approches choisies par les utilisateurs n'ont pas toutes les mêmes forces et faiblesses. Ainsi, l'antépénultième variation de l'utilisateur *L.* sort en rang 1 chez l'utilisateur *G*, alors qu'à l'inverse, l'antépénultième variation de l'utilisateur *G.* sort en rang 1 chez l'utilisateur *L.*.

5.7.3.1.2 Rang moyen et médian Si l'on se concentre sur les rangs moyens du tableau récapitulatif 5.9, on pourrait en déduire que l'ensemble le plus efficace est celui de l'utilisateur *L.* devant respectivement ceux des utilisateurs *F.*, *T.* et enfin *G.*. Cependant, comme nous l'avons vu précédemment, il est plus parlant de calculer la moyenne en écartant les valeurs aberrantes, sans les trois plus mauvais résultats, ce qui donne des résultats très différents : le rang moyen est toujours le plus intéressant chez l'utilisateur *L.*, suivi désormais par les utilisateurs *F.* puis *G.*, et enfin *T.*

	Utilisateur *T.*	Utilisateur *G.*	Utilisateur *L.*	Utilisateur *F.*
Rang moyen	15,58	64,58	4,41	14,06
Rang moyen sans les 3 dernières valeurs	6,69	3,42	1,57	2,15
Rang médian	4	1	1	1

TABLEAU 5.9 – Tableau résumant les rangs médians et moyens des utilisateurs sur notre cohorte de variations pathogènes

On retrouve ces aspects en calculant la médiane qui est un outil statistique plus parlant en cas de données aberrantes, puisqu'avec ces résultats on observe que le rang médian est 1 pour trois utilisateurs, et seulement 4 pour l'utilisateur *T.*. Ces valeurs de moyennes corrigées ainsi que de médiane permettent de bien représenter l'efficacité d'un ensemble de *rulesets*.

5.7.3.1.3 Nombre de variants dans le top 3 Comme présenté dans la section précédente, les moyennes corrigées ainsi que les médianes permettent de présenter globalement l'efficacité. Pour plus de précisions, nous avons relevé le nombre de variations présent parmi les 3 premières variations, ces résultats sont présentés dans la figure 5.24.

Nous pouvons constater sur ce graphique que le *ruleset* de l'utilisateur *T.* semble présenter de moins bonnes performances que les 3 autres. Ces résultats permettent également de

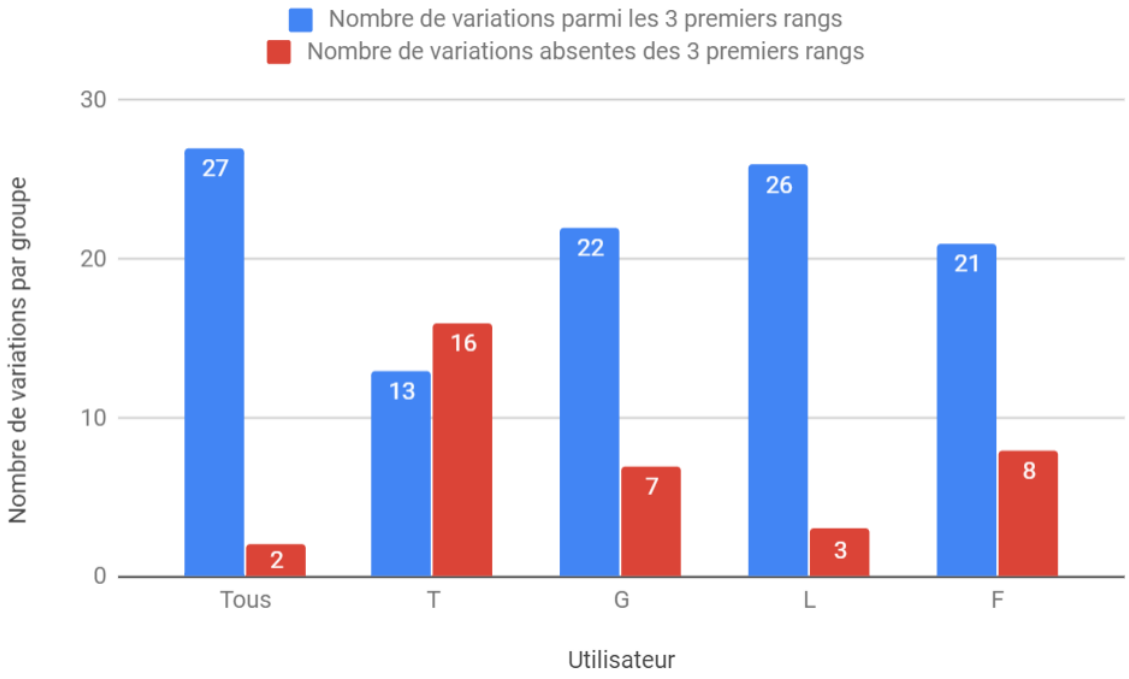


FIGURE 5.24 – Positionnement des variations pathogènes parmi les premiers rangs

constater la complétion entre les approches des différents utilisateurs, puisque la combinaison des ensembles est meilleure que chacun des ensembles. Les variations qui ne sont pas placées dans le top 3 pour la combinaison des *rulesets* sont la variation bleue et la variation verte présentées dans le tableau 5.8 et qui seront détaillées dans la section 5.7.3.2.

5.7.3.1.4 Comparaison des *rulesets* entre eux Pour une comparaison plus directe des approches employées par les utilisateurs, on peut regarder laquelle obtient les meilleurs scores en analysant le nombre de fois pour chaque approche où elle est la meilleure, la moins bonne, ou enfin entre les deux. Ces résultats sont présentés dans la figure 5.25. Les égalités sont considérés comme des *ex aequo* et comptent donc toutes les deux dans la catégorie la plus élevée.

On peut constater sur ces figures que l'approche de l'utilisateur *T.* semble ici aussi moins performante que les autres, puisqu'elle est le moins souvent la plus efficace et le plus souvent la moins efficace. Ce point est discuté dans la section suivante plus en détail. Pour distinguer les trois autres approches, il faut préciser les critères. En effet, si l'on considère la meilleure approche comme étant celle classant le plus de fois la variation d'intérêt au meilleur rang, alors l'approche de l'utilisateur *G.* est la meilleure. Cependant, cette approche est aussi celle qui a les moins bons résultats si l'on regarde le nombre de fois où l'approche a été la plus mauvaise. Si l'on considère cette définition, c'est alors l'approche de l'utilisateur *L.* qui peut être considérée comme la meilleure et la plus fiable.

5.7.3.1.5 Comparaison des résultats par rapport à *QueryOR* De la même manière que *GenSCor*, *QueryOR* fonctionne *via* les règles de priorisation saisies par les utilisateurs. Nous ne pouvons transposer les règles de nos utilisateurs pour deux raisons majeures : les fonctionnalités sont très différentes, il est par exemple impossible de traduire les règles multiples ou les règles diminuant le score qui représentent la majorité des règles saisies. De plus, les annotations disponibles sont également très différentes, la plupart des critères choisis par nos utilisateurs ne sont pas dans l'annotation proposée par *QueryOR*. Afin d'avoir une mesure objective de performance, nous avons utilisé un ensemble de règles proposé par les

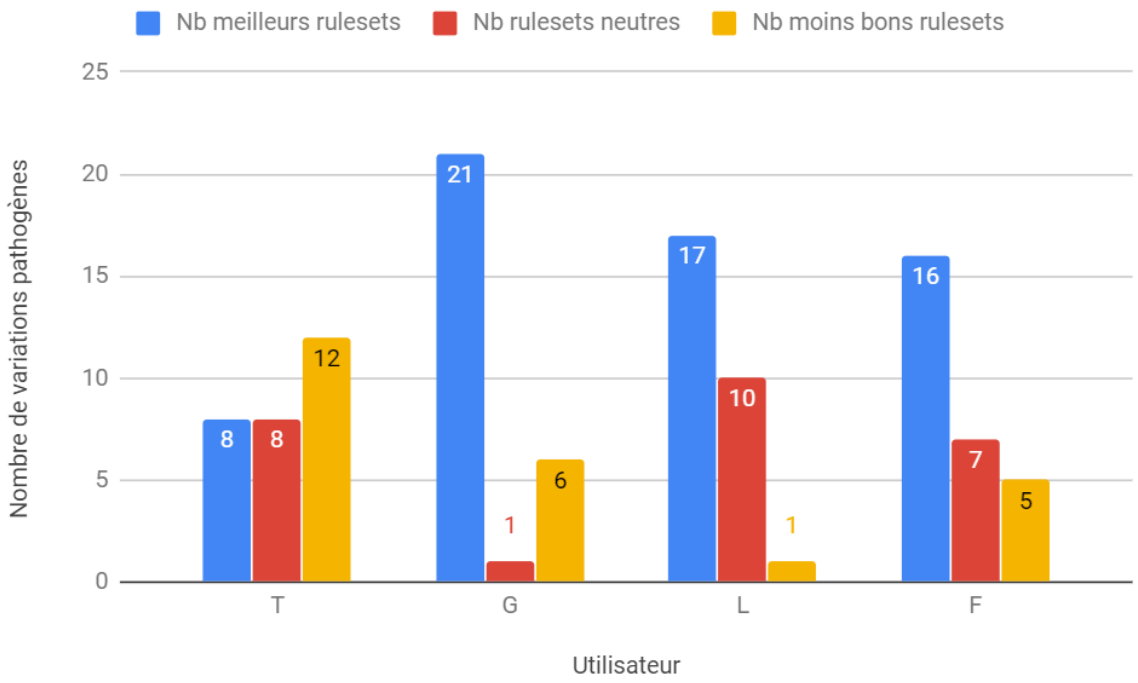


FIGURE 5.25 – Comparaisons des approches utilisées par les différents utilisateurs

développeurs pour cibler les variations rares et pathogènes, ce qui correspond parfaitement à notre cas d'utilisation. Nous avons testé un sous échantillon représentatif des variations pathogènes de notre cohorte et calculé leur rang. Les résultats sont présentés dans le tableau 5.10.

Rang moyen	982
Rang médian	791
3 Meilleurs rangs	1, 29, 36
3 Moins bons rangs	1393, 2254, 4452

TABLEAU 5.10 – Tableau résumant les rangs obtenus avec *QueryOR* sur notre *corpus* de variations pathogènes

Deux remarques sont à ajouter à ces résultats. Pour l'un des fichiers d'analyse d'exome, *QueryOR* retourne une erreur lors de l'annotation, il a donc été impossible d'analyser ce fichier. Ce premier point est un problème très important, car il est inenvisageable de ne pas analyser les données d'un patient. De plus, le seul rang 1 obtenu est de façon surprenante sur une variation du gène *QARS* (dont l'atteinte est sur le mode autosomique récessif). Elle n'aurait cependant pas été retenue, car la seconde variation n'apparait pas dans les résultats, alors qu'elle est bien présente dans le fichier VCF chargé sur le site.

Les performances de *QueryOR* semblent donc très inférieures à celles de *GenSCor* sur notre cohorte puisqu'aucune des variations pathogènes n'auraient pu être extraites en utilisant cet outil et le jeu de paramètres par défaut proposé par ses auteurs.

5.7.3.2 Discussion

Avant de chercher à conclure sur l'efficacité des *rulesets* des différents utilisateurs et donc sur l'efficacité de notre approche et de notre prototype, il faut chercher à expliquer les données extrêmes ou aberrantes rencontrées, c'est-à-dire les variations donnant des résultats très différents des autres.

5.7.3.2.1 Différence d'approche de l'utilisateur *T.* La maîtrise du biais de rang détaillé dans la section 5.7.3.2.4 permet de ne pas surévaluer les performances des *rulesets*, cependant, ce biais pénalise tout de même les résultats de l'utilisateur *T.*, qui a groupé toutes ses approches dans un seul *ruleset*. En effet, les meilleurs résultats des différentes approches sont rassemblés, ce qui tend à augmenter le rang de la variation d'intérêt. Cet utilisateur *T.* a une utilisation différente de l'outil, qui induit des métriques difficilement comparables avec celles des autres utilisateurs. Il envisage un nombre plus important de variations mais ne les analyse pas toutes en détail, il s'intéresse ainsi à des valeurs de scores assez basses, comme nous pourrons en discuter dans la section suivante.

5.7.3.2.2 Données aberrantes Comme nous l'avons constaté dans la section résultats 5.7.3.1.1, plusieurs variations ont été très mal priorisées par les *rulesets* des utilisateurs. Trois cas sont intéressants à étudier. Le premier est le cas de la variation colorée en bleu dans le tableau 5.8, celle qui a été très difficile à trouver pour tous les utilisateurs. Il s'agit en effet d'un cas très complexe à faire ressortir, puisque c'est une variation homozygote sur le gène TAF8 (*c.903-1G>A*) qui crée un site accepteur d'épissage. Les outils classiques, tels que les fréquences alléliques ou le score *VaRank* ne sont pas ici des bons indicateurs, la variation étant présente plusieurs fois dans les bases de données à l'état hétérozygote, notamment à 10 reprises dans *gnomAD*. Il était cependant possible de la faire ressortir, puisque l'utilisateur *L.* la classe $7^{ème}$ dans son *ruleset* dédié aux variations récessives. Ce classement est en partie dû à la pondération forte de l'haplotype "homozygote", ce qui n'est pas le cas chez les autres utilisateurs. Cet exemple permet de montrer à quel point il est important de couvrir les différents cas possibles et de présenter collégialement les variations retenues.

Il faut néanmoins rappeler qu'avec la stratégie particulière de l'utilisateur *T.*, en n'ajoutant un nombre qui n'est pas un multiple de 10 en cas de variation homozygote, et en dépit du rang lointain de la variation sur le gène *TAF8*, elle ressort ainsi relativement facilement, attirant l'œil et permettant son analyse même si elle n'est pas bien priorisée. Ainsi, si les variations bien priorisées seront analysées de façon certaine, celles qui présentent un plus mauvais rang peuvent l'être également, car des stratégies particulières ou l'appui du descriptif clinique du patient peuvent permettre de surmonter un rang inadapté en faisant ressortir la variation aux yeux de l'utilisateur.

La seconde source de données aberrantes est la paire de variations récessives sur le gène *QARS* (*p.Gly45Val* et *p.Tyr243Asp*). Cette paire a globalement posé beaucoup moins de problème aux utilisateurs, les différents couples de rang sont 2 :27 , 9 :7, 1 :73 et 1 :12. La première variation était annotée "Pathogène" par ClinVar ce qui permet de la faire ressortir relativement facilement. La seconde était plus complexe, reprenant les difficultés de la variation localisée sur *TAF8*, avec une fréquence allélique moins utile qu'à la normale. Cependant, même si les résultats présentent cette paire comme difficile à mettre en évidence, dans les faits, elle a été trouvée rapidement par les généticiens, qui ont simplement recherché d'autres variations dans le gène *QARS* puisqu'il se classe en rang 1 ou 2 pour trois utilisateurs. Pour le quatrième utilisateur, le processus était également simple, les deux variations étant situé dans le top 10. Cet exemple illustre la difficulté pour la mise en évidence des variations récessives et la nécessité de classer au moins une des variations dans les premières variations.

Enfin, le troisième et dernier exemple ne concerne qu'un utilisateur, *G.*. Il s'agit d'une variation dans le gène *KIF11* (*c.2548-1G>A*) qui réalise 2 de ses 5 plus mauvais rangs, alors que cette variation est trouvée en rang 1 par tous les autres. Il s'agit ici d'une surestimation

de la fréquence allélique comme paramètre de tri et de seuil trop stringeant, mettant plus en avant les variations qui ne sont pas répertoriées dans aucune des bases de données contrôles ou locales. Or, biologiquement, il s'agit d'une analyse en trio, les deux parents plus l'enfant, mais la mère était également atteinte de la pathologie. L'allèle est donc vu 2 fois dans notre base de données locale, une fois chez la mère et une fois chez l'enfant. Ce point peut-être considéré comme un biais lié à notre annotation, mais la prise en compte des données de fréquences alléliques locales étant une pratique courante, cela implique que la gestion des apparentés soit traitée plus finement. Ce dernier exemple permet de montrer l'importance de lier le cas à ses indications cliniques, qui peuvent changer ponctuellement la façon de définir les règles. C'est également un argument de plus pour une analyse qui doit être la plus adaptable possible par l'utilisateur.

5.7.3.2.3 Valeurs cumulées Lors de la présentation des résultats, les valeurs ont également été présentées en prenant en compte les *rulesets* des 4 utilisateurs. Ce cumul correspond à la réunion d'harmonisation entre les différents généticiens permettant de décider collégialement des variations dont l'étude est à approfondir. Cette réunion existait bien entendu avec la méthode traditionnelle, elle existe toujours avec *GenSCor* sur les données prospectives qui continuent d'être générées. Cependant, pour ces analyses rétrospectives, elle est simulée par la mise en commun des *rulesets* et des résultats, avec notamment l'étude du meilleur rang tout utilisateur confondus.

5.7.3.2.4 Biais potentiels Le premier biais est méthodologique, il est lié à notre approche de test. En effet, nous ne testons ici que les variations pathogènes déjà connues, ce qui ne nous permet pas de conclure sur des éventuelles variations qui auraient été manquées avec les approches traditionnelles et qui seraient ressorties avec l'approche *GenSCor*. Les données issues des analyses d'exomes rendus négatifs ont été réanalysées par les généticiens et des pistes intéressantes sont en cours d'étude. Cependant, le très long délai entre l'analyse des données d'exomes et le rendu concret du résultat, qui est de plus d'un an, nous empêche d'inclure ce type de résultats dans ce travail.

Il existe plusieurs biais liés au rang. Le premier est dû à l'emploi de plusieurs *rulesets* pour chaque utilisateur, ce qui amène donc à plusieurs variations de rang 1. Ce biais était anticipé et maitrisé en ne considérant que le rang dans le *ruleset* dédié à ce type de variation. Nous avons constaté *a posteriori* que cette gestion n'était pas utile, car les *rulesets* dédiés étaient dans tous les cas ceux qui fournissaient le meilleur rang (potentiellement *ex aequo*). Un second biais de rang est lié à la présence de deux variations chez un seul patient, ce qui diminue forcément la plupart des valeurs. Nous avons laissé ce biais actif parce qu'il ne peut pas faire surestimer les résultats et que son impact est minime.

5.7.3.3 Conclusion sur l'efficacité

Cette étude permet de montrer que cette approche est au moins aussi efficace que l'approche manuelle, puisque les variations pathogènes sont retrouvées et classées parmi les premières variations. En cumulant les résultats des différents utilisateurs, toutes les variations sont classées par au moins un des utilisateurs parmi les 7 premières variations. Seules 2 variations sur 29 ne sont pas dans les 3 premières, il s'agit de deux variations récessives complexes à prioriser qui sont toutes deux classées $7^{èmes}$.

Des études sont en cours pour montrer le gain d'efficacité potentielle sur les variations manquées par la méthode traditionnelle. Cette première étude a néanmoins permis de montrer que tous les *rulesets* ne se valent pas, celui de l'utilisateur *T.* étant moins performant dans plusieurs cas. Enfin, elle permet également de montrer l'intérêt d'une concertation entre les spécialistes au moment de déterminer les variations candidates, car l'ensemble des *rulesets* donne des résultats cumulés nettement meilleurs que ceux obtenus individuellement par chaque utilisateur. Ces résultats s'expliquent par les différences d'approches des utilisateurs,

produisant des *rulesets* avec des points forts et des points faibles différents, se complétant mutuellement.

5.7.4 Efficience

L'efficience est divisée en deux aspect dans notre cadre d'étude : le temps passé pour analyser l'exome d'un patient et la charge mentale qu'une telle analyse représente pour le généticien.

5.7.4.1 Temps d'analyse

Comme nous l'avons présenté dans la partie méthodologie, nous avons concentré notre analyse du temps sur le temps machine ainsi que sur le temps de mise en place d'un premier score.

Sur ce dernier point, nous avons observé trois utilisateurs lors de leur prise en main du prototype et de la définition de leur premier score. Les temps sont très variables, 27 minutes, 19 minutes et 12 minutes. Cependant la validité de ces valeurs n'est pas satisfaisante, car les *rulesets* mis en place lors de ces sessions chronométrées ne sont pas les mêmes que les *rulesets* terminaux, qui ont subi de nombreuses mises à jour depuis. Ainsi, les *rulesets* utilisés pour les tests vont jusqu'à la $13^{ème}$ version. Ces données sont donc considérées comme inexploitables et ne seront pas analysées plus avant.

En revanche, le temps pris par le logiciel pour calculer les scores peut-être analysé. Chaque condition prend environ 0.4 seconde à être testée sur une analyse d'exome moyenne contenant 55 000 variations sur un disque dur de type SSD. Comme les *rulesets* sont constitués en moyenne de 45 conditions, leur application prend autour d'une vingtaine de secondes par patient et par *ruleset*. L'étape de pré-analyse prend au total 4 minutes pour une série classique de 12 patients. Nous avons cependant obtenu des temps 3 à 4 fois supérieurs pour des configurations de machines plus classiques, avec une fréquence moindre du processeur et des disques durs fonctionnant avec des technologies classiques. Ce temps ne prend pas en compte le temps de développement du *rulesets* ni celui de l'analyse réelle des variations les mieux classées.

5.7.4.2 Notion de R50

Nous avons observé chez différents utilisateurs l'usage d'une valeur de score seuil, en-dessous de laquelle ils n'analysaient pas les variations. Ces valeurs sont évidemment différentes d'un utilisateur à l'autre et sont spécifiques à chaque *ruleset*. Nous avons constaté que ces valeurs seuils définies empiriquement par les utilisateurs correspondent environ à la moitié du score maximum possible du *ruleset*. Pour vérifier la validité de cette méthode, nous avons calculé le rang à 50 % du meilleur score, que l'on appelle R50, pour chaque variation. Nous avons ensuite regardé pour chacune des variations pathogènes si la valeur de son rang était inférieure à celle du R50. Les résultats sont présentés dans la figure 5.26. L'objectif est de trouver un indicateur permettant d'obtenir un seuil plus neutre que celui utilisé empiriquement par l'utilisateur. Il est cependant important de maintenir un lien entre l'indicateur et la valeur réellement utilisée. On peut constater sur cette figure que les utilisateurs *F.* et *L.* ont tout deux une variation en dehors du R50, il s'agit pour le premier de la variation sur le gène *TAF8* et pour le second d'une des deux variations sur le gène *QARS*. Étant donné que ces deux variations sont manquées, on peut considéré l'indicateur fiable pour ces deux utilisateurs. Concernant l'utilisateur *G.* deux variations sont manquées, celle sur *TAF8* ainsi que la variation sur le gène BCL11A. Ces deux variations sont également considérées manquées étant donné leur classement, le R50 peut ici aussi être considéré fiable et représentatif de ce qu'analyse réellement l'utilisateur. Enfin, concernant l'utilisateur *T.*, on constate que le R50 semble être un indicateur inutilisable, puisque près de la moitié des variations pathogènes sont en dehors. Cependant, ce problème peut être imputable à un mécanisme de *ruleset* particulier.

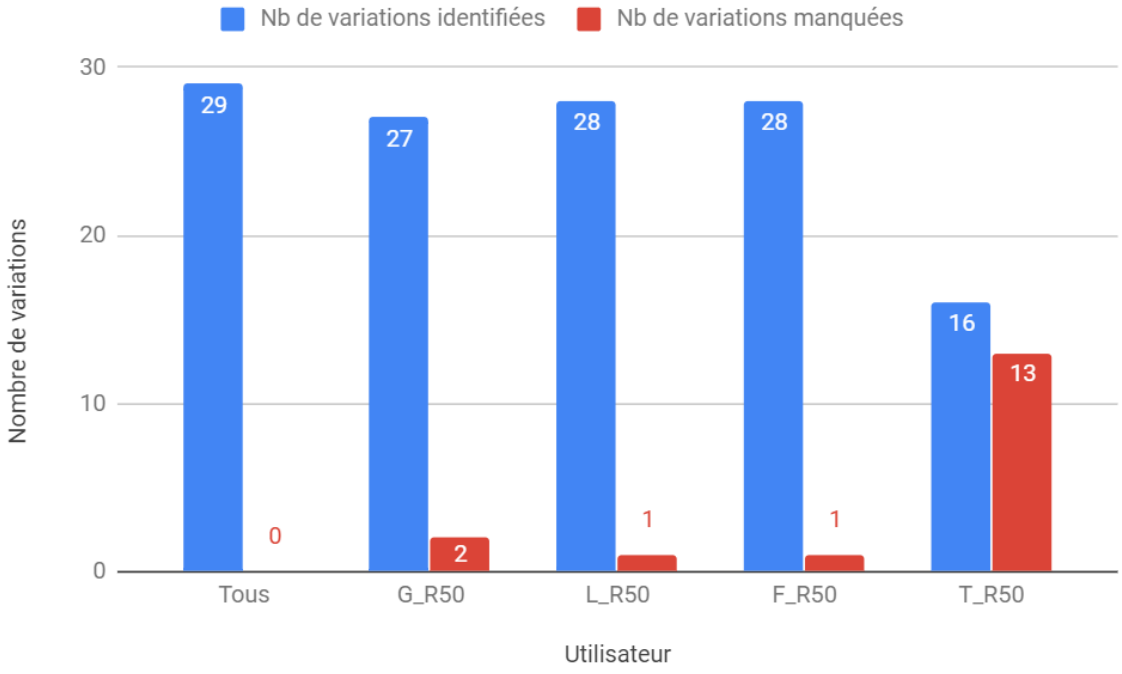


FIGURE 5.26 – Comparaisons des R50 des différents utilisateurs

En effet, la distribution des variations avec des hauts scores est particulière, indiquant un très faible nombre de variations à hauts scores, s'expliquant par la présence de peu de règles à fort coefficient. On constate de plus des valeurs très faibles de R50. Enfin, empiriquement, le seuil utilisé par cet utilisateur était plus bas que le R50 calculé. Basé sur ces informations, nous avons calculé un nouveau seuil pour cet utilisateur, plus proche de celui qu'il utilise empiriquement, à hauteur de 25% du meilleur score. Ce nouveau seuil est beaucoup plus cohérent avec les valeurs des autres utilisateurs, comme on peut le voir dans la figure 5.27, c'est donc ce seuil qui sera utilisé pour cet utilisateur afin de comparer la charge mentale.

5.7.4.3 Charge mentale

Nous avons précédemment défini la notion de R50 et constaté que les variations pathogènes sont très majoritairement comprises dans ce seuil. Nous utiliserons dans cette partie le terme R50 pour parler de l'indicateur, même si dans les faits le R50 de l'utilisateur *T.* est un R25. Cet indicateur permet d'estimer la quantité de variations qui sont réellement à analyser, ce qui permet d'approximer la charge mentale de l'utilisateur. Dans cette partie, les valeurs sont additionnées pour les utilisateurs disposant de plusieurs *rulesets*, puisque chacun correspond à une approche de recherche différente, les variations des différents *rulesets* sont analysées.

Le détail des R50 moyens par *ruleset* est présenté dans la figure 5.28, les valeurs cumulées par utilisateur sont présentées dans la figure 5.29. On constate dans le détail un effet très marqué de l'approche, les *rulesets* dédiés au chromosome X présentent des nombres de variations bien inférieurs aux autres, ce qui est cohérent avec l'approche, puisque beaucoup moins de variations sont à analyser.

Les valeurs cumulées par utilisateur correspondent aux nombres de variations que l'utilisateur doit interpréter pour être exhaustif dans son analyse. L'ensemble de *rulesets* le plus performant est celui de l'utilisateur *L.*, qui doit analyser une moyenne de 37 variations pour être exhaustif dans son analyse, soit 0,05 % des variations d'un exome. Le moins performant concernant le R50 est l'ensemble de l'utilisateur *G.*, dont le R50 moyen est à 130 variations, ce qui correspond à 0,26 % des variations d'un exome. Il est intéressant de noter que l'utilisateur *G.* utilise un seuil empirique plus faible, correspondant à environ 40 % du meilleur score.

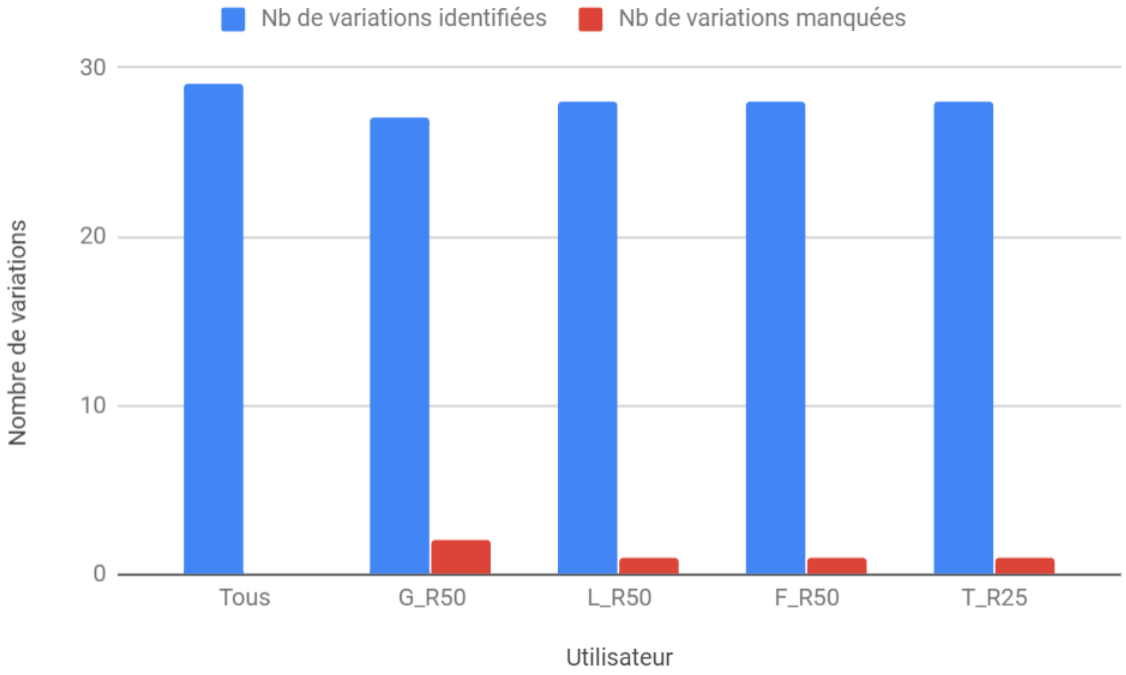


FIGURE 5.27 – Récapitulatif des performances des indicateurs de seuils utilisés pour chaque utilisateur.

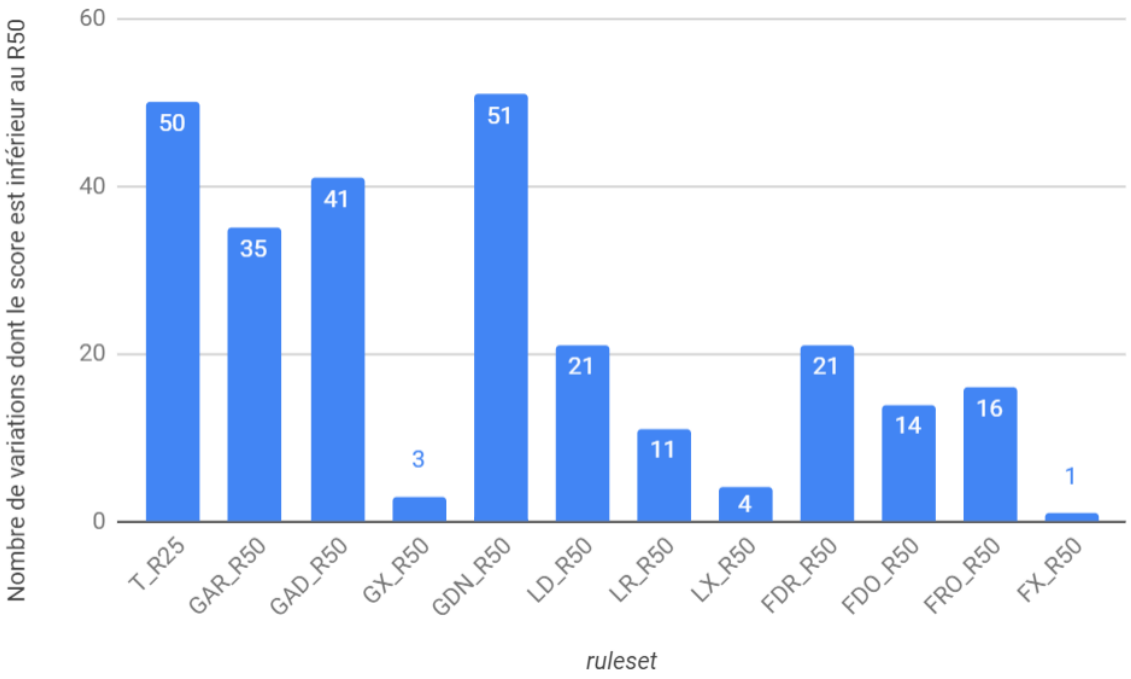


FIGURE 5.28 – R50 moyen pour chaque *ruleset*

L'utilisation de ce seuil permet de passer à 114 variations à analyser, ce qui est plus proche de la quantité réellement analysée par cet utilisateur. Ce changement de seuil ne change bien sûr pas le nombre de variations hors de l'indicateur, elles sont toujours au nombre de deux, sur les gènes *TAF8* et *BCL11A*.

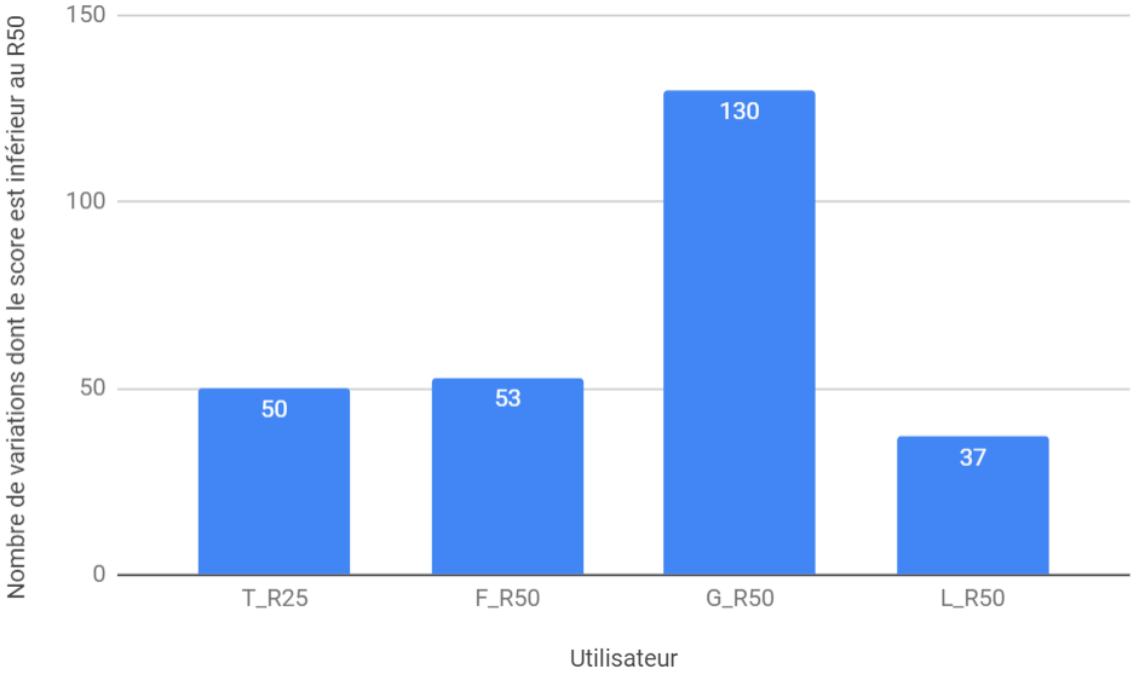


FIGURE 5.29 – R50 moyen cumulé par utilisateur

5.7.4.4 Discussion

Nous avons rencontré plusieurs problèmes lors de l'analyse de l'efficience, qui ont limité notre capacité à acquérir des données et à évaluer l'approche. En premier lieu, l'absence de mesure concernant le temps pris par l'analyse des variant est un manque certain. Cependant, l'autoévaluation par les utilisateurs, si elle manque de validité scientifique concernant le temps réellement pris, est un indicateur intéressant. En effet, elle nous permet de constater un ressenti moins contraignant avec le prototype, ce qui est encourageant concernant son utilisation à long terme et son utilisabilité. Ces points sont également renforcés par le fait que les 4 utilisateurs continuent d'utiliser le prototype dans leurs analyses de routine.

Nous avons également constaté l'intérêt et les limites du R50. Son intérêt est évident, il permet une comparaison de l'efficience des *rulesets*, cependant, nous avons observé ses limites en terme d'adaptabilité, en particulier sur le *ruleset* de l'utilisateur *T.*. Le besoin d'un indicateur est important pour l'évaluation des scores, le R50 peut être utilisé comme référence s'il est corroboré par les seuils empiriques des utilisateurs, mais il est capital de vérifier sur un échantillon de données pathogènes connues son efficacité, comme nous l'avons fait. Il peut alors être nécessaire de changer le seuil pour adapter au mieux l'indicateur au *ruleset*.

5.7.4.5 Conclusion sur l'efficience

Nous avons démontré que l'utilisation d'un seuil est un indicateur valide pour étudier la quantité de variations à analyser. La valeur de ce seuil est cependant à préciser en fonction de l'approche, même si 50 % du meilleur score est une approximation valable dans la majorité des cas. La corrélation avec les pratiques empiriques des utilisateurs effectuant l'analyse est importante, puisqu'elle permet de rester proche des valeurs actuellement utilisées et de ne pas biaiser les conclusions.

L'étude de cet indicateur nous permet de conclure à une très nette amélioration de l'efficience au niveau de la charge mentale, la quantité de variations à analyser par exome étant très fortement réduite. En moyenne, les utilisateurs analysent 75 variations par exome, ce qui correspond à environ 0,12 % des variations présentes dans un exome. Cette quantité permet

néanmoins de rester confiant dans les résultats et d'assurer une certaine exhaustivité même en cas de résultat négatif si les scores sont bien conçus et l'analyse discutée collégialement. Cela permet aussi de ne faire qu'une seule analyse par patient, sans avoir besoin d'écarter les filtres au fur et à mesure en cas d'analyse négative, le temps d'analyse devrait donc en être réduit.

La mesure du temps restant incomplète, nous ne pouvons affirmer qu'il y ait effectivement un gain de temps. Le ressenti des utilisateurs semble l'indiquer, mais des études plus approfondies sont nécessaires pour le démontrer et le quantifier. Nous pouvons en revanche constater un temps machine nécessaire faible, de quelques secondes par fichier. Ce temps se situe dans les standards des outils de la bibliographie, d'autant plus qu'il n'y a pas ici de temps de chargement à ajouter.

5.7.5 Résultats rétrospectifs et prospectifs

En plus de l'étude principale sur la cohorte que nous avons décrite, nous avons obtenu des données prospectives, résultant de l'utilisation du prototype *GenSCor* sur des analyses d'exomes qui n'avaient pas été préalablement traitées avec la méthode manuelle. Parmi ce nouveau *corpus*, seules les variations clairement pathogènes sont comptabilisées, les délais de vérifications ne permettant pas d'inclure les variations où des doutes sont à lever à l'aide d'études familiales. Le tableau de ces nouvelles variations est présenté dans le tableau 5.11, avec les rangs correspondants obtenus par chaque utilisateur. Le moins bon rang obtenu par un utilisateur est 30, pour une variation récessive homozygote. Les résultats sont très cohérents avec les données que nous avons pu collecter sur notre cohorte de variations connues.

Patient	Gène	Type	Variation	Rang *L.*	Rang *G.*	Rang *F.*	Rang *T.*
18A	PIK3CA	Faux sens	c.323G>A *p.Arg108His*	1	1	1	1
19A	CHD7	Faux sens	c.6749A>G *p.Lys2250Arg*	1	2	1	26
23A	RAF1	Faux sens	c.788T>A *p.Val263Asp*	1	1	1	1
30A	DCC	Non sens	c.823C>T *p.Arg275Ter*	1	1	1	1
31A	ENG	Décalage CL	c.1116_1119delAAAG *p.Lys374SerfsTer6*	9	1	1	5
33A	ATP2A2	Décalage CL	c.1692delC *p.Thr565LeufsTer6*	1	1	4	6
36A	BPTF	Non sens	c.619C>T *p.Arg207Ter*	1	1	1	14
39A	CHD7	Non sens	c.5428C>T *p.Arg1810Ter*	1	1	1	1
40A	PTPN11	Faux sens	c.923A>C *p.Asn308Thr*	1	3	1	1
41A	BCOR	Décalage CL	c.4936dupC *p.Leu1646ProfsTer6*	1	1	1	2
42A	KIT	Décalage CL	c.2849dupT *p.Val951GlyfsTer98*	1	2	1	4
43A	POMT1	V. d'épissage	c.280+1G>T	1	30	11	2

TABLEAU 5.11 – Tableau récapitulatif des variations pathogènes prospectives associées aux rangs obtenus par les utilisateurs. "Décalage CL" signifie décalage du cadre de lecture, "V. d'épissage" désigne une altération d'un site d'épissage.

On peut ainsi constater que ces variations sont priorisées de façon relativement constante par les différents utilisateurs. Ces données indiquent une efficacité au moins égale à la méthode manuelle, car ces variations sont relativement simples à mettre en évidence, quelle que soit la méthode. Il est cependant intéressant de noter que leur mise en lumière est ici très importante, puisque la variation pathogène est classée au premier rang dans près de 70 % des cas. On peut également noter que chacune de ces variations est classée au premier rang par au moins un des utilisateurs.

En plus de ce *corpus*, le laboratoire a participé au projet hospitalier de recherche clinique inter-régional HUGODIMS, visant à développer les capacités d'analyses d'exomes des CHU et au cours duquel les ADN d'une cohorte d'une centaine de patient atteints de déficience intellectuelle dont l'étiologie est inconnue ont été séquencés. Les données finales ne sont pas encore disponibles, mais cette cohorte est très prometteuse, car les analyses ont été effectuées par d'autres laboratoires et portent sur des variations beaucoup plus délicates à détecter. Cela permet donc de comparer GenSCor à d'autres approches de recherches ainsi que de tester ces performances sur des variations plus complexes. Nous disposons de 3 patients de cette cohorte qui ont pu bénéficier de l'analyse d'un génome complet à la suite d'une analyse d'exome par la méthode manuelle infructueuse. Il s'avère que les variations causales étaient bien présentes dans les données d'exomes mais n'avaient pas été isolées par les généticiens. Nous avons utilisé les *rulesets* de nos utilisateurs sur ces données d'exomes, les résultats sont présentés dans le tableau 5.12.

Patient	Gène	Type	Variation	Rang *L.*	Rang *G.*	Rang *F.*	Rang *T.*
TT	MN1	Non sens	c.3778G>T *p.Glu1260Ter*	19	1	2	72
CL	INTS11	Faux sens	c.1204A>G *p.Lys402Glu*	58	2	15	130
CL	INTS11	V. d'épissage	c.1313-9G>A	587	561	461	399
39A	TMEM70	V. d'épissage	c.317-2A>G (*hmz*)	1	18	2	1

TABLEAU 5.12 – Tableau récapitulatif des variations pathogènes rétrospectives associées aux rangs obtenus par les utilisateurs. "V. d'épissage" désigne une altération d'un site d'épissage.

On peut constater dans ce tableau que les *ruleset* des utilisateurs priorisent bien la variation sur le gène *MN1* ainsi que la variation homozygote sur le gène *TMEM70*. Concernant la paire de variations sur le gène récessif *INTS11*, l'une des deux était très délicate à prioriser, car son annotation contenait peu d'informations évoquant son caractère pathogène. En revanche, l'autre variation est relativement bien priorisée par les utilisateurs *F.* et surtout *G.*, le couple de variations aurait donc été analysé par ses deux utilisateurs.

5.7.6 Discussion générale

Après l'analyse des résultats et les différentes conclusions partielles évoquées, nous pouvons maintenant recontextualiser l'ensemble de nos données en reprenant (1) les règles de conception de *Gendomus*, (2) l'utilisation de différentes fonctionnalités de *GenSCor* par les généticiens et enfin (3) les modifications à apporter.

5.7.6.1 Règles de conception

Les règles de conception que nous avons définies en spécialisant celles proposées par les auteurs de *GenDomus* nous permettent plusieurs conclusions. La comparaison entre les outils en est facilitée, même si les règles peuvent être subjectives, il n'empêche qu'elle permettent de constater objectivement des différences de fonctionnalités. Elles sont également garantes de l'adéquation entre le prototype et l'approche que nous avons choisie, ce qui nous permet de poser des conclusions à l'échelle de l'approche et non simplement sur l'outil.

Enfin, elles ne sont pas pour autant spécifiques à ce contexte d'utilisation et peuvent permettre une première évaluation des différents outils bioinformatiques dédiés à la génétique qui sont employés par des utilisateurs non spécialisés. Le bon respect de ces règles n'empêche pas un réel besoin d'évolution de l'interface, qui pourra notamment s'appuyer sur l'ensemble des défauts relevés et présentés ci-dessous.

5.7.6.2 Opérateur *OU* et coefficient

L'analyse de l'usage du prototype permet de voir que ces deux fonctionnalités, l'opérateur ***OU*** et le coefficient, ne sont pas utilisées par nos 4 testeurs. L'opérateur ***OU*** a été implanté à la demande des utilisateurs, après une première phase de test. Deux hypothèses peuvent s'envisager, son utilisation peut-être trop complexe ou inadaptée à la demande des utilisateurs, ou le temps correspondant au développement a permis aux utilisateurs de trouver un moyen de contourner efficacement le problème. Cet opérateur n'est pas indispensable, puisque une duplication d'une règle avec un opérateur ***ET*** a le même résultat, il permet cependant d'éviter de traiter plusieurs fois une condition, donc d'améliorer les performances en termes temporel, ce qui peut être pertinent sur des machines peu performantes. Une étude auprès des utilisateurs est maintenant nécessaire pour comprendre les raisons de cette non-utilisation, avant d'éventuellement améliorer ou supprimer cette fonctionnalité.

Le coefficient quand à lui est très peu utilisé, seul l'utilisateur *F.* l'a utilisé pour 4 règles réparties sur 3 *rulesets*. Il lui est préféré un ajustement de la valeur du score, plus flexible. Il reste cependant utilisé dans les phases de développement ainsi que comme stockage de règles. Dans les deux cas, c'est la valeur de coefficient 0 qui est utilisée : dans le premier cas pour tester les effet d'un *ruleset* sans certaines règles dans le premier cas, dans le second cas pour garder des règles à n'utiliser que dans certains cas particuliers pour le second cas. Cette fonctionnalité pourrait donc être simplifiée, en ne conservant que la possibilité de supprimer temporairement les règles, puisque c'est l'emploi actuel de ce coefficient.

5.7.6.3 Gestion des variants récessifs

Nous avons constaté, notamment avec les variations sur les gènes *TAF8* et *QARS*, que malgré des *rulesets* dédiés, les variations récessives pouvaient rester plus complexes à capturer et à prioriser convenablement. Ce phénomène est en partie dû au manque d'informations nettes dans l'annotation permettant de mettre en évidence les variations pathogènes récessives. Ce point peut donc être amélioré avec une annotation plus complète. Cependant, des fonctionnalités dédiées pourraient simplifier le processus. L'une des pistes envisagées est de grouper les variations par gènes lors de l'export, afin de visualiser plus facilement les paires de variations pouvant avoir des effets pathologiques. Cependant, certains gènes peuvent contenir plus d'une dizaine de variations, un score positif peut par exemple être utilisé comme seuil pour éviter de perdre le bénéfice de la priorisation. Les variations homozygotes peuvent être priorisées en l'état *via* des stratégies plus ou moins complexes, comme la simple pondération importante de l'haplotype comme l'a proposé l'utilisateur *L.* à une gestion plus complexe des congruences comme décrit par l'utilisateur *T.*

5.7.6.4 Gestion des CNV

Les CNV peuvent actuellement être pris en charge par *GenSCor*, le prototype n'étant pas spécifique. Cette étude se concentre sur les SNV car le *pipeline* de détection et d'annotation des CNV est actuellement en cours d'élaboration. Cependant, le compte-rendu annoté de CNV étant également un fichier plat tabulaire, le fonctionnement sera exactement le même. La mise en relation de ces deux comptes-rendus est une étape supplémentaire à atteindre pour avoir une vue plus exhaustive des possibles variations et capturer des cas plus complexes, avec par exemple un CNV sur un allèle et un SNV sur l'autre. Cette jonction est relativement bien mise en pratique par Varaft et devra être ajoutée à *GenSCor*.

Les points d'évolution du prototype *GenSCor* se situent sur ces deux derniers aspects, la gestion des variations récessives et la gestion des variations de type CNV. D'autres études sont également à prévoir pour quantifier le gain d'efficacité et étudier le gain de temps.

Chapitre 6

Conclusion

Nos travaux se situent dans le domaine de la bio-informatique, plus précisément dans le contexte du diagnostic clinique des maladies génétiques rares. Globalement, ces maladies sont dues à un ou plusieurs variants dans le code génétique de l'individu, dont l'expression se traduit par un phénotype plus ou moins caractéristique, et des conséquences cliniques souvent graves. Identifier les variants mis en causes chez un patient donné est fondamental à la fois pour la recherche clinique et pour le patient en particulier, permettant d'affiner le diagnostic et le suivi médical, de proposer un conseil génétique adapté aux apparentés, et peut-être de proposer des traitements dans les années à venir.

6.1 Contextualisation

Le domaine de la génétique est dans une phase d'évolution rapide depuis l'arrivée des techniques de séquençage NGS. Ces techniques ont permis d'atteindre le stade de l'analyse globale du génome humain, pour autoriser un séquençage individualisé, prémisses d'une aide au diagnostic par exemple. C'est ainsi que de nombreuses bases de données génétiques ont été constituées, mettant ainsi à la disposition des professionnels du domaine un savoir de plus en plus riche.

Les techniques NGS génèrent des quantités de données très importantes, qui doivent être traitées informatiquement, ce qui a longtemps été le point limitant du processus. Dans le contexte de la génétique des maladies rares, l'amélioration des *pipelines* bio-informatiques ces dernières années a déplacé ce point limitant à l'analyse par les généticiens des fichiers de résultats. En effet, ces derniers doivent extraire une ou des variations pathogènes de fichiers contenant plusieurs dizaines de milliers de variations.

Pour les aider dans cette tâche, ils disposent de deux outils principaux : une annotation adaptée et complète, permettant d'ajouter du savoir aux variations et de faciliter leur classement, pathogène ou non, et une méthode de priorisation, souvent constituée de filtres successifs destinés à écrémer la liste des variations pour ne garder que celles ayant le plus de chance d'être pathogènes. Cependant, l'annotation est un processus informatique, qui requiert dans la plupart des cas l'intervention d'un bio-informaticien. Cet aspect limite grandement les possibilités des généticiens, qui ne peuvent pas expérimenter facilement en testant de nouvelles bases de données, chaque test demandant le développement de scripts dédiés. La tâche de priorisation est elle aussi sous-optimale : d'une part, les filtres successifs augmentent le risque de faux-négatif et obligent à plusieurs analyses successives en élargissant petit à petit les mailles du filet, d'autre part, ils doivent être redéfinis à chaque analyse, car ils varient en fonction de l'annotation, de la pathologie recherchée et des variations présentes. Enfin, cette phase d'analyse et de priorisation dépend grandement de la phase d'annotation, puisque ce sont ces annotations qui vont permettre de prioriser efficacement les variations.

On peut résumer le processus complet utilisé dans le domaine du diagnostic génétique des maladies rares par le schéma présenté figure 6.1

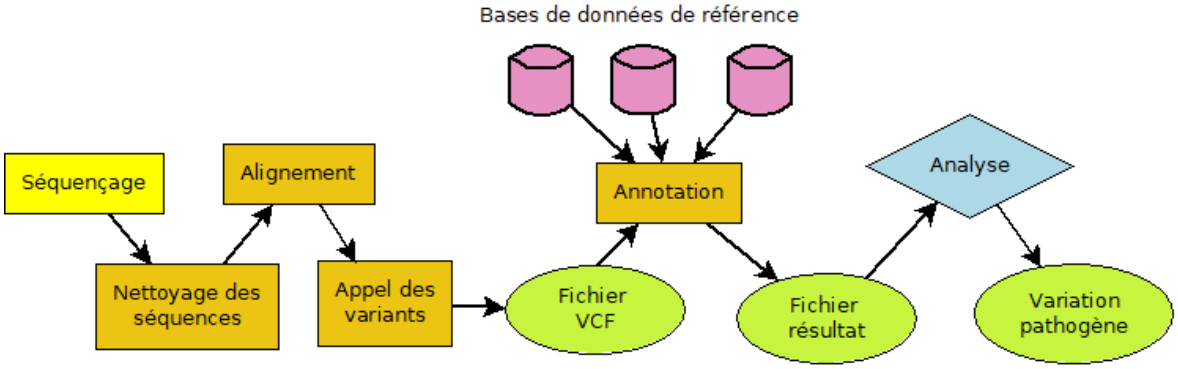


FIGURE 6.1 – Schéma du processus global de diagnostic des maladies rares.

Notre travail a pour but d'améliorer l'efficacité du travail d'analyse du généticien. Suite à une observation fine de la tâche, nous avons cherché à exploiter le savoir-faire du généticien pour lui fournir des outils adaptables permettant d'explorer les données en restreignant le recours au bio-informaticien.

6.2 Résumé de nos travaux

Au cours de ces travaux, nous avons abordés plusieurs thématiques : la bio-informatique, la génétique, l'Interaction Homme Machine et plus particulièrement les techniques de *End-User Programming* et les bases de données en abordant le problème de la construction d'ontologies. Nous avons fait un survol de l'état de l'art de ces différents points puis nous nous sommes particulièrement penché sur les problématiques de la liberté d'expérimentation des généticiens et de leur dépendance vis-à-vis des bio-informaticiens. Nous avons abordé plus spécifiquement deux points de cette problématique, l'étape de construction de l'annotation et l'étape d'analyse des variations annotées.

Concernant l'étape d'annotation, le nombre croissant de bases de données dédiées pouvant être reliées à toute recherche diagnostique clinique, ainsi que la fréquence des mises à jour de ces bases de données nous ont orienté vers la possibilité de création d'ontologies personnelles dédiées. Nous avons étudié les méthodes de création automatique et semi-automatique d'ontologies, ce qui nous a permis de constater la faible place des experts métiers lors de ces processus. Ces méthodes existantes ne convenant pas à notre contexte d'utilisation, nous avons élaboré notre propre méthode.

Nous avons dans un premier temps vérifié les postulats déduits de l'observation du travail des généticiens. Le plus important était que les généticiens construisent un modèle mental de leur domaine très proche d'un modèle de base de données, ce qui ouvrirait la voie à la construction directe d'ontologie personnelle par le généticien, sans aide de l'informaticien. Ainsi, nous avons mis en place une première expérience auprès d'experts du domaine aux profils variés afin de tester leur façon de décrire le domaine. Nous avons conçu cette expérience pour vérifier le postulat et, dans le cas de la validation de celui-ci, observer les structures syntaxiques et le vocabulaire utilisés par les utilisateurs.

Cette expérience a validé le postulat et nous a apporté une importante quantité de données. Nous avons étudié ces résultats dans l'objectif de trouver des correspondances entre les notions de bases de données et le vocabulaire ou les structures sémantiques utilisés par les généticiens. En nous appuyant sur ces études, nous avons développé un prototype, COPUNG, dont l'objectif était de valider notre méthode de création d'ontologie en nous appuyant sur des exports de bases de données et sur le modèle mental de l'utilisateur pour créer le modèle de l'ontologie. Ce prototype s'appuie sur les principes de la programmation avec exemple : l'utilisateur expert se voit proposer une approche guidée pour créer son ontologie, tout en ayant la possibilité de visualiser en permanence les résultats de ses choix. Les concepts manipulés sont exprimés dans le langage de l'expert, tel que nous avons pu le construire lors de notre première expérience. Au cours du test du prototype auprès de généticiens, nous avons pu constater la facile prise en main par les utilisateurs et nous avons étudié les ontologies produites.

Nous avons ainsi pu constater que ces ontologies étaient exploitables, bien que sous-optimales. Nous avons proposé des améliorations permettant de simplifier le processus et d'améliorer l'optimisation des ontologies produites. Nous n'avons pas développé le prototype plus avant, les tâches restantes relevant principalement de l'ingénierie.

Nous avons alors concentré nos efforts sur l'analyse des données annotées.

Ce second travail a d'abord nécessité l'étude détaillée des différents outils de la littérature scientifique permettant d'assister les généticiens dans leur tâche de priorisation des variations identifiées par séquençage NGS. Ces outils ne nous ont pas paru adaptés au domaine de la génétique des maladies rares. Ils souffrent ainsi de trois limites majeures :

— le manque de liberté sur les critères d'annotation, réduisant la liberté de l'utilisateur et

excluant l'utilisation de données spécifiques à une pathologie par exemple.

— la disponibilité des outils et leurs contraintes d'utilisation, requérant souvent de long temps de téléchargement et posant des soucis légaux importants.

— une méthode de priorisation en filtres successifs peu adaptée, qui augmente le risque de faux-négatifs et diminue drastiquement l'efficacité et l'efficience de la priorisation.

Nous avons alors proposé notre propre méthode de priorisation, basée sur un système de règles programmables par l'utilisateur, dans une démarche de programmation avec exemple. À partir du jeu de données issu de l'annotation, il s'agit d'enregistrer les opérations successives qui permettent de prioriser les données en fonctions de critères spécifiques définis par l'utilisateur expert. Nous avons défini le langage à base de règles permettant de construire le programme de priorisation, qui peut être sauvegardé et ré-utilisé sur des jeux de données d'origines différentes. On peut ainsi tester facilement des approches de recherches différentes, s'adapter à des modes de productions de données variés, ou encore prendre réellement en compte les annotations particulières de l'utilisateur. Cette méthode de priorisation n'est pas spécifique au domaine des maladies rares et peut s'appliquer à de nombreux autres domaines.

Nous avons validé cette méthode en construisant un prototype, GenSCor, que nous avons fait tester à des experts généticiens, puis nous avons analysé les résultats produits notamment en nous appuyant sur une cohorte de variations pathogènes connues. Les résultats ont montré que les généticiens réussissaient à prioriser les variations pathogènes, puisque sur les 29 variations pathogènes étudiées, 27, soit 93% sont placées au premier rang par au moins 1 des 4 utilisateurs et toutes sont placées parmi les 10 premières par au moins un utilisateur. Les contraintes du domaine, notamment le faible nombre d'utilisateur disponibles, la diversité des approches d'un utilisateurs à l'autre et l'impossibilité d'avoir un utilisateur naïf plus d'une fois ne nous ont pas permis de quantifier pour le moment le gain d'efficacité par rapport aux méthodes précédemment employées par les généticiens, mais nous avons pu prouver qu'il n'y a avait pas de perte d'efficacité en utilisant *GenSCor*. Nous avons cependant montré que pour au moins trois cas, des variations complexes qui avaient été manquées par l'analyse manuelle sont bien priorisées en utilisant *GenSCor*, ce qui laisse présager un gain d'efficacité, en particulier sur les variations complexes, délicates à isoler manuellement.

Concernant l'efficience, nous avons pu démontrer que la charge mentale des utilisateurs était réduite puisque le nombre de variations à analyser pour garantir une exhaustivité est considérablement réduit, sans avoir besoin de relancer l'analyse de multiples fois comme c'est le cas avec les approches de priorisation basées sur des filtres successifs. Le ressenti des utilisateurs vient appuyer ces résultats puisque tous indiquent une plus grande satisfaction et un meilleur confort d'usage en utilisant *GenSCor* par rapport à leur ancienne méthode. Enfin, nous n'avons pas pu quantifier le gain de temps de notre approche autrement que par les ressentis des utilisateurs indiquant un gain de temps perçu. Néanmoins le cumul de ces deux résultats indique une efficience supérieur de *GenSCor* par rapport aux méthodes précédentes. Cette supériorité est appuyée par le fait que les utilisateurs qui ont testé le prototype continuent de l'utiliser dans leurs analyses après la fin de l'expérience.

Notre méthode a prouvé sa robustesse, puisqu'elle a supporté avec succès des approches de recherches et des utilisations très variées d'un testeur à l'autre. Elle a également permis l'apparition de nouveaux mécanismes non envisagés, tel que l'utilisation de la parité pour faire ressortir un certain type de variation plus facilement comme l'a fait un des utilisateurs.

Grâce à ces deux approches, nous pouvons proposer un nouveau processus de recherche génétique clinique pour les maladies rares, qui prend en compte les deux outils proposés pour augmenter l'efficacité et l'efficience des généticiens, tout en limitant leur dépendance aux experts bio-informaticien dans leur tâche d'analyse (figure 6.2).

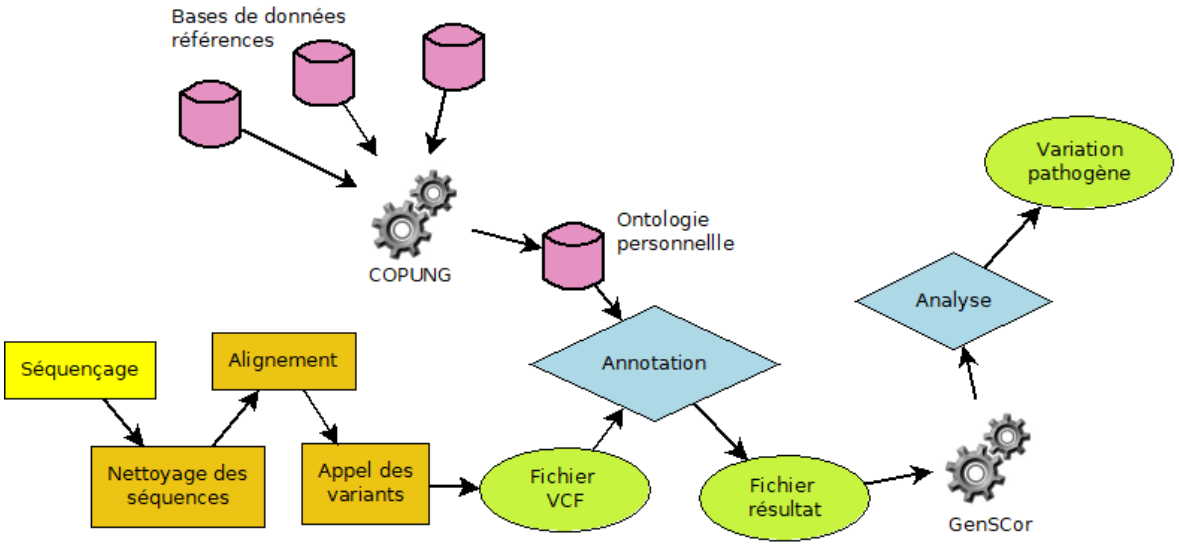


FIGURE 6.2 – Schéma du positionnement effectif de nos contributions sur le processus global

6.3 Perspectives de recherches et d'améliorations

Les perspectives concernant les approches proposées dans ce travail ainsi que les prototypes réalisés sont nombreuses. En ce qui concerne COPUNG, les étapes de développement supplémentaires sont à venir, afin de proposer un meilleur accompagnement des généticiens tout au long du processus et d'améliorer l'optimisation des ontologies produites.

Une phase de développement est également nécessaire pour créer concrètement l'annotateur utilisant ces ontologies et fournir toutes les fonctionnalités permettant d'assurer un fonctionnement en routine de l'outil. Ce développement aura pour but de répondre à ces différentes questions : comment intégrer facilement des mises à jours de bases de données dans les données sources ? Comment gérer les conflits entre données issues de bases de données différentes ? Comment prendre en compte facilement la modification de structure de bases de données sources ? Comment et jusqu'où historiser le travail effectué pour le réutiliser facilement ? Cette phase s'assimile principalement à de l'ingénierie, mais n'en reste pas moins complexe.

En plus de ces suites de développement, il serait intéressant de creuser l'applicabilité de cette méthode pour d'autres thématiques. En effet, si la génétique apparait comme un contexte très différents des disciplines informatiques concevant des ontologies, le contexte d'un expert métier sans compétences informatiques, mais avec à sa disposition des bases de données et qui souhaite les fusionner peut être retrouvé dans d'autres secteurs.

Concernant la seconde partie de nos travaux, de nombreuses pistes sont ouvertes, dont certaines sont déjà en cours. Le test de *GenSCor* sur une cohorte plus importante de variations, elles-mêmes plus complexes et identifiées par d'autres laboratoires, avec potentiellement des pratiques très différentes s'impose. La collecte de ces données est actuellement en cours. L'étude longitudinale d'une cohorte d'utilisateur plus significative est également prévue, afin de vérifier que l'approche est utilisable en dehors du laboratoire, mais aussi d'étudier les différents comportements des utilisateurs et de mesurer le bénéfice en temps.

Cette seconde étude demande des développements supplémentaire sur le prototype. En effet, nous avons constaté la différence importante de performance entre des configurations puissantes de machines et des machines plus classiques. Si ces différences ne posent pas de problèmes majeurs à l'échelle de l'analyse d'exome, elles peuvent empêcher le passage à l'échelle du génome. De même, les résultats nous ont indiqué que certaines fonctionnalités n'étaient pas ou peu utilisées par notre panel de testeur. Il convient de les rendre plus utilisables, ou de les supprimer si elles n'apportent pas d'aide. Nous avons également constaté les limites de *GenSCor* sur certains aspects, notamment la gestion des variations récessives, qui doit

donc être améliorée. Enfin, une étude sur des variations de type CNV et également prévue rapidement. Une présentation commune des résultats pourrait être intéressante et doit être réfléchie.

De la même façon que COPUNG, il serait intéressant d'utiliser ces travaux dans d'autres domaines. *GenSCor* ne comporte aucune fonctionnalités spécifique à la génétique, il peut donc être utilisé dans des disciplines variées. En effet, cette méthode permet de prioriser des entités en prenant en compte une multitude de critères non spécifiques, ce qui peut être pertinent ailleurs que dans le contexte des maladies génétiques rares.

Un exemple d'utilisation est le travail en budget limité, notamment le cas de l'entretien des lampadaires à l'échelle départementale. Au niveau départemental, une structure administrative nommée "syndicat énergie" gère les parcs de lampadaires des différentes communes. Tous les ans, cette structure a un budget défini alloué pour remplacer les lampadaires trop âgés ou inadaptés. Pour cela, le personnel administratif s'appuie sur des multiples tableaux fournis par les communes, contenant un lampadaire par ligne et une information par colonne. Ces tableaux sont tous construits sur le même modèle, puisque c'est le syndicat qui fournit les modèles de tableaux. Les informations sont nombreuses, on peut citer par exemple le type de route, l'année de mise en service, le type de mât, le type de lampe, ou encore l'état général. Afin de trouver quels lampadaires remplacer, le personnel administratif cherche à les trier du moins pertinent au plus urgent. Cette tâche se fait manuellement, c'est-à-dire que le personnels parcours l'ensemble du fichier, examine les lampadaires un à un et estime pour chacun s'il est à remplacer ou pas. La liste de lampadaires extraite est ensuite réaffinée pour correspondre au budget, en supprimant les moins urgents jusqu'à être dans le budget. Le processus est délicat, long et très peu efficient. Un outil de priorisation tel que *GenSCor* permettrait de le simplifier considérablement, en définissant les règles adaptées, il serait possible d'obtenir un score associé au besoin de remplacement. Il suffirait ensuite d'utiliser le budget comme seuil pour obtenir la liste des lampadaires à remplacer. Cet exemple trivial a réellement été rencontré en 2012, mais *GenSCor* n'existait pas encore, le processus de sélection fût donc fastidieux.

Bibliographie

Adzhubei, Ivan A et al. (2010). "A method and server for predicting damaging missense mutations". In : *Nature methods* 7.4, p. 248.

Adzhubei, Ivan, Daniel M Jordan et Shamil R Sunyaev (2013). "Predicting functional effect of human missense mutations using PolyPhen-2". In : *Current protocols in human genetics* 76.1, p. 7-20.

Ahmed, K Mueen et Bandar Al Dhubaib (2011). "Zotero: A bibliographic assistant to researcher". In : *Journal of Pharmacology and Pharmacotherapeutics* 2.4, p. 303.

Akita, Kohei, Toshimitsu Tanaka et Yuji Sagawa (2018). "SliT: Character Input System Using Slide-in and Tap for Smartwatches". In : *International Conference on Human-Computer Interaction*. Springer, p. 3-16.

Alemán, Alejandro et al. (2014). "A web-based interactive framework to assist in the prioritization of disease candidate genes in whole-exome sequencing studies". In : *Nucleic acids research* 42.W1, W88-W93.

Alliance maladies rares (2016). *L'errance de diagnostic - ERRADIAG*. url : `https://www.alliance-maladies-rares.org/lerrance-de-diagnostic/`.

Aluru, Srinivas et Nagakishore Jammula (2013). "A review of hardware acceleration for computational genomics". In : *IEEE Design & Test* 31.1, p. 19-30.

Aranguren, Mikel Egana (2005). "Ontology design patterns for the formalisation of biological ontologies". In : *University of Manchester*, p. 81.

Arenas et al. (2012). *Direct mapping of relationnal data to RDF*. url : `https://www.w3.org/TR/rdb-direct-mapping`.

Ashburner, Michael et al. (2000). "Gene ontology: tool for the unification of biology". In : *Nature genetics* 25.1, p. 25.

Bada, Michael et al. (2004). "A short study on the success of the Gene Ontology". In : *Web Semantics: Science, Services and Agents on the World Wide Web* 1.2, p. 235-240.

Barrasa, Jesus, Oscar Corcho et Asuncion Gomez-pérez (2004). "R2O, an Extensible and Semantically based Database-to-Ontology Mapping Language". In : *in In Proceedings of the 2nd Workshop on Semantic Web and Databases(SWDB2004*. Springer, p. 1069-1070.

Bertoldi, Loris et al. (2017). "QueryOR: a comprehensive web platform for genetic variant analysis and prioritization". In : *BMC bioinformatics* 18.1, p. 225.

Biosoftware, Interactive (2014). *Alamut Batch*. url : `https://www.interactive-biosoftware.com`.

Bodrunova, Svetlana S et Alexandr Yakunin (2018). "Impact of menu complexity upon user behavior and satisfaction in information search". In : *International Conference on Human Interface and the Management of Information*. Springer, p. 55-66.

Braun, Simone et al. (2007). "Ontology Maturing: a Collaborative Web 2.0 Approach to Ontology Engineering." In : *Ckc* 273.

Brewster, Christopher et al. (2007). "Dynamic iterative ontology learning". In : *6th International Conference on Recent Advances in Natural Language Processing*.

Brickley, Dan, Ramanathan V Guha et Brian McBride (2014). "RDF Schema 1.1". In : *W3C recommendation* 25, p. 2004-2014.

Buitelaar, Paul, Philipp Cimiano et Bernardo Magnini (2005). *Ontology learning from text: methods, evaluation and applications*. T. 123. IOS press.

Buitelaar, Paul, Philipp Cimiano, Stefania Racioppa et al. (2006). "Ontology-based information extraction with soba". In : *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*.

Burnett, Margaret M et Christopher Scaffidi (2014). *The Encyclopedia of Human-Computer Interaction*.

Card, Stuart K (1983). *The psychology of human-computer interaction*. Crc Press.

Card, Stuart K, Thomas P Moran et Allen Newell (1980). "The keystroke-level model for user performance time with interactive systems". In : *Communications of the ACM* 23.7, p. 396-410.

Carlisle, James H (1976). "Evaluating the impact of office automation on top management communication". In : *Proceedings of the June 7-10, 1976, national computer conference and exposition*. ACM, p. 611-616.

Centre National de Ressources Textuelles et Lexicales (2019). *Etymologie du mot score*. url : `https://www.cnrtl.fr/etymologie/score`.

Cingolani, Pablo et al. (2012). "A program for annotating and predicting the effects of single nucleotide polymorphisms, SnpEff: SNPs in the genome of Drosophila melanogaster strain w1118; iso-2; iso-3". In : *Fly* 6.2, p. 80-92.

Clark, Michelle M et al. (2018). "Meta-analysis of the diagnostic and clinical utility of genome and exome sequencing and chromosomal microarray in children with suspected genetic diseases". In : *NPJ genomic medicine* 3.

Commission nationale de l'informatique et des libertés (2017). *Les données génétiques*. La documentation française. isbn : 978-2-11-145187.

Compeau, Phillip EC, Pavel A Pevzner et Glenn Tesler (2011). "How to apply de Bruijn graphs to genome assembly". In : *Nature biotechnology* 29.11, p. 987.

Conti, Elena et Elisa Izaurralde (2005). "Nonsense-mediated mRNA decay: molecular insights and mechanistic variations across species". In : *Current opinion in cell biology* 17.3, p. 316-325.

Cooper, Gregory M et Jay Shendure (2011). "Needles in stacks of needles: finding disease-causal variants in a wealth of genomic data". In : *Nature Reviews Genetics* 12.9, p. 628.

Cooper, Gregory M, Eric A Stone et al. (2005). "Distribution and intensity of constraint in mammalian genomic sequence". In : *Genome research* 15.7, p. 901-913.

Cypher, Allen et Daniel Conrad Halbert (1993). *Watch what I do: programming by demonstration*. MIT press.

Das, sundara et Cyganiak (2012). *R2RML : RDB to RDF mapping language*. url : `https://www.w3.org/TR/r2rml`.

Davydov, Eugene V et al. (2010). "Identifying a high fraction of the human genome to be under selective constraint using GERP++". In : *PLoS computational biology* 6.12, e1001025.

De Ligt, Joep et al. (2012). "Diagnostic exome sequencing in persons with severe intellectual disability". In : *New England Journal of Medicine* 367.20, p. 1921-1929.

De Nicola, Antonio et Michele Missikoff (2016). "A lightweight methodology for rapid ontology engineering". In : *Communications of the ACM* 59.3, p. 79-86.

De Nicola, Antonio, Michele Missikoff et Roberto Navigli (2005). "A proposal for a unified process for ontology building: UPON". In : *International Conference on Database and Expert Systems Applications*. Springer, p. 655-664.

— (2009). "A software engineering approach to ontology building". In : *Information systems* 34.2, p. 258-275.

Desmet, François-Olivier et al. (2009). "Human Splicing Finder: an online bioinformatics tool to predict splicing signals". In : *Nucleic acids research* 37.9, e67-e67.

Desvignes, Jean-Pierre et al. (2018). "VarAFT: a variant annotation and filtration system for human next generation sequencing data". In : *Nucleic acids research* 46.W1, W545-W553.

Dimou, Anastasia, Miel Vander Sande et al. (2013). "Extending R2RML to a Source-independent Mapping Language for RDF". In : *Proceedings of the 2013th International Conference on Posters & Demonstrations Track - Volume 1035*. ISWC-PD'13. Sydney, Australia : CEUR-WS.org, p. 237-240. URL : http://dl.acm.org/citation.cfm?id=2874399.2874459.

Dimou, Anastasia, Miel Vander Sande et al. (2014). "RML: a generic language for integrated RDF mappings of heterogeneous data". eng. In : *7th Workshop on Linked Data on the Web, Proceedings*. Seoul, Korea, p. 5.

Drumond, Lucas et Rosario Girardi (2008). "A Survey of Ontology Learning Procedures." In : *WONTO*. Sous la dir. de Frederico Luiz Gonçalves de Freitas et al. T. 427. CEUR Workshop Proceedings. CEUR-WS.org. URL : http://dblp.uni-trier.de/db/conf/wonto/wonto2008.html%5C#DrumondG08.

Duruisseau, Mickael et al. (2018). "VisUML: a live UML visualization to help developers in their programming task". In : *International Conference on Human Interface and the Management of Information*. Springer, p. 3-22.

England, Genomics (2018). *Genomics England PanelApp*. Accessed: 2019-05-02. URL : %5Curl%7Bhttps://panelapp.genomicsengland.co.uk%7D.

Ermilov, Ivan, Sören Auer et Claus Stadler (2013). "User-driven Semantic Mapping of Tabular Data". In : I-SEMANTICS '13, p. 105-112. DOI : 10.1145/2506182.2506196. URL : http://doi.acm.org/10.1145/2506182.2506196.

Ezkurdia, Iakes et al. (2014). "Multiple evidence strands suggest that there may be as few as 19 000 human protein-coding genes". In : *Human molecular genetics* 23.22, p. 5866-5878.

Frühwirth, Thomas, Wolfgang Kastner et Lukas Krammer (2018). "A methodology for creating reusable ontologies". In : *2018 IEEE Industrial Cyber-Physical Systems (ICPS)*. IEEE, p. 65-70.

Gargis, Amy S et al. (2015). "Good laboratory practice for clinical next-generation sequencing informatics pipelines". In : *Nature biotechnology* 33.7, p. 689.

Geoffroy, Véronique et al. (2015). "VaRank: a simple and powerful tool for ranking genetic variants". In : *PeerJ* 3, e796.

Girard, Patrick (2001). "Bringing programming by demonstration to CAD users". In : *Your wish is my command*. Elsevier, p. 135-VII.

Goubali, Olga et al. (2016). "Designing Functional Specifications for Complex Systems". In : *International Conference on Human-Computer Interaction*. Springer, p. 166-177.

Grantham, Richard (1974). "Amino acid difference formula to help explain protein evolution". In : *Science* 185.4154, p. 862-864.

Gruber, Thomas R (1993). "A translation approach to portable ontology specifications". In : *Knowledge acquisition* 5.2, p. 199-220.

Gutbell, Ralf, Lars Pandikow et Arjan Kuijper (2018). "Web-based visualization component for geo-information". In : *International Conference on Human Interface and the Management of Information*. Springer, p. 23-35.

Haase, Peter et al. (2008). "The neon ontology engineering toolkit". In : *WWW*.

Hamosh, Ada et al. (2005). "Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders". In : *Nucleic acids research* 33.suppl_1, p. D514-D517.

Handschuh, Siegfried, Steffen Staab et Raphael Volz (2003). "On Deep Annotation". In : *Proceedings of the 12th International Conference on World Wide Web*. WWW '03. Budapest, Hungary : ACM, p. 431-438. ISBN : 1-58113-680-3. DOI : 10.1145/775152.775214. URL : http://doi.acm.org/10.1145/775152.775214.

Hart, Steven N et al. (2015). "VCF-Miner: GUI-based application for mining variants and annotations stored in VCF files". In : *Briefings in bioinformatics* 17.2, p. 346-351.

Herzog, Reinhard, Michael Jacoby et Ivana Podnar Žarko (2016). "Semantic interoperability in IoT-based automation infrastructures". In : *at-Automatisierungstechnik* 64.9, p. 742-749.

Hewett, Thomas T et al. (1992). *ACM SIGCHI curricula for human-computer interaction*. ACM.

Hildebrandt, C et al. (2018). "Ontology Building for Cyber-Physical Systems: A domain expert-centric approach". In : *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*. IEEE, p. 1079-1086.

Hou, Wen-jun et al. (2018). "User defined eye movement-based interaction for virtual reality". In : *International Conference on Cross-Cultural Design*. Springer, p. 18-30.

Ihemedu-Steinke, Quinate Chioma et al. (2018). "VR Evaluation of Motion Sickness Solution in Automated Driving". In : *International Conference on Virtual, Augmented and Mixed Reality*. Springer, p. 112-125.

Iniguez Jarrin, Carlos et al. (2017). "Guidelines for Designing User Interfaces to Analyze Genetic Data. Case of Study: GenDomus". In : *International Conference on Evaluation of Novel Approaches to Software Engineering*. Springer, p. 3-22.

Iñiguez-Jarrìn, Carlos et al. (2017). "GenDomus : Interactive and Collaboration Mechanisms for diagnosing Genetic Diseases." In : *ENASE*, p. 91-102.

Jansen, Sandra et al. (2017). "De novo truncating mutations in the last and penultimate exons of PPM1D cause an intellectual disability syndrome". In : *The American Journal of Human Genetics* 100.4, p. 650-658.

Jirkovsky, Václav, Marek Obitko et Vladimír Mařík (2016). "Understanding data heterogeneity in the context of cyber-physical systems integration". In : *IEEE Transactions on Industrial Informatics* 13.2, p. 660-667.

Kahn, Ken (1996). "Seeing systolic computations in a video game world". In : *Proceedings 1996 IEEE Symposium on Visual Languages*. IEEE, p. 95-101.

Karczewski, Konrad J, Laurent C Francioli et al. (2019). "Variation across 141,456 human exomes and genomes reveals the spectrum of loss-of-function intolerance across human protein-coding genes". In : *BioRxiv*, p. 531210.

Karczewski, Konrad J, Ben Weisburd et al. (2016). "The ExAC browser: displaying reference data information from over 60 000 exomes". In : *Nucleic acids research* 45.D1, p. D840-D845.

Kim, Sanghee et al. (2002). "Artequakt: Generating tailored biographies from automatically annotated fragments from the web". In :

Kircher, Martin et al. (2014). "A general framework for estimating the relative pathogenicity of human genetic variants". In : *Nature genetics* 46.3, p. 310.

Kochinke, Korinna et al. (2016a). "Systematic phenomics analysis deconvolutes genes mutated in intellectual disability into biologically coherent modules". In : *The American Journal of Human Genetics* 98.1, p. 149-164.

— (2016b). "Systematic phenomics analysis deconvolutes genes mutated in intellectual disability into biologically coherent modules". In : *The American Journal of Human Genetics* 98.1, p. 149-164.

Koeda, Masanao et al. (2018). "Development of wireless surgical knife attachment with proximity indicators using ArUco marker". In : *International Conference on Human-Computer Interaction*. Springer, p. 14-26.

Köhler, Sebastian, Leigh Carmody et al. (2018). "Expansion of the Human Phenotype Ontology (HPO) knowledge base and resources". In : *Nucleic acids research* 47.D1, p. D1018-D1027.

Köhler, Sebastian, Sandra C Doelken et al. (2013). "The Human Phenotype Ontology project: linking molecular biology and disease through phenotype data". In : *Nucleic acids research* 42.D1, p. D966-D974.

Kotis, Konstantinos et George A Vouros (2006). "Human-centered ontology engineering: The HCOME methodology". In : *Knowledge and Information Systems* 10.1, p. 109-131.

Laborda, Cristian Pérez de et Stefan Conrad (2005). "Relational.OWL: A Data and Schema Representation Format Based on OWL". In : *Proceedings of the 2Nd Asia-Pacific Conference on Conceptual Modelling - Volume 43*. APCCM '05. Newcastle, New South Wales, Australia : Australian Computer Society, Inc., p. 89-96. isbn : 1-920-68225-2. url : http://dl.acm.org/citation.cfm?id=1082276.1082287.

Landrum, Melissa J et al. (2015). "ClinVar: public archive of interpretations of clinically relevant variants". In : *Nucleic acids research* 44.D1, p. D862-D868.

Larroque, Béatrice et al. (2008). "Neurodevelopmental disabilities and special care of 5-year-old children born before 33 weeks of gestation (the EPIPAGE study): a longitudinal cohort study". In : *The Lancet* 371.9615, p. 813-820.

Lee, Tae-Ho et Hyuk-Jae Lee (2018). "A New Virtual Keyboard with Finger Gesture Recognition for AR/VR Devices". In : *International Conference on Human-Computer Interaction*. Springer, p. 56-67.

Lefrançois, Maxime, Antoine Zimmermann et Noorani Bakerally (2017). "Génération de RDF à partir de sources de données aux formats hétérogènes". In :

Letienne, Lucile (2019). "Intérêt de la technique de l'exome dans la recherche de nouveaux variants géniques impliqués dans la déficience intellectuelle". Thèse de doct. Université de Poitiers, Faculté de Médecine et Pharmacie.

Letondal, Catherine (2001). "Interaction et programmation". Thèse de doct.

Li, Quan et Kai Wang (2017). "InterVar: clinical interpretation of genetic variants by the 2015 ACMG-AMP guidelines". In : *The American Journal of Human Genetics* 100.2, p. 267-280.

Li, Xuemei et Mike Thelwall (2012). "F1000, Mendeley and traditional bibliometric indicators". In : *Proceedings of the 17th international conference on science and technology indicators*. T. 2. Science-Metrix et OST Montréal, Canada, p. 451-551.

Lieberman, Henry (2001). *Your wish is my command: Programming by example*. Morgan Kaufmann.

Maedche, Alexander et Steffen Staab (2004). "Ontology learning". In : *Handbook on ontologies*. Springer, p. 173-190.

Mather, Cheryl A et al. (2016). "CADD score has limited clinical validity for the identification of pathogenic variants in noncoding regions in a hereditary cancer panel". In : *Genetics in Medicine* 18.12, p. 1269.

Maulik, Pallab K et al. (2011). "Prevalence of intellectual disability: a meta-analysis of population-based studies". In : *Research in developmental disabilities* 32.2, p. 419-436.

McKusick, Victor A (2014). *Mendelian inheritance in man: catalogs of autosomal dominant, autosomal recessive, and X-linked phenotypes*. Elsevier.

McKusick, Victor Almon et al. (1966). *Heritable disorders of connective tissue*. T. 467. Mosby St. Louis.

McLaren, William et al. (2016). "The ensembl variant effect predictor". In : *Genome biology* 17.1, p. 122.

Menolli, André Luís Andrade et al. (2013). "An Incremental and Iterative Process for Ontology Building." In : *ONTOBRAS*. Citeseer, p. 215-220.

Meyer, Ronald et al. (2018). "Gesture-Based Vehicle Control in Partially and Highly Automated Driving for Impaired and Non-impaired Vehicle Operators: A Pilot Study". In : *International Conference on Universal Access in Human-Computer Interaction*. Springer, p. 216-227.

Mwaniki, Michael K et al. (2012). "Long-term neurodevelopmental outcomes after intrauterine and neonatal insults: a systematic review". In : *The Lancet* 379.9814, p. 445-452.

Ng, Pauline C et Steven Henikoff (2003). "SIFT: Predicting amino acid changes that affect protein function". In : *Nucleic acids research* 31.13, p. 3812-3814.

Niemiec, Emilia et Heidi Carmen Howard (2016). "Ethical issues in consumer genome sequencing: Use of consumers' samples and data". In : *Applied & translational genomics* 8, p. 23-30.

Noy, Natalya Fridman, Monica Crubézy et al. (2003). "Protégé-2000: an open-source ontology-development and knowledge-acquisition environment." In : *AMIA Annual Symposium Proceedings*. T. 2003. American Medical Informatics Association, p. 953-953.

Noy, Natalya Fridman, Deborah L McGuinness et al. (2001). *Ontology development 101: A guide to creating your first ontology*.

O'Connor, Martin J., Christian Halaschek-Wiener et Mark A. Musen (2010). "Mapping Master: A Flexible Approach for Mapping Spreadsheets to OWL". In : *International Semantic Web Conference*.

Oliveira Camenar, Leticia Maria de, Diego de Faria do Nascimento et Leonelo Dell Anhol Almeida (2018). "A Method for Analyzing Mobility Issues for People with Physical Disabilities in the Context of Developing Countries". In : *International Conference on Universal Access in Human-Computer Interaction*. Springer, p. 3-17.

Orphanet (2019). *Prévalence des maladies rares : données bibliographiques*. url : %5Curl%7Bhttps://www.orpha.net/%7D.

Overton, James A et al. (2015). "ROBOT: A command-line tool for ontology development." In : *ICBO*.

Petasis, Georgios et al. (2011). "Ontology population and enrichment: State of the art". In : *Knowledge-driven multimedia information extraction and ontology evolution*. Springer-Verlag, p. 134-166.

Pierra, Guy (2003). "Context-explication in conceptual ontologies: the PLIB approach." In : *ISPE CE*. Citeseer, p. 243-253.

Pierra, G et al. (1998). "Exchange of component data: the PLIB (ISO 13584) model, standard and tools". In : *Proceedings of the CALS EUROPE* 98, p. 160-176.

Pinto, H Sofia, Steffen Staab et Christoph Tempich (2004). "DILIGENT: Towards a fine-grained methodology for Distributed, Loosely-controlled and evolvInG". In : *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI 2004)*. T. 110, p. 393.

Pollard, Katherine S et al. (2010). "Detection of nonneutral substitution rates on mammalian phylogenies". In : *Genome research* 20.1, p. 110-121.

Quang, Daniel, Yifei Chen et Xiaohui Xie (2014). "DANN: a deep learning approach for annotating the pathogenicity of genetic variants". In : *Bioinformatics* 31.5, p. 761-763.

Quinodoz, Mathieu et al. (2017). "DOMINO: using machine learning to predict genes associated with dominant disorders". In : *The American Journal of Human Genetics* 101.4, p. 623-629.

Ran, F Ann et al. (2013). "Genome engineering using the CRISPR-Cas9 system". In : *Nature protocols* 8.11, p. 2281.

Richards, C Sue et al. (2008). "ACMG recommendations for standards for interpretation and reporting of sequence variations: Revisions 2007". In : *Genetics in Medicine* 10.4, p. 294.

Richards, Sue et al. (2015). "Standards and guidelines for the interpretation of sequence variants: a joint consensus recommendation of the American College of Medical Genetics and Genomics and the Association for Molecular Pathology". In : *Genetics in medicine* 17.5, p. 405.

Riché-Piotaix, Quentin et al. (2018). "Example Based Programming and Ontology Building: A Bioinformatic Application". In : *International Conference on Human-Computer Interaction*. Springer, p. 101-108.

Roeleveld, Nel et Gerhard A Zielhuis (1997). "The prevalence of mental retardation: a critical review of recent literature". In : *Developmental Medicine & Child Neurology* 39.2, p. 125-132.

Royer, Kevin (2015). “Vers un entrepôt de données et des processus: le cas de la mobilité électrique chez EDF”. Thèse de doct. ISAE-ENSMA Ecole Nationale Supérieure de Mécanique et d’Aérotechique-Poitiers.

Rutherford, Kim et al. (2000). “Artemis: sequence visualization and annotation”. In : *Bioinformatics* 16.10, p. 944-945.

Salgado, David et al. (2016). “UMD-predictor: a high-throughput sequencing compliant system for pathogenicity prediction of any human cDNA substitution”. In : *Human mutation* 37.5, p. 439-446.

Salzberg, Steven L (2018). “Open questions: How many genes do we have?” In : *BMC biology* 16.1, p. 94.

Sanger, Frederick, Steven Nicklen et Alan R Coulson (1977). “DNA sequencing with chain-terminating inhibitors”. In : *Proceedings of the national academy of sciences* 74.12, p. 5463-5467.

Schwarz, Jana Marie et al. (2014). “MutationTaster2: mutation prediction for the deep-sequencing age”. In : *Nature methods* 11.4, p. 361.

Schwarze, Katharina et al. (2018). “Are whole-exome and whole-genome sequencing approaches cost-effective? A systematic review of the literature”. In : *Genetics in Medicine*.

Sequeda, Juan et Daniel P. Miranker (2015). “Ultrawrap Mapper: A Semi-Automatic Relational Database to RDF (RDB2RDF) Mapping Tool”. In :

Sherry, Stephen T et al. (2001). “dbSNP: the NCBI database of genetic variation”. In : *Nucleic acids research* 29.1, p. 308-311.

Siepel, Adam et al. (2005). “Evolutionarily conserved elements in vertebrate, insect, worm, and yeast genomes”. In : *Genome research* 15.8, p. 1034-1050.

Simperl, Elena et Markus Luczak-Rösch (2014). “Collaborative ontology engineering: a survey”. In : *The Knowledge Engineering Review* 29.1, p. 101-131.

Smith, Barry et al. (2007). “The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration”. In : *Nature biotechnology* 25.11, p. 1251.

Smith, Michael K (2004). “OWL web ontology language guide”. In : *http://www. w3. org/TR/owl-guide/*.

Srour, Myriam et Michael Shevell (2014). “Genetics and the investigation of developmental delay/intellectual disability”. In : *Archives of disease in childhood* 99.4, p. 386-389.

Stojanovic, Ljiljana, Nenad Stojanovic et Raphael Volz (2002). “Migrating Data-intensive Web Sites into the Semantic Web”. In : *Proceedings of the 2002 ACM Symposium on Applied Computing*. SAC ’02. Madrid, Spain : ACM, p. 1100-1107. isbn : 1-58113-445-2. doi : 10.1145/508791.509008. url : http://doi.acm.org/10.1145/508791.509008.

Suárez-Figueroa, Mari Carmen (2010). “NeOn Methodology for building ontology networks: specification, scheduling and reuse”. Thèse de doct. Informatica.

Suárez-Figueroa, Mari Carmen, Asunción Gómez-Pérez et Mariano Fernández-López (2012). “The NeOn methodology for ontology engineering”. In : *Ontology engineering in a networked world*. Springer, p. 9-34.

Thousand Genomes Project Consortium et al. (2015). “A global reference for human genetic variation”. In : *Nature* 526.7571, p. 68.

Turner, Tychele N et al. (2016). “denovo-db: A compendium of human de novo variants”. In : *Nucleic acids research* 45.D1, p. D804-D811.

Venter, J Craig et al. (2001). “The sequence of the human genome”. In : *science* 291.5507, p. 1304-1351.

Vrandečić, Denny et al. (2005). “The DILIGENT knowledge processes”. In : *Journal of Knowledge Management* 9.5, p. 85-96.

Wache, H. et al. (2001). “Ontology-Based Integration of Information - A Survey of Existing Approaches”. In : p. 108-117.

Wall, Dennis P et al. (2010). “Genotator: a disease-agnostic tool for genetic annotation of disease”. In : *BMC medical genomics* 3.1, p. 50.

Wang, Kai, Mingyao Li et Hakon Hakonarson (2010). "ANNOVAR: functional annotation of genetic variants from high-throughput sequencing data". In : *Nucleic acids research* 38.16, e164-e164.

Watson, James D et Francis HC Crick (1953). "The structure of DNA". In : *Cold Spring Harbor symposia on quantitative biology*. T. 18. Cold Spring Harbor Laboratory Press, p. 123-131.

Westerinen, Andrea et Rebecca Tauber (2017). "Ontology development by domain experts (without using the "O" word)". In : *Applied Ontology* 12.3-4, p. 299-311.

Wieczorek, Dagmar (2018). "Autosomal dominant intellectual disability". In : *medizinische genetik* 30.3, p. 318-322.

Wolfsberg, Tyra G, Johanna McEntyre et Gregory D Schuler (2001). "Guide to the draft human genome". In : *Nature* 409.6822, p. 824.

World Health Organization and others (2011). *World report on disability 2011*. World Health Organization.

Zerbino, Daniel R et al. (2017). "Ensembl 2018". In : *Nucleic acids research* 46.D1, p. D754-D761.

Annexe A

Liste des acronymes

ACM Association for Computing Machinery. 45

ACMG *American College of Human Genetics*. 27, 31, 38

ADN Acide DésoxyriboNucléique. 4, 5, 10, 11, 18, 19, 21–25, 28, 31, 47, 131

API *Application Programming Interface*. 56

ARN Acide RiboNucléique. 18, 19

ARNm ARN messager. 18–20

CADD *Combined Annotation Dependent Depletion*. 31, 36, 85, 100

CHARGE Coloboma, Heart defect, Atresia choanae, Retarded growth and development, Genital hypoplasia, Ear anomalies/deafness. 15

CHU Centre Hospitalo-Universitaire. 97, 131

CIGAR *Compact Idiosyncratic Gapped Alignment Report*. 25, 26

CNV *Copy Number Variation*. 34, 88, 89, 132, 138

COPUNG *Constructeur d'Ontologies Pour Utilisateur Novice Généticien*. iii, 64, 66, 135, 137, 138

CSS Cascading Style Sheet. 66

DANN *Deleterious Annotation of genetic variants using Neural Networks*. 100

DI Déficience Intellectuelle. iii, 6, 10, 14, 15, 27, 28, 36, 39, 99

ESE *Exonic Splicing Enhancer*. 19

ESS *Exonic Splicing Silencer*. 19

EUD End-User Development. 46

EUP *End-User Programming*. 46, 47, 93, 101, 135

GERP *Genomic Evolutionary Rate Profiling*. 36, 37

GO *Gene Ontology*. 39, 41, 44, 52, 56, 152

GPU *Graphics Processing Unit*. 5

GRC *Genome Reference Consortium*. 25

HCI Human Computer Interaction. 45

HPO *Human Phenotype Ontology*. v, 41, 44, 59–62, 71, 72, 92, 99, 152

HTML HyperText Markup Language. 66

IHM Interaction Homme Machine. iii, 8, 45, 46, 92, 101, 135

lncRNA *long non coding RNA*. 18

MIM *Mendeleian Inheritance in Man*. 38

MySQL My Structured Query Language. 66

NCBI *National Center for Biotechnology Information*. 25, 38, 39

NGS *Next Generation Sequencing*. 4, 21, 24, 100, 134, 135

NMD *Nonsense Mediated mRNA Decay*. 20

OBO *Open Biomedical Ontologies*. 55–57

OMIM *Online Mendeleian Inheritance in Man*. i, v, vi, 6, 38, 39, 44, 59–63, 71, 73, 76, 99, 112, 116, 117, 152

OWL *Web Ontology Language*. 40, 56, 152

PCR *Polymerase Chain Reaction*. 22, 23

PHP PHP Hypertext Preprocessor. 66

PLIB *Parts Library*. 40

RAM *Random Access Memory*. 4

RDFS *Ressource Description Framework Schema*. 40

RIHN Références des actes Innovants Hors Nomenclature. 28

SIFT *Sorting Intolerant From Tolerant*. 37, 91, 100

SNP *Single Nucleotide Polymorphism*. 39

SNV *Single Nucleotide Variation*. 26, 39, 72–75, 132

SPARQL *SPARQL Protocol And Rdf Query Language*. 51

SysID *Systems biology approaches to Intellectual Disability*. 6, 10, 15, 39, 88, 99, 114, 152

UCSC *University of Carolina Santa-Cruz*. 25

VCF *Variant Call Format*. 34, 35, 80, 81, 84, 87, 89–91, 93, 123

VEP *Variant Effect Predictor*. 34, 35, 152

Annexe B

Glossaire

bam format de fichier bioinformatique contenant les séquences alignées binarisées. 25, 26, 87

caller logiciel bio-informatique effectuant le *variant call*. 26

checkbox boîte à cocher présentant un choix non exclusif entre plusieurs options. 67

cluster groupe de fragments clonés lors du séquençage avec des technologies Illumina. 23, 24, 44

combobox liste déroulante présentant des choix finis et un champs de texte libre. 103

deep learning ensemble de méthodes d'apprentissage automatique. 32

fastq format de fichier bioinformatique contenant les *reads* issus du séquençage. v, 24, 25

flow cell support physique de la réaction chimique de séquençage. 4, 22, 23

genome browser logiciel permettant de visualiser le génome, d'y naviguer et d'y voir des informations supplémentaires. 87

machine learning apprentissage automatique, domaine de l'intelligence artificielle visant à faire apprendre à des machines à partir de grandes quantités de données. 31, 32, 36, 37, 44, 50, 51

mapping mise en correspondance de deux modèles de données. 42–45, 56

package archive comprenant tous les fichiers informatiques nécessaire à l'ajout d'une nouvelle fonctionnalité à un logiciel hôte. 35

paired-end type de séquençage ou les deux brins sont séquencés ensemble. 23, 25

parsing opération informatique d'extraction du contenu d'un fichier en vu d'un traitement informatique. 64–66, 105

phred score de qualité permettant d'évaluer le séquençage de l'ADN. 6, 24, 25, 93

pipeline enchainement d'outils et de scripts bio-informatiques connectant les sorties d'une étape aux entrées de l'étape suivante. 4, 8, 24, 39, 56, 93, 97, 98, 109, 132, 134

plugin greffon logiciel destiné à apporter une fonctionnalité supplémentaire à un logiciel hôte existant. 56

proxy logiciel interfaçant deux réseaux, très souvent un réseau d'entreprise ou privé et le Web. 88

radiobox boîte à cocher présentant un choix exclusif entre plusieurs options. 66

read traduit par "lecture", séquence d'ADN correspondant à la lecture d'un fragment, généralement de 150 paires de bases. v, 4, 5, 13, 24–26

sam type de fichier bio-informatique contenant les séquence alignées non binarisées. 25, 26

scoring fait d'associer un score à chaque entité dans un objectif de tri ou de seuil. 83, 84, 88, 100, 101

trimmer opération bio-informatique consistant à couper les bases à l'extrémité d'un *read*. 24

variant call opération bio-informatique consistant à extraire la liste des variations à partir des séquences alignées. 5, 149

Annexe C

Table des adresses URL

Outil	URL
WebANNOVAR	http://wannovar.wglab.org/
Ensembl	https://www.ensembl.org/index.html
RefSeq	https://www.ncbi.nlm.nih.gov/refseq/
Microsoft Access	https://products.office.com/fr-fr/access
Google sheets	https://www.google.fr/intl/fr/sheets/about/
Framacalc	https://accueil.framacalc.org/fr/
PubMed	https://www.ncbi.nlm.nih.gov/pubmed/
Alliance maladies rares	https://www.alliance-maladies-rares.org/lerrance-de-diagnostic/
Direct Mapping	https://www.w3.org/TR/rdb-direct-mapping
Alamut	https://www.interactive-biosoftware.com
R2RML	https://www.w3.org/TR/r2rml
CNRTL	https://www.cnrtl.fr/etymologie/score
PanelAPP	https://panelapp.genomicsengland.co.uk
Orphanet	https://www.orpha.net/
OWL	http://www.w3.org/TR/owl-guide/
Variant Effect Predictor	https://www.ensembl.org/info/docs/tools/vep/index.html
Exac	http://exac.broadinstitute.org/
VaRank	http://www.lbgi.fr/VaRank/
ANNOVAR	http://annovar.openbioinformatics.org/en/latest/
Varaft	https://varaft.eu
QueryOR	http://queryor.cribi.unipd.it/cgi-bin/queryor/mainpage.pl
VCF-Miner	http://bioinformaticstools.mayo.edu/research/vcf-miner/
BierApp	http://bierapp.babelomics.org/
dbSNP	https://www.ncbi.nlm.nih.gov/snp/
Human Phenotype Ontology	https://hpo.jax.org/app/
Systems biology approaches to Intellectual Disability	https://sysid.cmbi.umcn.nl
Online Mendeleian Inheritance in Man	https://www.omim.org
Gene Ontology	http://geneontology.org/
GnomAD	https://gnomad.broadinstitute.org

Tableau C.1 – Tableau récapitulatif des adresses url des outils utilisés

Annexe D

rulesets des utilisateurs

Listing D.1 – *ruleset* de l'utilisateur *T*.

```
1;CLINSIG/;Contains//;enign//;BASE//;0/;Decrease;100
1;Varank_score/;<//;40//;BASE//;0/;Decrease;50
1;Intervar/;Contains//;enign//;BASE//;0/;Decrease;100
1;Nbre_homozygote/;>//;3//;BASE//;0/;Decrease;100
1;Nbre_homozygote/;>//;1//;BASE//;0/;Decrease;50
1;Nbre_heterozygote/;>//;5//;BASE//;0/;Decrease;100
1;Nbre_allele/;>//;10//;BASE//;0/;Decrease;100
1;Allele_gnomad/;>//;50//;BASE//;0/;Decrease;180
1;Couverture/;<//;25//;BASE//;0/;Decrease;50
1;Presence_pc/;<//;25//;BASE//;0/;Decrease;50
1;gnomad_all/;>//;50//;BASE//;0/;Decrease;180
1;Frequence_allele/;>//;0.02//;BASE//;0/;Decrease;100
1;CLINSIG/;Contains//;athogenic//;BASE//;0/;Increase;200
1;Effet_mutation/;=//;stop_gained//;BASE//;0/;Increase;50
1;Effet_mutation/;=//;frameshift_variant//;BASE//;0/;Increase;50
1;Effet_mutation/;=//;start_lost//;BASE//;0/;Increase;50
1;Effet_mutation/;=//;stop_lost//;BASE//;0/;Increase;50
1;Effet_mutation/;=//;stop_retained_variant//;BASE//;0/;Increase;50
1;Denovo-db/;<//;20//;BASE//;0/;Increase;25
1;Denovo-db/;>//;-20//;BASE//;0/;Increase;25
1;Effet_mutation/;=//;splice_acceptor_variant//;BASE//;0/;Increase;160
1;Effet_mutation/;=//;splice_donor_variant//;BASE//;0/;Increase;160
1;Couverture/;<//;10//;BASE//;0/;Decrease;100
1;Presence_pc/;<//;10//;BASE//;0/;Decrease;100
1;panel/;Don't␣contains//;.//;BASE//;0/;Increase;20
1;phenotype_PanelApp/;Don't contains//;.//;BASE//;0/;Increase;20
1;syndrome_panelApp/;Don't␣contains//;.//;BASE//;0/;Increase;20
1;CLINSIG/;=//;Uncertain_significance//;BASE//;0/;Increase;10
1;CLINSIG/;=//;Conflicting_interpretations_of_pathogenicity//
␣␣␣␣␣␣␣␣;BASE//;0/;Increase;10
1;CLINSIG/;=//;Conflicting_interpretations_of_pathogenicity,_risk_factor//
␣␣␣␣␣␣␣␣;BASE//;0/;Increase;10
1;Phred_score/;<//;200//;BASE//;0/;Decrease;150
1;Effet_mutation/;Contains//;UTR//;BASE//;0/;Decrease;100
1;Effet_mutation/;Contains//;stream//;BASE//;0/;Decrease;100
1;Effet_mutation/;=//;intron_variant//;BASE//;0/;Decrease;100
1;gnomad_all/;<=//;2//;BASE//;0/;Increase;40
1;Allele_gnomad/;<=//;2//;BASE//;0/;Increase;40
1;Allele_gnomad/;>//;100//;BASE//;0/;Decrease;120
1;gnomad_all/;>//;100//;BASE//;0/;Decrease;120
1;nb_index/;>//;10//;BASE//;0/;Decrease;180
1;nb_index/;=//;0//;BASE//;0/;Increase;50
1;nb_patho/;!=//;0//;BASE//;0/;Increase;200
```

```
1;Effet_mutation/;=//;.//;BASE//;0/;Decrease;500
1;Phred_score/;>/;250/;BASE/;0/;Increase;10
1;Intervar/;Contains/;significance/;BASE/;0/;Increase;20
1;Intervar/;Contains/;athogenic/;BASE/;0/;Increase;50
1;CLNSIGCONF/;Contains/;enign/;BASE/;0/;Decrease;30
1;Effet_mutation/;=/;splice_region_variant/;BASE/;0/;Increase;80
1;Haplotype/;=/;hom/;BASE/;0/;Increase;55
```

Listing D.2 – *ruleset* dominant (LD) de l'utilisateur *L*.

```
1;Allele_gnomad/Nbre_allele/;<=/<=/;3/2/;AND/AND/;0/0/;Augmenter;200
1;Allele_gnomad/Nbre_allele/;contient/<=/;./2/
        ;AND/AND/;0/0/;Augmenter;200
1;Couverture/;<=/;10/;AND/;0/;Diminuer;300
1;Presence_pc/;<=/;30/;AND/;0/;Diminuer;300
1;CLINSIG/;contient/;athogenic/;AND/;0/;Augmenter;100
1;CLINSIG/;=/;Benign/;AND/;0/;Diminuer;100
1;CLINSIG/;=/;Likely_benign/;AND/;0/;Diminuer;30
1;CLINSIG/;=/;Benign/Likely_benign/;AND/;0/;Diminuer;100
1;Intervar/;contient/;athogenic/;AND/;0/;Augmenter;100
1;Intervar/;=/;Benign/;AND/;0/;Diminuer;100
1;Effet_mutation/;contient/;intron/;AND/;0/;Diminuer;300
1;Effet_mutation/;contient/;prime/;AND/;0/;Diminuer;300
1;Effet_mutation/;contient/;intragenic/;AND/;0/;Diminuer;300
1;Effet_mutation/;contient/;intergenic/;AND/;0/;Diminuer;300
1;Effet_mutation/;contient/;stream/;AND/;0/;Diminuer;300
1;Effet_mutation/;=/;non_coding_exon_variant/;AND/;0/;Diminuer;300
1;Effet_mutation/;=/;TF_binding_site_variant/;AND/;0/;Diminuer;300
1;Varank_score/;=/;100/;AND/;0/;Augmenter;100
1;Varank_score/;=/;80/;AND/;0/;Augmenter;80
1;Varank_score/;=/;50/;AND/;0/;Augmenter;50
1;Varank_score/;=/;40/;AND/;0/;Augmenter;40
1;Domino_score/;>=/;0.7/;AND/;0/;Augmenter;100
1;dbscSNV_ADA/;>=/;0.8/;AND/;0/;Augmenter;100
1;Hypothese_transmission/;contient/;MONOALLELIC/;AND/;0/;Augmenter;100
1;Hypothese_transmission/;contient/;monoallelic/;AND/;0/;Augmenter;100
1;Allele_gnomad/;>/;3/;AND/;0/;Diminuer;500
1;Phred_score/;</;30/;AND/;0/;Diminuer;500
1;role_DI/;contient/;non/;AND/;0/;Diminuer;150
```

Listing D.3 – *ruleset* lié au chromosome X (LX) de l'utilisateur *L*.

```
1;ID_MT/;ne contient pas/;chrX/;AND/;0/;Diminuer;1000
1;Allele_gnomad/Nbre_allele/;<=/<=/;25/12/;AND/AND/;0/0/;Augmenter;200
1;Allele_gnomad/Nbre_allele/;contient/<=/;./12/
        ;AND/AND/;0/0/;Augmenter;200
1;Couverture/;<=/;10/;AND/;0/;Diminuer;300
1;Presence_pc/;<=/;30/;AND/;0/;Diminuer;300
1;CLINSIG/;contient/;athogenic/;AND/;0/;Augmenter;100
1;Intervar/;contient/;athogenic/;AND/;0/;Augmenter;100
1;Intervar/;=/;Benign/;AND/;0/;Diminuer;100
1;Effet_mutation/;contient/;intron/;AND/;0/;Diminuer;100
1;Effet_mutation/;contient/;prime/;AND/;0/;Diminuer;100
1;Effet_mutation/;contient/;intragenic/;AND/;0/;Diminuer;100
1;Effet_mutation/;contient/;intergenic/;AND/;0/;Diminuer;100
1;Effet_mutation/;contient/;stream/;AND/;0/;Diminuer;100
1;Effet_mutation/;=/;non_coding_exon_variant/;AND/;0/;Diminuer;100
1;Effet_mutation/;=/;TF_binding_site_variant/;AND/;0/;Diminuer;100
1;Varank_score/;=/;100/;AND/;0/;Augmenter;100
1;Varank_score/;=/;80/;AND/;0/;Augmenter;80
```

```
1;Varank_score/;=/;50/;AND/;0/;Augmenter;50
1;Varank_score/;=/;40/;AND/;0/;Augmenter;40
1;dbscSNV_ADA/;>=/;0.8/;AND/;0/;Augmenter;100
```

Listing D.4 – *ruleset* récéssif (LR) de l'utilisateur *L*.

```
1;Allele_gnomad/Nbre_allele/;<=/<=/;25/15/;AND/AND/;0/0/;Augmenter;200
1;Allele_gnomad/Nbre_allele/;contient/<=/;./15/;AND/AND/;0/0/
        ;Augmenter;200
1;Allele_gnomad/;>/;25/;AND/;0/;Diminuer;500
1;Nbre_allele/;>=/;30/;AND/;0/;Diminuer;200
1;Nbre_homozygote/;>=/;3/;AND/;0/;Diminuer;200
1;CLINSIG/;contient/;athogenic/;AND/;0/;Augmenter;150
1;Intervar/;contient/;athogenic/;AND/;0/;Augmenter;150
1;Couverture/;<=/;10/;AND/;0/;Diminuer;500
1;Presence_pc/;<=/;30/;AND/;0/;Diminuer;500
1;Effet_mutation/;contient/;prime/;AND/;0/;Diminuer;200
1;Effet_mutation/;contient/;stream/;AND/;0/;Diminuer;200
1;Effet_mutation/;contient/;intragenic/;AND/;0/;Diminuer;200
1;Effet_mutation/;contient/;intergenic/;AND/;0/;Diminuer;200
1;Effet_mutation/;contient/;intron/;AND/;0/;Diminuer;200
1;Effet_mutation/;=/;TF_binding_site_variant/;AND/;0/;Diminuer;200
1;Effet_mutation/;=/;non_coding_exon_variant/;AND/;0/;Diminuer;200
1;Hypothese_transmission/;contient/;BIALLELIC/;AND/;0/;Augmenter;100
1;Varank_score/;=/;100/;AND/;0/;Augmenter;100
1;Varank_score/;=/;80/;AND/;0/;Augmenter;80
1;Varank_score/;=/;50/;AND/;0/;Augmenter;50
1;Varank_score/;=/;40/;AND/;0/;Augmenter;40
1;Haplotype/;=/;hom/;AND/;0/;Augmenter;200
1;DANN_pred/;>=/;0.600/;AND/;0/;Augmenter;50
1;role_DI/;contient/;oui/;AND/;0/;Augmenter;100
1;Revel/;<=/;0.700/;AND/;0/;Diminuer;100
1;dbscSNV_ADA/;>=/;0.8/;AND/;0/;Augmenter;100
```

Listing D.5 – *ruleset de novo* (GDN) de l'utilisateur *G*.

```
1;Phred_score/;</;200/;BASE/;0/;Decrease;500
1;Couverture/;</;10/;BASE/;0/;Decrease;500
1;Presence_pc/;</;30/;BASE/;0/;Decrease;500
1;Nbre_allele/;>/;2/;AND/;0/;Decrease;500
1;nb_index/;>/;1/;AND/;0/;Decrease;500
1;nb_index/;=/;0/;BASE/;0/;Increase;200
1;nb_patho/;!=/;0/;BASE/;0/;Increase;200
1;Varank_score/;=/;./;BASE/;0/;Decrease;500
1;Allele_gnomad/;=/;0/;BASE/;0/;Increase;200
1;Allele_gnomad/;>=/;1/;BASE/;0/;Decrease;500
1;Hypothese_transmission/Domino_score/;Contains/>=/
        ;MONO/0.5/;AND/OR/;0/1/;Increase;100
1;Hypothese_transmission/;Contains/;BIALLELIC/;BASE/;0/;Decrease;100
1;Hypothese_transmission/;Contains/;both/;BASE/;0/;Increase;100
1;ID_MT/;Contains/;chrX/;BASE/;0/;Decrease;300
1;PLI/Varank_score/;>=/>=/;0.9/100/;AND/AND/;0/0/;Increase;50
1;RS_id/;=/;./;AND/;0/;Increase;100
1;Varank_score/;>=/;100/;AND/;0/;Increase;160
1;Varank_score/;<=/;10/;BASE/;0/;Decrease;50
1;Varank_score/PLI/;>=/</;100/0.9/;AND/AND/;0/0/;Decrease;100
1;Effet_mutation/;=/;splice_donnor_variant/;BASE/;0/;Increase;100
1;Effet_mutation/;=/;splice_acceptor_variant/;BASE/;0/;Increase;100
1;Effet_mutation/;=/;disruptive_inframe_deletion/;BASE/;0/;Increase;40
1;Effet_mutation/GERP++_pred/;=/>=/;inframe_deletion/6/
```

```
        ;AND/AND//;0/0/;Increase;50
1;Effet_mutation/;=//;upstream_gene_variant//;BASE//;0/;Decrease;200
1;Effet_mutation/;=//;downstream_gene_variant//;BASE//;0/;Decrease;200
1;Effet_mutation/;=//;5_prime_UTR_variant//;BASE//;0/;Decrease;200
1;Effet_mutation/;=//;5_prime_UTR_premature_start_codon_gain_variant//
        ;BASE//;0/;Decrease;200
1;Effet_mutation/;=//;3_prime_UTR_variant//;AND//;0/;Decrease;200
1;Effet_mutation/;=//;synonymous_variant//;BASE//;0/;Decrease;200
1;Effet_mutation/dbscSNV_ADA/;=/>=//;synonymous_variant/0.1//
        ;AND/AND//;0/0/;Increase;250
1;Effet_mutation/dbscSNV_ADA/;=/>=//;synonymous_variant/0.1//
        ;AND/AND//;0/0/;Increase;250
1;Effet_mutation/;=//;missense_variant//;BASE//;0/;Increase;50
1;Effet_mutation/dbscSNV_ADA/dbscSNV_RF/;=/>/>//
        ;splice_region_variant/0.2/0.2//;AND/AND/AND//;0/0/0/;Increase;50
1;GERP++_pred/;>//;0.8//;BASE//;0/;Increase;20
1;phastCons_mammalian_pred/;>=//;0.6//;AND//;0/;Increase;5
1;phylop_mammalian_score/;>=//;0.8//;AND//;0/;Increase;5
1;Grantham/;>=//;80//;AND//;0/;Increase;5
1;Groupe_DI/;!=//;.//;BASE//;0/;Increase;20
1;GTEX_v6_tissue/;>=//;0.1//;BASE//;0/;Increase;20
1;Denovo-db/;!=//;./;BASE//;0/;Increase;20
1;Interpro domaine/;!=//;./;BASE//;0/;Increase;20
1;M-CAP/;>=//;0.025//;BASE//;0/;Increase;30
1;CADD_pred/;>=//;20//;BASE//;0/;Increase;30
1;CADD_pred/;>=/;30/;BASE/;0/;Increase;50
1;Revel/;>=//;0.4//;BASE//;0/;Increase;30
1;CLINSIG/;Contains//;benign//;AND//;0/;Decrease;100
1;CLINSIG/;Contains//;pathogenic//;AND//;0/;Increase;100
1;Gene_details/;!=/;./;BASE/;0/;Decrease;100
```

Listing D.6 – *ruleset* autosomique dominant (GAD) de l'utilisateur *G*.

```
1;Phred_score/;<//;200//;BASE//;0/;Decrease;500
1;Couverture/;<//;10//;BASE//;0/;Decrease;500
1;Presence_pc/;<//;30//;BASE//;0/;Decrease;500
1;Nbre_allele/;>//;2//;AND//;0/;Decrease;500
1;nb_index/;>//;2//;AND//;0/;Decrease;500
1;nb_index/;=//;1//;BASE//;0/;Increase;200
1;nb_patho/;!=//;0//;BASE//;0/;Increase;200
1;Varank_score/;=//;.//;BASE//;0/;Decrease;500
1;Allele_gnomad/;>//;1//;BASE//;0/;Decrease;200
1;Allele_gnomad/;=//;1//;BASE//;0/;Increase;50
1;Allele_gnomad/;=//;0//;BASE//;0/;Increase;200
1;Hypothese_transmission/Domino_score/;Contains/>=//;MONO/0.5//
        ;AND/OR//;0/1/;Increase;50
1;Hypothese_transmission/;Contains//;BIALLELIC//;BASE//;0/;Decrease;50
1;Hypothese_transmission/;Contains//;both//;BASE//;0/;Increase;50
1;ID_MT/;Contains//;chrX//;BASE//;0/;Decrease;300
1;PLI/Varank_score/;>=/>=//;0.9/100//;AND/AND//;0/0/;Increase;50
1;Varank_score/;>=//;100//;AND//;0/;Increase;160
1;Varank_score/;<=//;10//;BASE//;0/;Decrease;50
1;Varank_score/PLI/;>=/<//;100/0.9//;AND/AND//;0/0/;Decrease;100
1;Effet_mutation/;=//;splice_donnor_variant//;BASE//;0/;Increase;100
1;Effet_mutation/;=//;splice_acceptor_variant//;BASE//;0/;Increase;100
1;Effet_mutation/;=//;disruptive_inframe_deletion//;BASE//;0/;Increase;40
1;Effet_mutation/;=//;upstream_gene_variant//;BASE//;0/;Decrease;100
1;Effet_mutation/;=//;downstream_gene_variant//;BASE//;0/;Decrease;100
1;Effet_mutation/;=//;5_prime_UTR_variant//;BASE//;0/;Decrease;100
1;Effet_mutation/;=//;5_prime_UTR_premature_start_codon_gain_variant//
```

```
        ;BASE//;0/;Decrease;100
1;Effet_mutation/;=//;3_prime_UTR_variant//;AND//;0/;Decrease;100
1;Effet_mutation/;=//;synonymous_variant//;BASE//;0/;Decrease;200
1;Effet_mutation/dbscSNV_ADA/;=/>=//;synonymous_variant/0.1//
        ;AND/AND//;0/0/;Increase;250
1;Effet_mutation/dbscSNV_ADA/;=/>=//;synonymous_variant/0.1//
        ;AND/AND//;0/0/;Increase;250
1;Effet_mutation/GERP++_pred/;=/>=//;inframe_deletion/6//
        ;AND/AND//;0/0/;Increase;50
1;Effet_mutation/;=//;missense_variant//;BASE//;0/;Increase;50
1;Effet_mutation/dbscSNV_ADA/dbscSNV_RF/;=/>/>//
        ;splice_region_variant/0.2/0.2//;AND/AND/AND//;0/0/0/;Increase;50
1;GERP++_pred/;>//;0.8//;BASE//;0/;Increase;20
1;phastCons_mammalian_pred/;>=//;0.6//;AND//;0/;Increase;5
1;phylop_mammalian_score/;>=//;0.8//;AND//;0/;Increase;5
1;Grantham/;>=//;80//;AND//;0/;Increase;5
1;Groupe_DI/;!=//;.//;BASE//;0/;Increase;20
1;GTEX_v6_tissue/;>=//;0.1//;BASE//;0/;Increase;20
1;Denovo-db/;!=//;.//;BASE//;0/;Increase;20
1;Interpro domaine/;!=//;.//;BASE//;0/;Increase;20
1;M-CAP/;>=//;0.025//;BASE//;0/;Increase;30
1;CADD_pred/;>=//;20//;BASE//;0/;Increase;30
1;CADD_pred/;>=/;30/;BASE/;0/;Increase;50
1;Revel/;>=//;0.4//;BASE//;0/;Increase;30
1;CLINSIG/;Contains//;benign//;AND//;0/;Decrease;100
1;CLINSIG/;Contains//;pathogenic//;AND//;0/;Increase;100
1;Gene_details/;!=/;./;BASE/;0/;Decrease;100
```

Listing D.7 – *ruleset* autosomique récessif (GAR) de l'utilisateur *G*.

```
1;Phred_score/;<//;200//;BASE//;0/;Decrease;500
1;Couverture/;<//;10//;BASE//;0/;Decrease;500
1;Presence_pc/;<//;30//;BASE//;0/;Decrease;500
0;Nbre_allele/;=//;1//;AND//;0/;Increase;200
1;Nbre_allele/;>//;2//;AND//;0/;Decrease;500
1;nb_index/;>//;5//;AND//;0/;Decrease;500
1;nb_index/;<//;3//;BASE//;0/;Increase;200
1;nb_patho/;!=//;0//;BASE//;0/;Increase;200
1;Varank_score/;=//;.//;BASE//;0/;Decrease;500
1;Allele_gnomad/;=//;0//;BASE//;0/;Increase;200
1;Allele_gnomad/;<//;100//;BASE//;0/;Increase;50
1;Allele_gnomad/;>//;100//;BASE//;0/;Decrease;300
1;Presence_pc/Allele_gnomad/;>=/=//;80/0//;AND/AND//;0/0/;Increase;100
1;Presence_pc/Allele_gnomad/;>=/<//;80/100//;AND/AND//;0/0/;Increase;75
1;Hypothese_transmission/Domino_score/;Contains/<=//;BIALLELIC/0.3//
        ;AND/OR//;0/1/;Increase;100
1;Hypothese_transmission/;Contains//;MONOALLELIC//;BASE//;0/;Decrease;200
1;ID_MT/;Contains//;chrX//;BASE//;0/;Decrease;200
1;Effet_mutation/dbscSNV_ADA/dbscSNV_RF/;=/>/>//
        ;splice_region_variant/0.2/0.2//;AND/AND/AND//;0/0/0/;Increase;50
1;Varank_score/;>=//;100//;AND//;0/;Increase;160
1;Varank_score/PLI/;>=/<//;100/0.9//;AND/AND/AND/;0/0/;Increase;100
1;Effet_mutation/;=//;splice_donnor_variant//;BASE//;0/;Increase;100
1;Effet_mutation/;=//;splice_acceptor_variant//;BASE//;0/;Increase;100
1;Effet_mutation/;=//;disruptive_inframe_deletion//;BASE//;0/;Increase;40
1;Effet_mutation/;=//;upstream_gene_variant//;BASE//;0/;Decrease;100
1;Effet_mutation/;=//;downstream_gene_variant//;BASE//;0/;Decrease;100
1;Effet_mutation/;=//;5_prime_UTR_variant//;BASE//;0/;Decrease;100
1;Effet_mutation/;=//;5_prime_UTR_premature_start_codon_gain_variant//
        ;BASE//;0/;Decrease;100
```

```
1;Effet_mutation/;=//;3_prime_UTR_variant//;AND//;0/;Decrease;100
1;Effet_mutation/;=//;synonymous_variant//;BASE//;0/;Decrease;200
1;Effet_mutation/dbscSNV_ADA/;=/>=//;synonymous_variant/0.1//
        ;AND/AND//;0/0/;Increase;250
1;Effet_mutation/dbscSNV_ADA/;=/>=//;synonymous_variant/0.1//
        ;AND/AND//;0/0/;Increase;250
1;Effet_mutation/PLI/;=/<//;splice_donor_variant/0.9//
        ;AND/AND//;0/0/;Increase;100
1;Effet_mutation/PLI/;=/<//;splice_acceptor_variant/0.9//
        ;AND/AND//;0/0/;Increase;100
1;Effet_mutation/;=//;missense_variant//;BASE//;0/;Increase;50
1;Effet_mutation/dbscSNV_ADA/dbscSNV_RF/;=/>/>//
        ;splice_region_variant/0.2/0.2//;AND/AND/AND//;0/0/0/;Increase;50
1;GERP++_pred/;>//;0.8//;BASE//;0/;Increase;20
1;phastCons_mammalian_pred/;>=//;0.6//;AND//;0/;Increase;5
1;phylop_mammalian_score/;>=//;0.8//;AND//;0/;Increase;5
1;Grantham/;>=//;80//;AND//;0/;Increase;5
1;Groupe_DI/;!=//;.//;BASE//;0/;Increase;100
1;GTEX_v6_tissue/;>=//;0.1//;BASE//;0/;Increase;20
1;Interpro domaine/;!=//;.//;BASE//;0/;Increase;20
1;M-CAP/;>=//;0.025//;BASE//;0/;Increase;30
1;CADD_pred/;>=//;20//;BASE//;0/;Increase;30
1;CADD_pred/;>=/;30/;BASE/;0/;Increase;50
1;Revel/;>=//;0.4//;BASE//;0/;Increase;30
1;CLINSIG/;Contains//;benign//;AND//;0/;Decrease;100
1;CLINSIG/;Contains//;pathogenic//;AND//;0/;Increase;100
1;Gene_details/;!=/;./;BASE/;0/;Decrease;100
```

Listing D.8 – *ruleset* lié au chromosome X (GX) de l'utilisateur *G*.

```
1;Phred_score/;<//;200//;BASE//;0/;Decrease;500
1;Couverture/;<//;10//;BASE//;0/;Decrease;500
1;Presence_pc/;<//;30//;BASE//;0/;Decrease;500
1;Presence_pc/Allele_gnomad/ID_MT/ID_MT/;>=/=//Contains//
        ;80/0//chrX//;AND/AND//AND/AND/;0/0/0/0/;Increase;200
1;Presence_pc/Allele_gnomad/ID_MT/ID_MT/;>=/<//Contains//
        ;80/10//chrX//;AND/AND//AND/AND/;0/0/0/0/;Increase;100
1;ID_MT/nb_index/;Contains/>//;chrX/10//;AND/AND//;0/0/;Decrease;200
1;ID_MT/nb_index/;Contains/<//;chrX/4//;AND/AND//;0/0/;Increase;200
1;nb_patho/;!=//;0//;BASE//;0/;Increase;200
1;Varank_score/;=//;.//;BASE//;0/;Decrease;500
1;ID_MT/Allele_gnomad/;Contains/=//;chrX/0//;AND/AND//;0/0/;Increase;200
1;ID_MT/allele_gnomad_male/allele_gnomad_female/;Contains/=/<//
        ;chrX/./10//;AND/AND/AND//;0/0/0/;Increase;150
1;ID_MT/;Don't␣contains//;chrX//;BASE//;0/;Decrease;500
1;Effet_mutation/;=//;splice_acceptor_variant//;BASE//;0/;Increase;100
1;Effet_mutation/;=//;splice_donnor_variant//;BASE//;0/;Increase;100
1;Effet_mutation/;=//;disruptive_inframe_deletion//;BASE//;0/;Increase;40
1;Effet_mutation/GERP++_pred/;=/>=//;inframe_deletion/6//
␣␣␣␣␣␣␣␣;AND/AND//;0/0/;Increase;50
1;Effet_mutation/;=//;3_prime_UTR_variant//;AND//;0/;Decrease;100
1;Effet_mutation/;=//;5_prime_UTR_premature_start_codon_gain_variant//
␣␣␣␣␣␣␣␣;BASE//;0/;Decrease;100
1;Effet_mutation/;=//;5_prime_UTR_variant//;BASE//;0/;Decrease;100
1;Effet_mutation/;=//;downstream_gene_variant//;BASE//;0/;Decrease;100
1;Effet_mutation/;=//;upstream_gene_variant//;BASE//;0/;Decrease;100
1;Effet_mutation/;=//;synonymous_variant//;BASE//;0/;Decrease;200
1;Effet_mutation/dbscSNV_ADA/;=/>=//;synonymous_variant/0.1//
␣␣␣␣␣␣␣␣;AND/AND//;0/0/;Increase;250
1;Effet_mutation/dbscSNV_ADA/;=/>=//;synonymous_variant/0.1//
```

```
␣␣␣␣␣␣␣␣;AND/AND//;0/0/;Increase;250
1;Effet_mutation/dbscSNV_ADA/dbscSNV_RF/;=/>/>//
␣␣␣␣␣␣␣␣;splice_region_variant/0.2/0.2//;AND/AND/AND//;0/0/0/;Increase;50
1;Effet_mutation/;=//;missense_variant//;BASE//;0/;Increase;50
1;Varank_score/;>=//;100//;AND//;0/;Increase;160
1;PLI/;>=//;0.9//;BASE//;0/;Increase;50
1;PLI/Varank_score/;>=/>=//;0.9/100//;AND/AND//;0/0/;Increase;50
1;GTEX_v6_tissue/;>=//;0.1//;BASE//;0/;Increase;20
1;Groupe_DI/;!=//;.//;BASE//;0/;Increase;100
1;Revel/;>=//;0.4//;BASE//;0/;Increase;30
1;CADD_pred/;>=//;20//;BASE//;0/;Increase;30
1;CADD_pred/;>=//;30//;BASE//;0/;Increase;50
1;M-CAP/;>=//;0.025//;BASE//;0/;Increase;30
1;GERP++_pred/;>//;0.8//;BASE//;0/;Increase;20
1;phastCons_mammalian_pred/;>=//;0.6//;AND//;0/;Increase;5
1;phylop_mammalian_score/;>=//;0.8//;AND//;0/;Increase;5
1;Grantham/;>=//;80//;AND//;0/;Increase;5
1;Interpro␣domaine/;!=//;.//;BASE//;0/;Increase;20
1;CLINSIG/;Contains//;pathogenic//;AND//;0/;Increase;100
1;CLINSIG/;Contains//;benign//;AND//;0/;Decrease;100
1;Gene_details/;!=//;.//;BASE//;0/;Decrease;100
1;Nbre_allele/;>/;2/;BASE/;0/;Decrease;500
```

Listing D.9 – *ruleset* dominant "recherche" (FDR) de l'utilisateur *F*.

```
1;Nbre_allele/;<=//;2//;AND//;0/;Augmenter;30
1;Allele_gnomad/;=//;0//;0//;;Augmenter;40
1;Allele_gnomad/;<=//;10//;AND//;0/;Augmenter;10
1;CLINSIG/;contient//;athogen//;AND//;0/;Augmenter;40
1;RS_id/;contient//;rs//;AND//;0/;Diminuer;30
1;Couverture/;<=//;10//;AND//;0/;Diminuer;100
1;Presence_pc/;<=//;30//;AND//;0/;Diminuer;50
2;Varank_score/;>=//;10//;AND//;0/;Augmenter;10
1;Nbre_allele/;>=//;39//;AND//;0/;Diminuer;100
1;role_DI/;contient//;oui(vert)//;AND//;0/;Augmenter;30
1;Grantham/;>//;80//;AND//;0/;Augmenter;20
1;Effet_mutation/;contient//;prime//;AND//;0/;Diminuer;100
1;Allele_gnomad/;>=//;15//;AND//;0/;Diminuer;100
1;Effet_mutation/;contient//;intron//;AND//;0/;Diminuer;50
1;Sift_pred/;=//;D//;AND//;0/;Augmenter;5
1;Provean_pred/;=//;D//;AND//;0/;Augmenter;5
1;Polyphen_HDIV_pred/;=//;D//;AND//;0/;Augmenter;5
1;Polyphen_HVAR_pred/;=//;P//;AND//;0/;Augmenter;5
1;LRT_pred/;=//;D//;AND//;0/;Augmenter;5
1;Mutation_taster_pred/;=//;D//;AND//;0/;Augmenter;5
1;FATHMM_pred/;=//;D//;AND//;0/;Augmenter;5
1;FATHMM_MKL_pred/;=//;D//;AND//;0/;Augmenter;5
1;Meta_SVP_pred/;=//;D//;AND//;0/;Augmenter;5
1;Presence_pc/;<//;20//;AND//;0/;Diminuer;100
1;esp_all/;ne contient pas//;.//;AND//;0/;Diminuer;50
1;1000g_all/;ne contient pas//;.//;AND//;0/;Diminuer;50
1;Varank_score/Nbre_allele/Allele_gnomad/;>=/<=/<=//
        ;60/2/10//;AND/AND/AND//;0/0/0/;Augmenter;50
1;Effet_mutation/Nbre_allele/Allele_gnomad/;contient/<=/<=//
        ;acceptor/2/10//;AND/AND/AND//;0/0/0/;Augmenter;30
1;Effet_mutation/Nbre_allele/Allele_gnomad/;contient/<=/<=//
        ;donor/2/10//;AND/AND/AND//;0/0/0/;Augmenter;30
1;CLINSIG/CLNSIGCONF/CLNREVSTAT/Nbre_allele/
        ;contient/contient/contient/<=//
        ;athogen/athogen/criteria_provided
```

```
        ,_multiple_submitters,_no_conflicts/4//
        ;AND/AND/AND/AND//;0/0/0/0/;Augmenter;100
1;Varank_score/Nbre_allele/;>=/>//;50/3//;AND/AND//;0/0/;Diminuer;50
1;Varank_score/Allele_gnomad/ID_MT/;>=/>=/ne contient pas//
        ;50/10/chrX//;AND/AND/AND//;0/0/0/;Diminuer;50
1;Intervar/;contient//;athogen//;AND//;0/;Augmenter;30
1;CADD_pred/;>=//;20//;AND//;0/;Augmenter;30
1;Effet_mutation/;contient//;downstream//;AND//;0/;Diminuer;100
1;Effet_mutation/;contient//;upstream//;AND//;0/;Diminuer;100
1;Varank_score/Nbre_allele/Allele_gnomad/RS_id/
        ;>=/<=/contient/ne contient pas//;60/2/./rs//
        ;AND/AND/AND/AND//;0/0/0/0/;Augmenter;50
1;Effet_mutation/Allele_gnomad/RS_id/;contient/contient/ne contient pas//
        ;acceptor/./rs//;AND/AND/AND//;0/0/0/;Augmenter;30
1;Effet_mutation/Allele_gnomad/RS_id/;contient/contient/ne contient pas//
        ;donor/./rs//;AND/AND/AND//;0/0/0/;Augmenter;30
1;Denovo-db/Allele_gnomad/RS_id/
        ;ne contient pas/contient/ne contient pas//
        ;././rs//;AND/AND/AND//;0/0/0/;Augmenter;30
1;Domino_score/;>=//;0.60//;AND//;0/;Augmenter;20
```

Listing D.10 – *ruleset* dominant "OMIM" (FDO) de l'utilisateur *F*.

```
1;Nbre_allele/;<=//;2//;AND//;0/;Augmenter;30
1;Allele_gnomad/;contient//;.//;AND//;0/;Augmenter;40
1;Allele_gnomad/;<=//;10//;AND//;0/;Augmenter;10
1;CLINSIG/;contient//;athogen//;AND//;0/;Augmenter;40
1;RS_id/;contient//;rs//;AND//;0/;Diminuer;30
1;Couverture/;<=//;10//;AND//;0/;Diminuer;100
1;Presence_pc/;<=//;30//;AND//;0/;Diminuer;50
2;Varank_score/;>=//;10//;AND//;0/;Augmenter;10
1;Nbre_allele/;>=//;4//;AND//;0/;Diminuer;100
1;role_DI/;contient//;oui(vert)//;AND//;0/;Augmenter;30
1;Grantham/;>//;80//;AND//;0/;Augmenter;20
1;Effet_mutation/;contient//;prime//;AND//;0/;Diminuer;100
1;Allele_gnomad/;>=//;15//;AND//;0/;Diminuer;100
1;Effet_mutation/;contient//;intron//;AND//;0/;Diminuer;50
1;Sift_pred/;=//;D//;AND//;0/;Augmenter;5
1;Provean_pred/;=//;D//;AND//;0/;Augmenter;5
1;Polyphen_HDIV_pred/;=//;D//;AND//;0/;Augmenter;5
1;Polyphen_HVAR_pred/;=//;P//;AND//;0/;Augmenter;5
1;LRT_pred/;=//;D//;AND//;0/;Augmenter;5
1;Mutation_taster_pred/;=//;D//;AND//;0/;Augmenter;5
1;FATHMM_pred/;=//;D//;AND//;0/;Augmenter;5
1;FATHMM_MKL_pred/;=//;D//;AND//;0/;Augmenter;5
1;Meta_SVP_pred/;=//;D//;AND//;0/;Augmenter;5
1;Presence_pc/;<//;20//;AND//;0/;Diminuer;100
1;Hypothese_transmission/;contient//;MONO//;AND//;0/;Augmenter;30
1;Transmission_HPO/;contient//;Autosomal dominant inheritance//
        ;AND//;0/;Augmenter;30
1;esp_all/;ne contient pas//;.//;AND//;0/;Diminuer;50
1;1000g_all/;ne contient pas//;.//;AND//;0/;Diminuer;50
1;Varank_score/Nbre_allele/Allele_gnomad/;>=/<=/<=//;60/4/10//
        ;AND/AND/AND//;0/0/0/;Augmenter;50
1;Effet_mutation/Nbre_allele/Allele_gnomad/;contient/<=/<=//
        ;acceptor/4/10//;AND/AND/AND//;0/0/0/;Augmenter;30
1;Effet_mutation/Nbre_allele/Allele_gnomad/;contient/<=/<=//
        ;donor/4/10//;AND/and/and//;0/0/0/;Augmenter;30
1;CLINSIG/CLNSIGCONF/CLNREVSTAT/Nbre_allele/
        ;contient/contient/contient/<=//
```

```
        ;athogen/athogen/criteria_provided
        ,_multiple_submitters,_no_conflicts/4//
        ;AND/AND/AND/AND//;0/0/0/0/;Augmenter;100
1;Varank_score/Nbre_allele/;>=/>//;50/4//;AND/AND//;0/0/;Diminuer;50
1;Varank_score/Allele_gnomad/ID_MT/;>=/>=/ne contient pas//
        ;50/10/chrX//;AND/AND/AND//;0/0/0/;Diminuer;50
1;Intervar/;contient//;athogen//;AND//;0/;Augmenter;30
1;CADD_pred/;>=//;20//;AND//;0/;Augmenter;30
1;Effet_mutation/;contient//;downstream//;AND//;0/;Diminuer;100
1;Effet_mutation/;contient//;upstream//;AND//;0/;Diminuer;100
1;role_DI/;contient//;oui(orange)//;AND//;0/;Augmenter;20
1;role_DI/;contient//;oui(rouge)//;AND//;0/;Diminuer;10
1;CLINSIG/Allele_gnomad/Nbre_allele/;contient/</<=//;athogen/10/4//
        ;AND/AND/AND//;0/0/0/;Augmenter;40
1;Varank_score/Nbre_allele/Allele_gnomad/;>=/<=/contient//;60/4/.//
        ;AND/AND/AND//;0/0/0/;Augmenter;50
1;Effet_mutation/Nbre_allele/Allele_gnomad/;contient/<=/contient//
        ;acceptor/4/.//;AND/AND/AND//;0/0/0/;Augmenter;30
1;Effet_mutation/Nbre_allele/Allele_gnomad/;contient/<=/contient//
        ;donor/4/.//;AND/AND/AND//;0/0/0/;Augmenter;30
1;CLINSIG/Allele_gnomad/Nbre_allele/;contient/contient/<=//
        ;athogen/./4//;AND/AND/AND//;0/0/0/;Augmenter;40
1;Allele_gnomad/;=/;0/;BASE/;0/;Augmenter;40
```

Listing D.11 – *ruleset* récessif "OMIM" (FRO) de l'utilisateur *F*.

```
1;Nbre_allele/;<=//;2//;AND//;0/;Augmenter;30
1;Allele_gnomad/;=//;0//;AND//;0/;Augmenter;40
1;Allele_gnomad/;<=//;50//;AND//;0/;Augmenter;10
1;CLINSIG/;contient//;athogen//;AND//;0/;Augmenter;70
1;Couverture/;<=//;10//;AND//;0/;Diminuer;100
1;Presence_pc/;<=//;30//;AND//;0/;Diminuer;50
2;Varank_score/;>=//;10//;AND//;0/;Augmenter;10
1;Nbre_allele/;>=//;8//;AND//;0/;Diminuer;100
1;role_DI/;contient//;oui//;AND//;0/;Augmenter;30
1;Grantham/;>//;80//;AND//;0/;Augmenter;30
1;Effet_mutation/;contient//;prime//;AND//;0/;Diminuer;100
1;Allele_gnomad/;>=//;51//;AND//;0/;Diminuer;40
1;Effet_mutation/;contient//;intron//;AND//;0/;Diminuer;50
1;Sift_pred/;=//;D//;AND//;0/;Augmenter;5
1;Provean_pred/;=//;D//;AND//;0/;Augmenter;5
1;Polyphen_HDIV_pred/;=//;D//;AND//;0/;Augmenter;5
1;Polyphen_HVAR_pred/;=//;P//;AND//;0/;Augmenter;5
1;LRT_pred/;=//;D//;AND//;0/;Augmenter;5
1;Mutation_taster_pred/;=//;D//;AND//;0/;Augmenter;5
1;FATHMM_pred/;=//;D//;AND//;0/;Augmenter;5
1;FATHMM_MKL_pred/;=//;D//;AND//;0/;Augmenter;5
1;Meta_SVP_pred/;=//;D//;AND//;0/;Augmenter;5
1;Presence_pc/;<//;20//;AND//;0/;Diminuer;100
1;Hypothese_transmission/;contient//;BI//;AND//;0/;Augmenter;30
1;Transmission_HPO/;contient//;Autosomal recessive inheritance//
        ;AND//;0/;Augmenter;30
1;Effet_mutation/Nbre_allele/Allele_gnomad/;contient/<=/<=//
        ;acceptor/4/10//;AND/AND/AND//;0/0/0/;Augmenter;30
1;Effet_mutation/Nbre_allele/Allele_gnomad/;contient/<=/<=//
        ;donor/4/10//;AND/AND/AND//;0/0/0/;Augmenter;30
1;CLINSIG/CLNSIGCONF/CLNREVSTAT/Nbre_allele/
        ;contient/contient/contient/<=//
        ;athogen/athogen/criteria_provided
        ,_multiple_submitters,_no_conflicts/50//
```

```
        ;AND/AND/AND/AND//;0/0/0/0/;Augmenter;100
1;Varank_score/Allele_gnomad/;>=/>=//;50/100//;AND/AND//;0/0/;Diminuer;50
1;Intervar/;contient//;athogen//;AND//;0/;Augmenter;30
1;CADD_pred/;>=//;20//;AND//;0/;Augmenter;30
1;Varank_score/Nbre_allele/Allele_gnomad/;>=/<=/<=//;60/4/50//
        ;AND/AND/AND//;0/0/0/;Augmenter;30
1;Effet_mutation/;contient//;uptream//;AND//;0/;Diminuer;50
1;Effet_mutation/;contient//;downstream//;AND//;0/;Diminuer;50
1;Effet_mutation/Nbre_allele/Allele_gnomad/;contient/<=/contient//
        ;donor/4/.//;AND/AND/AND//;0/0/0/;Augmenter;30
1;Effet_mutation/Nbre_allele/Allele_gnomad/;contient/<=/contient//
        ;acceptor/4/.//;AND/AND/AND//;0/0/0/;Augmenter;30
1;Varank_score/Nbre_allele/Allele_gnomad/;>=/<=/contient//
        ;60/4/.//;AND/AND/AND//;0/0/0/;Augmenter;30
1;Varank_score/Nbre_allele/Nbre_homozygote/Allele_gnomad/Couverture/
        ;>=/=/=/<=/>=//;10/2/1/50/20//;AND/AND/AND/AND/AND//
        ;0/0/0/0/0/;Augmenter;40
1;Varank_score/Nbre_allele/Nbre_homozygote/Allele_gnomad/Couverture/
        ;>=/=/=/contient/>=//;10/2/1/./20//;AND/AND/AND/AND/AND//
        ;0/0/0/0/0/;Augmenter;40
```

Listing D.12 – *ruleset* lié au chromosome X (FX) de l'utilisateur *F*.

```
2;Varank_score/;>=//;60//;AND//;0/;Augmenter;30
1;Allele_gnomad/;contient//;.//;AND//;0/;Augmenter;40
1;Allele_gnomad/;<=//;10//;AND//;0/;Augmenter;10
1;Couverture/;<=//;10//;AND//;0/;Diminuer;100
1;Presence_pc/;<=//;30//;AND//;0/;Diminuer;50
1;Effet_mutation/;contient//;donor//;AND//;0/;Augmenter;30
1;Effet_mutation/;contient//;acceptor//;AND//;0/;Augmenter;30
2;Varank_score/;>=//;10//;AND//;0/;Augmenter;10
1;Nbre_allele/;>=//;4//;AND//;0/;Diminuer;100
1;role_DI/;ne contient pas//;Rouge//;AND//;0/;Augmenter;10
1;Grantham/;>//;80//;AND//;0/;Augmenter;20
1;Effet_mutation/;contient//;prime//;AND//;0/;Diminuer;100
1;Allele_gnomad/;>=//;15//;AND//;0/;Diminuer;100
1;Effet_mutation/;contient//;intron//;AND//;0/;Diminuer;50
1;Sift_pred/;=//;D//;AND//;0/;Augmenter;5
1;Provean_pred/;=//;D//;AND//;0/;Augmenter;5
1;Polyphen_HDIV_pred/;=//;D//;AND//;0/;Augmenter;5
1;Polyphen_HVAR_pred/;=//;P//;AND//;0/;Augmenter;5
1;LRT_pred/;=//;D//;AND//;0/;Augmenter;5
1;Mutation_taster_pred/;=//;D//;AND//;0/;Augmenter;5
1;FATHMM_pred/;=//;D//;AND//;0/;Augmenter;5
1;FATHMM_MKL_pred/;=//;D//;AND//;0/;Augmenter;5
1;Meta_SVP_pred/;=//;D//;AND//;0/;Augmenter;5
1;Presence_pc/;</;30//;AND//;0/;Diminuer;100
1;Hypothese_transmission/;contient//;X-LINKED//;AND//;0/;Augmenter;20
1;Transmission_HPO/;contient//;X-linked//;AND//;0/;Augmenter;20
1;allele_gnomad_male/;=//;0//;AND//;0/;Augmenter;30
1;ID_MT/;ne contient pas//;chrX//;AND//;0/;Diminuer;100
1;esp_all/;ne contient pas//;.//;AND//;0/;Diminuer;30
1;1000g_all/;ne contient pas//;.//;AND//;0/;Diminuer;30
1;CLINSIG/ID_MT/;contient/contient//;athogen/chrX//
        ;AND/AND//;0/0/;Augmenter;50
1;ID_MT/Nbre_allele/Nbre_homozygote/;contient/<=/=//;chrX/3/1//
        ;AND/AND/AND//;0/0/0/;Augmenter;30
1;Effet_mutation/;contient//;downstream//;AND//;0/;Diminuer;50
1;Effet_mutation/;contient//;upstream//;AND//;0/;Diminuer;50
1;ID_MT/Presence_pc/;contient/>=//;chrX/95//;AND/and//;0/0/;Augmenter;30
```

```
1;Allele_gnomad/;=/;0/;BASE/;0/;Augmenter;40
```