Pilotage d'algorithmes pour la reconnaissance en ligne d'arythmies cardiaques

François Portet

N° d'ordre: 3290

THÈSE

Présentée devant

devant l'Université de Rennes 1

pour obtenir

le grade de : Docteur de l'Université de Rennes 1
Mention Informatique

par

François Portet

Équipe d'accueil : projet DREAM (IRISA) et projet SEPIA (LTSI)
École Doctorale : Matisse
Composante universitaire : IFSIC

Titre de la thèse :

Pilotage d'algorithmes
pour la reconnaissance en ligne d'arythmies cardiaques

soutenue le 6 décembre 2005 devant la commission d'examen

Présidente :	Monique	Thonnat	DR INRIA Sophia-Antipolis
Rapporteurs :	Michel	Dojat	IR INSERM Grenoble (HDR)
	Jim	Hunter	Pr. Université d'Aberdeen
Examinateurs :	Guy	Carrault	Pr. Université de Rennes 1 (co-dir. thèse)
	Marie-Odile	Cordier	Pr. Université de Rennes 1 (dir. thèse)
	René	Quiniou	CR INRIA Rennes (co-encadrant)

Remerciements

Cette thèse est l'occasion pour moi de remercier toutes les personnes qui ont contribué à ce travail ainsi que celles que j'ai pu rencontrer durant ces années de doctorat.

Tout d'abord, je tiens à remercier Monique Thonnat, directrice de recherche INRIA à Sophia-Antipolis, spécialiste du pilotage de programmes, pour m'avoir fait l'honneur de présider mon jury.

Je remercie également mes deux rapporteurs Michel Dojat, ingénieur de recherche INSERM à Grenoble et Jim Hunter professeur à l'université d'Aberdeen, dont les commentaires et les remarques éclairées ont contribué à l'enrichissement du document final. J'ai précédemment eu l'occasion de les rencontrer et j'ai pu apprécier leur expérience, leur disponibilité et leur gentillesse. J'ai été particulièrement touché que Jim Hunter accepte de rapporter une thèse qui ne soit pas dans sa langue d'origine et fasse le déplacement jusqu'à Rennes.

Cette thèse est le fruit d'une collaboration étroite entre l'équipe DREAM de l'IRISA et l'équipe SEPIA du LTSI. Je remercie ma directrice de thèse Marie-Odile Cordier (MAO) professeur à l'université de Rennes 1 pour m'avoir accueilli dans l'équipe DREAM. Merci aussi à René Quiniou CR INRIA dont la disponibilité fut sans faille notamment pour l'apprentissage des arythmies (et les nombreux réapprentissages ...). J'accorde aussi toute ma reconnaissance à mon autre directeur de thèse Guy Carrault professeur à l'université de Rennes 1 pour m'avoir accueilli au LTSI et pour tous les instants qu'il m'a accordés notamment lors de mes moments de doute.

J'adresse des remerciements tout particuliers à mes deux collègues de bureau Elisa et Alban qui ont eu la lourde tâche de me supporter durant trois années notamment lorsque je pestais contre Matlab .... Je n'oublierai pas les efforts d'Elisa pour me convertir à la fièvre du jeu et ceux d'Alban pour me faire enfin rire à une de ses blagues (désolé mais elles ne sont vraiment pas drôles ...) ni même l'hymne de notre bureau : « sex over the phone » ! C'est grâce à eux que j'ai pu travailler dans la bonne humeur (quelques fois jusqu'à 6h du matin). Sans eux le bureau est bien vide.

J'ai eu la chance d'appartenir à deux équipes de recherche de deux laboratoires différents. Je n'ai malheureusement pas pu être présent au LTSI aussi souvent que je l'aurais voulu. Cependant, j'ai pu faire la connaissance d'Alfredo avec qui j'ai pu travailler. J'ai rarement eu l'occasion de voir quelqu'un qui allie compétence, gentillesse et disponibilité aussi naturellement et j'espère que nous pourrons de nouveau travailler ensemble. Je n'oublierai pas non plus les réunions SEPIA qui m'ont permises de connaître et d'apprécier Emmanuel, Jérôme, Lotfi et tous les autres membres de cette chaleureuse

équipe.

Les trois années que j'ai passées au sein de l'équipe DREAM de l'IRISA ont aussi été très fructueuses tant professionnellement qu'humainement. Les séminaires d'équipe « Tahiti », les réunions d'équipe et les pauses café ont été des moments très conviviaux. Ce fut un plaisir de vivre dans une équipe aussi sympathique où la personnalité de chacun est appréciée et où il est toujours facile de parler d'autre chose que du boulot. Je remercie ainsi Sophie qui m'a souvent aidé pour mes dossiers académiques et Véronique pour m'avoir fait goûter un de ses chocolats qu'elle garde secrètement dans son tiroir ..., Yves pour m'avoir souvent emmené à la piscine alors qu'il ne semblait pas avoir tellement envie de plonger ..., Pascal mon nouveau collègue de bureau dont l'enthousiasme scientifique est tellement communicatif, le binôme Alex (président) et Ronan et leurs disputes de couple, et Xavier le petit dernier.

Merci aussi à Marie-Noëlle et Céline, les assistantes de projet pour leur bonne humeur et tout ce qu'elles ont fait pour moi (sans que je me sois rendu compte de tout). Merci aussi à l'Atelier pour leur disponibilité et leur rapidité, notamment Philippe Aoustin et Philippe Lecler que j'ai embêtés plus souvent que les autres.

J'adresse également un grand merci aux stagiaires, Mickaël Pencolé et Fabien Bonic pour leur travail indispensable et leur professionnalisme.

Ces années de doctorat ont aussi été pour moi l'occasion de me lancer dans l'enseignement et de travailler en équipe pédagogique. Merci aux enseignants qui ont pu me transmettre leur expérience : Claudio (INSA), Stéphane (INSA), Danièle (INSA), Laurent (IFSIC) et Gilles (IFSIC)

Par ailleurs, j'ai eu l'occasion de m'investir dans diverses associations de jeunes chercheurs notamment l'ADOC qui a fortement contribué à mon épanouissement au sein de l'IRISA. C'est grâce à l'ADOC que j'ai rencontré les doctorants qui ont fait parti de mon univers à l'IRISA. Merci aussi aux Nicomaciens et à la CJC. C'est lors de réunion CJC que j'ai vraiment pris conscience des compétences que le doctorat apporte et fait développer. C'est surtout la CJC qui m'a permis de comprendre que le doctorat est plus un travail scientifique et professionnel qu'une poursuite d'études, n'en déplaise à certains responsables de troisième cycle.

Je ne peux pas conclure sans remercier Cécile qui, malgré à sa patience légendaire ..., m'a supporté durant ma rédaction et a toujours sut me « coacher » dans mes moments de doute durant toutes ces années. Je n'y serais pas arrivé sans elle.

Table des matières

Introduction

Les pathologies cardiovasculaires provoquent chaque année 17 millions de décès à travers le monde (Mackay et Mensah, 2004). Elles représentent en France la première cause de morbidité et de mortalité (311,5 décès/an pour 100 000 habitants) (Beaufils et coll., 1999). Parmi ces pathologies cardiovasculaires, les infarctus du myocarde représentent 10% des décès dans le monde et sont provoqués par une ischémie (défaut prolongé d'apport sanguin). En présence d'une ischémie, le tissu cardiaque tend à perdre ses propriétés de contractilité et le cœur ne peut plus assurer sa fonction de pompe ce qui provoque l'infarctus. L'infarctus du myocarde est accompagné de troubles graves du rythme cardiaque, appelés *arythmies cardiaques*, tels que la fibrillation ventriculaire. Ces arythmies se caractérisent par un fort degré d'urgence et explique qu'il est primordial de pouvoir prendre en charge très tôt les patients souffrant d'un infarctus. De plus, les arythmies mineures informent sur l'état de récupération cardiaque des patients et doivent donc être détectées notamment pour prévenir une dégénérescence possible en arythmies sévères. C'est pourquoi, les Unités de Soins Intensifs pour Coronariens (USIC) ont été créées dans les années 60 afin d'accueillir les patients atteints de pathologies cardiovasculaires dans l'urgence, de déceler les complications au plus tôt et d'assurer une surveillance continue. En effet, le traitement de ces pathologies nécessite un personnel médical aguerri et du matériel spécialisé pour une intervention thérapeutique rapide. Pour assurer une surveillance constante de jour comme de nuit, les USIC sont équipées de systèmes de monitorage de patient. À travers les signaux physiologiques recueillis, tel que l'électrocardiogramme (ECG) ou la pression artérielle, ces systèmes ont pour tâche de détecter les situations alarmantes, telles que la tachycardie ventriculaire, nécessitant une intervention médicale. Au départ bornés au simple affichage des signes vitaux, ces systèmes ont bénéficié, au fil des années, de l'évolution de l'intelligence artificielle pour conduire au concept de *monitorage intelligent* (Mora et coll., 1993).

Les systèmes de monitorage intelligent tels que Vie-Vent (Miksch et coll., 1996) et NéoGanesh (Dojat et coll., 1997) comprennent généralement deux parties distinctes : une partie bas-niveau d'*abstraction temporelle* dédiée à l'acquisition, au traitement et à l'analyse des signaux physiologiques, et une partie haut-niveau de *diagnostic médical* qui infère un diagnostic à partir des informations transmises par l'abstraction temporelle et d'une base de connaissances. Ainsi, le personnel médical est assisté 24 heures sur 24 dans la tâche de surveillance et bénéficie d'une aide au diagnostic. Le but étant, à terme, de proposer un diagnostic qui puisse apporter une aide suffisante pour permettre aux

médecins non spécialistes ou débutants de prendre les bonnes décisions. Cette partie de diagnostic médical est souvent constituée d'un système expert. La plupart des systèmes de monitorage intelligent reposent sur une architecture flexible pouvant s'adapter aux situations variables rencontrées en milieu clinique. Cette adaptation peut se faire à différents niveaux de la chaîne de traitement, par exemple en sélectionnant les ressources utiles au diagnostic médical, en détectant les artefacts ou en focalisant le diagnostic sur l'analyse d'événements particuliers. Cependant, peu de systèmes intègrent une adaptation à tous les niveaux de la chaîne de traitement et les informations générées par le moniteur, telles que le diagnostic médical, sont rarement exploitées pour optimiser les algorithmes de traitement de l'information. De plus, un nombre important de fausses alarmes générées en milieu clinique sans signification médicale subsiste (Tsien et Falcker, 1997), ce qui amène à une perte de confiance du personnel envers le matériel (Soulas, 2001). Un système de monitorage inférant un diagnostic médical fiable tout en générant peu de fausses alarmes reste encore un objectif à atteindre.

Tous ces objectifs ont été pris en compte dans ce mémoire qui présente une évolution du système de monitorage Calicot (Wang, 2002; Carrault et coll., 2003) destiné à la reconnaissance des arythmies cardiaques. Il décrit le développement d'un nouveau système de monitorage intelligent, appelé IP-Calicot (*Integrated Piloting and Cardiac Arrhythmias Learning for Intelligent Classification of On-line Tracks*) capable, grâce à un module de pilotage d'algorithmes, d'utiliser les informations du contexte courant, telles que le bruit de ligne et le diagnostic médical, pour modifier sa chaîne de traitement. Ce travail se rapproche de ceux portant sur le pilotage de programmes en traitement du signal (Shekhar et coll., 1994) et de ceux dédiés à la conception de systèmes structurellement auto-adaptatifs (Karsai et Sztipanovits, 1999). Cependant, l'originalité du pilote conçu est de prendre explicitement en compte la spécificité du monitorage cardiaque. Le pilotage d'algorithmes doit agir aux trois niveaux décrits ci dessous.

1. Au niveau du *diagnostic médical* : le diagnostic médical repose sur une description des signaux physiologiques dans un langage précis. Cependant, certaines arythmies peuvent être caractérisées à partir d'une description plus abstraite nécessitant moins de calcul. De plus, en fonction des situations (p. ex. trop de bruit sur la ligne), certaines caractéristiques du signal peuvent ne pas être disponibles. Le pilote doit donc choisir, en ligne, le langage de description afin de consommer moins de ressources et assurer un diagnostic médical à différentes granularités.

2. Au niveau des *tâches d'abstraction du signal* : la description des signaux est réalisée par un ensemble de tâches spécialisées dans l'extraction de caractéristiques particulières du signal. Ces tâches extraient différents types d'ondes et d'événements présents dans les signaux. Cependant, lorsque les signaux sont bruités, l'extraction des caractéristiques du signal peut être erronée et fournir une fausse description du signal qui amène à un diagnostic médical défaillant. Le pilote doit donc, en fonction du contexte courant, désactiver les tâches d'extraction de caractéristiques qui ne peuvent pas se poursuivre sans erreurs et activer les autres tâches.

3. Au niveau des *algorithmes de traitement du signal* : chaque tâche d'extraction peut être réalisée par plusieurs algorithmes différents. Cependant, certains algorithmes sont plus adaptés que d'autres au contexte courant. Le pilote doit donc choisir, pour chaque tâche, les algorithmes de traitement du signal les plus adéquats en fonction du contexte courant.

Le premier chapitre de ce mémoire s'attache à présenter le domaine d'application. Après avoir introduit l'électrocardiographie et les arythmies cardiaques, un état de l'art des dernières orientations de la recherche concernant les systèmes de monitorage intelligent est présenté. Les avancées effectuées dans ce domaine et les limites des systèmes actuels sont notamment exposées.

Pour bien comprendre les enjeux et les principes du pilotage d'algorithmes, le chapitre 2 présente un état de l'art des recherches liées au pilotage d'algorithmes. Cet état de l'art permet de mettre en exergue les concepts retenus pour la mise en œuvre d'un pilote dans IP-Calicot. Ce nouveau système est une évolution de Calicot dont les fondements sont présentés en deuxième partie de ce chapitre.

L'intégration du pilotage dans Calicot fait l'objet du chapitre 3. Ce chapitre détaille l'architecture du nouveau système de monitorage IP-Calicot, le système de pilotage et les bases de connaissances qu'il utilise. On montre en particulier comment la reconnaissance d'une arythmie peut être effectuée avec différentes granularités. Ceci a naturellement conduit à l'apprentissage des arythmies selon différents langages de description.

Le pilotage d'algorithmes consiste à choisir, dans une base d'algorithmes, un algorithme à utiliser pour une situation donnée (contexte). Pour cela il faut acquérir les règles qui permettent au pilote d'associer un contexte à un algorithme particulier. C'est l'objet du chapitre 4 qui se focalise sur l'acquisition de ces règles. Plutôt que de faire appel à des experts pour décrire les règles de pilotage, une méthode originale d'interprétation, fondée sur une analyse en composantes principales, permet d'étudier les performances des algorithmes sur un ensemble d'ECG représentatifs de contextes cliniques, et d'en dériver les règles de pilotage.

Le chapitre 5 présente les résultats du pilotage obtenus sur des électrocardiogrammes cliniques bruités. Les résultats sont exposés selon les trois niveaux de pilotage (diagnostic médical, tâches d'abstraction et algorithmes de traitement du signal). L'apport du pilotage d'algorithmes de traitement du signal est analysé en profondeur puis la qualité des règles de reconnaissance d'arythmies est évaluée. Enfin, les performances globales du système IP-Calicot sont comparées aux performances du système sans pilote sur des ECG bruités représentatifs de situations réelles bruitées.

Enfin, le chapitre 6 nous amène au cœur du système de monitorage et s'attache à décrire la mise en œuvre de IP-Calicot. Les choix de mise en œuvre, de représentation des données et des concepts, et d'exécution du pilotage sont décrits et discutés.

Chapitre 1

Monitorage en Cardiologie

1.1 Introduction

Ce chapitre présente le contexte de l'étude : le monitorage de patient en unité de soins intensifs. Il débute donc naturellement par la présentation du contexte applicatif qu'est l'Unité de Soins Intensifs pour Coronariens (USIC) et des arythmies cardiaques en section 1.2. Le mécanisme à l'origine de l'activité cardiaque et des arythmies ainsi que les problèmes d'acquisition et de traitement de l'électrocardiogramme sont introduits pour permettre de comprendre les problèmes de base du traitement automatique des arythmies. La section 1.3 présente le monitorage de patient et notamment les systèmes de monitorage intelligent. Les différents blocs de base sont détaillés au regard des différents travaux de la littérature.

1.2 Contexte médical : Les arythmies en Unité de Soins Intensifs pour Coronariens

1.2.1 Introduction

Les pathologies cardiovasculaires provoquent chaque année 17 millions de décès à travers le monde d'après l'OMS (Mackay et Mensah, 2004) et représente en France la première cause de mortalité (311,5 décès/an pour 100 000 habitants) (Beaufils et coll., 1999). La mort subite est un risque constant de l'activité cardiaque, quelle que soit la stabilité apparente de la situation cardiaque initiale, et nécessite une intervention immédiate à l'aide de moyens de ressuscitation tel qu'un défibrillateur. Certains de ces accidents cardiaques sont prévisibles mais la plupart surviennent de façon inopinée. La prévision et la découverte des facteurs pouvant amener à de tels risques reste un enjeu majeur. Le traitement de certaines maladies cardiovasculaires nécessite un personnel médical aguerri et du matériel spécialisé pour une intervention thérapeutique rapide. L'Unité de Soins Intensifs pour Coronariens (USIC) a ainsi été créée pour prendre en charge les patients atteints de maladies cardiovasculaires graves.

Parmi ces pathologies cardiovasculaires, les attaques cardiaques représentent la

deuxième cause de décès (33%) après les pathologies des coronaires (43%). Ces pathologies cardiaques sont principalement provoquées par les ischémies et les arythmies. Les arythmies sont des troubles du rythme cardiaque dont certaines, telles que la fibrillation ventriculaire, peuvent entraîner des morts subites. Pour diagnostiquer les arythmies, la méthode la plus répandue est l'analyse de l'électrocardiogramme. Cette analyse est effectuée par des cardiologues spécialisés appelés rythmologues, qui décèlent sur les différents tracés le mécanisme de l'activité électrique cardiaque. L'électrocardiogramme peut être en partie analysé automatiquement, c'est pourquoi toutes les USIC possèdent des systèmes de monitorage dont le but est de détecter les arythmies sévères nécessitant une intervention immédiate du personnel hospitalier. Cependant, les systèmes doivent faire face à une quantité de bruit non négligeable générée par le milieu hospitalier. La réalisation de systèmes de monitorage capables de reconnaître un grand nombre d'arythmies dans un contexte bruité en milieu USIC est donc un objectif majeur et c'est dans ce cadre que nous plaçons notre étude.

Cette section débute par l'introduction du contexte USIC en section 1.2.2. Puis, après une présentation du système cardiaque en section 1.2.3, l'électrocardiographie, qui permet d'obtenir une image de l'activité électrique cardiaque, est détaillée en section 1.2.4. Enfin, les troubles du rythme et de la conduction sont décrits en section 1.2.5.

1.2.2 Unité de Soins Intensifs pour Coronariens

L'Unité de Soins Intensifs pour Coronariens (USIC) est une structure des hôpitaux, spécialement organisée pour une activité cardiovasculaire d'urgence. Elle prend en charge, 24h sur 24, des malades souffrants d'une pathologie cardiovasculaire susceptible d'entraîner une défaillance cardiovasculaire aiguë. La mise en place de ces structures a permis au cours des quarante dernières années d'amener la mortalité hospitalière des infarctus du myocarde de 30% à 10% (Beaufils et coll., 1999). Les patients admis au sein d'une USIC sont surveillés en continu par l'observation régulière et méthodique de différents paramètres dont le caractère anormal peut entraîner des alarmes.

Les USIC contiennent des chambres de dimensions suffisantes pour permettre l'accueil et la circulation des appareils d'investigation et de traitement. Au centre de l'USIC se trouve un espace réservé aux systèmes de monitorage : l'unité centrale. Dans cet espace, se tient en permanence un infirmier qui peut surveiller l'ensemble des patients et intervenir en cas d'alarme. Chaque lit doit être équipé d'un moniteur électrocardiographique à deux tracés qui peut enregistrer au moins 24 heures de signaux (Beaufils et coll., 1999). Ce dispositif est relié à l'unité centrale. Ce système de monitorage émet donc des alarmes dans la chambre des patients mais aussi dans l'unité centrale. En outre, l'USIC doit disposer d'un appareil de mesure de pression invasif, d'un appareil de mesure de pression non invasif de la pression artérielle, d'appareils de saturométrie et bien sûr de défibrillateurs.

Généralement, le séjour en USIC est court, l'objectif principal étant de stabiliser l'état du patient puis, dès qu'il ne nécessite plus de suivi intensif, de le transférer dans une autre unité.

1.2.3 Activité cardiaque

Le système cardio-vasculaire assure la circulation du sang qui permet les échanges respiratoires et nutritifs indispensables à la vie. Depuis la découverte fondamentale par William Harvey de l'existence de la *petite circulation* et de la *grande circulation*, de nombreux travaux sont venus enrichir la connaissance de la circulation sanguine.

La grande circulation représente la vascularisation (ou perfusion) de toutes les cellules du corps, hormis les poumons. La petite circulation est celle qui concerne uniquement les poumons. Dans la grande circulation, l'oxygène du sang artériel est consommé par les cellules ce qui produit le sang veineux. À travers les capillaires des poumons, le sang veineux se recharge en oxygène et s'artérialise.

La circulation proprement dite est assurée par un organe ayant le rôle de « pomper » et distribuer le sang : le **cœur**. Il assure un échange régulier entre la petite et la grande circulation suivant un cycle bien précis. La régularité de ces échanges est commandée par un stimulus électrique, dépendant du système nerveux autonome[1], qui parcours le cœur du nœud sinusal à l'apex[2] pour déclencher la contraction des différentes chambres qui constituent le cœur. Ce stimulus peut être observé en mesurant les différences de potentiel de plusieurs électrodes à la surface du corps. L'interprétation de ces différences de potentiel relève du domaine de l'*électrocardiographie*.

Le cycle cardiaque ne tolère pas les interruptions car certaines cellules meurent lorsqu'elles ne sont plus alimentées en sang artériel. C'est notamment le cas des cellules du cerveau, c'est pourquoi les défaillances cardiaques sont si fatales. Par la suite, le cœur est présenté selon trois axes : l'anatomie, l'activité mécanique et l'activité électrique.

1.2.3.1 Anatomie du cœur

Le cœur est un organe intrathoracique situé entre les deux poumons au carrefour des grosses artères (aorte et artère pulmonaire) et grosses veines (veines caves et pulmonaires) de l'organisme. Sa structure est composée de 3 épaisseurs : l'endocarde, surface externe, où passent nerfs et vaisseaux sanguins ; l'épicarde, membrane séreuse formant la paroi interne du péricarde[3] ; et le myocarde, partie véritablement active du cœur.

Dans les fibres du myocarde, il y a principalement deux tissus qui jouent un rôle complémentaire dans le cycle cardiaque. L'un est dédié à la mécanique musculaire, c'est le **myocarde commun**, l'autre engendre et conduit l'excitation (la commande de contraction), c'est le **myocarde différencié** (ou circuit nodal).

Myocarde commun Le myocarde commun est essentiellement composé de cellules musculaires (ou myocytes). Il est réparti en quatre chambres creuses liées entre elles : les oreillettes droite et gauche et les ventricules droit et gauche (voir Figure 1.1). L'oreillette et ventricule gauches (resp. droits) communiquent entre eux par la valve mitrale (resp. tricuspide).

La paroi des oreillettes est très mince et celle du ventricule droit ne dépasse pas quelques millimètres. Le ventricule gauche, quant à lui, possède une paroi de plus d'un

[1] Partie du système nerveux innervant notamment le cœur, les poumons, le tube digestif et les organes

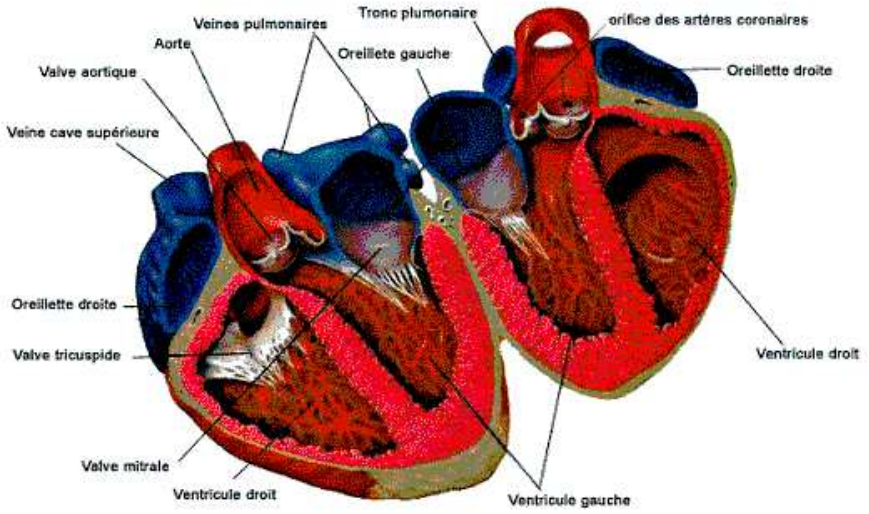


FIG. 1.1 – Structure anatomique du cœur.

centimètre. Les deux masses auriculaire et ventriculaire sont séparées par les structures fibreuses des anneaux auriculo-ventriculaires et du septum fibreux. Le septum inter-ventriculaire, épais d'une quinzaine de millimètres, est constitué de l'adossement du ventricule gauche, en majorité, et de l'adossement du ventricule droit ;

Myocarde différencié La contraction du myocarde est déclenchée par une excitation électrique (le potentiel d'action) conduite et générée par la structure du myocarde différencié. Les cellules nodales du myocarde différencié sont présentes sur l'ensemble des cavités cardiaques de l'oreillette droite à l'oreillette gauche jusqu'aux ventricules (voir Figure 1.2).

Ce circuit comprend essentiellement cinq structures.

- Les cellules du nœud sinusal, ou nœud de Keith et Flack (Keith et Flack, 1906), situées dans la paroi de l'oreillette droite, génèrent le rythme cardiaque normal. L'activité du nœud sinusal est dépendante du système nerveux autonome.
- Les voies de conduction intra-auriculaire comprennent trois voies sino-nodales (antérieure, moyenne et postérieure) qui parcourent l'oreillette droite du nœud sinusal au nœud auriculo-ventriculaire et le faisceau de Bachmann qui dérive de la voie sino-nodale antérieure et se dirige vers l'oreillette gauche.
- Les cellules du nœud auriculo-ventriculaire (nœud AV), ou nœud de Tawara (Tawara, 1906; Tawara, 2000), situées dans la partie basse et antérieure de la cloison inter-auriculaire, bloquent pendant une courte pause l'activité électrique provenant des oreillettes puis la communique au faisceau de His (Kent, 1914).

génitaux

[2]extrémité inférieure du cœur

[3]Sac membraneux qui enveloppe le cœur

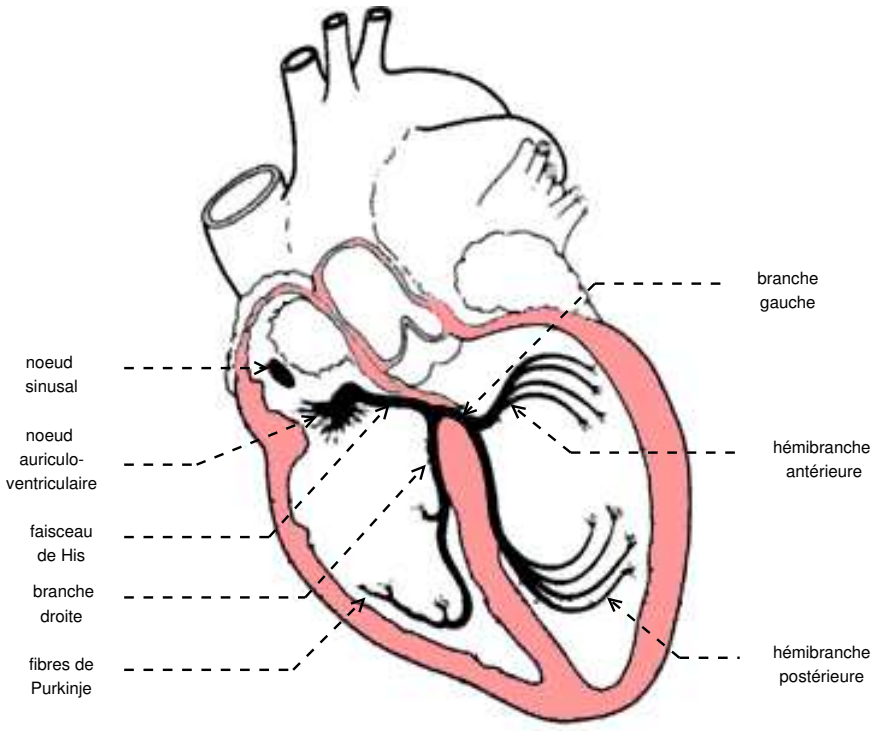


FIG. 1.2 – Myocarde différencié ou circuit nodal.

- Les cellules du faisceau de His constituent un cordon partant du nœud AV jusqu'à la jonction du septum inter-ventriculaire et du septum fibreux. Le faisceau se divise ensuite en deux branches :
  - la branche droite qui descend sur la face droite du septum inter-ventriculaire jusqu'à l'apex où elle se sépare en petites ramifications,
  - la branche gauche qui se sépare en deux hémibranches : l'hémibranche postérieure et l'hémibranche antérieure (il existe d'autres petites ramifications de moindre importance).
- Les fibres de Purkinje (Purkinje, 1845; Matousek et Posner, 1969) se situent à la terminaison des branches gauches et de la branche droite et tapissent les cavités des deux ventricules. Elles irriguent les myocytes[4] ventriculaires.

D'autres voies de conduction intracardiaque peuvent court-circuiter le circuit normal. Il s'agit des faisceaux de Kent, des fibres de Mahaim et des fibres de James que l'on peut voir sur le schéma de la figure 1.3 (tiré de (Blondeau et Hiltgen, 1980)). Leur existence est suspectée en présence de certains symptômes pathologiques.

[4]cellules myocardiques

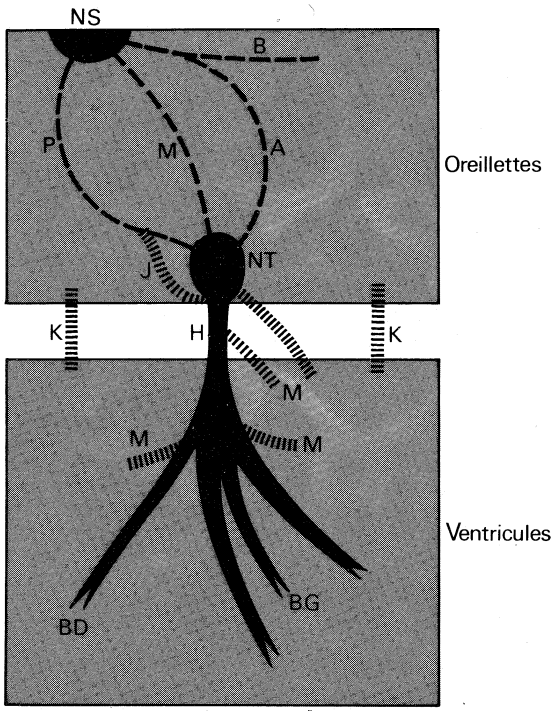


FIG. 1.3 – Schéma des voies de conduction intracardiaques. Tissus spécifiques : NS nœud sinusal, NT nœud AV, H faisceau de His, BD branche droite, BG branche gauche. Voies intra-auriculaires : A voie sino-nodale antérieure, M voie moyenne, P voie postérieure, B faisceau de Bachmann. Voies accessoires : K faisceau de Kent, J fibres de James, M fibres de Mahaim.

L'activité de pompe du cœur peut être vue sous deux aspects : l'un, mécanique, mettant en jeu la contractilité des cavités cardiaques et les ouvertures et fermetures des valves des artères et veines ; l'autre, électrique, mettant en jeu la conduction électrique de la commande de contraction du muscle cardiaque à travers celui-ci. Ces deux aspects sont détaillés dans les sections suivantes.

1.2.3.2 Activité mécanique cardiaque

Le cycle de la circulation sanguine se répète constamment et se divise en deux périodes : la systole et la diastole. La systole est la période correspondant à l'éjection du sang dans la grande et petite circulation. Elle se décompose en trois phases : la systole auriculaire, la contraction ventriculaire isovolumique et la systole ventriculaire. La diastole est la phase de relaxation du cœur, pendant laquelle il se remplit de sang. Cette période est composée de deux phases : la relaxation ventriculaire isométrique et la phase de repos. La figure 1.4 illustre la succession des différentes phases détaillées ci-dessous.

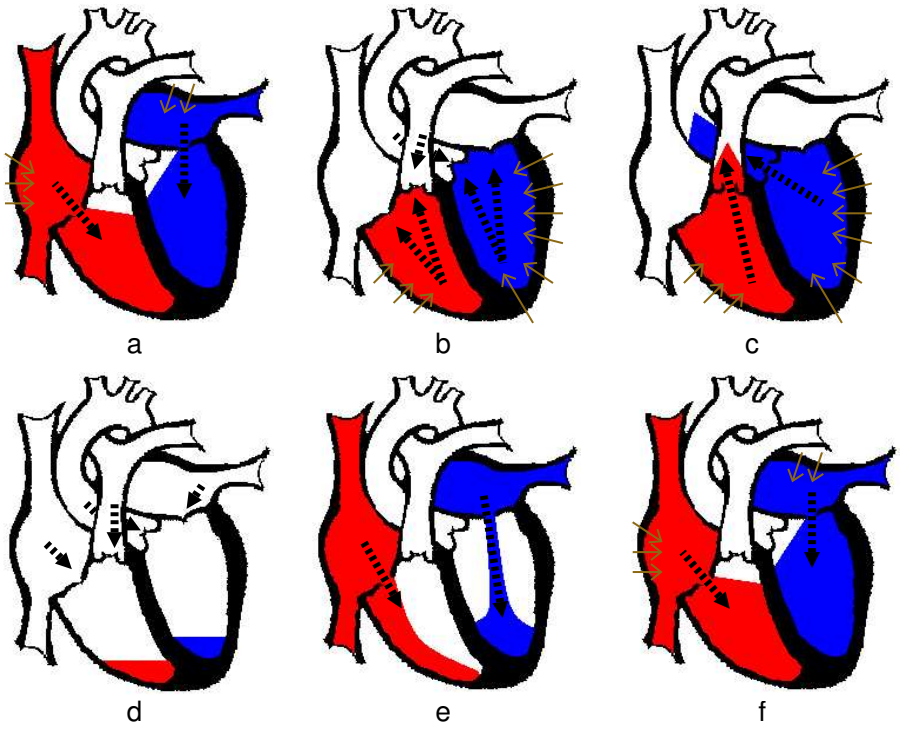


FIG. 1.4 – Cycle mécanique cardiaque. a) systole auriculaire, b) contraction isovolumique des ventricules, c) systole ventriculaire, d) relaxation des ventricules, e) repos, et f) nouvelle systole auriculaire. Les flèches pleines indiquent la contraction musculaire et les flèches en pointillés, la direction de la pression sanguine.

a et f) **La systole auriculaire** est la contraction des oreillettes lorsque celles-ci sont remplies de sang. La pression exercée par le muscle auriculaire ferme les orifices des veines caves et pulmonaires et provoque le versement du sang auriculaire dans les ventricules (Figure 1.4) par les valves tricuspide et mitrale. Cependant, la majorité du remplissage des ventricules survient passivement pendant la diastole.

b) **La contraction ventriculaire isovolumique** commence lorsque les cellules musculaires du myocarde ventriculaire se contractent. La pression ferme les valves tricuspide et mitrale et les valvules sigmoïdes restent fermées pendant quelques dizaines de centièmes de secondes. Comme il n'y a pas d'éjection ventriculaire, la pression augmente fortement dans les ventricules.

c) **La systole ventriculaire** commence lorsque les pressions dans les ventricules dépassent les pressions dans l'artère pulmonaire et l'aorte. Les valvules sigmoïdes s'ouvrent et l'éjection commence. Le sang oxygéné emprunte la crosse aortique et le sang désoxygéné le tronc pulmonaire. La pression aortique atteint un pic (pression artériel systolique) puis redescend jusqu'à la fin de la systole.

d) **La relaxation ventriculaire** fait suite à la systole. Les ventricules se relâchent,

la pression chute jusqu'à être inférieure à celle exercée dans l'aorte et l'artère pulmonaire. En conséquence, les valvules sigmoïdes se ferment et, lorsque la pression devient inférieure à celle des oreillettes, les valves tricuspide et mitrale s'ouvrent.

e) **La phase de repos** est celle pendant laquelle le sang des veines caves et pulmonaires s'écoule librement dans les ventricules via les oreillettes.

1.2.3.3 Activité électrique cardiaque

Le mécanisme cardiaque, qui comprend l'expulsion du sang et l'ouverture-fermeture des valves, fonctionne uniquement grâce aux contractions du myocarde. Ces contractions sont déclenchées par la propagation de proche en proche du *potentiel d'action* à travers les cellules myocardiques. Chaque cellule myocardique réagit à un stimulus électrique grâce à une membrane semi-perméable aux ions. Au repos, l'intérieur de la membrane cellulaire est chargé négativement par rapport à l'extérieur qui est pris comme référence. Dans cet état électrique stable on dit que la cellule est *polarisée*. De l'extérieur, on n'enregistre aucune activité électrique.

Lorsque la cellule est stimulée électriquement, les propriétés de la membrane se modifient et sa perméabilité aux ions augmente. Les échanges ioniques à travers la membrane des cellules myocardiques donnent naissance au potentiel d'action.

La figure 1.5 montre l'effet des échanges ioniques transmembranaires sur le potentiel d'action. La phase 0 est caractérisée par les ions sodium qui concourent à l'établissement d'un déséquilibre électrique entre le secteur extra-cellulaire et intra-cellulaire. Cette phase correspond à la *dépolarisation*. La phase 1 est le résultat de l'accroissement brutal de la perméabilité membranaire au sodium, ainsi, le potentiel de la membrane passe de -90mV à 40mV, c'est la systole électrique. Durant la phase 2, l'entrée des ions calcium à l'intérieur de la membrane permet le maintien en plateau de la dépolarisation. Durant la phase 3, l'accroissement de conductance au potassium est responsable d'une négativation des charges intra-cellulaires, et donc de la *repolarisation* cellulaire. La phase 4 correspond à la phase de repos, c'est la diastole électrique. On obtient l'équilibre avec une différence de potentiel négative.

L'excitabilité de la cellule, qui est la capacité d'une cellule myocardique à conduire un potentiel d'action, c'est-à-dire à générer un potentiel d'action en réponse à une stimulation, peut être décomposée en trois phases spécifiques.

- La période réfractaire absolue (P.R.A.) : période pendant laquelle tout stimulus externe n'a aucun effet sur la cellule (aucune excitation possible).
- La période réfractaire effective (P.R.E.) : période incluant la P.R.A., on y ajoute une phase pendant laquelle la cellule peut être stimulée mais ne conduit pas.
- La période réfractaire relative (P.R.R.) : période pendant laquelle seul un stimulus puissant peut générer un potentiel d'action.

De plus, le stimulus originel doit être supérieur au seuil d'excitabilité pour pouvoir déclencher le processus et doit être bien plus puissant encore en période réfractaire relative. On voit donc que la *commande* de contraction des cellules myocardiques est temporisée par les périodes réfractaires. La période globale du cycle cardiaque est ainsi

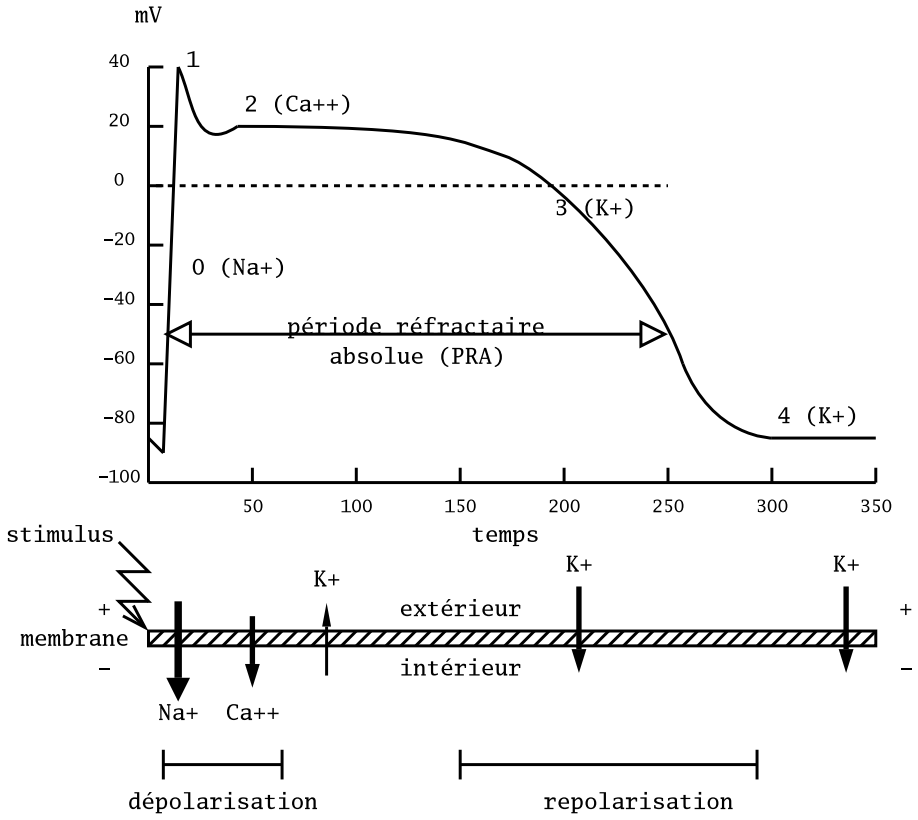


FIG. 1.5 – Les mouvement ioniques transmembranaires donnent naissance au potentiel d'action (myocytes ventriculaires).

soumise aux périodes réfractaires de chaque cellule. L'activation de proche en proche des cellules ne se fait que dans un certain sens (des cellules nodales jusqu'aux cellules myocardiques les plus éloignées) établissant un front directionnel de la dépolarisation du myocarde. La période réfractaire, jouant le rôle tampon d'empêcher un retour de stimulation dans le sens inverse, contribue à stabiliser l'activité électrique myocardique.

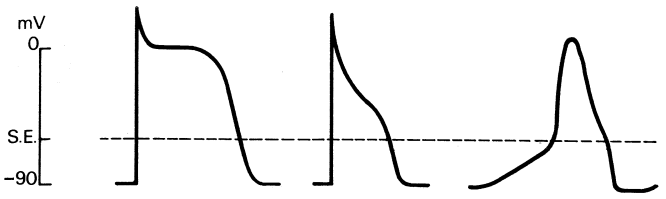


FIG. 1.6 – Diverses formes du potentiel d'action. De gauche à droite : potentiel d'une cellule du myocarde commun ventriculaire, potentiel d'une cellule du myocarde commun auriculaire, potentiel d'une cellule nodale. S.E. : Seuil d'Excitabilité.

Sur la figure 1.6, on peut voir les différences de comportement du potentiel d'action selon qu'il s'agisse d'une cellule nodale ou musculaire myocardique. Sur les cellules nodales, il n'y a jamais de phase de repos car l'excitabilité est toujours entretenue. Cette excitation est dépendante du système sympathique et parasympathique.

Séquence normale d'activation cardiaque Dans une séquence normale d'activation cardiaque telle que décrite par Tawara (Tawara, 1906; Tawara, 2000), le stimulus de départ du cycle cardiaque est généré par les cellules nodales du nœud sinusal. La propriété d'automaticité des cellules nodales du nœud génère un potentiel régulier et d'amplitude suffisante pour exciter les cellules myocardiques des oreillettes. Cette activité, communiquée de proche en proche, est stoppée au nœud AV qui le communique ensuite au faisceau de His puis à la branche droite et aux branches gauches puis aux fibres de Purkinje qui irriguent les myocytes ventriculaires. On peut ainsi suivre le cycle mécanique. Tout d'abord les cellules des oreillettes sont dépolarisées, ce qui provoque leur contraction (systole auriculaire), l'onde traverse le nœud auriculo-ventriculaire et le septum inter-ventriculaire, puis le ventricule droit puis gauche se dépolarisent (contraction ventriculaire isovolumique et systole ventriculaire). Enfin, les cellules du myocarde entre en repolarisation (relaxation ventriculaire) puis stabilisation (phase de repos).

1.2.4 Électrocardiographie de surface

Le corps humain étant électriquement conducteur, les potentiels d'actions générés lors de l'activité électrique cardiaque peuvent être recueillis par des électrodes placées sur la peau. L'enregistrement de cette activité électrique du cœur, sur un plan frontal (par les dérivations des membres) et sur un plan horizontal (par les dérivations précordiales), est un électrocardiogramme (ECG). L'ECG est un outil diagnostic permettant de détecter les pathologies cardiaques rythmiques, musculaires, les problèmes extra-cardiaques métaboliques, médicamenteux, hémodynamiques et autres.

Les électrodes peuvent être utilisées selon deux modes, l'un dit bipolaire où le potentiel d'une électrode est soustrait à une autre, et l'autre dit unipolaire où le potentiel d'une électrode est pris par rapport à un point de référence qui est généralement une moyenne du potentiel de toutes les autres électrodes utilisées ou une électrode éloignée de toute activité électrique (la masse). Toute activité électrique se dirigeant vers l'électrode est enregistrée par une déflexion positive et toute activité s'en éloignant est enregistrée par une déflexion négative.

Dans les sections suivantes, les différentes dérivations standards de l'ECG clinique et l'enregistrement ambulatoire de Holter sont présentés. Puis, la nomenclature des ondes et intervalles utilisée pour l'analyse de l'ECG est détaillée. Enfin, la dernière section introduit les artefacts visibles sur l'ECG que l'on rencontre en utilisation clinique.

1.2.4.1 Électrocardiogramme à douze dérivations

L'ECG est un enregistrement de surface de l'activité électrique du cœur, par des électrodes reliées à un électrocardiographe qui amplifie le signal électrique. Les tissus se trouvant entre le cœur et les électrodes parasitent le signal, le tracé électrocardiographique n'est donc qu'une estimation de l'activité électrique générée par le cœur.

L'ECG standard est enregistré sur 12 dérivations (six dérivations des membres et six précordiales), avec une vitesse de déroulement du papier à 25 mm par seconde et une amplitude de 10 mm pour 1 mV.

Dérivations bipolaires des membres Les dérivations bipolaires des membres permettent d'étudier l'activité électrique du cœur sur le plan frontal. Elles ont été déterminées par Einthoven (Einthoven, 1906) au début du vingtième siècle et restent encore utilisées aujourd'hui. Ces trois dérivations sont déduites des trois électrodes posées sur les membres. Soit VL le potentiel sur le bras gauche, VR le potentiel sur le bras droit et VF le potentiel sur la jambe gauche, les trois dérivations sont :

- DI (dérivation I) dont la différence de potentiel respecte l'équation $DI = VL - VR$,
- DII (dérivation II) avec $DII = VF - VR$,
- DIII (dérivation III) avec $DIII = VF - VL$.

Ces trois dérivations constituent le triangle d'Einthoven et chaque dérivation est une arête du triangle (voir Figure 1.7).

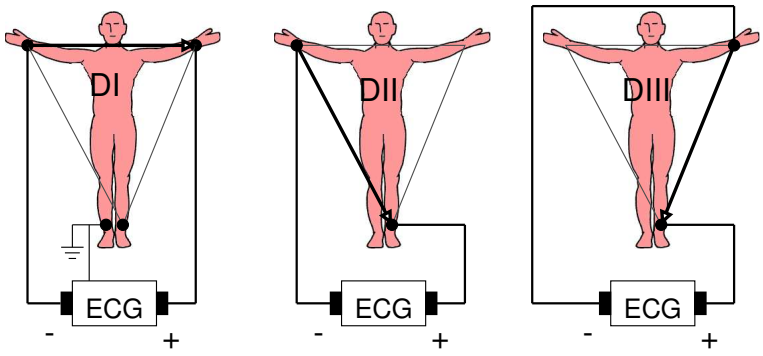


FIG. 1.7 – Montage d'Einthoven pour l'enregistrement des dérivations bipolaires des membres.

Le triangle d'Einthoven est formé par les trois électrodes posées sur le bras droit le bras gauche et la jambe gauche. Sur la jambe droite est posée une électrode qui sert de référence (la masse). Le cœur se trouve au centre du triangle et les trois dérivations bipolaires permettent l'enregistrement sous trois angles différents.

Dérivations unipolaires des membres Les dérivations unipolaires des membres permettent d'étudier l'activité électrique du cœur sur le plan frontal. Elles ont été

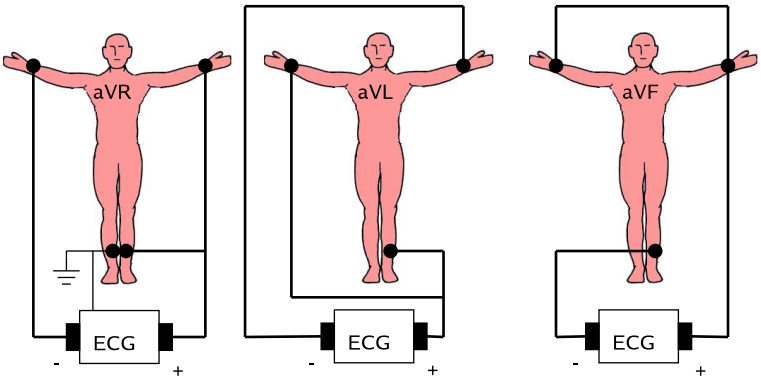


FIG. 1.8 – Montage de Goldberger pour l'enregistrement des dérivations unipolaires des membres augmentés.

déterminées par Wilson (Wilson et coll., 1934) en 1934 et améliorées par Goldberger (Goldberger, 1942). Les dérivations périphériques de Goldberger se servent des mêmes électrodes qu'Einthoven. Chaque électrode est prise comme pôle positif avec pour référence négative les 2 autres électrodes. Les droits dérivations sont :

- aVL (a comme *augmented*) dont la différence de potentiel respectent l'équation $aVL = VL - \frac{VR+VF}{2} = 1,5VL$ car selon la loi de Kirchhoff $VL + VR + VF = 0$;
- aVR avec $aVR = 1,5VR$;
- aVF avec $aVF = 1,5VF$.

Ces trois dérivations constituent trois vecteurs passant au centre du triangle d'Einthoven (voir Figure 1.8).

Dérivations précordiales Ce sont des dérivations unipolaires, mises au point par Wilson (Wilson et coll., 1944). Elles sont posées sur le thorax et sont désignées par la lettre V suivie du numéro de leur emplacement. Le potentiel de l'électrode exploratrice est pris par rapport à la moyenne des potentiels VL, VR et VF.

Six points, définis par Wilson, permettent d'obtenir les dérivations V1 à V6. Leur emplacement est représenté sur la figure 1.9.

Les dérivations précordiales ont deux caractéristiques qui les distinguent fondamentalement des dérivations des membres : elles mesurent l'activité électrique cardiaque dans le plan horizontal et sont posées à proximité du cœur.

1.2.4.2 Enregistrement ambulatoire Holter

L'enregistrement électrocardiographique de longue durée selon la méthode de Holter (Holter, 1961) permet de compléter l'électrocardiogramme standard pour le diagnostic des troubles du rythme et de la conduction. Il consiste à enregistrer un électrocardiogramme de surface modifié pendant au moins 24 heures. L'ECG Holter se différencie de l'ECG standard, de par son contexte d'observation à la fois ambulatoire et longue

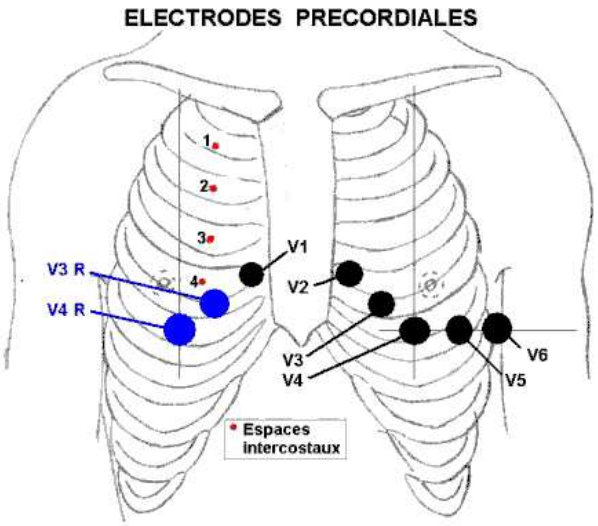


FIG. 1.9 – Position des électrodes précordiales.

durée. Il est par conséquent générateur de situations souvent difficiles en termes de reconnaissance d'événements ou de diagnostic, ceci pour deux raisons :

- la diversité et le caractère aléatoire des dégradations pouvant affecter ponctuellement ou durablement les informations d'ECG,
- la modulation des caractéristiques morphologiques et temporelles du signal ECG, relative à l'activité régulatrice du système cardio-vasculaire en réponse aux besoins métaboliques du sujet.

Durant l'enregistrement, le patient doit tenir un journal d'activité. Ainsi, le cardiologue peut analyser sur une longue période (de 24 à 48 heures) l'évolution du rythme cardiaque en fonction des activités du patient. Par exemple, une augmentation du rythme sur l'enregistrement sera normale si le patient pratiquait une activité physique à ce moment là. Les dérivations utilisées ne sont pas les mêmes que pour l'ECG standard (voir Figure 1.10). Elles sont de type bipolaires thoraciques qui, contrairement aux dérivations périphériques, sont à la fois peu sensibles aux courants d'action musculaires (EMG) s'additionnant au signal ECG et très sensibles aux aspects ischémiques rencontrées en cas d'insuffisance coronaire. La position des électrodes est déterminée en partie en fonction de la pathologie suspectée.

Les électrodes sont reliées à un enregistreur à bande magnétique ou à mémoire solide (numérique) que le patient garde à la ceinture pendant son activité quotidienne. L'enregistrement est ensuite analysé par le médecin. Étant donnée la très grande quantité de données à analyser (2160 mètres d'ECG pour 24 heures), il existe trois méthodes d'analyse (Adamec et Adamec, 2000).

- La lecture manuelle, inventée par N.J. Holter, consiste à visualiser l'enregistrement en vitesse accélérée sur un oscilloscope. Les battements normaux s'affichent comme une image fixe tandis que les anomalies (extrasystole, pause, tachycardie)

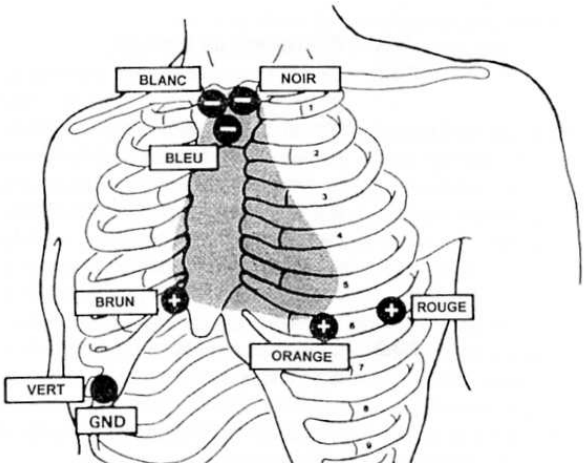


FIG. 1.10 – Position recommandée des électrodes pour l'enregistrement Holter.

apparaissent en dehors ou déforment l'image normale.
- La lecture semi-automatique consiste à apprendre automatiquement les éléments essentiels de l'enregistrement. Lors de la lecture, le système s'arrête sur chaque nouvelle anomalie que le médecin doit classer.
- La lecture automatique utilise des techniques variables (templates matching, réseaux neuronaux, etc.) pour détecter les anomalies. Cette méthode, très rapide, ne se passe pas d'une interprétation et vérification humaine.

Les enregistrements Holter représentent un immense réservoir de données médicales et la plupart des bases de données de la recherche en cardiologie électrocardiographique sont constituées de ce type d'examen[5].

1.2.4.3 Ondes et intervalles

Les différentes ondes et les intervalles les séparant ont permis de dresser le standard ECG. Toute interprétation de l'électrocardiogramme se réfère aux caractéristiques de forme et de largeur d'onde du signal ECG normal présentées Figure 1.11, sont détaillées par la suite.

Onde P : dépolarisation des oreillettes L'onde P correspond à la dépolarisation des oreillettes depuis le nœud sinusal vers le nœud atrio-ventriculaire. C'est l'onde qui précède le complexe QRS.

Complexe QRS : dépolarisation des ventricules Le complexe QRS est une imbrication de 3 ondes accolées qui suivent l'onde P et qui correspondent à la dépolarisation des ventricules. Par définition, l'onde Q est la première onde négative, l'onde

[5]http ://www.physionet.org/physiobank/

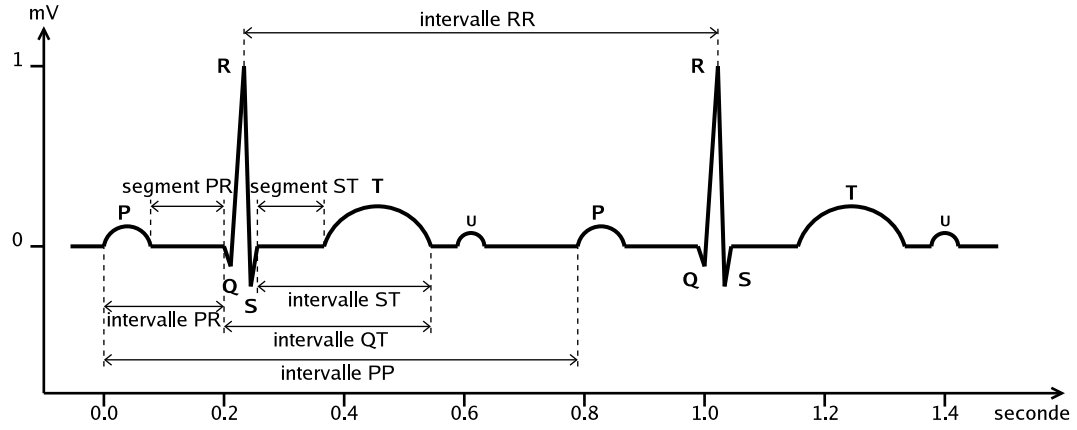


FIG. 1.11 – ECG normal avec les notations usuelles de l'électrocardiographie clinique.

R la première onde positive du complexe, et l'onde S la première onde négative après l'onde R. Toute onde supplémentaire, positive ou négative, sera appelée R', S', R", etc. La figure 1.12 donne les différents aspects que peut prendre le QRS.

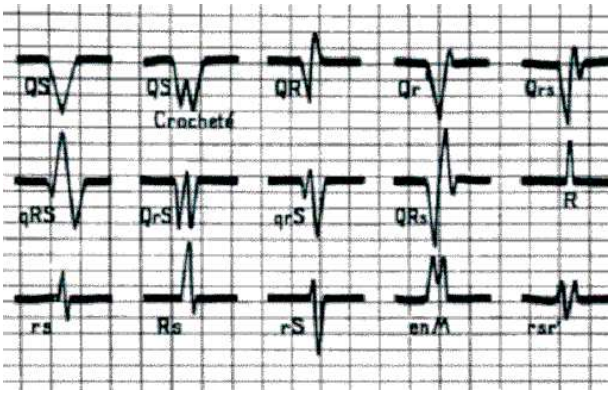


FIG. 1.12 – Schémas des différents aspects du complexe QRS et de leur notation.

Onde T : repolarisation des ventricules L'onde T correspond à la repolarisation des ventricules. Cette onde succède au complexe QRS après retour à la ligne isoélectrique (sauf pathologie particulière).

Onde U : repolarisation des fibres de Purkinje ? L'onde U est une onde positive qui suit l'onde T, visible essentiellement en précordiale (surtout en V2 et V3), dont la signification est discutée (repolarisation prolongée des cellules du réseau de Purkinje ou à un facteur mécanique correspondant à la relaxation du myocarde).

Intervalle RR : fréquence des battements cardiaques L'intervalle RR correspond au délai entre deux dépolarisations des ventricules. C'est cet intervalle qui permet de calculer la fréquence cardiaque.

Intervalle PP : période de polarisation des oreillettes L'intervalle PP correspond au délai entre deux dépolarisations des oreillettes.

Segment PR : pause du nœud AV Le segment PR correspond au délai entre la fin de la dépolarisation des oreillettes et le début de celle des ventricules. C'est le temps pendant lequel l'onde de dépolarisation est bloquée au niveau du nœud AV.

Intervalle PR : durée de conduction auriculo-ventriculaire L'intervalle PR correspond au délai entre le début de la dépolarisation des oreillettes et celle des ventricules. C'est le temps de propagation de l'onde de dépolarisation jusqu'aux cellules myocardiques ventriculaires.

Intervalle QT : durée de systole ventriculaire Cet intervalle correspond au temps de systole ventriculaire, qui va du début de l'excitation des ventricules jusqu'à la fin de leur relaxation.

Segment ST : durée de stimulation complète des ventricules Le segment ST correspond à la phase pendant laquelle les cellules ventriculaires sont toutes dépolarisées, le segment est alors isoélectrique.

1.2.4.4 Artefacts visibles sur l'électrocardiogramme

Sur tout enregistrement électrocardiographique il peut apparaître des événements indésirables pouvant brouiller le tracé et, parfois, induire en erreur le diagnostic final. Ces bruits sont reconnaissables par l'œil expérimenté qui les identifie avant d'effectuer son diagnostic. Ces perturbations ont fait l'objet d'études (Moody et coll., 1984) et restent, pour certaines, encore difficiles à traiter de manière automatique. Les effets indésirables peuvent avoir plusieurs sources : techniques, physiques, pathologiques, ou pharmacologiques. Nous allons surtout développer l'aspect technique et physique des bruits et artefacts présents sur les tracés électrocardiographiques notamment sur les tracés Holter.

En partant du principe que les bruits fréquents en électrocardiographie sont des bruits additifs, les caractéristiques de ces bruits auxquelles nous allons nous attacher sont : l'amplitude, la périodicité, et la bande spectrale. Les artefacts prennent une place particulière dans ce chapitre puisqu'ils induisent des modifications des performances des algorithmes chargés de l'analyse automatique de l'électrocardiogramme.

Bruits techniques Le matériel utilisé lors de l'enregistrement doit être manipulé avec précaution car il peut être source de bruits lors de l'enregistrement. Les plus courants sont présentés ci dessous.

Bruit dû au secteur Le réseau de distribution électrique peut parfois brouiller le signal électrocardiographique avec une onde dont l'harmonique principale est à 50 Hz (voir Figure 1.13).

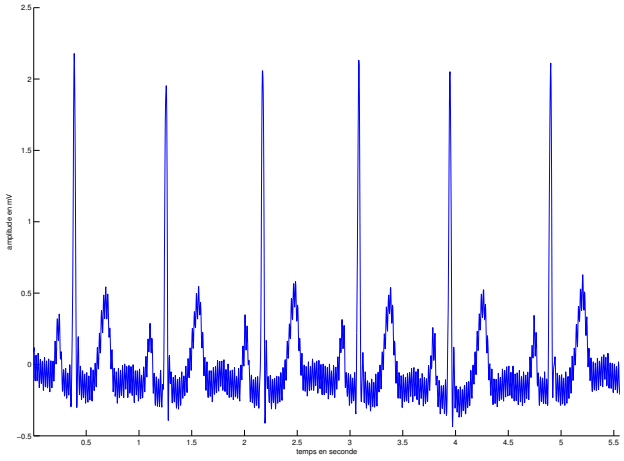


FIG. 1.13 – Signal électrocardiographique perturbé par le secteur

Ce type de bruit apparaît sur tout l'enregistrement et peut être assez fort mais il s'élimine facilement avec un filtre sélectif car c'est un bruit haute fréquence à bande étroite.

Bruit dû aux mouvements d'électrodes Lorsque les électrodes sont connectées incorrectement, des sauts brusques de la ligne de base apparaissent. L'effet sur le tracé peut aller de la simple diminution d'amplitude à l'apparition de pics lorsque les électrodes sont en contact intermittent avec la peau. Ces pics peuvent parfois être confondus avec les ondes du tracé normal (voir Figure 1.14).

Ce type de bruit intermittent à bande spectrale large s'élimine difficilement car son énergie se trouve dans la même gamme de fréquence que le complexe QRS.

Autres bruits courants Parmi les bruits courants on peut citer les artefacts dus aux mouvements des câbles électriques, la saturation des instruments de mesure, les mauvais câblages, les artefacts dus au port de vêtements synthétiques, etc.

Artefacts physiques Les artefacts physiques sont dus aux activités électriques du corps humain telles que les commandes de contraction des muscles ou la respiration.

Mouvements de la ligne de base Lors de l'enregistrement de l'électrocardiogramme, l'activité respiratoire peut faire osciller la ligne de base de l'ECG à un rythme

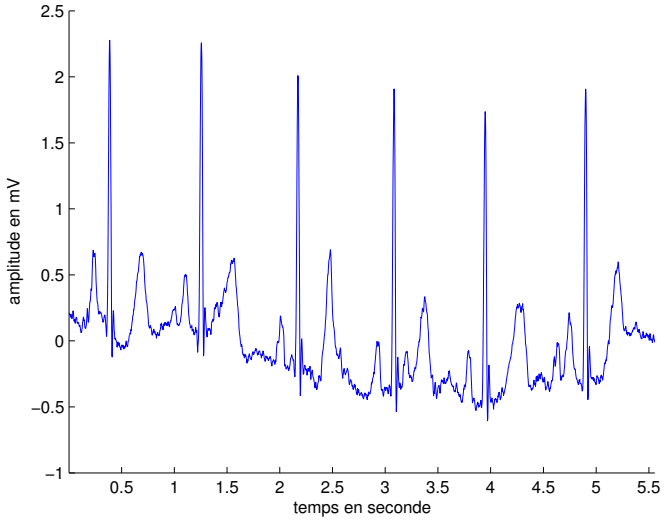


FIG. 1.14 – Bruit dû aux mouvements des électrodes

régulier (voir 1.15). D'autres perturbations peuvent avoir pour effet de déplacer temporairement la ligne de base comme, par exemple, les mauvais contacts entre la peau et les électrodes.

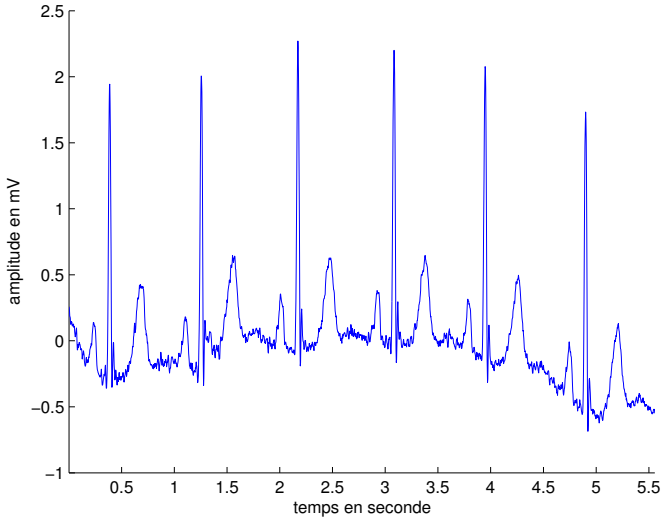


FIG. 1.15 – Mouvements de la ligne de base

Ces perturbations sont généralement peu gênantes pour l'analyse de l'ECG et peuvent être en grande partie filtrées car leur énergie se situe dans une bande de fré-

quence basse, qui empiète peu sur celle de l'ECG normal.

Bruit myoéletrique ou tremblement somatique La contraction d'un muscle est commandée par une dépolarisation des cellules musculaires et, bien que les électrocardiographes soient construits pour être surtout sensibles aux fréquences du myocarde, l'ECG enregistre les contractions des muscles squelettiques. L'aspect le plus courant est une oscillation à haute fréquence (voir 1.16) liée à la tension musculaire d'un sujet qui n'est pas convenablement détendu.

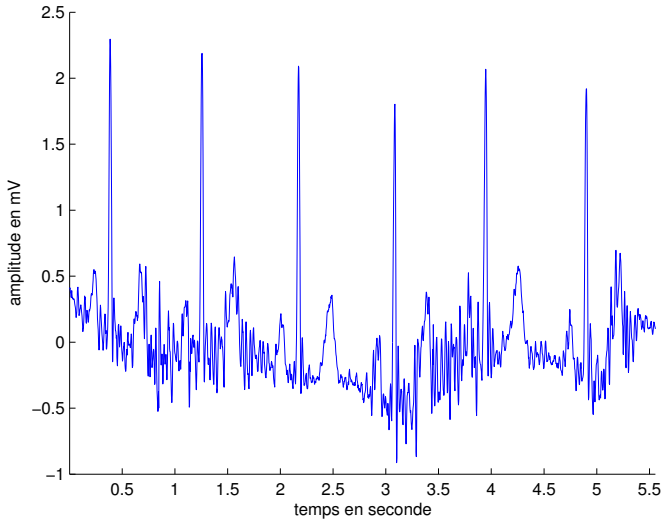


FIG. 1.16 – Bruit musculaire

Ces perturbations sont assez gênantes lorsque le patient bouge beaucoup ou lorsqu'il frissonne, elles peuvent noyer les ondes P et T et empêcher un diagnostic fiable. L'apparition de ces perturbations dépend de l'état du patient, s'il est très tendu ou atteint de maladie de Parkinson, l'enregistrement peut être de mauvaise qualité sur toutes les voies de l'ECG.

Autres artefacts altérants l'ECG Certaines maladies généralisées peuvent affecter le tracé électrocardiographique. L'hyperthyroïdie, l'ischémie, l'hypokaliémie (prolongement de l'intervalle QT, onde T aplatie), modifient l'électrocardiogramme. L'usage de médicament, notamment la digoxine pour bloquer la conduction AV et ralentir la fréquence cardiaque, a aussi son effet sur le tracé. La digitaline provoque un abaissement du segment ST avec inversion des ondes T et tend à raccourcir l'intervalle QT.

1.2.5 Troubles du rythme et de la conduction cardiaque

Sous cette dénomination on regroupe les arythmies cardiaques et les blocs cardiaques. Le meilleur outil pour diagnostiquer une arythmie est l'électrocardiogramme

(Lake, 1990). Dans l'analyse de l'ECG, les pathologies ou anomalies sont détectées et classées en fonction de leur déviation par rapport au rythme idéal qu'est le rythme sinusal (voir section 1.2.3.3). Chaque déviation visible sur l'ECG peut être attribuée à une anomalie physiologique. Ainsi, les blocs cardiaques sont dus à un défaut de conduction de l'onde de dépolarisation à travers le myocarde différencié et les arythmies sont générées par un foyer ectopique prenant le relais ou supplantant le nœud sinusal. Ces pathologies ne sont pas exclusives, un patient peut être atteint d'arythmies et de blocs cardiaques. Les sections suivantes détaillent le rythme sinusal, les blocs cardiaques et certaines arythmies cardiaques.

Rythme sinusal Le rythme sinusal est le rythme normal cardiaque. Il correspond à une activation physiologique des oreillettes, puis des ventricules, à partir du nœud sinusal. Son rythme est compris entre 60 à 80 battements par minute avec un intervalle régulier entre des battements normaux. Le cœur s'accélère normalement lors de l'activité physique, dans les circonstances physiologiques qui exigent un surcroît de demande métabolique ou sous l'effet des émotions ou d'excitants tels que café, tabac, alcool.

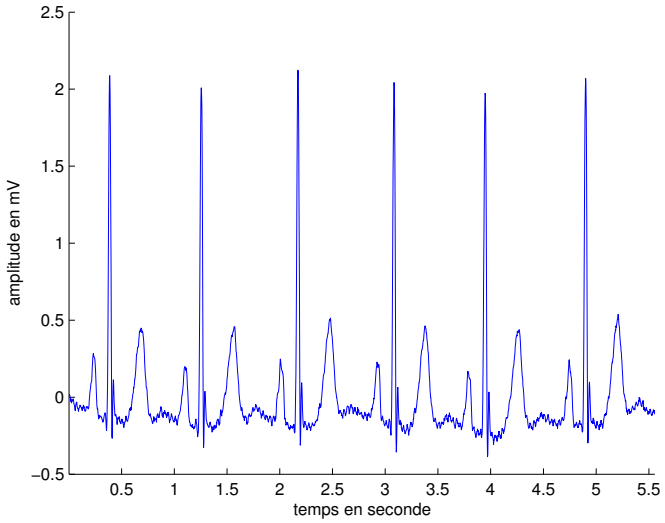


FIG. 1.17 – Exemple d'ECG normal.

Blocs cardiaques Les blocs cardiaques sont dus à une rupture de conduction du myocarde différencié qui altère la dépolarisation du myocarde. Ces ruptures peuvent être plus ou moins sévères : freinantes (allongement du temps de parcours), intermittentes (un stimulus sur 2 ou 3 est conduit), ou complète (aucune conduction). Nous allons détailler les quatre types de blocs cardiaques :

Bloc Sino-Auriculaire (Bloc SA) Le nœud sinusal peut ne pas transmettre de stimulus aux cellules des oreillettes. La conséquence est qu'au moins un cycle complet n'est pas effectué. Après la pause, due au bloc, le cycle reprend normalement si aucun autre foyer ectopique n'a déclenché de contraction.

Blocs Auriculo-Ventriculaire (BAV) On appelle BAV l'altération de la conduction du stimulus de dépolarisation entre les oreillettes et les ventricules. On distingue trois degrés de sévérité.

- Les BAV de premier degré provoquent l'allongement du segment PR de façon égale à chaque cycle.
- Les BAV de deuxième degré traduisent l'absence momentanée d'onde QRS après une onde P normale. Lorsque les segments PR précédents sont normaux, on parle de Mobitz de type II (voir Figure 1.18). Lorsque les segments précédents vont en s'augmentant, on parle de phénomènes de Wenckebach (ou Mobitz de type I).
- Les BAV de troisième degré sont dit complets, c'est-à-dire qu'aucune dépolarisation auriculaire ne parvient aux ventricules. Un foyer ectopique ventriculaire ou jonctionnel joue alors le rôle de pacemaker. Le foyer est identifiable par la forme et la fréquence des battements. Les activités auriculaire et ventriculaire sont complètement dissociées.

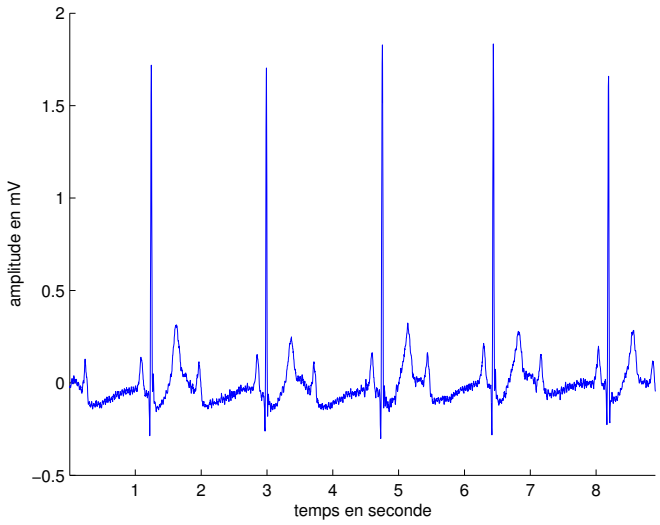


FIG. 1.18 – Exemple de Mobitz de type II.

Blocs de branche Le bloc de branche est dû au blocage de la dépolarisation dans une des branches du faisceau de His. Un bloc dans l'une ou l'autre branche provoque un retard dans la dépolarisation du ventricule auquel elle appartient. La dépolarisation des ventricules est désynchronisée et le complexe QRS est élargi (voir Figure 1.19).

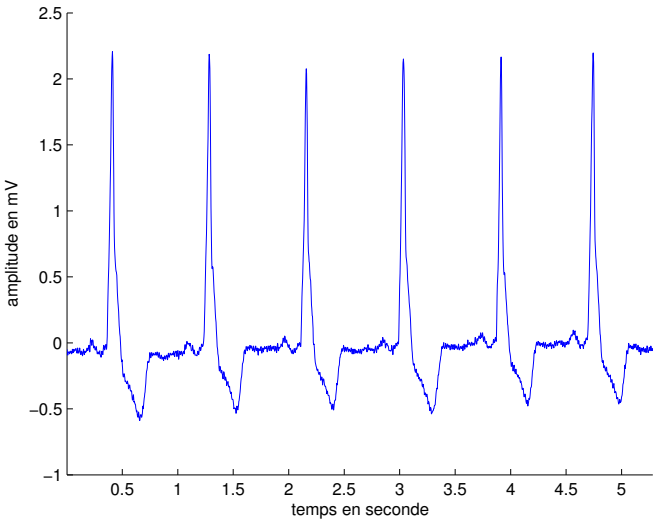


FIG. 1.19 – Exemple de bloc de branche gauche.

Arythmies Les arythmies proprement dites relèvent de l'entrée en jeu d'un « foyer » ectopique qui peut se situer dans n'importe quelle portion du cœur, ou de la formation d'un circuit électrique (appelé réentrée notamment par les faisceaux de Kent), dont la localisation peut être auriculaire, jonctionnelle (entre oreillettes et ventricules) ou ventriculaire. Les paragraphes suivants présentent quelques arythmies dites supraventriculaires (auriculaires ou jonctionnelles) et des arythmies ventriculaires.

Arythmies supraventriculaires

La tachycardie supraventriculaire (TSV) (maladie de Bouveret) correspond à un mécanisme de réentrée qui peut soit être localisé à l'intérieur du nœud auriculo-ventriculaire (réentrée nodale), soit couvrir un large circuit empruntant dans un sens la voie de conduction normale (nœud AV et faisceau de His) et dans l'autre le faisceau de Kent. La tachycardie est rapide (entre 180 et 220 pulsations par minute) parfaitement régulière, à début et fin brusques. Les complexes QRS sont normaux et l'activité auriculaire se superpose à ces derniers ou les suit. Les accès se répètent à des intervalles variables, mais l'évolution reste bénigne car il n'existe pas, sauf association occasionnelle, d'anomalie des valves, du muscle cardiaque ou des coronaires.

La fibrillation auriculaire (FA) est une désorganisation totale de l'activité électrique auriculaire avec perte de la contraction mécanique rythmée des oreillettes. Les oreillettes sont parcourues de multiples ondes électriques, qui se propagent de façon anarchique à une très grande vitesse, supérieure à 350 par minute. Les ondes auriculaires de fibrillation (f) sont en grande partie bloquées dans le nœud auriculo-ventriculaire qui joue le rôle de filtre, protégeant les ventricules d'une activité trop rapide. Le choc élec-

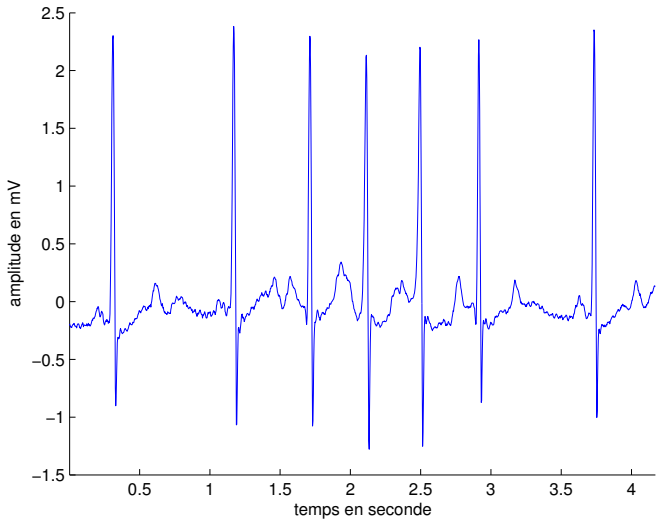


FIG. 1.20 – Exemple d'accès de tachycardie supraventriculaire.

trique externe est le moyen le plus efficace pour rétablir le rythme sinusal (emploi d'un défibrillateur). La FA peut être traitée par des soins médicamenteux (antiarythmiques) et s'ils se révèlent inappropriés, il devient nécessaire d'implanter un défibrillateur.

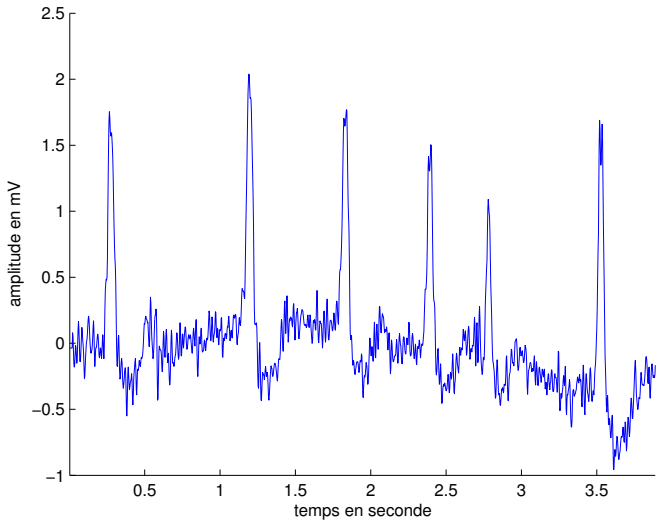


FIG. 1.21 – Exemple de fibrillation auriculaire.

Arythmies ventriculaires

L'arythmie extrasystolique est la plus fréquente. Les extrasystoles sont des battements ectopiques, uniques ou répétés, provenant d'un seul ou de plusieurs foyers qui peuvent entraîner des sensations désagréables de ratés, de coups dans la poitrine, d'arrêts de cœur ou de palpitations. Les extrasystoles sont des phases systoliques en trop qui seront apparentes sur l'ECG par un complexe QRS large (Figure 1.22). Les extrasystoles ne constituent habituellement pas en elles-mêmes un facteur de gravité, leur pronostic dépend de l'état cardiaque qui peut être absolument normal (extrasystoles dites bénignes) ou pathologique. Lorsqu'il existe un double foyer ventriculaire, on parle de **doublet ventriculaire** (Figure 1.23).

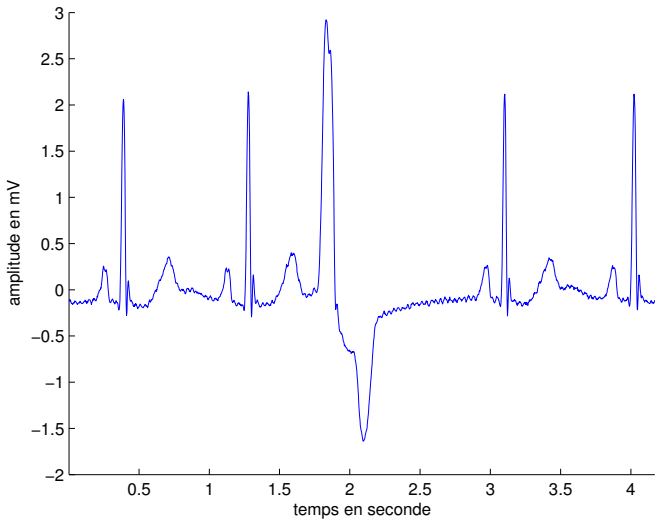


FIG. 1.22 – Exemple d'extrasystole.

Les bigéminismes et trigéminismes sont des rythmes à deux commandes. La commande de base (généralement sinusale) est interrompue par des battements d'origine ectopique. Lorsque l'on se trouve en présence d'un bigéminisme les QRS qui appartiennent au rythme de base sont suivis d'un QRS d'origine ectopique avec une succession de 1/1. On parle de trigéminisme lorsqu'on est en présence d'une succession 2/1. Les figures 1.24 et 1.25 présentent respectivement un exemple de rythme de bigéminisme et de trigémisme dans le cas d'un rythme sinusal avec un foyer ectopique ventriculaire.

Les tachycardies ventriculaires (TV) représentent les arythmies les plus graves, elles compromettent souvent l'hémodynamique et peuvent dégénérer en fibrillation ventriculaire létale. Les complexes QRS sont toujours élargis et le rythme est rapide. L'accès de tachycardie ventriculaire persistante est une urgence médicale, l'arrêt de la crise pouvant être obtenu par injection intraveineuse d'un antiarythmique ou par choc électrique

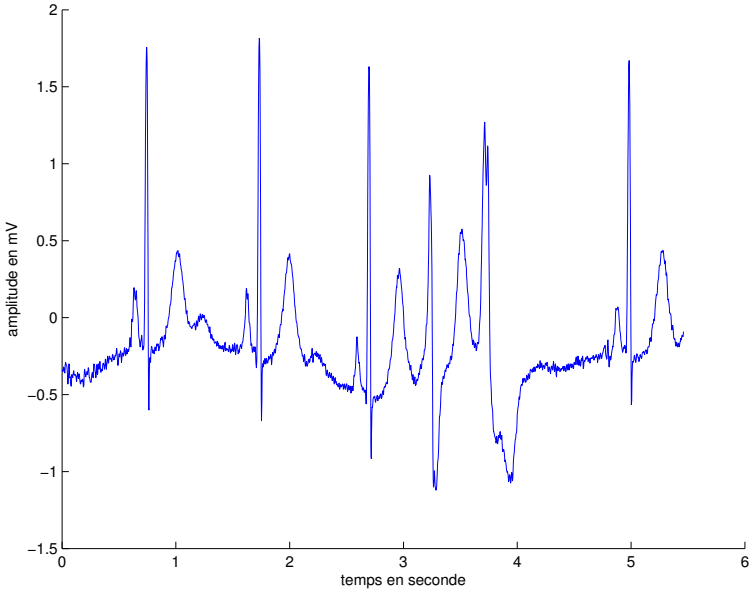


FIG. 1.23 – Exemple de doublet ventriculaire.

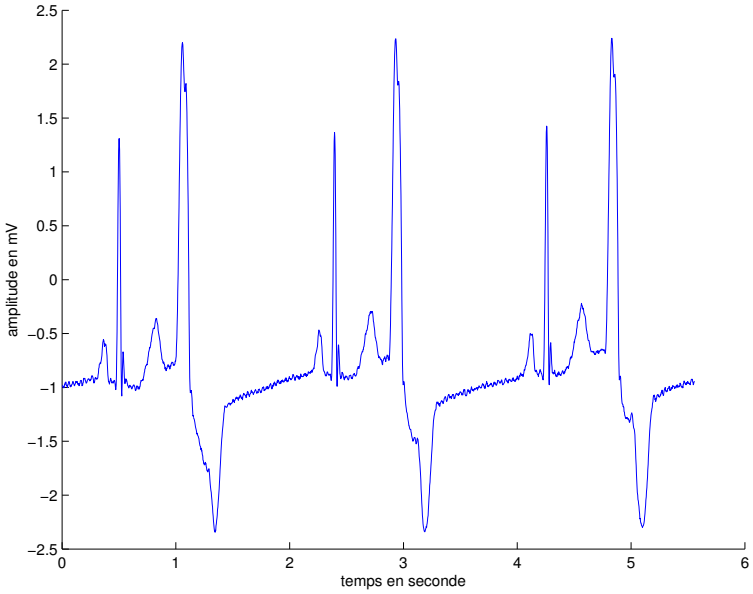


FIG. 1.24 – Exemple de rythme de bigéminisme.

externe. Quand les récidives de tachycardie ne sont pas prévenues par la médication antiarythmique, le recours à des méthodes non pharmacologiques est légitime : exérèse chirurgicale de la zone arythmogène, défibrillateur implantable.

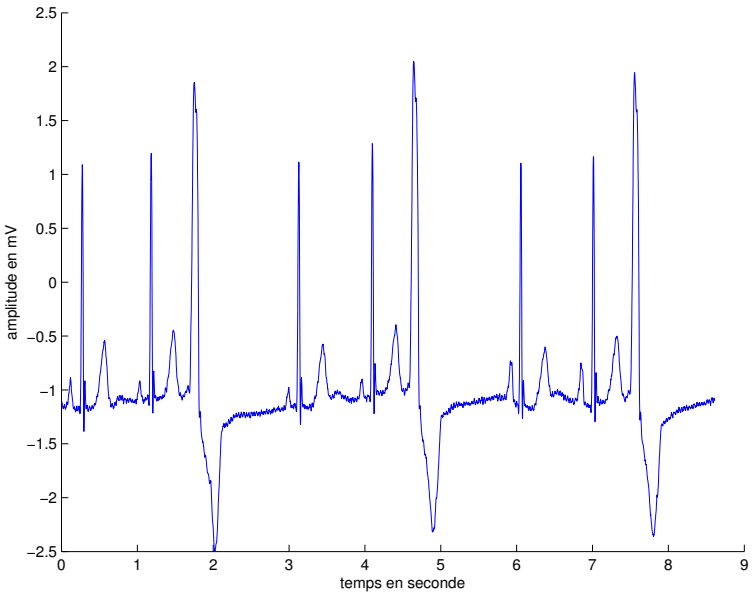


FIG. 1.25 – Exemple de rythme de trigéminisme.

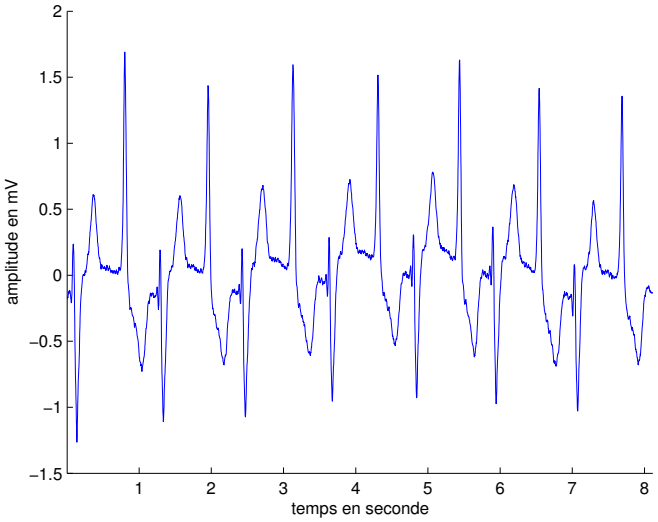


FIG. 1.26 – Exemple de tachycardie ventriculaire.

La fibrillation ventriculaire (FV) est la cause principale de mort subite. Elle peut apparaître d'emblée, comme manifestation d'instabilité électrique, dans les premières minutes ou heures d'un infarctus du myocarde aigu ou succéder à une autre arythmie ventriculaire chez les malades ayant une cardiopathie sévère. Les contractions complètement anarchiques des ventricules aboutissent très rapidement à une inefficacité

cardio-circulatoire qui est létale en l'absence de manœuvres de réanimation (massage cardiaque, ventilation assistée, choc électrique externe). Ces dernières doivent être entreprises en quelques minutes pour éviter les complications cérébrales secondaires à une privation d'oxygène prolongée au niveau des cellules.

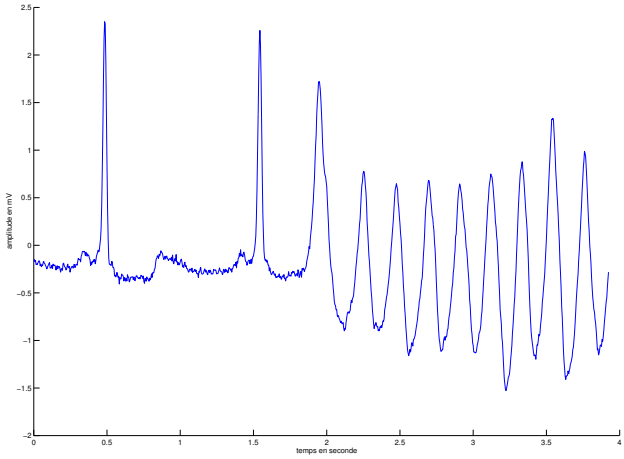


FIG. 1.27 – Exemple d'accès de fibrillation ventriculaire.

Diagnostic des arythmies Le problème principal dans le diagnostic des arythmies cardiaques est de trouver le siège du foyer ectopique qui peut être soit supraventriculaire (auriculaire ou jonctionnel) ou ventriculaire. Pour ce faire le cardiologue décrit l'électrocardiogramme selon les ondes et intervalles définis précédemment en section 1.2.4.3. Ce paragraphe a pour but de présenter les deux principales méthodes pour diagnostiquer des arythmies à partir d'ECG : l'analyse de la morphologie des QRS et l'analyse des relations entre l'activité auriculaire et ventriculaire.

Analyse morphologique Les arythmies supraventriculaires, naissent au-dessus de la bifurcation du faisceau de His. Elles sont théoriquement caractérisées par des QRS fins car l'activation ventriculaire reste normale (donc rapide).

Les arythmies ventriculaires sont au contraire caractérisées par l'existence de complexes QRS élargis et déformés. L'activation naît en un point quelconque du myocarde ventriculaire et entraîne une activation anormale de la masse ventriculaire. La durée du QRS est supérieure à 120 ms, avec un aspect de bloc de branche.

Dans certains cas, cette différenciation entre arythmie supraventriculaire ou ventriculaire peut être très difficile. C'est le cas par exemple des arythmies supraventriculaires associées à un bloc de branche préexistant, les QRS sont alors élargis quelque soit le foyer et peuvent être confondus avec une arythmie ventriculaire.

Relations entre l'activité auriculaire et ventriculaire L'analyse de l'activité électrique auriculaire est parfois difficile, surtout quand le rythme ventriculaire est rapide. En cas d'arythmies supraventriculaires, il existe le souvent une lésion constante entre l'activité auriculaire et ventriculaire. En cas d'arythmies ventriculaires, on observe le plus souvent une dissociation entre l'activité auriculaire et l'activité ventriculaire.

1.3 Systèmes de monitorage de patient

1.3.1 Introduction

La définition du monitorage au sens large est difficile à donner tant les objectifs et les techniques employées divergent en fonction des applications. En effet, le monitorage est appliqué à la surveillance du trafic routier, d'un réseau de télécommunication, d'une centrale nucléaire, etc. Le monitorage de systèmes dynamiques peut être défini comme la surveillance en continu d'un système par une opération de haut niveau qui analyse toutes les situations rencontrées, communique avec un opérateur et propose des actions à effectuer sur le système (Cauvin et coll., 1998). L'opération de monitorage consiste donc principalement à détecter les pannes, à retrouver la chaîne causale qui les ont provoquées et à proposer une intervention par une alarme.

Dans les applications industrielles, les systèmes surveillés peuvent être entièrement définis ou modélisés (du moins théoriquement) à un degré d'abstraction plus ou moins élevé. Il n'en va pas de même pour le monitorage de patient où le système surveillé est le corps humain. La médecine n'étant pas une science exacte, la connaissance utilisée pour modéliser le système reste incomplète et approximative. Même si les organes commencent à être bien compris, les effets des différents organes les uns sur les autres et les facteurs psychologiques ne sont pour l'instant pas modélisables (Le Beux et coll., 1994). Le monitorage de patient est donc un champ d'application riche pour les techniques de l'intelligence artificielle capables de raisonner sur des informations incomplètes ou incertaines et aider au diagnostic. Ce sont ces techniques qui sont présentées par la suite.

L'opération de monitorage en milieu hospitalier est tout d'abord décrite en section 1.3.2 Elle explique comment les systèmes de monitorage s'intègrent dans la chaîne de monitorage et comment l'évolution de l'intelligence artificielle a conduit au concept de *monitorage intelligent*. La section 1.3.3 présente le télémonitorage qui est une application émergente du monitorage de patient. En section 1.3.4, les blocs de base d'un système de monitorage intelligent (SMI) sont décrits en détail. Enfin, la section 1.3.5 détaille les types d'adaptation qu'un SMI doit gérer et comment le pilotage d'algorithmes peut permettre une gestion souple de ces différentes adaptations.

1.3.2 Monitorage de patient

L'activité de monitorage est une activité humaine courante des unités de soins intensifs. Elle consiste à surveiller un patient en continu par un ensemble d'observables

(signaux physiologiques, observations cliniques, tests de laboratoire, etc.) afin d'interpréter les situations alarmantes nécessitant une intervention médicale (Marsh, 1990).

Le monitorage médical est particulièrement dédié aux patients dont la pathologie peut évoluer rapidement et présenter des risques conséquents de mort subite, tels que les patients victimes d'une attaque cardiaque ou ayant subi une opération chirurgicale. Pour cette raison, l'objectif d'une unité de soins intensifs est de concentrer toutes les ressources humaines et matérielles à la surveillance des signes avant-coureurs d'une dégradation brusque du patient et à l'intervention rapide pour stabiliser son état (Pierson, 1998). Même s'il est difficile de donner une définition consensuelle du monitorage médical, tant les applications et les champs disciplinaires couverts sont nombreux, il convient de le définir par ses objectifs les plus généraux :

> *Le monitorage de patient est une évaluation continue et répétitive des signes physiologiques du patient en temps réel pour guider la prise de décisions –incluant les interventions thérapeutiques– et évaluer l'application de ces interventions.* (Hess, 1990)

Il convient de noter que, dans cette thèse, le terme « monitorage de patient » se rapporte à l'*activité humaine* de surveillance d'un patient et que les « système de monitorage » sont des *outils* qui s'intègrent dans la chaîne de surveillance pour assister le personnel médical. La figure 1.28 représente l'environnement classique du monitorage de patient tel que présenté dans (Mora et coll., 1993).

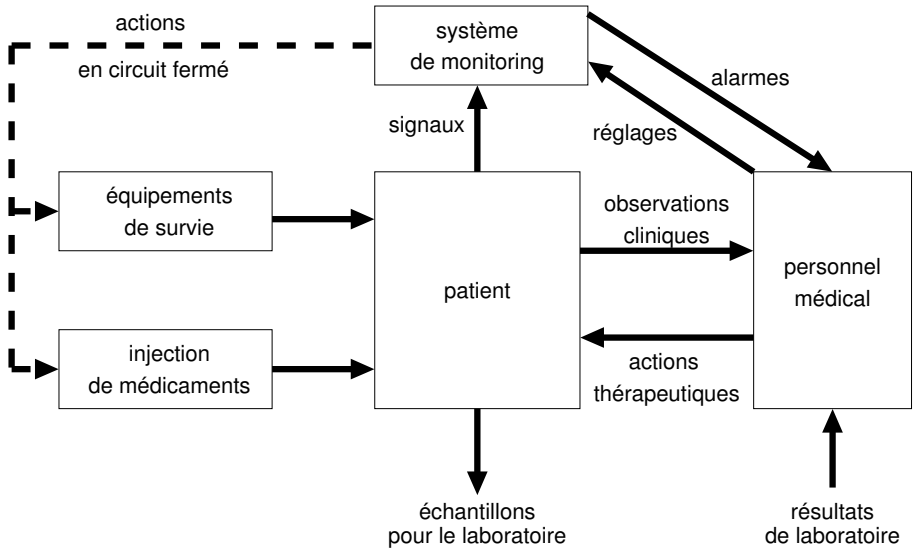


FIG. 1.28 – Environnement clinique du monitorage de patient

L'apparition de l'ordinateur en clinique a permis de révolutionner les pratiques. En unité de soins intensifs, les premiers systèmes de monitorage furent d'abord dédiés à l'automatisation de la collecte de données et l'affichage de signaux multisources, tels que l'électrocardiogramme et la pression artérielle. Ces systèmes de monitorage des signes

vitaux permettent au praticien d'avoir une vision instantanée des mesures physiologiques du patient pour repérer les variations anormales nécessitant une intervention médicale immédiate (Blitt, 1990). Aucun de ces systèmes n'est en mesure d'analyser les relations entre les données et les pathologies alors que de nombreux mécanismes de diagnostic peuvent être automatisés. C'est pourquoi, l'intégration de mécanismes de raisonnement et de diagnostic a été entreprise pour soulager le personnel médical d'une grande part de la charge cognitive dans un environnement où les décisions doivent être prises d'urgence. Depuis la fin des années 80, les systèmes de monitorage se sont perfectionnés pour gérer plus de paramètres, un diagnostic en ligne, des bases de connaissances et intégrer d'autres outils de l'intelligence artificielle (IA), ce qui a conduit au concept de *système de monitorage intelligent* (SMI) (Siregar et coll., 1989; Coiera, 1993; Mora et coll., 1993).

Les bénéfices attendus par les SMI sont nombreux, les principaux sont cités dans la liste qui suit.

- La *surveillance en continue 24h sur 24* est l'un des plus importants. Cette surveillance ininterrompue permet de pallier les défauts de surveillance humaine (manque de personnel, distraction par d'autres activités cliniques) et rassure les patients. La surveillance est donc accrue par la combinaison de la surveillance humaine et artificielle.
- L'*aide au diagnostic* apporte un soutien aux médecins en proposant des diagnostics médicaux et en donnant la chaîne causale qui a amené à ce diagnostic. L'aide au diagnostic peut aussi prédire l'évolution de l'état du patient et peut ainsi alarmer le personnel le plus tôt possible.
- L'*aide à la thérapie* est aussi importante dans les unités de soins intensifs où le temps accordé à la stratégie médicale peut être réduit par l'urgence de l'intervention. Les SMI intègrent aussi le suivi de la thérapie appliquée, le thérapeute est ainsi assisté dans toutes les étapes de décisions.
- La *gestion d'un grand nombre de capteurs* permet d'utiliser le maximum d'information disponible. Une gestion adéquate permet de donner une vision de l'évolution des paramètres sur plusieurs échelles temporelles alors que le simple affichage ne donne qu'une vision instantanée.
- L'informatisation permet aussi de conserver une trace des traitements administrés et de donner une *procédure à suivre* pour les traitements. Par exemple, le projet NEONATE (Hunter et coll., 2003) a pour objectif d'aider le personnel hospitalier en unité de soins intensifs néonatale en proposant des procédures de soins adaptées (*Guidelines*). Les auteurs proposent de créer automatiquement de telles procédures à partir des données de routine acquises (Fuchsberger et coll., 2005).

Grâce à des mécanismes de gestion de connaissances et de raisonnement, les SMI proposent des diagnostics et des thérapies. Le personnel médical est ainsi assisté et peut consacrer plus de temps aux patients et à la stratégie médicale. Ces systèmes n'ont bien évidement pas pour but de remplacer le personnel, qui lui seul est en mesure d'apprécier la nécessité des soins, mais plutôt de l'aider à plusieurs niveaux de la chaîne de monitorage notamment en l'avertissant par des alarmes lorsque l'état du patient

devient préoccupant et lors de la prise de décisions (Mora et coll., 1993).

1.3.3 Télémonitorage

Parallèlement aux progrès accomplis dans les systèmes à base de connaissances, l'émergence de systèmes de communication de plus en plus miniaturisés et haut débit a permis d'initier un grand nombre de travaux dans le domaine du télémonitorage (Meystre, 2005). Le télémonitorage consiste à acquérir à distance des informations sur un patient (vidéo, signaux physiologiques, etc.) de façon à surveiller en continu, ou à intervalles réguliers, ses signes vitaux pour une intervention immédiate en cas de crise.

Le télémonitorage répond principalement aux besoins de surveillance médicale des personnes âgées. L'espérance de vie des populations occidentales augmentant, les besoins en hospitalisation deviennent de plus en plus importants. Le télémonitorage apporte une surveillance médicale des personnes âgées sans nécessiter une prise en charge hospitalière lourde et coûteuse et leur permet de continuer à vivre dans leur milieu habituel. Un exemple de recherche dans le domaine de la surveillance de personnes âgées à domicile est donné dans (Duchêne et coll., 2005). L'objectif est d'extraire des motifs de différents capteurs d'un système de monitorage des personnes âgées dans un habitat. Ces motifs permettent de reconnaître les changements inhabituels de comportement pour détecter automatiquement une dégradation de l'état du patient.

Le suivi à distance de thérapies est aussi un objectif du télémonitorage. Par exemple, le système VIE-DIAB (Popow et coll., 2003) permet aux patients d'envoyer des informations journalières au médecin via un téléphone cellulaire. Le médecin peut ainsi suivre le traitement et envoyer des conseils par messagerie (SMS).

Une autre des applications importantes du télémonitorage est la surveillance des patients souffrant de troubles cardiaques. Étant donné qu'au moins deux tiers des décès d'origine cardiaque surviennent hors de l'hôpital, la conception de matériels de monitorage ambulatoire capables de détecter les facteurs précoces de mort subite et de contacter les urgences est un sujet de recherche très actuel (Rubel et coll., 2004; Liszka et coll., 2004). Ainsi, les auteurs de (Liszka et coll., 2004) proposent un système de monitorage des arythmies cardiaques combiné à un système de localisation GPS qui permet aux unités d'urgence d'intervenir le plus rapidement possible sur les lieux de la crise.

Le schéma classique d'un système de télémonitorage est donné Figure 1.29. Ces systèmes comprennent généralement :

- un matériel mobile porté par le patient qui acquiert les signaux physiologiques à partir de capteurs placés sur la peau ;
- un centre de calcul, qui analyse les multiples paramètres et déclenche des alarmes à destination du patient ou d'une unité d'intervention d'urgence ;
- un système de communication qui échange des données avec le centre médical (signaux, messages, alarmes, etc.).

Le projet de monitorage ambulatoire Amon (*Alert portable telemedical MONitor*) (Anliker et coll., 2004) qui propose un système compact de bracelet recueillant différents signaux physiologiques sur une seule surface du corps (le poignet) respecte ce

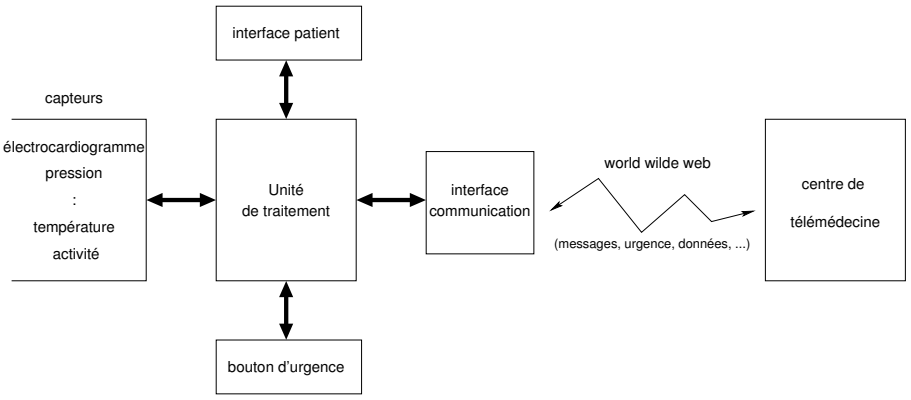


FIG. 1.29 – Système classique de télémonitorage.

schéma. Les signaux acquis sont traités pour détecter en temps réel les situations alarmantes qui sont communiquées au patient via une interface utilisateur et à un centre de télémédecine via une liaison cellulaire.

Les bénéfices attendus du télémonitorage sont nombreux. Bien entendu, la rapidité d'intervention et la localisation permettent d'éviter les morts subites. De plus, en terme de traitement, le médecin peut avoir une vision journalière de l'évolution du patient et améliorer sa thérapie. Pour les personnes âgées, leur cadre de vie est amélioré en leur laissant la possibilité de continuer à vivre dans leur habitation. Enfin, le nombre de trajets du patient au médecin est réduit, épargnant ainsi du temps. Cependant, les systèmes de télémonitorage ne sont qu'au début de leur développement. Ils doivent faire face à un coût de conception et d'utilisation élevé. De plus, les capteurs utilisés doivent être robustes au bruit et l'acquisition multisource pour la fusion de données est limitée par le nombre de capteurs réellement utilisables de manière journalière sans gêne pour le patient. Pour que le télémonitorage s'implante dans les pratiques, des avancées dans le pronostic médical doivent être faites afin d'intervenir sur les lieux de crise au plus tôt. En outre, l'analyse des signaux et le diagnostic médical doivent permettre de discriminer les situations pathologiques du bruit ambiant très important dans les systèmes embarqués mobiles. Le développement du télémonitorage dépend donc directement des avancées faites dans le domaine des SMI.

1.3.4 Systèmes de monitorage intelligent (SMI)

La fonction de monitorage fait appel à de nombreuses capacités de traitement et d'analyse. La conception des SMI se base sur le découpage de cette fonction en plusieurs étapes successives similaires à celles effectuées par les praticiens. Dans (Uckun, 1993), cinq tâches sont définies pour un SMI :

1. collecte des données en temps réel,
2. diagnostic de la situation observée,

3. prédiction de l'évolution du système surveillé,
4. construction d'un plan d'actions avec réaction rapide dans les cas les plus urgents,
5. pour les systèmes bouclés, exécution du plan d'actions.

Ce sont les étapes primaires et essentielles du monitorage intelligent, chacune peut être décomposée ou raffinée selon les méthodes de monitorage.

La conception d'un SMI implique aussi différentes méthodes de raisonnement successives. Ainsi, dans (Dojat et Sayettat, 1996), les auteurs mettent au point un modèle de raisonnement imitant celui du praticien hospitalier. Cette décomposition du raisonnement du praticien montre bien les trois types de raisonnement auxquels un système de surveillance fait appel :

– le diagnostic,
– la planification de la thérapie à adopter,
– le monitorage du patient et du résultat de la thérapie

Dans notre vision des SMI nous adoptons une décomposition en quatre étapes représentées par la figure 1.30 :

1. acquisition des signes vitaux du patient tels que l'électrocardiogramme, la pression artérielle, les tests de laboratoire,
2. extraction des caractéristiques essentielles de ces signes vitaux,
3. élaboration d'un diagnostic médical à partir des caractéristiques extraites,
4. recommandation d'une thérapie et application au patient.

Les effets de la thérapie adoptée se reflètent dans les signes vitaux et permettent de changer de thérapie si celle-ci se révèle inefficace.

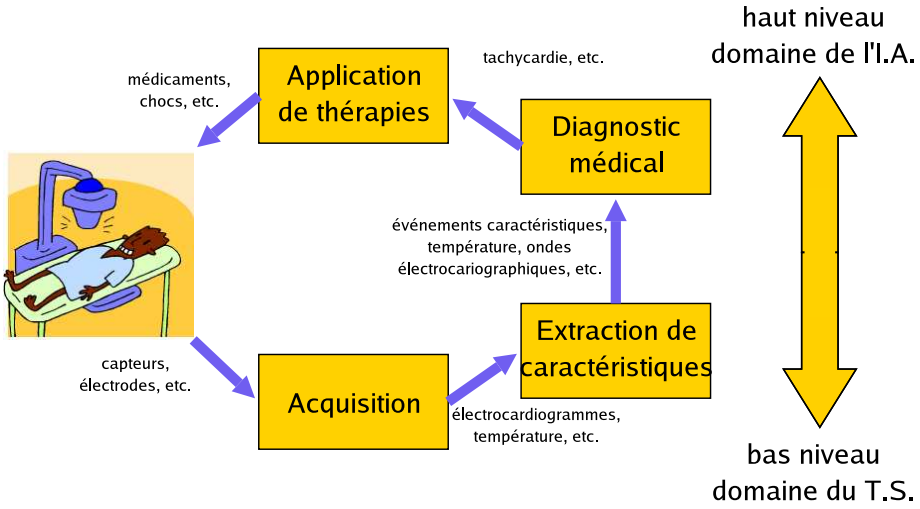


FIG. 1.30 – Les quatre étapes classiques du monitorage médical.

Dans la plupart des SMI tels que GUARDIAN (Larsson et Hayes-Roth, 1998) ou NÉOGANESH (Dojat et coll., 1997), une séparation peut être faite entre :

- les étapes de bas niveau, dédiées à l'acquisition et l'analyse des signaux physiologiques, qui transforment les signaux acquis en observations caractéristiques ;
- les étapes de haut niveau qui infèrent un diagnostic et proposent une thérapie à partir des observations de la partie bas niveau et d'une base de connaissances.

Les étapes haut niveau font intervenir des techniques de l'intelligence artificielle tandis que l'acquisition et l'analyse des signaux font intervenir des outils de traitement du signal. Cependant la frontière entre les deux niveaux tend à disparaître car de plus en plus d'algorithmes de traitement du signal utilisent de la connaissance notamment pour la fusion de données (Thoraval et coll., 1997; Hernández et coll., 1999).

La plupart de ces systèmes fonctionnent en circuit ouvert. C'est-à-dire que les thérapies proposées sont appliquées par le personnel hospitalier selon leur validité. D'autres systèmes fonctionnent en circuit fermé et appliquent directement les thérapies (injection de médicaments par exemple). C'est le cas du système d'assistance respiratoire NéoGanesh (Dojat et coll., 1997) qui déduit le niveau de récupération physique du patient pour faire varier le degré d'assistance de la ventilation. C'est, en outre, un des rares systèmes expérimentaux à avoir été transféré en industrie. En effet, très peu de recherches pourtant de qualité, ont abouti vers un transfert industriel. Ceci est principalement dû à une focalisation de la recherche sur certains des aspects du monitorage alors que tous doivent être pris en compte pour obtenir un système utilisable. Pour la conception des SMI on peut considérer trois niveaux de recherche.

1. Au niveau de l'architecture du système :
   - concevoir un système modulaire avec une architecture extensible et et des composants réutilisables,
   - fonctionner en boucle ouverte ou en boucle fermée,
   - fonctionner en temps réel,
   - fonctionner dans un environnement à ressources limitées (système embarqué).
2. Au niveau de l'analyse des données :
   - limiter le nombre de fausses alarmes,
   - traiter des données multidimensionnelles, temporelles, incomplètes, erronées et hétérogènes,
   - sélectionner les données d'intérêt en fonction du contexte médical et de la validité des données,
   - visualiser les données et le diagnostic d'une manière compacte et complète.
3. Au niveau du raisonnement :
   - gérer des connaissances pratiques et théoriques,
   - raisonner à partir de données qualitatives et quantitatives, (signaux, âge, etc.) continues ou événementielles (signaux, résultats de laboratoire, etc.)
   - effectuer la tâche de monitorage même si les connaissances sont incomplètes,
   - utiliser toutes les informations disponibles,
   - proposer un diagnostic médical à partir de données incertaines et hétérogènes avec des contraintes temps réel (être robuste au bruit),
   - suivre l'évolution du patient sur le long et le court terme,
   - proposer des thérapies pour améliorer l'état du patient.

La conception d'un SMI implique donc des connaissances variées dans différents champs scientifiques en traitement du signal, systèmes temps réel, intelligence artificielle, automatique, génie logiciel et électronique. Qui plus est, la tâche de monitorage doit pouvoir s'adapter aux patients, aux pathologies et au milieu clinique hostile. Les systèmes doivent donc être les plus adaptatifs possible tant au niveau du diagnostic qu'au niveau du traitement des données. Ces niveaux d'adaptation font l'objet des sections suivantes mais tout d'abord, les constituants des SMI sont détaillés. Les méthodes d'extraction de caractéristiques sont présentées en section 1.3.4.1. L'accent est aussi porté sur le filtrage des artefacts, en section 1.3.4.2, et la gestion des alarmes, en section 1.3.4.3, qui sont deux aspects importants pour atteindre un système robuste au bruit et riche en informations pertinentes. Pour rendre les systèmes intelligents, il est nécessaire qu'ils puissent gérer de la connaissance. Les différents types de connaissance employés dans les SMI sont détaillés en section 1.3.4.4. L'acquisition de la connaissance, qui est une étape difficile et importante dans la réalisation d'un système intelligent, est présentée en section 1.3.4.5. Enfin, le raisonnement temporel, qui permet le diagnostic médical, la prédiction et la proposition de thérapie, est présenté en section 1.3.4.6.

1.3.4.1 Extraction de caractéristiques

La première étape du monitorage est d'extraire les caractéristiques sur lesquelles raisonner à partir des signaux physiologiques. Par exemple, le rythme cardiaque est obtenu à partir des signaux électrocardiographiques. Cette extraction est essentielle dans le sens où c'est elle qui va *décrire* les signaux observés dans un langage qui permet d'effectuer un raisonnement. Dans cette étape, plusieurs niveaux de traitement peuvent se compléter. On trouve principalement la détection d'événements caractéristiques à partir des signaux (tel que l'électrocardiogramme ou électroencéphalogramme), le franchissement de seuils de paramètres physiologiques, et le calcul de tendances à partir des signaux (telles que l'augmentation du rythme cardiaque ou de la pression systolique).

L'extraction des événements caractéristiques des signaux consiste à détecter les éléments qui les composent. De l'électrocardiogramme on peut chercher à détecter et classer les différentes ondes qui le composent ainsi que les intervalles temporels entre les ondes. Les techniques appliquées couvrent un large champ du traitement du signal et intègrent de mécanismes de l'intelligence artificielle. Elles vont de la transformation non-linéaire (Pan et Tompkins, 1985) aux réseaux de neurones (Silipo et Marchesi, 1998) en passant par l'analyse fréquentielle par bancs de filtres (Afonso et coll., 1999) ou par ondelettes (Kadambe et coll., 1999), par des procédés stochastiques, de la logique floue ou encore du *pattern matching*. Ces traitements fournissent une description des signaux qui peut être directement utilisée pour le raisonnement médical.

Une autre méthode consiste à décrire les paramètres mesurés par des tendances. L'extraction de tendances s'effectue généralement sur des paramètres mesurés régulièrement qui décrivent l'état du patient tels que la mesure journalière de la température d'un patient ou la mesure toutes les secondes d'un rythme cardiaque dans le cadre d'une anesthésie. La technique consiste à découper le signal en périodes et à calculer un segment de droite qui se rapproche le plus possible du signal. Les paramètres observés

peuvent ainsi être décrits en terme : d'état stable, d'état instable, d'évolution croissante ou d'évolution décroissante durant les intervalles de temps considérés.

La description des paramètres par des tendances nécessite des templates de tendances pour la classification des états et doit être considérée d'une manière relative : une décroissance d'un paramètre peut aussi bien être pathologique que saine (Miksch et coll., 1996). Les auteurs de (Morik et coll., 2000) utilisent ainsi des tendances qui ont une signification selon le contexte et non d'une manière absolue. Par ailleurs, le choix de la taille de la fenêtre temporelle de calcul est toujours délicat. Dans (Calvelo et coll., 2000), les auteurs proposent une méthode d'extraction de tendances à partir de données brutes temporelles utilisées dans le projet AIDDIAG (Vilhelm et coll., 1996). Cette méthode de pré-traitement permet d'exprimer les données sous forme symbolique pour l'apprentissage et la reconnaissance en ligne. De plus, elle définit automatiquement la fenêtre de calcul de manière à ce qu'elle soit ni trop courte ni trop longue dans une représentation sous forme de filtre linéaire simplifiant le calcul en ligne.

À partir de cette méthode, les auteurs de (Sharshar et coll., 2005) proposent une nouvelle approche d'abstraction des données multisources. Grâce à une analyse en composantes principales, la contribution de chaque signal selon différents intervalles de temps est classée en fonction de sa tendance. À chaque tendance est attribuée une lettre. Ces lettres forment ensuite des mots puis des séquences de mots. Ces séquences peuvent ensuite être utilisées par un système d'apprentissage tel que *Think !* (Vilhelm, 1998; Vilhelm et coll., 2000) pour apprendre les séquences ayant une signification clinique.

La plupart des méthodes proposées sont utilisées sur des mesures directes des symptômes (pression artérielle, rythme cardiaque, etc.). Le calcul de tendances sur les signaux d'ECG ou de pression artérielle se révèle plus délicat car il est très difficile d'exprimer des ondes sous forme de segments sans retirer toute l'information qu'elles contiennent.

1.3.4.2 Filtrage des artefacts

Le milieu clinique est très hostile aux systèmes embarqués. Les interférences, la manipulation du matériel, les mouvements du patient, provoquent du bruit sur les enregistrements et perturbent très fortement les algorithmes de traitement du signal chargés de l'extraction de caractéristiques. Les observations fournies par l'extraction de caractéristiques sont alors erronées et les diagnostics proposés peuvent devenir incohérents. L'amélioration de la robustesse du diagnostic médical au bruit est donc un enjeu majeur.

Pour limiter les effets des artefacts sur le diagnostic médical, il existe deux méthodes souvent complémentaires. L'une consiste à séparer les sources physiologiques des sources de bruit par filtrage numérique pour améliorer la qualité des signaux, et donc de l'extraction de caractéristiques. Mais la séparation entre les sources utiles et le bruit devient problématique lorsque le bruit est trop important ou trop difficile à distinguer des signaux physiologiques. C'est pourquoi, lorsqu'on se trouve dans un cas multivoie redondant, il est préférable de sélectionner les voies non contaminées par le bruit (Dawant et coll., 1993; Vila et coll., 1997).

Une approche complémentaire se situe au niveau du diagnostic médical. Dans le pro-

jet PATRICIA (Moret-Bonillo et coll., 1993), les règles de détection d'artefacts consistent à vérifier les relations causales entre les informations pour éliminer les contradictions dans le raisonnement. De plus, des approches utilisant le raisonnement temporel peuvent réduire le nombre de diagnostics aberrants. Par exemple, dans NÉOGANESH (Dojat et Sayettat, 1996) un mécanisme d'*aggregation* et de *forgetting* est utilisé pour vérifier la cohérence du diagnostic temporel. Si, pendant une période donnée, plusieurs diagnostics différents sont proposés, le raisonnement temporel agrégera les diagnostics les plus vraisemblables et oubliera les diagnostics ponctuels qui peuvent être provoqués par du bruit transitoire. Cette technique permet de réduire les effets transitoires des informations erronées sur le diagnostic.

Dans ces systèmes, le problème de la réduction des effets du bruit est vu comme étant uniquement du domaine du traitement du signal ou uniquement du domaine de l'intelligence artificielle et peu de recherches ont tiré parti du diagnostic médical et de l'analyse du bruit de ligne. Pourtant ces informations sont utiles pour adapter les algorithmes de traitement du signal. Cette adaptation pourrait être effectuée par un système piloté qui sélectionne les algorithmes les plus adaptés à une situation donnée.

1.3.4.3 Gestion des alarmes

Quelque soit le domaine d'application, l'objectif principal d'un système de monitorage est de générer des alarmes prévenant, en industrie, l'opérateur d'une panne du système surveillé et, en clinique, le personnel médical d'une aggravation de l'état du patient. Dans tous les cas, le nombre d'alarmes peut devenir important et l'objectif est alors de réduire la surcharge cognitive en intégrant des priorités dans les alarmes et en filtrant les redondances (Cauvin et coll., 1998).

En monitorage de patient, les fonctions physiologiques sont monitorées à travers les paramètres physiologiques mesurés qui reflètent plus ou moins ces fonctions. Par exemple, le rythme cardiaque est souvent mesuré à l'aide de l'électrocardiogramme. Le monitorage contribue à maintenir les paramètres physiologiques dans des mesures « normales ». Cependant, ces paramètres peuvent varier avec de grandes amplitudes tout en restant « normaux » d'un point de vue médical, alors que le système de monitorage déclenchera une alarme à chaque franchissement de seuil (Chambrin, 2001). C'est notamment le cas des systèmes industriels car les constructeurs utilisent des seuils bas pour rendre leur systèmes plus sûrs et insistent plutôt sur l'attention à porter sur le branchement du matériel et sur les manipulations du personnel hospitalier (Agilent Technologies, 2000). Le nombre d'alarmes intempestives sans signification médicale est l'une des grandes limitations des systèmes de monitorage. Plusieurs études ont été entreprises pour mesurer les effets de ces fausses alarmes sur l'environnement clinique (Lawless, 1994; Cropp et coll., 1994; Aaron et coll., 1996; Tsien et Falcker, 1997; Biot et coll., 2003). Dans (Tsien et Falcker, 1997), les auteurs ont examiné la pertinence de 2942 alarmes durant 298 heures de monitorage et ont montré que seuls 8% des alarmes ont une signification clinique.

Les conséquences sur les soins apportés aux patients sont multiples. Le bruit causé par les alarmes empêche le repos des patients et augmente le stress du personnel (Aaron

et coll., 1996). Par ailleurs, l'étude réalisée dans (Soulas, 2001) montre que le nombre de fausses alarmes réduit la confiance du personnel médical envers le matériel et amène souvent le personnel à modifier les paramètres sans faire appel aux médecins voire même à désactiver certaines alarmes. Dans (Cropp et coll., 1994) et (Biot et coll., 2003), les auteurs montrent que seuls 50% des alarmes sonores critiques sont reconnues par le personnel hospitalier et que leur activation systématique perturbe plus la qualité des soins qu'elle ne l'améliore. Pour réduire le nombre de fausses alarmes, Chambrin (Chambrin, 2001) propose deux voies.

La première voie, qualifiée d'*organisationnelle*, met en exergue le fait qu'il n'y pas de priorité qui accompagne la génération d'alarmes. Les sons produits ou les couleurs affichées ne permettent pas de différencier les degrés d'urgence des alarmes (Cropp et coll., 1994). L'utilisation de standard pour définir les différents niveaux d'urgence serait une solution. Ainsi l'algorithme ST/AR de la société Agilent Technologies (Agilent Technologies, 2000) propose une classification des arythmies cardiaques en fonction de leur urgence (alarmes rouges et alarmes jaunes). Les auteurs du SMI en anesthésie, SENTINEL (Lowe et coll., 2001) utilisent la théorie des croyances pour générer des alarmes ou des avertissements selon l'appartenance des *possibilités* et *croyances* à un ensemble flou. Une interprétation est possible lorsqu'aucune contradiction n'existe alors qu'une croyance est générée par les éléments qui la confirment. Lorsque le système émet une alarme (ou un avertissement), les tendances des signaux qui l'ont provoquée sont affichées en couleur selon leur degré d'appartenance à un ensemble flou de templates de tendances. Cette méthode permet une génération plus spécifique des alarmes. De plus, la transparence du système (par l'affichage des diagnostics avec les faits qui les ont impliqués et leur degré de confiance) augmente la confiance du praticien envers le système.

La deuxième voie, qualifiée de *technique*, insiste surtout sur la réduction des effets des artefacts sur la génération d'alarmes. Le milieu hospitalier étant très bruité (mouvement du patient, manipulation du personnel, etc.), nombre de fausses alarmes sont dues aux artefacts. Hormis la résolution classique par filtrage, le concept d'alarme intelligente est né pour faire face à ce problème. La gestion intelligente des alarmes a pour but de répondre aux problèmes classiques : rigidité des seuils, fonctionnement en tout ou rien, déclenchement de plusieurs alarmes pour une même cause, etc. Le schéma de construction d'alarmes intelligentes tel que donné dans (Mora et coll., 1993) est représenté Figure 1.31.

À partir des signaux acquis, de leur traitement et des données patient, un contexte est déterminé. Ce contexte permet de choisir en ligne les alarmes à activer et leurs paramètres (degré d'urgence, etc.). Chaque signal acquis est testé pour vérifier sa validité (câble déconnecté, etc.) puis les caractéristiques sont extraites et l'analyse de tendances permet de supprimer les redondances. À partir de ces informations et d'une base de connaissances sur le patient, des alarmes intelligentes peuvent être générées qui donnent non seulement la cause des alarmes mais aussi des suggestions sur la thérapie à appliquer.

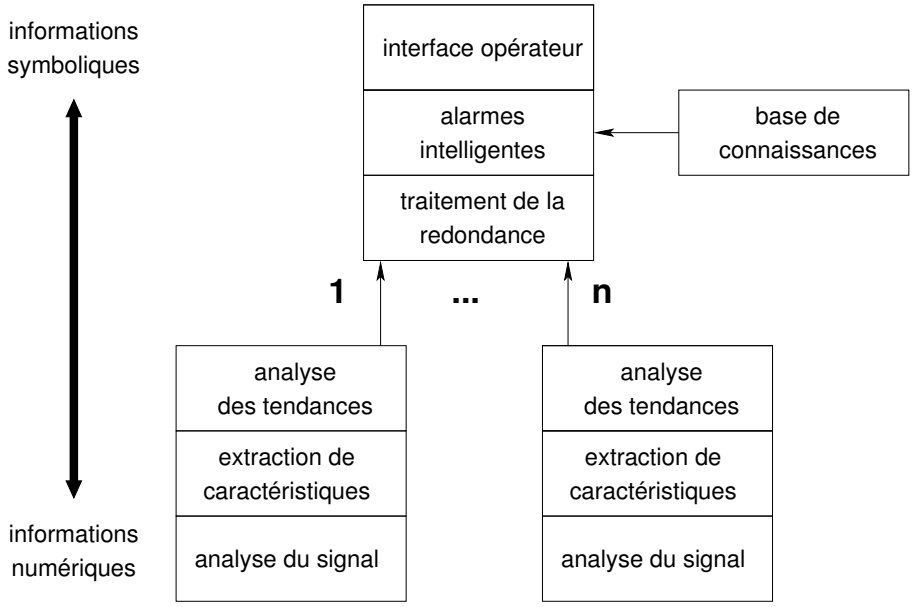


FIG. 1.31 – Intégration d'alarmes intelligentes.

1.3.4.4 Représentation de la connaissance

Le médecin diagnostique une maladie grâce à ses connaissances médicales. Un SMI procède de la même manière. Un système est intelligent lorsqu'il est amené à manipuler des connaissances. Ainsi, en monitorage médical, l'interprétation des signaux physiologiques est réalisée par un système à base de connaissances. La connaissance médicale est souvent classée en deux catégories : la connaissance de *surface* (ou *superficielle*) et la connaissance *profonde* (ou *fondamentale*).

Connaissance de surface On regroupe sous le terme de connaissance superficielle l'ensemble des facteurs analysables par le médecin (température, palpitation, antécédents, tracé électrocardiographique, etc.). La représentation de la connaissance de surface permet de lier directement les observations aux conclusions. C'est ce type de connaissance qui permet de passer des symptômes au diagnostic. Cette méthode d'inférence est similaire à celle d'un expert qui contrairement au novice, n'a pas besoin de construire un raisonnement pour arriver à la conclusion mais reconnaît directement la pathologie à partir des observations cliniques. L'avantage de ce type de connaissance est qu'il permet aux experts d'introduire directement leurs heuristiques dans une base de règles expertes, qui reste donc relativement simple à construire. Cela dit, la formulation des règles expertes dans les formalismes des systèmes n'est pas toujours évidente. De plus, la connaissance de surface ne permet pas d'expliquer les causes de la pathologie et reste inopérante lorsque les cas traités sortent de l'expertise. Cependant, pour le monitorage de patient où les pathologies traitées sont bien connues, ce type de connaissance

permet d'inférer rapidement un diagnostic en respectant les contraintes temps réel.

La technique d'exploitation de la connaissance de surface la plus répandue est, sans conteste, depuis MYCIN (Shortliffe, 1976), le système expert (Figure 1.32).

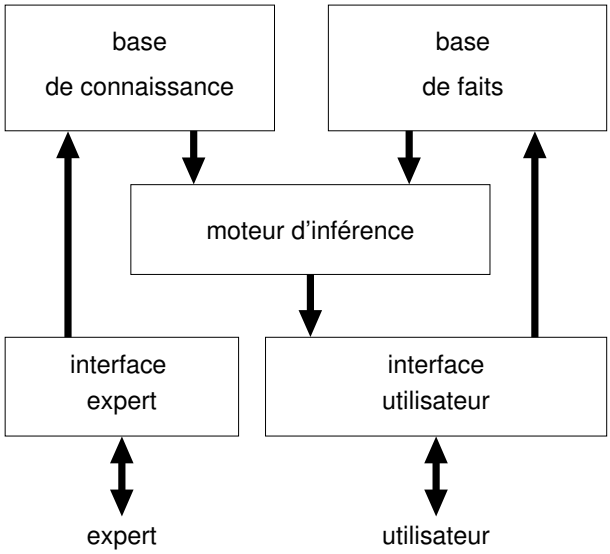


FIG. 1.32 – Système expert.

À partir d'une base de faits qui décrit le monde et d'une base de connaissances qui donne les règles qui régissent le monde, un moteur d'inférence met en relation les deux bases pour déduire de nouveaux faits. Le système expert est généralement très rapide car il sélectionne plus qu'il ne construit ses solutions. La plupart des systèmes de monitorage actuels utilisent des systèmes experts et des règles de surface sous des formes plus ou moins diversifiées.

Connaissance profonde La connaissance profonde est celle qui permet d'expliquer les phénomènes superficiels observables en retrouvant l'origine profonde des phénomènes. Elle consiste à employer des modèles des processus physiologiques. Cette modélisation peut être de type quantitatif ou qualitatif avec KARDIO (Bratko et coll., 1989), CARDIOLAB (Le Moulec, 1991), les modèles de TICKER (Hunter et Kirby, 1995), HOLMES (Guertin, 1996) ou encore une composition des deux (Hernández, 2000). Ces modèles sont développés pour étudier le fonctionnement d'un organe et sont très utilisés pour la simulation et la pédagogie. Par exemple, KARDIO est composé d'un modèle profond qui permet de simuler un ensemble exhaustif de connaissances de surfaces utilisables pour l'apprentissage de règles expertes. Les modèles profonds sont encore peu employés en monitorage mais semblent très prometteurs. Par exemple, les auteurs de CARMEN (Hernández et Carrault, 2004) proposent un modèle cardiaque semi-quantitatif, évolution du modèle qualitatif CARDIOLAB (Le Moulec, 1991; Siregar et coll., 1995), pour l'interprétation des battements cardiaques. Une fonction estime le

coût entre l'ECG analysé et le signal simulé par le modèle jusqu'à ce que les paramètres du modèle, mis à jour par des algorithmes évolutionnaires, minimisent suffisamment le coût. Ces paramètres sont ensuite utilisés pour expliquer les phénomènes observés.

L'un des avantages de la connaissance profonde par rapport à la connaissance de surface réside dans sa bonne tenue aux cas nouveaux. En effet, en connaissance de surface, il est difficile de demander à un expert de mettre régulièrement la base de connaissances à jour alors qu'un modèle peut traiter des cas nouveaux non prévus au départ. Par exemple, le système Vie-Vent (Miksch et coll., 1996) basé sur des règles expertes n'est pas capable de traiter les cas non prévus et nécessite donc une amélioration et une extension des règles qui le composent. De plus, les modèles profonds sont aussi utiles pour générer des scénarios à partir desquels une connaissance de surface peut être extraite et mise en mémoire afin d'inférer plus rapidement un diagnostic. Dans (Carrault et coll., 1999), le modèle Cardiolab (Siregar et coll., 1995) est proposé pour créer une base de données utilisée ensuite pour extraire des règles de reconnaissance d'arythmies. Mais, si cette connaissance profonde est efficace sur des systèmes simples tels que la surveillance de moteur ou des systèmes de régulation biochimique, elle devient inopérante lorsque les facteurs pathologiques sont multiples ou lorsque le facteur psychologique entre en jeu (Le Beux et coll., 1994). De plus, la complexité des modèles utilisés les rend, le plus souvent, difficiles à mettre en œuvre en temps réel. C'est pourquoi, même si les avancées effectuées dans la modélisation des organes sont prometteuses, la connaissance superficielle est généralement plus utilisée en monitorage médical, notamment par les systèmes experts. De plus, l'apparition des techniques d'apprentissage artificiel à partir de gros volumes de données rend la mise à jour de ces bases de connaissances des systèmes experts plus facile (Horn, 2001).

1.3.4.5 Acquisition de la connaissance

L'acquisition de la connaissance est un problème majeur des SMI. Outre le fait que l'acquisition d'expertise nécessite la disponibilité des experts pour mettre les bases de connaissances à jour, le langage dans lequel les connaissances sont exprimées est aussi problématique. Le langage doit être suffisamment descriptif, bien défini sémantiquement et consistant pour représenter l'ensemble des interactions mises en jeu dans le processus surveillé. Ce langage doit aussi être simple et intuitif pour permettre aux experts de traduire leurs connaissance dans le langage de représentation choisi. De plus, les connaissances sont souvent hétérogènes et sont issues de raisonnements mettant en jeu plusieurs sources temporelles (électrocardiogrammes, pression artérielle, etc.) statiques (poids, âge, etc.) ou événementielles (résultats de test de laboratoire, etc.). La multitude des représentations utilisées dans la littérature implique généralement des connaissances utilisables uniquement par le système pour lequel elles sont destinées entraînant ainsi un manque de transfert dans d'autres formalismes. Cependant, l'apparition des bases de données et l'explosion de la quantité de données stockées dans les hôpitaux ont conduit les recherches à s'orienter vers l'acquisition automatique des connaissances (Horn, 2001) par des techniques d'apprentissage symbolique et de fouilles

de données.

Un des objectifs de la fouille de données est d'extraire d'une grande masse de données les motifs caractéristiques de phénomènes. Ces techniques permettent de faire ressortir les phénomènes fréquents ou inconnus mettant en exergue les ensembles d'observations méritant l'attention de l'expert pour l'interprétation. Même s'il est illusoire de se passer d'experts pour acquérir la connaissance, les techniques de fouille de données et d'apprentissage artificiel réduisent considérablement le besoin d'expertise et ont une plus grande capacité de mise à jour et d'adaptation au patient.

Par exemple, les auteurs de (Wang et coll., 2001) et (Quiniou et coll., 2001) utilisent la programmation logique inductive (PLI) pour créer une base de connaissances de surface sous forme de chroniques, afin de reconnaître des arythmies cardiaques en ligne à partir d'un ECG. De même, dans (Fromont et coll., 2005; Fromont, 2005), la PLI est utilisée pour apprendre les motifs caractéristiques d'arythmies cardiaques sur des données temporelles multisources. L'acquisition automatique de la connaissance qu'ils utilisent a deux avantages.

- Elle permet de pallier les difficultés qu'ont certains experts du domaine à exprimer les règles de décision.
- Elle permet de paramétrer la complexité selon que l'on cherche à obtenir des règles facilement lisibles par le spécialiste, des règles minimales pour des raisons d'efficacité lors du diagnostic ou encore des règles robustes privilégiant les éléments faciles à déceler dans l'ECG.

Un bon exemple de système utilisant des techniques d'apprentissage artificiel est le système illustré par la figure 1.33 pour l'aide à la décision dans le cadre du monitorage hémodynamique en unité de soins intensifs (Morik et coll., 2000).

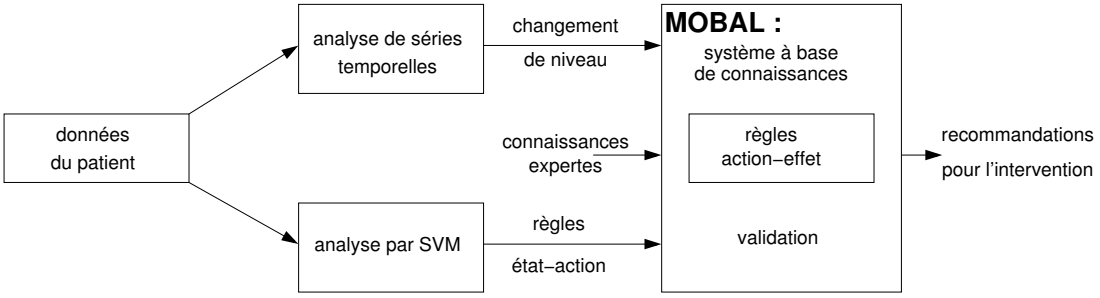


FIG. 1.33 – Architecture du système proposé par Morik et coll. (Morik et coll., 2000).

Les signes vitaux (signaux physiologiques), mesurés toutes les minutes, sont analysés en séries temporelles en utilisant la modélisation ARMA. Cette analyse permet de caractériser l'état du patient en terme de tendances et les artefacts. À partir de ces mêmes signes vitaux, l'analyse par SVM (*support vector machine*) permet d'instancier des propositions d'injection de médicaments qui peuvent améliorer l'état du patient. Les règles acquises par SVM ont été apprises à partir des données collectées sur le système d'information clinique du City Hospital de Dortmund (Allemagne). Les ten-

dances du patient et les injections proposées sont utilisées en ligne par un système à base de connaissances, mise en œuvre dans le système MOBAL, qui vérifie si les actions recommandées, en fonction des tendances, vont contribuer à mettre le patient dans un état stable. Les connaissances développées dans MOBAL sont acquises par expertise. Ce système montre bien que les méthodes d'apprentissage artificiel (ici SVM), même si elles simplifient l'acquisition de connaissances et contribuent à la spécialisation des traitements, nécessitent tout de même la mise en place d'un système de connaissances expertes pour fixer les règles générales inhérentes au domaine d'application (Horn, 2001).

Les techniques de fouille de données sont aussi souvent utilisées pour découvrir des relations inconnues entre les paramètres physiologiques au sein d'une même pathologie. Ainsi, les auteurs de (Duchêne et coll., 2005) proposent une méthode d'extraction de motifs temporels dans un ensemble de variables multidimensionnelles et hétérogènes correspondant aux données physiologiques d'un patient surveillé à domicile. Cette méthode permet d'extraire les motifs récurrents et caractéristiques correspondant à un comportement type du patient et ceux s'écartant de la normale.

Les techniques de fouille de données sont très utiles en milieu médical pour extraire des informations de sources pour lesquelles il existe peu de connaissances *a priori*. L'intérêt est donc double car il s'agit non seulement d'extraire les motifs directement utilisables pour une application médicale mais aussi d'enrichir la connaissance médicale avec les relations trouvées entre les différentes sources. De plus, même si l'intervention d'experts est nécessaire pour valider les connaissances acquises, ces techniques ont le mérite de simplifier grandement l'acquisition de connaissance. Ainsi, les auteurs de (Guyet et coll., 2005) proposent un système d'apprentissage multiagent assisté par un médecin. Trois types d'agents collaborent pour segmenter des signaux en différents intervalles temporels, classer les segments, apprendre des scénarios pour prévoir les états du patient en fonction des événements précédents. Le médecin fournit les annotations de départ pour l'apprentissage et intervient ensuite pour désambiguïser certaines classifications.

1.3.4.6 Raisonnement temporel

Les SMI appliqués à la médecine intègrent une partie de raisonnement temporel très importante. Dans l'activité de monitorage clinique, le médecin doit diagnostiquer l'état courant du patient, prévoir son évolution future et planifier les thérapies à adopter pour amener le patient dans un état voulu. Il doit adapter sa stratégie en fonction de l'historique du patient, le temps passé dans chaque état, etc. Ceci oblige à raisonner à plusieurs échelles temporelles et à construire son diagnostic régulièrement en fonction des nouvelles informations et des déductions faites à tous les niveaux d'abstraction. La figure 1.34, tirée de (Mora et coll., 1993), montre que la distribution des informations à travers le temps est très étendue.

Intégrer un mécanisme de raisonnement temporel (p. ex. systèmes experts temporels) dans un système de monitorage est donc un objectif crucial pour le diagnostic et l'aide à la décision. Si les premiers travaux effectués sur la logique temporelle par Allen

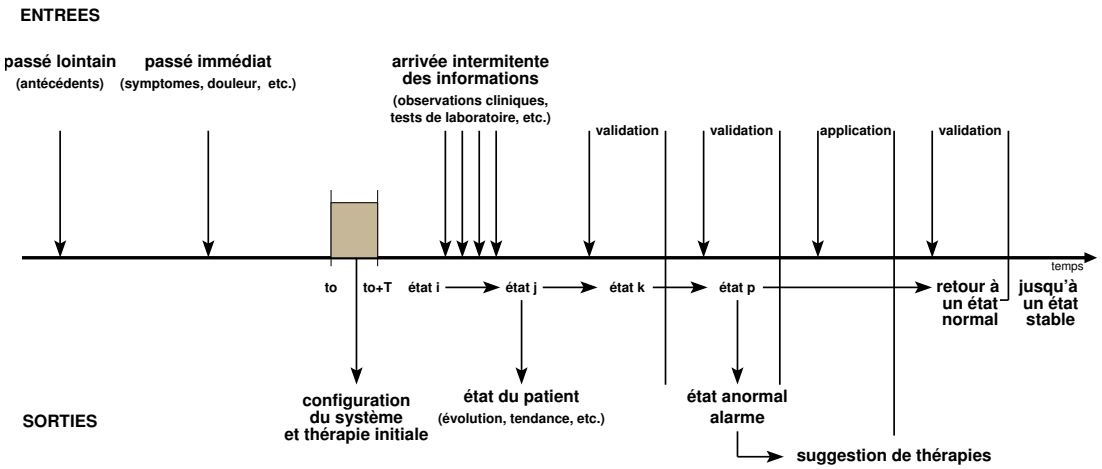


FIG. 1.34 – Distribution temporelle des informations et des actions d'un système de monitorage intelligent.

et McDermott formalisent les relations entre intervalles, le formalisme du raisonnement temporel classique souffre d'un manque de pouvoir d'expression qui ne peut permettre de décrire les situations complexes réelles (Dojat et Sayettat, 1996).

L'un des moyens de représenter les relations temporelles entre événements est d'utiliser les réseaux contraints temporellement (Dechter et coll., 1991). Par exemple, le formalisme des chroniques permet de représenter un ensemble d'événements contraints temporellement entre eux (Cordier et Dousson, 2000). Les contraintes sont exprimées sous forme d'intervalles et sont propagées au fil des occurrences d'événements. Dans (Carrault et coll., 2003), les chroniques sont utilisées pour reconnaître des arythmies cardiaques en ligne.

Le raisonnement temporel à base de modèles est aussi utilisé. Par exemple, le système HOLMES (Guertin, 1996) utilise un raisonnement abductif pour diagnostiquer des arythmies cardiaques très proche de celui utilisé dans l'ensemble de modèles TICKER (Hunter et Kirby, 1995). À partir des preuves fournies par l'extraction de caractéristiques du signal (c.-à-d. les ondes de l'ECG), HOLMES recherche l'ensemble des événements qui ont pu amener à ces preuves. Un modèle qualitatif du cœur permet de connaître les relations de cause à effet entre les différents constituants du cœur. Les événements sont les activations (dépolarisations) provoquées d'un constituant à un autre. Par exemple, l'événement `LBB->LV` traduit le fait que la dépolarisation de la branche gauche du faisceaux de His provoque la dépolarisation du ventricule gauche. Le système formule des hypothèses (liste d'événements qui ont pu se provoquer) et vérifie leur consistance. Pour éviter l'explosion combinatoire, des contraintes temporelles entre les événements permettent d'élaguer l'arbre de recherche et d'arriver ainsi plus rapidement à une explication (un diagnostic médical).

Dans (Gamper et Nejdl, 1997), les auteurs proposent un mécanisme de diagnostic temporel qui utilise des modèles de pathologies (ici l'hépatite B) sous forme de réseaux

de contraintes temporelles qualitatives. Les relations temporelles entre les différents états et symptômes de la maladie sont représentées par un graphe dont les nœuds sont les intervalles d'occurrence d'un état (p. ex. la période d'incubation) et les arcs les relations qualitatives entre les intervalles. Par exemple, la relation « l'intervalle I commence avant l'intervalle J et finit avant la fin de J » est représentée par (I avant J)∨(I rencontre J)∨(I recouvre J). Un mécanisme d'abstraction permet aussi d'agréger les observations consécutives d'un même phénomène (p. ex. deux événements informant l'incubation). Ainsi, les modèles sont indépendants du nombre réel d'observations. Cette représentation apporte un fort pouvoir d'expression mais les modèles restent assez complexes.

Le diagnostic médical demande parfois de raisonner à plusieurs échelles temporelles (p. ex. rapide en cas d'urgence, moyenne en phase de récupération, longue lors de suivis annuels). Les techniques classiques de raisonnement temporel ne sont pas assez expressives pour prendre en compte ces différents niveaux temporels. Par exemple, le *Cached Event Calculus* utilisé dans (Dojat et Chittaro, 1997) permet de déduire des états d'un patient à partir de données temporelles mais non de raisonner sur les états déduits pour retirer une tendance plus générale.

Pour raisonner sur plusieurs niveaux de granularité, les auteurs de (Shahar et Musen, 1993) proposent un raisonnement temporel, mise en œuvre dans le système RÉSUMÉ, basé sur trois mécanismes :

1. l'abstraction temporelle de points qui consiste à abstraire en une classe datée un ensemble de paramètres,
2. l'inférence temporelle qui permet de concaténer deux intervalles (ou points) qui se chevauchent en un nouvel intervalle,
3. l'interpolation temporelle qui relie deux intervalles espacés en un super intervalle.

Ces trois mécanismes sont définis d'une manière indépendante du domaine d'application mais leur fonctionnement effectif nécessite d'introduire de la connaissance sur le domaine. Ils permettent de raisonner à plusieurs niveaux de granularité en abstrayant des paramètres en points puis en intervalles puis en super intervalles afin d'obtenir une vision à différentes résolutions d'un phénomène (des symptômes au diagnostic par exemple). Le système TRENDX (Haimowitz et Kohane, 1996) analyse les données patient avec des templates de tendances. Ces templates consistent en un ensemble d'intervalles partiellement ordonnés dont les points finaux sont incertains. Les précisions relatives des différents templates sur les données permettent de discriminer les diagnostics possibles. Cependant, l'introduction d'incertitude rend le système très coûteux en calcul.

Un autre exemple de raisonnement à plusieurs niveaux de granularité est développé dans le système NÉOGANESH (voir Figure 1.35)

Les auteurs de (Dojat et Sayettat, 1996) mettent en œuvre un système de raisonnement intégrant des méta-connaissances, et des méta-règles afin d'instancier et produire de nouvelles règles. Par la suite, ces nouvelles règles permettent de produire de nouvelles informations pour le raisonnement temporel. Ils adoptent un modèle à événements-états, les évènements (par exemple un diagnostic) permettent l'instantiation d'un état (ou d'une alarme). Cet état est comparé aux états précédents par l'intermédiaire d'un

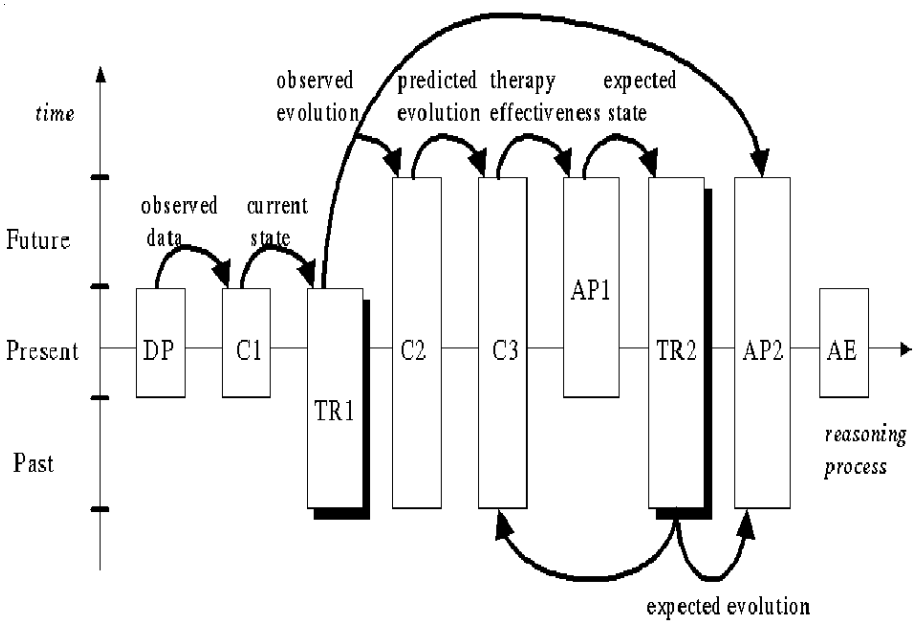


FIG. 1.35 – Modèle de raisonnement dans Dojat et Sayettat (Dojat et Sayettat, 1996). 1)-DP : Traitement du signal, acquisition des données, 2)-C1 : classification de l'état courant, 3)-TR1 : raisonnement temporel sur l'évolution observée, 4)-C2 : classification de l'évolution future, 5)-C3 : comparaison avec l'évolution espérée, 6)-AP1 : construction d'un plan d'action, 7)-TR2 : raisonnement temporel sur la cohérence de ce plan, 8)- AP2 : raffinement du premier plan, 9)-AE : Exécution du plan.

mécanisme d'*aggregation* de *forgetting* et de *break*. Ce mécanisme imite le raisonnement d'un praticien hospitalier qui reconnaît les états similaires d'un patient (aggregates), les interruptions (break) et oublie les événements non-importants (forgetting) volontairement en cas d'incohérence ou plus involontairement lorsqu'ils deviennent obsolètes. L'agrégation et l'oubli permettent de modifier dynamiquement la taille et la localisation de la fenêtre mettant en lumière les informations temporelles pertinentes. L'activation de ces deux mécanismes est dépendante du contexte et permet une interprétation de l'évolution globale du système pour la mise en place d'un plan de thérapie (action sur l'assistance ventillatoire). Ces mécanismes sont dépendants du domaine d'application dans le sens où ils font appel à deux prédicats *sameas* et *contradicts* qui ne peuvent être définis que de manière experte. De même la classification des états et l'évolution espérée requièrent des heuristiques entièrement dépendantes du domaine.

1.3.5 Vers des systèmes intelligents structurellement auto-adaptatifs

Les systèmes de monitorage médicaux sont développés pour des applications dans lesquelles il existe une très grande variabilité de patients observés. Le système doit être capable de raisonner avec un nombre d'informations qui peut être très variable (qui

dépend du matériel disponible de chaque hôpital) et souvent bruité ou incomplet. En effet, le milieu clinique est très hostile, le patient bouge, d'autres matériels peuvent provoquer des interférences, les électrodes peuvent se décoller, etc.

Toutes ces considérations impliquent que les systèmes de monitorage doivent pouvoir s'adapter aux différentes situations pour pouvoir assurer une qualité de surveillance optimale. Dans ce mémoire, différents types d'adaptation de la littérature sont abordés. Il s'agit de la prise en compte des spécificités du patient, la vérification de sources et l'adaptation aux contraintes temps réel et aux ressources de calcul. Cette section se termine par la présentation de quelques architectures de SMI et explique comment l'introduction du pilotage d'algorithmes permet une adaptation aisée.

1.3.5.1 Prise en compte du contexte patient

Contrairement à certains systèmes industriels qui raisonnent dans l'absolu, en médecine, la prise en compte du contexte patient permet aux tâches de décision et de contrôle d'adapter les traitements en fonction de l'évolution de l'état du patient ou de l'apparition d'un contexte spécifique. Plusieurs systèmes prennent en compte ce contexte.

- Simon (Dawant et coll., 1993) utilise le contexte de monitorage pour adapter le modèle de raisonnement au patient et pour la réduction et l'abstraction des données. Un autre but annoncé était de choisir les algorithmes et leurs paramètres en fonction du contexte de monitorage et des réserves en ressources.
- Patricia (Moret-Bonillo et coll., 1993) utilise un contexte naturel (données patient de départ) pour interpréter les valeurs numériques des variables monitorées, et un contexte inféré qui apporte des informations complémentaires nécessaires à l'établissement du diagnostic et de la thérapie.
- Le système de monitorage de patients diabétiques décrit dans (Bellazzi et coll., 1995) modélise les actions du médecin et du patient par deux modules de contrôle intelligent (voir définition dans (Bellazzi et coll., 1995) et références) représentés sous forme de circuits asservis. Le module de haut niveau, qui utilise des heuristiques et les paramètres monitorés, modifie la prescription médicale (période et quantité des doses d'insuline) et le module de bas niveau, basé logique floue et prédiction, adapte la prescription médicale pour l'injection d'insuline.
- Vie-Vent (Miksch et coll., 1996) utilise une définition du contexte pour sélectionner les données nécessaires et analyse les données acquises pour déterminer leur validité.
- Simbad (Soulas et coll., 1998; Le Certen et coll., 1998) utilisent des règles de sélection d'algorithmes de détection dérivées de (Friesen et coll., 1990). Simbad est constitué d'un analyseur de bruit de ligne qui permet de changer les paramètres des filtres et de commuter les détecteurs en fonction du bruit de ligne.
- Le modèle cardiaque Carmen (Hernández, 2000) modifie ses paramètres internes pour minimiser un coût entre le signal ECG analysé et le signal simulé par le modèle. Le modèle s'adapte donc au patient via le signal ECG.

Depuis les premières représentations de la connaissance par des règles absolues tirées de la littérature, les systèmes se sont spécialisés pour prendre en compte les variations intra et inter-patient qui, elles, sont des connaissances qui viennent de l'expérience des praticiens hospitaliers. Cette connaissance est plus dure à extraire car elle est beaucoup moins académique. Les chercheurs la représentent donc par des règles d'ajustement et d'adaptation du diagnostic (Dawant et coll., 1993; Moret-Bonillo et coll., 1993; Bellazzi et coll., 1995), de sélection de sources (Miksch et coll., 1996) ou de traitements (Soulas et coll., 1998) mais le plus facile à adapter reste le modèle profond. On voit bien la limite des systèmes à base de connaissances de surface qui résistent mal aux variations des patients observés. Cependant, un modèle tel que KARDIO (Bratko et coll., 1989) n'est pas adapté au monitorage en ligne dans le sens où son langage de description est trop précis pour les capacités actuelles de l'extraction de caractéristiques en ECG (Fromont et coll., 2003). Même si l'approche par modèles profonds est très prometteuse elle reste encore trop coûteuse en temps de calcul et les modèles profonds sont difficiles à modifier par rapport à la connaissance de surface. De plus, les règles utilisées en connaissance de surface sont plus interprétables que les paramètres des modèles.

1.3.5.2 Vérification et sélection de sources

Le nombre de variables d'entrée de la plupart des systèmes de monitorage peut être assez élevé. Ces données peuvent être redondantes, bruitées, voire inutiles dans certains contextes. La sélection et vérification de sources est une des phases essentielles d'un SMI (Mora et coll., 1993; Uckun, 1993). Cette sélection peut être faite selon trois principaux critères.

1. **La présence de bruit de ligne** implique d'utiliser uniquement les sources non bruitées pour effectuer un diagnostic. La sélection est surtout utilisée dans le cas de sources multiples et redondantes.
2. **L'identification du bruit de ligne** permet de filtrer au mieux la source du bruit courant de façon à la réparer, cette méthode est surtout appliquée au monosource.
3. **Le diagnostic médical en cours** permet de choisir les sources réellement nécessaires au diagnostic. Cela permet de réduire le nombre de sources à traiter ou d'augmenter la rapidité de calcul en cas d'urgence.

Plusieurs systèmes intègrent une phase de réduction ou de sélection de sources.

- SIMON (Dawant et coll., 1993) utilise un ensemble de modèles de fautes et d'artefacts qui permettent à son module d'abstraction temporelle de détecter quelles sont les sources contaminées afin de les extraire du traitement.
- VIE-VENT (Miksch et coll., 1996) intègre un module de sélection qui filtre les données nécessaires dans le contexte présent. Un module évalue ensuite la qualité des données en vérifiant si l'évolution des signaux reste dans un intervalle de valeurs (seuils adaptés). Sinon, les données sont « réparées » et ajustées lorsque cela est possible.
- SUTIL (Vila et coll., 1997) est un SMI des ischémies cardiaques qui détecte, sur trois voies d'ECG, différents types de bruit et sélectionne les voies les plus sûres.

Les informations extraites des voies sont ensuite fusionnées.
- GUARDIAN (Larsson et Hayes-Roth, 1998) intègre une phase de réduction qui d'une centaine de paramètres sélectionne ceux nécessaires au diagnostic.

D'une manière générale, les systèmes sélectionnent les sources soit d'un point de vue diagnostic soit d'un point de vue bruit de ligne. À ma connaissance, aucun système n'intègre les deux points de vue. Il y a pourtant un fort intérêt à concevoir un système capable de choisir les sources les plus importantes (ou sûres) et de raisonner sur cet ensemble restreint de sources en prenant en compte autant l'aspect vérification (bruit) que sélection (diagnostic).

1.3.5.3 Contraintes temps réel et limitation de ressources

Le monitorage en unité de soins intensifs requiert un fonctionnement temps réel sans faille. Le système ne peut se permettre de prévenir lorsqu'il est trop tard. Cependant, la charge de traitement à effectuer peut excéder les capacités de calcul qui peuvent être réduites dans un contexte de système embarqué. Dans de telles situations, le système doit adapter son raisonnement en fonction des ressources nécessaires pour exécuter les traitements les plus cruciaux. Par exemple, dans un contexte de monitorage cardiaque, en cas de surcharge calculatoire, le système doit pouvoir être capable de détecter un arrêt cardiaque ou une fibrillation et remettre à plus tard un diagnostic plus précis. Plusieurs systèmes sont capables de faire face à ces situations.
- GUARDIAN (Larsson et Hayes-Roth, 1998) s'appuie sur une architecture générique dédiée au monitorage de patient (Hayes-Roth et coll., 1995), chaque tâche du monitorage peut être effectuée par des méthodes choisies en fonction des ressources en calculs disponibles.
- Dans SIMON (Dawant et coll., 1993), le module d'acquisition sélectionne les sources à utiliser lorsque des contraintes de temps réel ne permettent pas l'acquisition de toutes les sources.
- NÉOGANESH (Dojat et coll., 1997) est un système multiagent qui utilise un modèle d'agent qui possède des capacités de réaction et des capacités cognitives qui lui permettent de donner une réponse rationnelle même lorsque le délai de réaction devient critique.

1.3.5.4 Architecture des systèmes de monitorage intelligent

La conception des SMI se base souvent sur une architecture objet ou multiagent (Sukuvaara et coll., 1993; Larsson et Hayes-Roth, 1998; Dojat et coll., 1997; Soulas, 2001; Mabry et coll., 2003; Mabry et coll., 2003) qui permet une bonne structuration des différents niveaux de traitement.

Les systèmes de monitorage de patient intègrent par nature différents niveaux de raisonnement, diverses méthodes de calcul et doivent posséder de bonnes capacités d'adaptation. Le nombre d'éléments qui communiquent entre eux nécessite une structure d'échange standardisée. C'est pourquoi de nombreux travaux ont été réalisés pour

standardiser l'architecture des systèmes de monitorage. Soulas (Soulas, 2001) propose une architecture de monitorage intelligent basée objet, reposant sur la définition d'une entité logicielle intelligente (voir Figure 1.36).

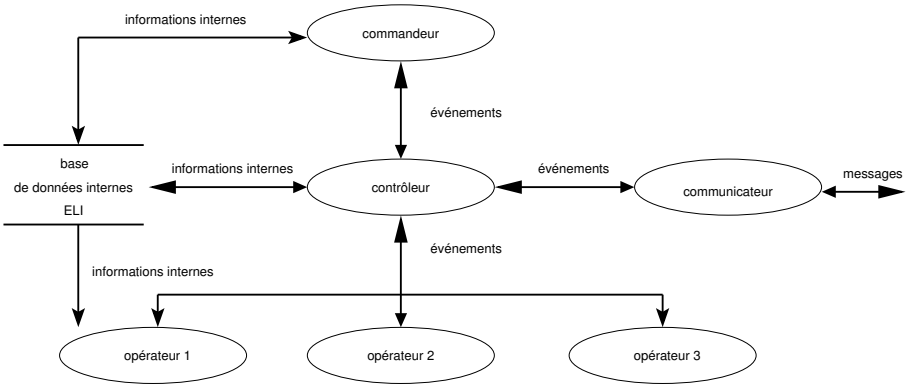


FIG. 1.36 – Diagramme de flots de données d'une entité logicielle intelligente.

Cette entité comporte un commandeur qui, à partir de la base de données interne, décide des actions à effectuer. Le contrôleur peut ensuite choisir les opérateurs à activer (algorithmes de traitement du signal, sauvegarde, etc.) et peut provoquer l'envoi de messages via le communicateur. Cette structure à été utilisée pour concevoir le système de monitorage SIMBAD (Le Certen et coll., 1998).

Dans (Hayes-Roth et coll., 1995), les auteurs proposent une architecture de logiciels spécifiques au domaine qui comprend :

- une architecture de référence qui décrit le fonctionnement général du système pour une application donnée,
- une bibliothèque de composants,
- une méthode de configuration d'application pour sélectionner et configurer les composants dans une architecture dédiée.

L'architecture de référence utilisée, illustrée par la figure 1.37, contient deux niveaux de raisonnement : le niveau physique qui interagit avec le monde extérieur et le niveau cognitif qui met en œuvre des raisonnements de plus haut niveau tels que l'estimation de la situation, le planning d'actions, etc.

Un comportement est choisi lorsque certains événements satisfont les conditions de déclenchement et qu'il est utile pour atteindre le but courant. L'exécution des différents comportements modifie la base d'information qui contient la connaissance sur le système, tel que les différents plans d'exécution possibles. Le méta-contrôleur choisit un comportement parmi ceux possibles et l'exécute avec les paramètres adaptés.

GUARDIAN repose sur une architecture de type « tableau noir » qui permet un échange centralisé des informations et une plus grande souplesse pour les modifications et futures évolutions. Le système NÉOGANESH proposé dans (Dojat et coll., 1997) utilise une architecture distribuée multiagent où chaque agent communique par message et possède ses propres mécanismes de raisonnement. La distribution de l'expertise médicale

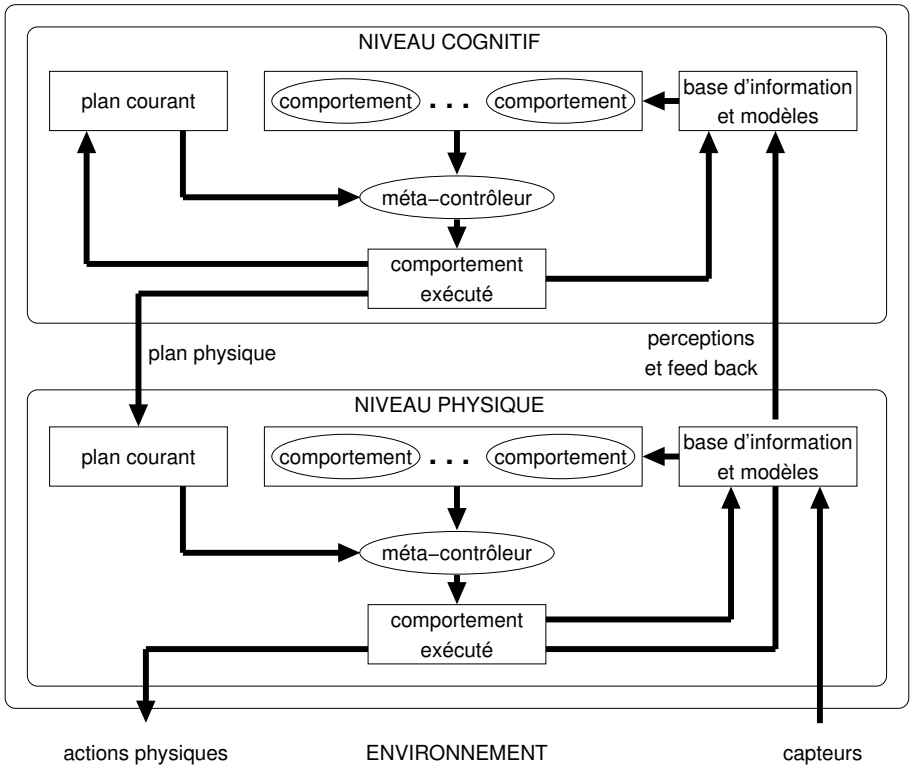


FIG. 1.37 – Architecture de référence d'un système adaptatif intelligent (Hayes-Roth et coll., 1995).

est illustrée par la figure 1.38.

Chaque agent raisonne à partir des informations qui lui sont envoyées par les autres. Par exemple, à partir des informations fournies par l'agent `DataProcessor`, l'agent `Classifier` déduit l'état courant du patient et communique cette information à l'agent `TemporalReasoner` qui raisonne à un niveau temporel supérieur. Toutes les informations sont envoyées à l'agent `ClinicianAgent` qui représente le médecin en charge du patient et qui peut agir sur le système. Cette structure peut s'adapter aux différentes situations, par exemple, lorsque l'agent `Classifier` détecte une situation alarmante, il envoie directement cette information à l'agent `ActionPlanner` pour qu'il puisse agir rapidement sur l'assistance respiratoire.

1.3.5.5 Amélioration de la robustesse du diagnostic par pilotage d'algorithmes

Tous les systèmes présentés ont montré à plus ou moins grande échelle des capacités d'adaptation au contexte. Par exemple, SIMBAD (Soulas et coll., 1998; Le Certen et coll., 1998) analyse le bruit de ligne pour sélectionner le meilleur algorithme capable d'effectuer une détection d'événements. SIMON (Dawant et coll., 1993) propose

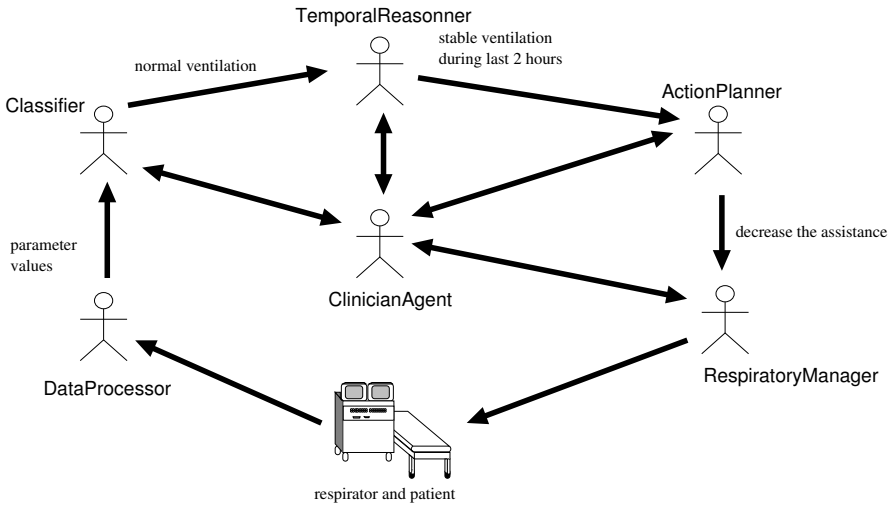


FIG. 1.38 – Distribution de l'expertise médicale entre les agents.

d'analyser le contexte pour choisir les algorithmes les plus adaptés aux ressources de calcul. Certains, tels que les auteurs du projet GUARDIAN (Hayes-Roth et coll., 1995), proposent une architecture dédiée capable de modifier en ligne le plan de traitement.

Cependant, toutes les architectures proposées développent leur propre cadre pour effectuer les adaptations et peu d'entre elles, proposent des adaptations à tous les niveaux de la chaîne de traitement, du traitement bas-niveau au raisonnement et diagnostic médical. Pour apporter plus de flexibilité et une meilleure structuration de l'architecture, il est intéressant de concevoir un système de monitorage comme un système de pilotage d'algorithmes.

Le pilotage d'algorithmes consiste à créer automatiquement une chaîne de traitement constituée d'algorithmes et à modifier cette chaîne au cours du temps pour réaliser une application donnée. La méthode utilisée consiste généralement à décomposer l'application en tâches primitives (Shekhar et coll., 1994; Karsai et Sztipanovits, 1999). Par exemple, une application de réception de signal hertzien peut se décomposer en tâches d'acquisition, de détection de porteuse, de démodulation et de filtrage. Pour chacune de ces tâches il existe plusieurs algorithmes différents (filtre RII ou RIF, démodulation par détection d'enveloppe ou multiplication de porteuse, etc.). Le pilote doit donc choisir l'algorithme le plus adapté et le paramétrer en fonction du contexte (bruit de ligne, indice de modulation, etc.).

La construction d'un tel système permet une bonne séparation entre les algorithmes et la connaissance nécessaire pour les piloter. La capacité de modification peut être exploitée pour sélectionner les sources valides, utiliser les algorithmes appropriés pour les traiter, extraire les informations pertinentes, sélectionner le niveau d'abstraction du raisonnement, etc. Par exemple, si un type de bruit est présent sur la ligne dans un contexte pathologique particulier, les algorithmes peuvent être modifiés pour répondre le mieux possible à la situation. L'ajout d'un pilote permet aussi de sélectionner les ob-

servations vraiment importantes pour le diagnostic. En effet, dans beaucoup de systèmes de monitorage, il existe des sources redondantes dont le traitement est très coûteux.

Le pilotage amène donc beaucoup de souplesse, des modifications aisées et un cadre de représentation des données et des traitements nécessaires à une bonne structuration de la connaissance.

1.4 Conclusion

Les différents systèmes de monitorage intelligent montrent que l'adaptabilité est une solution clé pour obtenir une reconnaissance des pathologies la plus fiable possible et diminuer le nombre de fausses alarmes. Ces avancées sont nécessaires pour que le monitorage de patient devienne un véritable support pour l'aide à la décision et qu'il ne soit pas une source de nuisance et de méfiance pour les patients et le personnel médical.

Pour atteindre ces objectifs, la conception des systèmes de monitorage demande beaucoup de connaissances et des mécanismes de raisonnement capables de les gérer. Le raisonnement doit se baser sur des informations fiables et doit être capable de raisonner avec des informations incomplètes pour permettre un diagnostic même en situation difficile. Les systèmes proposés permettent d'adapter leur chaîne de traitement et de raisonnement à différentes étapes du monitorage mais rarement à toutes. Cela en fait des systèmes dédiés, rarement généralisables, et dont l'architecture peut être très différente selon l'approche du problème. De plus, si l'introduction de connaissances dans les systèmes de monitorage intelligent a permis d'obtenir des diagnostics médicaux et de l'aide à la décision, peu d'efforts ont été accomplis pour utiliser cette connaissance afin d'optimiser la configuration interne du système.

L'approche par pilotage d'algorithmes semble être une réponse pertinente à ces défauts. Le pilotage d'algorithmes a pour but d'adapter une chaîne de traitement complète en fonction de la situation courante. Cette technique est donc prometteuse pour réaliser un système adaptatif intelligent. La conception même d'un tel système nécessite une bonne structuration de l'application et des connaissances, ce qui le rend plus facile à modifier et à comprendre. Ce pilotage présente un intérêt tant au niveau de l'optimisation de la chaîne de traitement qu'au niveau de l'économie de ressources. De plus, cette approche peut permettre d'autres types de pilotage tels que la sélection de voies en monitorage multisource ou encore la sélection des types de connaissances à utiliser (surface ou profonde). Par ailleurs, l'émergence du télémonitorage, va nécessiter des systèmes repartis capables de s'adapter à un grand nombre de situations. L'apport du pilotage d'algorithmes pourrait jouer un grand rôle en permettant une gestion efficace des ressources et des services.

Chapitre 2

Présentation du pilotage d'algorithmes et de CALICOT

2.1 Introduction

Le pilotage d'algorithmes de traitement du signal est une méthode prometteuse pour introduire une grande flexibilité et une adaptabilité paramétrique et structurelle dans les systèmes de monitorage médical. L'intégration d'un pilote dans un système doit permettre une modification dynamique de tous les niveaux de la chaîne de traitements. Ainsi, les opérations appliquées aux données d'entrée sont toujours les plus adaptées pour répondre à un objectif particulier (filtrage, détection, classification, etc.). La section 2.2 a pour objectif de définir le pilotage d'algorithmes et de présenter des exemples de systèmes de pilotage existants. Cet état de l'art permet de dégager les concepts de pilotage utilisés par la suite. Ce pilotage est destiné à être intégré dans le système de monitorage cardiaque CALICOT pour permettre l'adaptation dynamique de sa chaîne de traitements en fonction des données du patient. Les mécanismes de fonctionnement de CALICOT sont donc naturellement détaillés en section 2.3.

2.2 Pilotage d'algorithmes de traitement du signal

2.2.1 Introduction

Les systèmes à base d'algorithmes de traitement du signal sont largement utilisés dans les domaines de la communication, des nouvelles technologies, de la géologie, de la météorologie, de la robotique, de la médecine, etc. Pour certaines applications, le nombre de situations traitées peut devenir très important et obliger le concepteur du système à faire des choix concernant le type d'algorithme à utiliser pour qu'il traite le plus grand nombre de cas possible. Pour faire face au nombre de situations rencontrées, les algorithmes et systèmes adaptatifs ont vu le jour. Ces méthodes consistent à ajuster en ligne des paramètres en fonction de l'évolution des signaux pour les traiter au mieux. Cependant, ces méthodes ne permettent pas de modifier la chaîne de traitements en

cours pour changer d'algorithme. Une adaptivité plus haut niveau se situant au niveau des paramètres et des blocs de la chaîne de traitements (changement d'algorithmes) peut être apportée par un pilote d'algorithmes.

Le pilotage d'algorithmes consiste à manipuler un ensemble d'algorithmes et une base de connaissances de façon à constituer ou modifier une chaîne de traitements, en fonction de l'évolution de la situation courante, pour obtenir le meilleur traitement possible à un instant donné. Le pilote manipule une base d'algorithmes stockés sous une forme structurée. Cette structuration permet une plus grande réutilisation des algorithmes à l'heure où leur nombre croissant devient un handicap pour le non-spécialiste. Par exemple, des logiciels avec des bibliothèques générales de traitement du signal tels SCILAB[1] ou MATLAB[2] apportent une grande facilité d'utilisation.

Le pilote d'algorithmes introduit une grande flexibilité de l'architecture, certains traitements peuvent être désactivés en fonction des ressources de calcul, des traitements généraux peuvent être spécialisés pour en fonction des spécificités d'un signal, etc.

Avant de présenter le pilotage d'algorithmes, des généralités sur le traitement du signal sont rappelées en section 2.2.2. La section 2.2.3 détaille quelques systèmes de pilotage d'algorithmes de la littérature afin d'en extraire les concepts généraux en section 2.2.4. Enfin, les concepts retenus et utilisés par la suite dans notre approche sont présentés en section 2.2.5.

2.2.2 Architecture classique d'un système de traitement numérique du signal

Le traitement du signal a pour objectif d'extraire de l'information de signaux bruts provenant des sources à étudier. Les deux concepts fondamentaux de cette discipline sont : la notion de *signal* et de *traitement* (Shekhar et coll., 1994).

Un signal à une dimension est, dans le domaine numérique, une suite de nombres ordonnés. Un signal n'a pas de représentation figée. La représentation de l'évolution du signal au cours du temps est la plus connue mais d'autres représentations existent telles que la représentation dans l'espace fréquentiel ou temps-fréquence qui sont des projections permettant de mettre en valeur certaines caractéristiques du signal. Un signal peut être déterministe ou stochastique. Il peut être utile ou être un bruit selon qu'il transporte l'information. Le bruit, qui s'ajoute à tous signaux physiques, est généralement de type stochastique et brouille les signaux utiles.

Un traitement peut être vu comme une boîte noire prenant en entrée un signal et retournant une information ou un autre signal. La plupart des traitements utilisés sont linéaires et invariants. Dans le monde numérique, le calcul effectué à l'intérieur de la boîte est principalement réalisé par un algorithme (histogramme, filtrage, calcul de moyenne, etc.). Un algorithme de traitement du signal extrait de l'information d'un signal numérique (ou numérisé) ou le transforme de manière à rehausser la partie utile. Un algorithme peut être défini par son *but* (filtrer, compresser, détecter, etc.), par ses

[1] http ://www.rocq.inria.fr/scilab
[2] http ://www.mathworks.com

méthodes (compression par extrema d'ondelettes, par codage, etc.) et par ses *entrées-sorties* (p. ex. traitement d'une séquence d'images pour détecter un accident). À cela s'ajoutent les performances intrinsèques des algorithmes telles que la complexité, la qualité, la convergence, et des performances extrinsèques telles que la rapidité, la taille disque et mémoire, etc. Ces dernières performances sont très dépendantes du langage d'expression utilisé et de la machine sur laquelle l'algorithme s'exécute. Les traitements peuvent aussi être conçus de différentes manières selon les critères de qualité exigés qui rendent possible ou non leur exécution en temps réel. Enfin, les signaux peuvent être traités par flot (échantillon par échantillon) ou par bloc (segment par segment).

La conception d'un système de traitement du signal nécessite une connaissance précise de la succession de techniques à employer pour réaliser les différentes transformations et extractions qui amènent au but recherché. Cette succession est communément appelée la chaîne de traitements. L'exemple de la figure 2.1 représente une chaîne de traitements pour la réception d'un signal numérique.

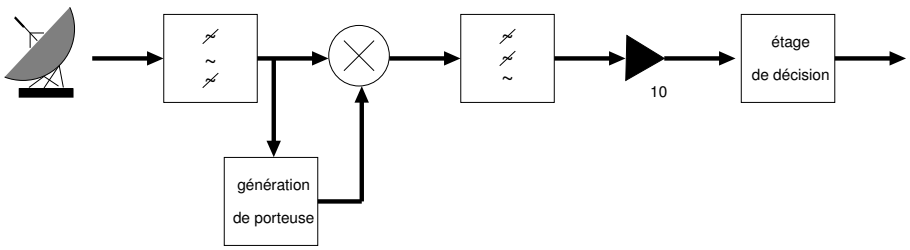


FIG. 2.1 – Chaîne de traitements pour la réception d'un signal numérique

Le signal est d'abord recueilli par une antenne puis séparé du bruit haute et basse fréquences. Le signal est ensuite démodulé par une multiplication de porteuse. Puis après un filtrage passe-bas et une multiplication par un gain, le signal est seuillé pour le transformer en information binaire.

La chaîne peut comporter un nombre très important de traitements avec un large ensemble de connexions avant et arrière. Les traitements impliqués peuvent être très simples ou très complexes selon les cas. Pour une application où les propriétés du signal sont bien définies, la conception s'oriente vers un système figé. Cependant, lorsque le signal n'est connu qu'approximativement, que les signaux ont des rapports signal-à-bruit (RSB) variables ou des comportements imprévus, les traitements à effectuer pour un même but peuvent être très différents. On peut alors choisir de mettre en place un système adaptatif qui fera varier des paramètres en fonction de certains critères (estimation du niveau de bruit par exemple). Cependant, certains signaux nécessitent des traitements radicalement différents au fil des situations rencontrées. Un système capable de modifier dynamiquement sa structure en utilisant différents types d'algorithmes pour un même but en fonction des situations rencontrées permettrait donc d'utiliser le traitement le plus adapté à la situation courante (Lesser et coll., 1993a). Ce besoin d'une adaptivité qui peut être non seulement perçue au niveau des paramètres, mais aussi au niveau de la chaîne de traitements, peut être réalisé en incluant un pilote

d'algorithmes au système ou, plus généralement, en développant l'application dans un environnement dédié, tel que la plate-forme LAMA (Moisan, 1998). Les plates-formes dédiées permettent des modifications aisées du système ce qui diminue les coûts des modifications sur des systèmes classiques (Karsai et Sztipanovits, 1999).

2.2.3 Systèmes de pilotage d'algorithmes de la littérature

2.2.3.1 Généralités

Le pilotage d'algorithmes fait partie du domaine du raisonnement qualitatif (supervision de systèmes). Il consiste à manipuler un ensemble d'algorithmes et une base de connaissances de façon à réaliser une chaîne de traitements de manière optimale. Si quelques recherches ont eu lieu dans le domaine du traitement du signal avec le développement de IPUS (Lesser et coll., 1995; Klassner et coll., 1998), CADDMAS (Sztipanovits et coll., 1998) basé sur MGA (Sztipanovits et coll., 1995; Karsai et Sztipanovits, 1999) et les recherches entreprises dans (Shekhar et coll., 1994), c'est surtout en traitement d'images, domaine pour lequel plus d'utilisateurs inexpérimentés peuvent être attirés, que l'on trouve des systèmes de pilotage avec notamment EXTI (Dalle et Dejean, 1998), BORG (Clouard et coll., 1998), OCAPI (Clément, 1990; Clément et Thonnat, 1993) et ses descendants PEGASE (Moisan et coll., 1997), MEDIA (Crubézy, 1999). Les deux disciplines étant très proches (les images sont des signaux), les concepts et les réflexions présentés dans ce document s'appuient sur des travaux provenant des deux domaines.

La notion de pilotage d'algorithmes n'est pas encore clairement définie dans la littérature car elle reste encore très dépendante de la culture scientifique et du domaine d'application. Dans (Moisan et Ziébelin, 2000), les auteurs proposent une définition qui entre dans les travaux de l'équipe ORION[3] de l'INRIA Sophia-Antipolis spécialisée dans le pilotage de programmes en traitement d'images.

> *Le pilotage de programmes vise à l'utilisation optimale d'un ensemble de programmes préexistants, indépendamment d'un domaine applicatif particulier.*

Si cette définition ne fait pas l'unanimité, notamment en ce qui concerne l'indépendance du domaine d'application dans la conception (Clément et Thonnat, 1993; Shekhar et coll., 1994; Lesser et coll., 1993b; Clouard et coll., 1998; Karsai et Sztipanovits, 1999), elle représente bien les objectifs d'un système de pilotage d'algorithmes. Il faut cependant noter que parmi les travaux présentés dans ce document, seule l'équipe ORION emploie explicitement le terme de *pilotage*. Dans (Dalle et Dejean, 1998), on parle de planification de traitements, dans (Lesser et coll., 1995; Klassner et coll., 1998), d'architecture pour l'intégration du traitement et de l'interprétation du signal, dans (Sztipanovits et coll., 1998), de systèmes structurellement auto-adaptatifs. Il s'ensuit que les techniques employées varient beaucoup selon les auteurs. En effet, les représentations de la connaissance peuvent être différentes selon les techniques de planification et les contraintes de temps peuvent imposer un système complètement automatique

[3] http ://www-sop.inria.fr/orion/

(système autonome temps réel) (Lesser et coll., 1995; Karsai et Sztipanovits, 1999) ou permettre un système semi-automatique dans le cas d'une aide à la décision (Clément et Thonnat, 1993; Shekhar et coll., 1997; Shekhar et coll., 1999).

Pour illustrer les différentes applications et faire ressortir les concepts communs, quelques systèmes de la littérature sont présentés. Il s'agit principalement :

- du système OCAPI (Clément, 1990; Clément et Thonnat, 1993) et de ses successeurs,
- de l'architecture IPUS (Lesser et coll., 1993b),
- de l'architecture Multigraphe (Sztipanovits et coll., 1998; Karsai et Sztipanovits, 1999).

2.2.3.2 OCAPI

OCAPI (Outils de Contrôle Automatique de Procédures Images) (Clément, 1990) est un système de pilotage d'algorithmes de vision par ordinateur. Il permet le développement de systèmes intelligents et généraux de traitement d'images qui peuvent manipuler des algorithmes de bas niveau.

Représentation de la connaissance Dans OCAPI les algorithmes de traitement d'images sont appelés *opérateurs élémentaires*. Ils sont représentés sous forme d'objets où chaque objet est associé à un but particulier. Chaque objet contient sa syntaxe d'appel, le format de ses entrées-sorties et d'autres informations complémentaires. Il existe aussi des *opérateurs complexes* représentés par un plan d'exécution d'opérateurs élémentaires. À chaque opérateur sont associées des règles d'initialisation et des règles d'ajustement qui permettent de modifier les valeurs de ses paramètres.

Outre les opérateurs disponibles, la base de connaissances d'OCAPI contient une description des problèmes que le système sait résoudre. Cette description est représentée sous forme de *but* spécifique à un traitement donné.

Le raisonnement d'OCAPI repose sur une séparation nette entre la connaissance liée au domaine du traitement de l'image –appelée *domaine indépendant*– et la connaissance liée au domaine d'application –appelée *domaine dépendant*–. La connaissance indépendante est détenue par les opérateurs et leurs règles d'ajustement et d'initialisation tandis que la connaissance dépendante est stockée dans les buts qui contiennent des règles d'évaluation des résultats. Cette décomposition a pour but de mimer le raisonnement d'un expert du traitement de l'image et d'un expert du domaine d'application (l'utilisateur) lors de la conception d'un système. L'expert image (domaine indépendant) va choisir et paramétrer les algorithmes jusqu'à ce que l'expert du domaine d'application (domaine dépendant) juge que les résultats obtenus sont satisfaisants. Par exemple, pour extraire une bactérie d'une image, l'expert image choisira les algorithmes de segmentation et les adaptera tant que l'expert biologiste ne jugera pas la qualité de segmentation suffisante.

Un dernier type de connaissance est représenté par le contexte qui contient des informations globales sur l'image à traiter : la taille de l'image, la résolution, l'absence ou la présence de tel ou tel objet, etc. Le contexte est fourni par l'utilisateur et est

utilisé pour l'évaluation des résultats. S'il est incomplet, des requêtes à l'utilisateur peuvent être générées pour demander des précisions.

Mécanismes de raisonnement OCAPI met principalement en œuvre deux types de raisonnement : la planification et le contrôle de l'exécution (voir Figure 2.2).

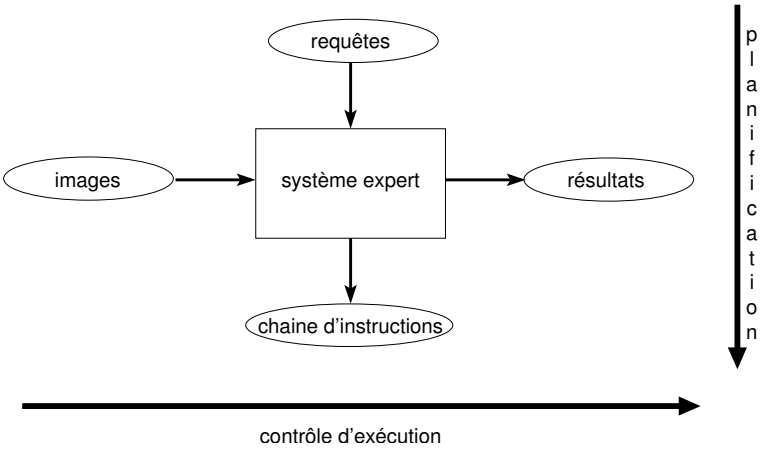


FIG. 2.2 – Planification et contrôle d'exécution du système OCAPI.

Le système est activé par une requête qui est composée : d'un but, des images à traiter et de contraintes sur les résultats. La planification décompose la requête en sous-buts jusqu'à obtenir la correspondance entre les sous-buts et les traitements réalisés par les opérateurs. Le contrôle de l'exécution cherche à obtenir des résultats optimaux. La sélection des opérateurs et des paramètres est faite par un mécanisme d'*essai-erreur*.

Moteur d'exécution Le fonctionnement du moteur d'exécution est illustré par l'algorithme 1

Algorithme 1 Boucle principale du moteur d'exécution d'OCAPI.

-1- *mise en correspondance globale*
Tant que *il y a des requêtes à traiter* **faire**
  -3- *choix de la requête à traiter*
  -5- *déclenchement des règles de choix*
  Tant que *la requête n'est pas satisfaite* **faire**
    -7- *choix effectif de l'opérateur à exécuter*
    -8- *paramétrisation de l'opérateur*
    -9- *exécution de l'opérateur*
    -10- *évaluation des résultats obtenus*
  Fin tant que
Fin tant que

Les étapes 1, 5 et 7 sont réalisées par le planificateur, la mise en correspondance globale vérifie que tous les buts demandés peuvent être effectivement traités par le système. La phase 5 génère un plan en classant les opérateurs disponibles puis la phase 7 choisit l'algorithme à exécuter pour chaque but.

En phase 8, le paramétreur détermine différemment les valeurs des paramètres de l'algorithme selon qu'il s'agisse d'une initialisation ou d'un ajustement. En phase 9, s'il s'agit d'un opérateur élémentaire celui-ci est directement exécuté, s'il s'agit d'un opérateur complexe, le moteur est appelé récursivement sur la décomposition de cet opérateur.

Le choix de la prochaine requête (3), l'évaluation des résultats (10) et le bouclage sont réalisés par le système expert (le pilote).

Ocapi possède les caractéristiques suivantes :

- une architecture de noyau de systèmes experts offrant des structures de représentation et de raisonnements généraux indépendants de toute bibliothèque spécifique de traitement d'images ;
- une description explicite des programmes des bibliothèques ;
- une description explicite des séquences de traitement d'images ;
- un mécanisme de choix entre les différents programmes et méthodes de traitement d'images ;
- une prise en compte de la connaissance sur l'évaluation des résultats et l'ajustement des paramètres ;
- une automatisation du mécanisme de contrôle de l'exécution.

Ocapi a été utilisé pour générer Progal (Clément, 1990) afin d'automatiser une chaîne de traitements d'images de galaxies et Planete (Thonnat et Moisan, 1995) destiné au pilotage temps réel d'un véhicule.

Ocapi a été suivi d'autres systèmes qui peuvent être vus comme des évolutions. Pour faciliter et accélérer leur développement, la plate-forme Lama a été créée. Elle inclut des bibliothèques de programmes, des interfaces graphiques, et d'autres outils qui permettent de constituer des bases de connaissances en utilisant le langage de description YAKL. Ce langage est un format pour les bases de données tout en restant compréhensible par les experts chargés de les construire. Il permet d'exprimer deux types de déclaration : structurelle (type d'opérateur, syntaxe d'appel, etc.) et des règles (règles d'initialisation, règles d'ajustement, etc.). Cette plate-forme, a permis de développer le système Pegase qui inclut des mécanismes de réparation absents d'Ocapi (qui ne permet pas de changer de plan lorsque celui utilisé échoue). Pegase a été utilisé, entre autre, pour la reconnaissance de cibles (Shekhar et coll., 1999), le traitement d'images médicales (Crubézy, 1999) et d'images satellitaires. D'autres systèmes, tels que Phenix qui possède une planification hiérarchique avec une planification dirigée par les opérateurs, et MedIA pour le traitement d'images médicales, ont ensuite été créés sur la plate-forme Lama.

2.2.3.3 IPUS

Dans IPUS (*Integrated Signal Processing and Signal Understanding*) (Lesser et coll., 1993b; Lesser et coll., 1993a; Lesser et coll., 1995), le but est de concevoir des systèmes de traitement et d'interprétation du signal qui permettent une interaction de haut niveau entre les solutions théoriques en traitement du signal et les solutions heuristiques en interprétation du signal. Sous les paradigmes classiques, les problèmes de réalisation de systèmes de traitement du signal amènent une explosion combinatoire en terme de réglage d'algorithmes. En général, le problème de réglage d'un algorithme avec un jeu de paramètres particulier est approprié à un environnement donné mais peut impliquer un contrôle non-linéaire ou être non soluble avec les techniques de contrôle classiques. Le système que les auteurs d'IPUS proposent permet non seulement la reconfiguration de la chaîne de traitements pour suivre l'évolution du signal mais permet aussi l'analyse de données mise en cache pour réduire l'incertitude sur l'interprétation du signal.

Deux prémisses régissent le système :

- La recherche d'une interprétation correcte des sorties implique la recherche simultanée (concourante) d'algorithmes de traitement du signal (SPA : *Signal processing algorithm*) et des paramètres de contrôle appropriés à l'environnement ;
- l'interaction entre ces recherches doit être structurée selon une théorie formelle qui explique comment l'usage inapproprié d'un SPA peut déformer sa sortie.

L'architecture générique de IPUS, représentée Figure 2.3, est instanciée pour un problème particulier avec des bases de connaissances et des algorithmes de traitement. Cette architecture est composée de quatre éléments clés : la *détection de désaccord*, le *diagnostic*, le *retraitement*, et le *diagnostic différentiel*, détaillés ci-dessous.

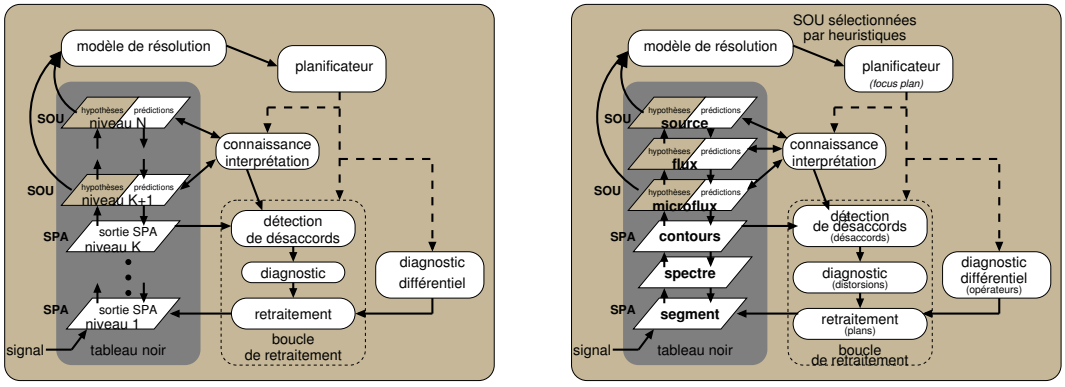


FIG. 2.3 – Architecture générique de IPUS et instance de cette architecture pour l'interprétation de sons. Les pointillés indiquent l'exécution de plans tandis que les flèches solides indiquent la transmission de données. Les parenthèses indiquent la connaissance nécessaire à la réalisation des actions.

Les algorithmes de traitement du signal sont vus comme des instances de SPA génériques (objets) avec des valeurs particulières de paramètres. Ces valeurs impliquent des capacités ou limitations à traiter une situation et à désambiguïser une interprétation.

Pour chaque bloc de signal à traiter, une configuration initiale de SPA est donnée. Les SPA sont choisis, non seulement en fonction de leur capacité à traiter les données mais aussi pour fournir des indications sur le moment où des objets inattendus apparaissent. Les SPA produisent des sorties (*correlates*) qui sont utilisées comme preuves pour soutenir une hypothèse d'interprétation particulière présente dans le contexte. Le contexte étant, dans cette étude l'ensemble des objets présents et leur orientation (par exemple la vitesse et la distance d'un objet spécifique).

Détection de désaccord Une fois la chaîne de traitements effectuée, la *détection de désaccord* est activée. L'architecture de IPUS contient deux types d'hypothèses d'interprétation du signal à partir des sorties des SPA : l'interprétation courante et la prédiction. Le désaccord intervient lorsque ces deux interprétations sont différentes et lorsque les sorties des SPA se contredisent. Cette détection est basée sur : les modèles des objets, les correspondances entre les sorties des SPA, et les caractéristiques du signal (dépendantes du domaine d'application).

Diagnostic de désaccord Si un désaccord apparaît, le *diagnostic* tente de l'expliquer en terme d'hypothèses de *distorsion*. Un modèle de SPA décrit comment la sortie d'un SPA est modifiée lorsqu'un de ses paramètres varie. Le diagnostic consiste à retrouver les distorsions qui expliquent le désaccord.

Retraitement et diagnostic différentiel Le *retraitement* sélectionne un nouvel ensemble de SPA qui permettent d'éliminer ou de réduire les distorsions à partir de plans de recherche et des modèles de SPA. Cette recherche est faite de manière itérative car les auteurs partent du principe qu'il est généralement impossible de prédire *a priori* les paramètres exacts à utiliser dans le retraitement. Lorsqu'il existe plusieurs interprétations concurrentes, le *diagnostic différentiel* sélectionne et exécute un retraitement pour détecter les caractéristiques qui peuvent discriminer entre différentes interprétations.

Contrôle IPUS utilise RESUN (Carver et Lesser, 1991) pour contrôler l'exécution des sources de connaissances. L'exécution est vue comme une procédure qui rassemble des preuves pour résoudre des sources d'incertitude (SOU : *Source Of Uncertainty*). La résolution de problème est dirigée par l'ensemble des interprétations et des SOU de chaque hypothèse dans le modèle de résolution de problème. Un planificateur réactif maintient le contrôle en exécutant des plans de contrôle. Les plans de contrôle sont des schémas qui définissent les stratégies et les SPA disponibles pour traiter et interpréter les données et pour résoudre les SOU. Le planificateur est soutenu par des heuristiques de Focus, dépendantes du contexte, qui permettent de concentrer le raisonnement sur des incertitudes particulières d'interprétation afin de les résoudre en priorité.

L'architecture générique d'IPUS a été instanciée particulièrement pour la reconnaissance de sons et pour la séparation de sources (téléphone, cris d'enfant, etc.) (Lesser et coll., 1995; Klassner et coll., 1998). Les instances de cette architecture étant très

dépendantes du domaine, la réutilisation des instances déjà composées reste très délicate, de même que la composition des applications, qui nécessite la présence des deux expertises en même temps (dépendante et indépendante du domaine d'application).

2.2.3.4 Architecture Multigraphe pour les systèmes structurellement auto-adaptatifs

Le système proposé par les auteurs de (Sztipanovits et coll., 1998; Karsai et Sztipanovits, 1999) vise à la création d'applications de traitement du signal structurellement auto-adaptatives. La conception d'un système de traitement du signal étant sujette à beaucoup de modifications coûteuses, une architecture générique qui permet une modification aisée du système entier est proposée. De plus, les systèmes statiques n'étant pas capables de prendre en compte des modifications de la chaîne de traitements, la possibilité de modifier automatiquement les traitements en cours de fonctionnement est prévue.

L'architecture générique proposée est basée sur l'architecture Multigraphe (MGA) (Sztipanovits et coll., 1995) qui possède trois niveaux d'abstraction : l'application, la synthèse de programmes (MIPS : *Model-Integrated Program Synthesis*) et le niveau méta. L'architecture simplifiée est représentée Figure 2.4.

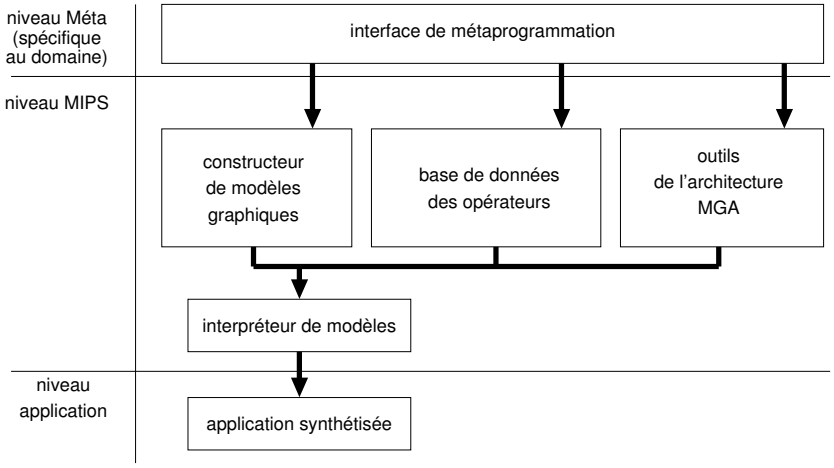


FIG. 2.4 – Architecture Multigraphe.

Le niveau Méta Le niveau Méta est dédié à fournir des outils pour générer des modèles spécifiques au domaine d'application pour les niveaux inférieurs.

Synthèse de programmes (MIPS) Le niveau MIPS possède trois principaux composants :

- le constructeur de modèles graphiques permet à l'utilisateur de concevoir l'architecture de son application,

– la base de données contient tous les blocs de base nécessaires à la construction de l'application,
– les outils de l'architecture Multigraphe permettent l'analyse de l'application en cours de traitement et notamment de générer l'*interpréteur de modèles*.

Les deux premiers composants sont génériques tandis que le troisième est très lié au domaine. Les instances spécifiques au domaine de l'environnement MIPS sont une suite d'outils nécessaires au support de la construction de modèles, l'analyse de modèles et la synthèse de programmes.

La base de données contient les opérateurs de traitement du signal. Il y a deux types d'opérateurs : les *opérateurs simples* qui représentent uniquement un traitement spécifique et les *opérateurs composés*. Les opérateurs composés contiennent des opérateurs simples et des opérateurs composés dont les connexions sont représentées par un graphe de flot de données. Lorsque qu'un opérateur composé peut être décrit de différentes manières, par exemple lorsqu'il peut fonctionner avec une ou plusieurs sources, il contient l'ensemble des plans alternatifs. La figure 2.5 représente un opérateur composé et ses différents constituants.

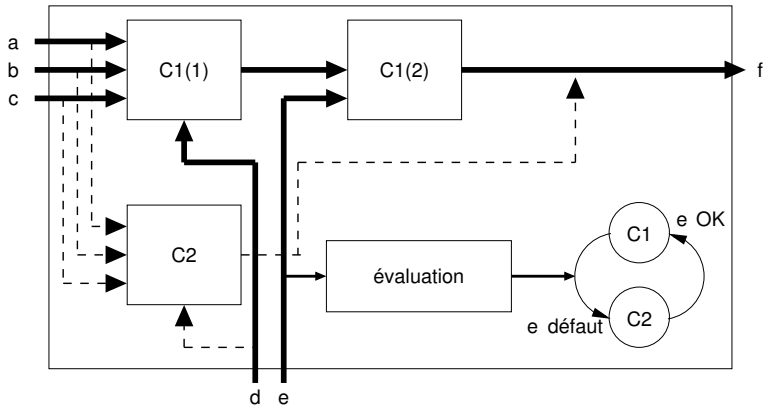


FIG. 2.5 – Exemple d'opérateur composé.

Dans cet exemple l'opérateur peut fonctionner soit avec les opérateurs C1(1) et C1(2) soit avec l'opérateur C2 seul. Lorsque l'information e n'est plus disponible (défaut de capteur), l'opérateur doit être reconfiguré. L'évaluateur active alors une transition de la machine à états finis qui représentent les différents plans alternatifs. Lorsque les données redeviennent disponibles, l'évaluateur permet de repasser en configuration initiale. C'est à travers les plans alternatifs des opérateurs composés que l'application est structurellement auto-adaptative.

Construction d'une application L'utilisateur construit une application à l'aide du constructeur de modèles graphiques. Celui-ci est ensuite traduit en un graphe de flot de données dont les nœuds représentent des traitements et les arcs des transferts de signaux. Les nœuds actifs sont des opérateurs tandis que les nœuds de données repré-

sentent des buffers de signaux pour le stockage et le partage. Ce graphe est ensuite envoyé à l'interpréteur de modèle qui utilise la base de données et construit incrémentalement le graphe exécutable à partir de la racine du graphe. L'interpréteur construit également une représentation de l'application sous forme d'objets. Cette représentation permet de stocker les références aux objets et aux composants du graphe exécutable et de maintenir les connexions avec les événements qui peuvent provoquer une reconfiguration. Si une reconfiguration est demandée alors la synthèse est redémarrée à partir du nœud incriminé.

L'architecture Multigraphe a été appliquée au domaine de l'aérospatial (MGA-DTOOL, MGA-RDS), de l'industrie chimique (IPCS) (Sztipanovits et coll., 1995) et du monitorage de systèmes dynamiques (CADDMAS) (Sztipanovits et coll., 1998).

Cette approche permet de construire des applications reconfigurables de manière très efficace car très proche du domaine d'application. Les objectifs visés sont, d'une part, l'aide à la construction d'applications d'une manière structurée, réutilisable et économique en modifications (le temps est réduit par la facilité de modification) et, d'autre part, la possibilité d'adaptation de manière décentralisée (au niveau des opérateurs composés) qui permet de réduire considérablement le nombre d'alternatives possibles avec un système centralisé.

Cependant, cette approche ne prend pas en compte la difficulté de réunir les deux types de connaissances (dépendante et indépendante du domaine) pour la construction et n'inclut pas d'étape d'interprétation du signal. Le système présenté possède beaucoup de propriétés du pilotage d'algorithmes de bas niveau mais il est plutôt destiné à la conception de systèmes tolérants aux fautes dont le choix des algorithmes est plus dirigé par une panne que par une recherche d'optimisation du traitement en fonction des données à traiter.

2.2.3.5 Discussion

Les solutions proposées pour concevoir un système de pilotage d'algorithmes dépendent beaucoup de l'application visée. Certaines solutions construisent une chaîne de traitements à partir de squelettes existants (Clément et Thonnat, 1993), d'autres demandent la réalisation de la chaîne par l'opérateur (Karsai et Sztipanovits, 1999). Pour évaluer les résultats, certains séparent les connaissances et n'utilisent que des connaissances dépendantes de l'application pour évaluer les résultats (Clément et Thonnat, 1993), d'autres utilisent des connaissances mixtes (Lesser et coll., 1993b), et certains incluent directement l'évaluation des résultats dans les opérateurs (Karsai et Sztipanovits, 1999).

En définitive, la construction d'une application de pilotage d'algorithmes amène à se poser les questions suivantes :

- Comment représenter la connaissances pour la résolution (base de règles, de plans, etc.) ?
- Comment représenter les algorithmes pour effectuer les traitements ?
- Comment prendre en comptes les informations contextuelles ?
- Comment enchaîner les traitements à exécuter (planification, plan fixe, etc.) ?

– Comment gérer l'exécution des actions ?
– Si le domaine d'application le permet, comment interpréter les résultats et modifier les paramètres ?

Les différentes techniques employées pour résoudre ces problèmes sont détaillées dans la section suivante.

2.2.4 Concepts généraux d'un système de pilotage d'algorithmes

L'objectif de cette section est de présenter les différentes techniques employées à tous les niveaux d'un système de pilotage d'algorithmes. Premièrement, le schéma général d'un système de pilotage d'algorithmes de traitement du signal est détaillé. Puis, les deux sections suivantes présentent les solutions apportées dans les deux grandes phases du pilotage : la génération de la chaîne de traitements et l'exécution. Enfin, une discussion donne une critique des techniques exposées.

2.2.4.1 Schéma général d'un système de pilotage d'algorithmes de traitement du signal

Des différents exemples de la littérature, on peut déjà faire ressortir les deux grandes phases du pilotage d'algorithmes :

– la phase de *génération de la chaîne de traitements* dédiée à la sélection, l'ordonnancement, la planification, l'édition de liens et l'initialisation des paramètres ;
– la phase d'*exécution* dédiée à l'exécution, le contrôle et l'interprétation des résultats.

La figure 2.6 illustre ce découpage.

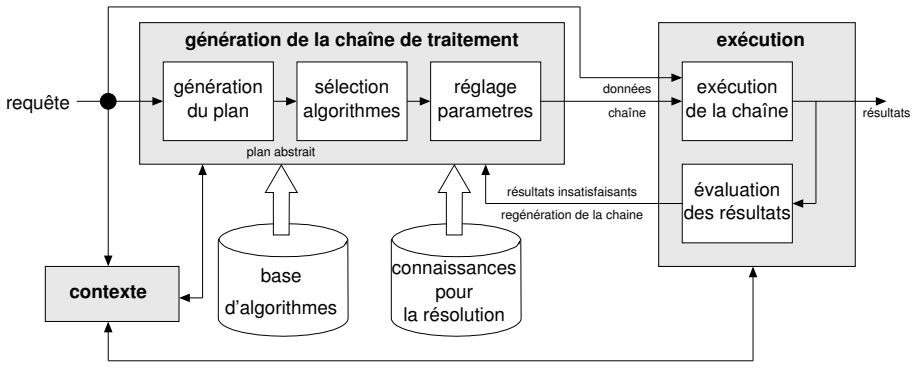


FIG. 2.6 – Schéma global d'un système de pilotage d'algorithmes.

Les données extérieures peuvent être regroupées sous le terme de requête (Shekhar et coll., 1999). Une requête peut contenir les données à traiter, le nom du traitement à appliquer et des données contextuelles (p. ex. le nombre d'objets à détecter).

Dans la phase de *génération de la chaîne de traitements*, la requête est analysée pour générer un plan de traitements. Le plan de traitements contient les étapes principales

à effectuer. L'opération suivante consiste à choisir les algorithmes et à les paramétrer. La chaîne obtenue est exécutée puis les résultats sont analysés. Si les résultats ne correspondent pas à ceux attendus, la chaîne est modifiée. La sélection, la paramétrisation et l'évaluation sont généralement définies par des bases de connaissances avec l'aide du contexte.

Le schéma exposé est très général et certaines étapes peuvent être ignorées. Par exemple dans MGA (Karsai et Sztipanovits, 1999), la génération de plan est laissée à la charge de l'utilisateur. Les sections suivantes décrivent les différentes techniques employées à chaque étape des deux grandes phases : la génération de la chaîne de traitements et l'exécution.

2.2.4.2 Génération de la chaîne de traitements

Dans la phase de génération de la chaîne de traitements, le pilote a pour rôle de réagir à une requête qui lui demande de réaliser un traitement sur des données. Le traitement original demandé est décomposé en sous traitements et ce découpage est souvent ramené à un problème de planification. Le pilote choisit et paramètre un ou plusieurs algorithmes de sa base en fonction de plans de traitementss fixes ou générés, des buts des algorithmes et d'un contexte de la situation courante. Une fois ces algorithmes ordonnancés, l'étape suivante consiste à les paramétrer.

Cette phase de génération est assez unanime dans la littérature, les différences résident surtout dans la manière de planifier : ensemble de squelettes de plan (PEGASE (Moisan, 1998), BORG (Clouard et coll., 1998)), construction d'un plan (EXTI (Dalle et Dejean, 1998)), exécution d'un plan fixe unique (MGA (Karsai et Sztipanovits, 1999)).

Pour effectuer cette génération, le pilote doit réunir trois principaux modules :

- une base d'algorithmes : Comment représenter les algorithmes pour effectuer les traitements ? Comment prendre en compte les informations contextuelles ?,
- une base de connaissances permettant de choisir les algorithmes : Comment représenter la connaissance pour la résolution ?,
- un générateur de plans : Comment enchaîner les traitements à exécuter ?.

Base et représentation des algorithmes Le pilote choisit dans une base les algorithmes nécessaires au but à atteindre. Pour mener à bien cette opération, la base et la représentation des algorithmes doivent être bien structurées de manière à pouvoir manipuler efficacement les algorithmes.

Dans OCAPI (Clément et Thonnat, 1993) et (Shekhar et coll., 1994), les algorithmes sont représentés sous forme d'objets. Chaque algorithme est associé à un but (filtrage, seuillage, etc.). Il contient une description de sa syntaxe d'appel (c.-à-d. nom de la méthode, arguments d'entrée-sortie) et d'autres informations telles que sa qualité, sa rapidité, etc. À chaque algorithme est associé des règles d'initialisation et d'ajustement. Par exemple, si trop peu d'événements ont été détectés, alors il est nécessaire de diminuer le seuil d'un certain pas ou d'un certain pourcentage. Des représentations d'opérateurs sont données Figure 2.7.

Dans ExTI (Dalle et Dejean, 1998) les données et les opérateurs de traitement d'image sont représentés par de la connaissance profonde : les opérateurs sont vus comme des constructeurs d'*indices visuels*. Les images sont décrites selon neuf champs et les opérateurs sont exprimés par les transformations qu'ils imposent à chacun des champs. Chaque opérateur impose cinq étapes de transformation à la donnée :

- l'identification détermine les sous-structures de la donnée à traiter,
- la mesure calcule une valeur à partir des sous-structures,
- la partition sépare en plusieurs classes les sous-structures en fonction de la mesure,
- le codage attribue une valeur aux classes,
- le regroupement rassemble les éléments de même valeur de codage.

L'opérateur est décrit en indiquant comment ces transformations sont réalisées sur les images. De cette description, on obtient une représentation fonctionnelle des opérateurs qui permet de les décomposer en un enchaînement de transformations de base.

Dans (Shekhar et coll., 1999), les algorithmes sont accompagnés de trois types de connaissances. La connaissance syntaxique (syntaxe d'appel, etc.), la connaissance sémantique (dans quel cas utiliser l'algorithme, comment évaluer les résultats, etc.) et la connaissance stratégique (stratégie de réparation, d'opération, etc.). Les algorithmes composent ensuite une application et chaque paramètre devient un paramètre de l'application. Le réglage des paramètres revient donc à chercher la configuration optimale dans l'espace des paramètres. Comme cette recherche peut être très longue, des connaissances spécifiques sont ajoutées pour restreindre l'espace de recherche.

Dans MGA (Karsai et Sztipanovits, 1999), les algorithmes sont représentés par des opérateurs simples ou des opérateurs composés. Les opérateurs simples sont uniquement destinés à réaliser une opération de traitement du signal. Les opérateurs composés contiennent un ensemble d'opérateurs simples et composés reliés sous forme de graphe, les arcs du graphe représentant les flot de signaux. Lors de la synthèse, les opérateurs sont compilés en programmes exécutables et traduits en modèle pour garder la trace de la configuration courante et permettre une modification des opérateurs complexes.

Générateur de plans Dans la plupart des applications qui se rapportent au pilotage d'algorithmes, la construction de la chaîne de traitements est ramenée à un problème de planification des actions qui utilise très souvent des squelettes de plans existants.

Ainsi, dans Ocapi (Clément et Thonnat, 1993; Shekhar et coll., 1994) le problème posé est décomposé en sous-problèmes selon une base de connaissances contenant les buts que le système sait résoudre. D'autre part, la base d'algorithmes contient les *opérateurs complexes* qui représentent des plans de traitements associés à un but particulier. La figure 2.7 représente le fonctionnement de leur système.

En réponse à une requête, le but à atteindre est inféré. Si aucun algorithme ne correspond directement au but, le but est décomposé en sous-buts jusqu'à ce qu'ils correspondent à des buts simples.

Dans Borg (Clouard et coll., 1998), les auteurs utilisent une modélisation des plans de traitements par un arbre de tâches à cinq niveaux d'abstraction. Une tâche est caractérisée par un but à atteindre, des contraintes, des images d'entrée, des images

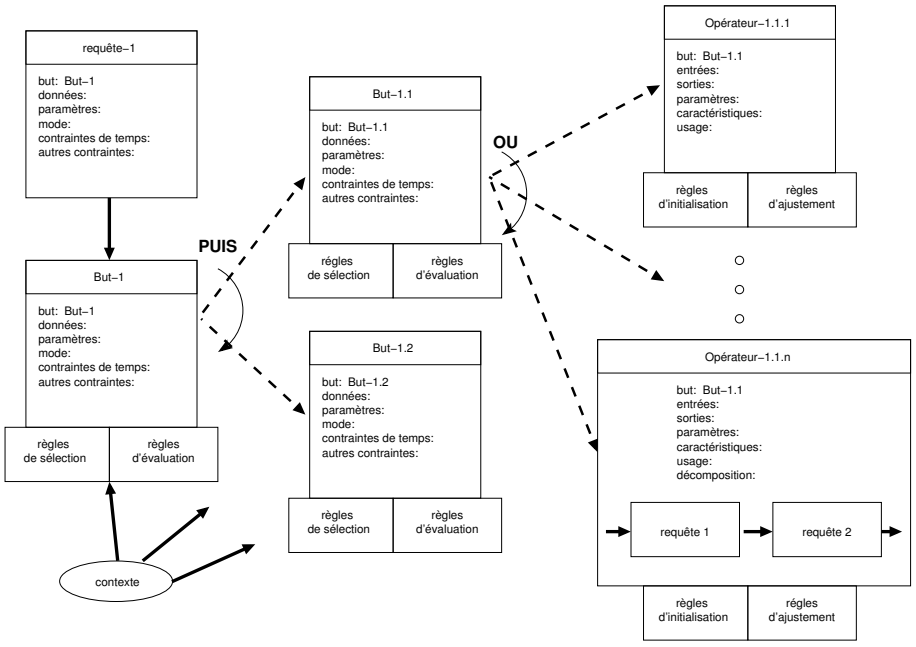


FIG. 2.7 – Entités mises en jeu lors de la génération de la chaîne de traitements dans OCAPI.

de sortie, et des règles d'évaluation. La décomposition des tâches en sous tâches est effectuée par des sources de connaissances, avec un contrôle de type tableau noir.

D'autres méthodes construisent le plan solution à partir des données. Dans EXTI (Dalle et Dejean, 1998), la requête utilisateur est traduite en une description de données particulières à travers une hiérarchie de langages (concepts → données → objectifs). Le planning utilisé consiste ensuite à trouver l'enchaînement de traitements simples qui amène de l'image initiale à l'image finale correspondant à l'objectif à atteindre.

Dans MGA (Karsai et Sztipanovits, 1999), il n'y a pas de planification. La chaîne de traitements est représentée sous forme de graphe de flot de données. La modification du plan est effectuée par les traitements eux mêmes qui contiennent la connaissance nécessaire pour modifier leur structure en fonction de différents événements (panne de capteur par exemple).

Base de connaissances pour la sélection d'algorithmes La sélection des algorithmes est une étape importante du processus qui demande une bonne connaissance de l'application.

Dans OCAPI (Clément et Thonnat, 1993; Shekhar et coll., 1994), le système doit répondre à une requête qui est décomposée en sous-buts. Chaque sous-but doit correspondre à un algorithme. Comme plusieurs algorithmes existent pour un même but, le choix des algorithmes est effectué à l'aide de règles de production qui dresse une liste ordonnée des algorithmes à utiliser. Ces règles n'intègrent aucune connaissance sur le

domaine.

Dans ExTI (Dalle et Dejean, 1998), les données d'entrée et de sortie sont traduites en indice visuel. Les transformations de base qui permettent de passer d'un indice à l'autre sont ensuite recherchées. La chaîne constituée est raffinée jusqu'à obtenir les opérateurs les plus simples. Des règles expertes éliminent ensuite certains opérateurs, les autres sont conservés pour amener vers une ou plusieurs solutions.

Dans MGA (Karsai et Sztipanovits, 1999), la chaîne de traitements est définie par l'utilisateur qui impose un fonctionnement général du système. La modification de la chaîne de traitements se fait à travers les opérateurs composés qui représentent une sous-chaîne de traitements. Un opérateur composé contient les différentes alternatives de sa composition. Par exemple, si un opérateur composé prend habituellement en entrée deux signaux et que la source du deuxième signal est interrompue, l'opérateur contient un plan de connexions alternatif qui lui permet de fonctionner avec une seule source. Un opérateur composé contient donc non seulement de la connaissance indépendante du domaine provenant des opérateurs simples qui le composent mais aussi de la connaissance spécifique provenant des plans alternatifs qui eux sont définis par l'utilisateur.

2.2.4.3 Exécution de la chaîne de traitements

Une fois le plan généré, le pilote doit exécuter tous les algorithmes tour à tour. Cette exécution peut être plus ou moins contrôlée selon les contraintes de temps et les connaissances disponibles sur les résultats. Pour effectuer l'exécution, le pilote doit donc posséder :

- un moteur d'exécution : Comment gérer l'exécution des actions ?,
- un moteur de contrôle d'exécution : Comment prendre en compte les informations contextuelles ? Si le domaine d'application le permet comment interpréter les résultats et modifier les paramètres ?

Moteur d'exécution Le moteur d'exécution a pour rôle de transformer la chaîne de traitements symboliques en programme effectif. Lorsque le moteur reçoit en entrée la chaîne de traitements à exécuter, seuls les traitements élémentaires sont exécutés. Les traitements composés sont décomposés pour exécuter chaque bloc un à un ou en parallèle si cela est possible.

Le moteur d'exécution doit aussi gérer la création de variables intermédiaires pour les connexions entres les algorithmes, le partage de fichier, l'exécution parallèle, le partage mémoire.

Par ailleurs, le moteur doit gérer différents formalismes, un algorithme peut aussi bien être écrit en C qu'en Scheme ou même en script *shell*. Dans Ocapi (Clément, 1990), une représentation Le_Lisp est utilisée pour exprimer les syntaxes d'appel avec une interface C et Fortran vers Le_Lisp. L'ensemble des algorithmes est donc exécuté sous un seul langage pour le moteur.

Dans MGA (Karsai et Sztipanovits, 1999), la chaîne de traitements, qui est représentée par un graphe, est exécutée de deux façons. Partant du principe que la compilation

complète est trop lente à la génération et qu'un interpréteur est trop lent à l'exécution, les nœuds du graphe (opérateurs) sont compilés au préalable tandis que le plan est interprété (Sztipanovits et coll., 1998).

Contrôle de l'exécution Le contrôle de l'exécution permet de s'assurer que le système tend bien vers les résultats voulus. Dans OCAPI (Clément et Thonnat, 1993) et BORG (Clouard et coll., 1998), le contrôle est assuré par un mécanisme d'*essai-erreur* qui consiste à relancer un traitement (après ajustement ou changement d'algorithmes) tant que les résultats ne correspondent pas à ceux attendus.

Pour évaluer les résultats, le moteur utilise les connaissances fournies par la requête et celles détenues par le *contexte* (Shekhar et coll., 1994; Clouard et coll., 1998). Le contexte contient des informations de type physique, qui renseignent sur les capteurs et les conditions d'acquisition, de type perceptif, qui décrivent les entités présentes, et de type sémantique qui définissent la notion d'objet. Un exemple de contexte est représenté Figure 2.8.

; Description physique	; Description perceptive	; Description sémantique
(length-of-image=256)	(type-of-image=density)	(object-mini-size=36)
(width-of-image=256)	(background ?=yes)	(object-number-of-classes=1)
(noise=low)	(background=homogenous)	(object-size=average)
(imaging-device=microscope)	(gray-level=discriminative)	(object-color=dark)
(quality=sharp)	(boundaries-contrast=high)	

FIG. 2.8 – Exemple de contexte extrait de BORG.

Dans (Shekhar et coll., 1999), les résultats d'une application sont évalués selon deux modes : spécialiste et utilisateur. Le mode spécialiste donne les résultats de chaque traitement de l'application qui doivent être évalués par l'utilisateur. En mode utilisateur, seul le résultat final doit être évalué par l'utilisateur qui ne possède pas les connaissances nécessaires pour évaluer les résultats de chaque étape du traitement. Enfin, PEGASE (Moisan et coll., 1997) intègre des règles de réparation si le traitement à été interrompu.

2.2.4.4 Discussion

Dans cette discussion, les solutions proposées sont analysées en regard de notre application qui est le monitorage temps réel de patient et dont les principales difficultés sont de choisir les algorithmes et d'évaluer les résultats.

Choix des algorithmes et séparation de la connaissance La plupart des systèmes de traitement du signal peuvent être vus comme un ensemble de boîtes noires interconnectées qui prennent en entrée un signal et en ressortent une information directement utilisable. Par exemple, un système de monitorage transforme des signaux électrocardiographiques en un diagnostic médical. Durant la conception, un expert du *traitement* du signal choisit et paramètre les algorithmes jusqu'à ce que les résultats correspondent aux attentes de l'expert médical en charge de l'*interprétation* du signal.

Nous avons donc, dans la conception, deux types de connaissances : les connaissances du *domaine indépendant* représentées par l'expert du traitement du signal et les connaissances du *domaine dépendant* représentées par l'expert médical (Shekhar et coll., 1994; Clouard et coll., 1998).

Cette mise en commun des savoirs est effectuée lors de la réalisation d'un système statique mais devient beaucoup plus problématique lorsqu'il s'agit de systèmes supervisés et adaptatifs, capables de modifier leur architecture. Hormis la difficulté habituelle d'une acquisition d'expertise, se pose le problème de savoir comment structurer ces deux types de connaissances pour le raisonnement. Dans la littérature on peut trouver deux visions différentes : l'une consiste à séparer au maximum les connaissances pour la généralisation, l'autre utilise au contraire une fusion pour optimiser le raisonnement dans le cadre d'une application spécifique.

Dans OCAPI (Clément et Thonnat, 1993; Shekhar et coll., 1994), les auteurs distinguent nettement les deux connaissances qui sont spécialisées d'une part dans le traitement du signal (domaine indépendant) et, d'autre part, dans l'interprétation du signal (domaine dépendant). Pour ces auteurs, dans le domaine du traitement du signal, le raisonnement est beaucoup plus dirigé vers le choix et le paramétrage d'un traitement pour un but donné. Dans le domaine de l'interprétation du signal, le raisonnement consiste plus à vérifier la correspondance des résultats avec un modèle ou avec une hypothèse. La séparation des types de connaissance et de raisonnements permet un développement plus approprié de chacun. La coopération entre les deux types de raisonnements est illustrée Figure 2.9.

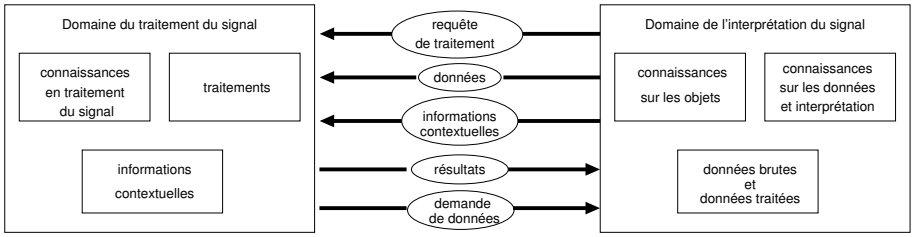


FIG. 2.9 – Coopération entre le traitement du signal et l'interprétation du signal.

Dans IPUS (Lesser et coll., 1993a; Lesser et coll., 1995), les auteurs proposent un paradigme pour structurer l'interaction entre traitement (domaine dépendant) et interprétation (domaine indépendant) du signal afin de permettre une interaction de haut niveau entre les solutions théoriques en traitement du signal et les solutions heuristiques en interprétation du signal. Dans ce paradigme la recherche de l'algorithme de traitement du signal approprié est intimement liée à la recherche de l'interprétation correcte des données de sortie du traitement du signal. Les sorties des algorithmes sont comparées aux prédictions pour fournir un degré de *désaccord*. Le désaccord est ensuite expliqué par les *distorsions* entre les résultats et les prévisions. Un algorithme est choisi s'il implique des transformations qui pourront réduire ou annuler les distorsions. Le choix est guidé par des connaissances liées au domaine d'application et aux connais-

sances en traitement du signal. La décision est donc prise vis-à-vis des deux types de connaissance.

Dans EXTI (Dalle et Dejean, 1998), ce sont les données qui guident le choix des algorithmes. La construction de la chaîne de traitements peut être vue comme la recherche des chemins de transformations (ici traitement d'image) qui permettent de passer de l'image initiale à l'image finale.

Dans MGA (Karsai et Sztipanovits, 1999), le choix d'algorithmes est complètement lié au domaine d'application. L'architecture Multigraphe générique est indépendante du domaine mais les instances de cette architecture sont réalisées en ajoutant des connaissances spécifiques (sous formes de modèles) au domaine d'application notamment pour le choix des traitements.

Il est intéressant de constater que la séparation entre les domaines est surtout assurée dans les systèmes de traitement d'images tels que OCAPI (Clément, 1990), EXTI (Dalle et Dejean, 1998) ou BORG (Clouard et coll., 1998), alors que les systèmes de traitement du signal tels que IPUS (Lesser et coll., 1993b) ou MGA (Karsai et Sztipanovits, 1999) utilisent plus de fusion de domaine. Cette différence est difficile à expliquer mais elle peut être due au fait que les opérations effectuées sur les images sont plus standard que celles effectuées sur des signaux qui peuvent être d'origine très variable. Qui plus est, le traitement de l'image est plus récent que le traitement du signal et l'on sait que pour améliorer les performances des traitements il est nécessaire de spécialiser de plus en plus les algorithmes. Le traitement du signal ayant une plus grande histoire, les algorithmes seraient donc plus spécifiques à un domaine particulier et donc moins généralisables. Notre application de traitement du signal électrocardiographique étant très spécifique, les connaissances utilisées pour le choix des algorithmes sont très liées au domaine.

Évaluation des résultats Dans la plus grande partie des cas, le mécanisme de pilotage repose surtout sur l'évaluation des résultats obtenus à chaque étape de traitement. Les solutions de pilotage d'algorithmes, notamment en traitement d'images, accorde une grande part à l'intervention de l'utilisateur dans la boucle d'évaluation des résultats. Si cette méthode est raisonnable dans le cas d'une aide à la décision, elle reste peu efficace lorsque des contraintes temps réel fortes sont présentes. Ainsi, pour le système PLANETE, destiné au pilotage temps réel d'un véhicule, la planification et l'exécution ont été simplifiées par rapport à OCAPI et la supervision très réduite (Thonnat et Moisan, 1995).

De plus, pour pouvoir interpréter les résultats il faut déjà avoir une idée précise de ce que l'on cherche à obtenir ce qui n'est pas toujours le cas. Selon les domaines, l'évaluation des résultats peut être assez ardue. Comment, par exemple, évaluer le succès d'une détection d'événements sans connaissance *a priori* du nombre d'événements à détecter ? Notre but étant de réaliser un système de monitorage autonome, l'évaluation ne peut se faire que d'une manière très basique reposant sur des connaissances *a priori*.

Part ailleurs, la plupart des systèmes proposés, tels que IPUS (Lesser et coll., 1993a) et PEGASE (Moisan et Thonnat, 1995), reposent sur une approche itérative par *essai-erreur* pour choisir les algorithmes adéquats. Cette approche est nécessaire pour des résultats optimaux mais très coûteuse lorsque des objectifs temps réel, notamment dans

un contexte USIC, imposent une solution rapide. Là encore, des connaissances *a priori* se révèlent moins générales mais plus efficaces. Ainsi, dans le domaine du monitorage cardiaque, le système SIMBAD (Soulas et coll., 1998; Le Certen et coll., 1998) inclut des règles de sélection d'algorithmes de détection de QRS. SIMBAD est constitué d'un analyseur de bruit de ligne qui permet de commuter les détecteurs en fonction du bruit de ligne.

Bien que les méthodes d'évaluation proposées soient efficaces pour obtenir de bons résultats, ces méthodes nécessitent une connaissance des résultats attendus et beaucoup de temps de calcul si le nombre d'erreurs est important. Elles ne sont donc pas adaptées aux contraintes temps réel, c'est pourquoi, MGA (Karsai et Sztipanovits, 1999) inclut l'évaluation des résultats à l'intérieur des opérateurs composés. Cette méthode ne nécessite donc pas de rebouclage et de régénération profonde de la chaîne de traitements.

La solution qui se rapproche le plus de nos contraintes temps réel et applicatives est celle proposée par (Sztipanovits et coll., 1998). Cependant, l'architecture proposée, même si elle reste modifiable, ne permet pas de séparer la connaissance nécessaire au pilotage de la construction des opérateurs complexes. De plus, les changements de structure sont provoqués par des événements (surtout des pannes de capteurs) et non par un mécanisme de raisonnement haut niveau qui prendrait en considération la nature des données traitées. En outre, le pilotage de tâches de haut niveau n'est pas abordé.

La solution retenue s'inspire des systèmes présentés mais utilise un plan fixe de traitements génériques pour des raisons de rapidité tout en modifiant les algorithmes utilisés (paramétrage ou changement d'algorithmes). Ces changements sont effectués par un pilote dont la connaissance est constituée de règles de pilotage et d'informations contextuelles initiales et remises à jour automatiquement. Les décisions du pilote reposent sur des connaissances dépendantes et indépendantes du domaine d'application.

2.2.5 Concepts retenus pour le pilotage d'un système de monitorage cardiaque

Cette section présente les concepts utilisés par la suite pour décrire les éléments du système de pilotage d'algorithmes de traitement du signal. Ces concepts sont définis à partir des différents travaux de la littérature et des contraintes inhérentes à un système de traitement du signal temps réel.

Une *application* de traitement du signal peut être décomposée en *tâches* interconnectées. Tout comme un opérateur composé, une tâche peut être réalisée par un ensemble d'algorithmes. Cet ensemble d'algorithmes est modifié en ligne par le pilote en fonction du contexte courant du système. Les tâches peuvent aussi être *activées* ou *désactivées* lorsque, par exemple, un signal ne fournit plus d'information ou est trop bruité pour être utilisable. Contrairement aux approches présentées, notre vision du pilotage centralise les informations puis déduit l'ensemble des modifications à effectuer sur la chaîne de traitements. De cette manière, l'ensemble des modifications à apporter à la chaîne de traitements de l'application à un temps donné est déduit à partir du même contexte.

Définition 2.1 (Tâche)
Une tâche est l'exécution d'un traitement réalisé par un ou plusieurs algorithmes.

Une tâche représente un traitement précis tel qu'une détection d'événements sur un signal. Pour réaliser ce traitement il existe souvent plusieurs algorithmes. Le pilote a pour rôle de choisir l'algorithme (ou l'ensemble d'algorithmes) le plus adéquat au contexte courant pour la tâche à réaliser. La tâche est proche des concepts d'opérateurs composés présentés précédemment.

Définition 2.2 (Contexte)
Le contexte est l'ensemble des informations qui influent sur l'application. Lorsqu'il est connu, ce contexte peut être utilisé pour piloter l'application.

La connaissance du contexte courant est utile pour paramétrer les algorithmes. Par exemple le niveau de bruit de ligne permet d'adapter les seuils de filtres. Mais ce contexte peut aussi contenir des informations de haut niveau dépendantes du domaine d'application tel que le type de données transmises (texte, image, vidéo, etc.), la pathologie d'un patient sous monitorage, etc.

Définition 2.3 (Application)
Une application est un ensemble de tâches interconnectées selon un schéma fixé pour réaliser une fonction précise.

Les applications de traitement du signal peuvent être complexes. Nous partons du principe qu'une décomposition en tâches permet de bien structurer le fonctionnement. Par exemple, une chaîne de réception peut être décomposée en trois tâches : réception du signal, filtrage du signal, détection-classification. Le contexte courant contient des informations telles que le niveau de bruit et le codage utilisé pour la transmission. Si le niveau de bruit augmente et que le codage des données change, la tâche de filtrage devra choisir un filtre plus sélectif et la tâche de détection-classification devra utiliser l'algorithme correspondant au codage courant.

Définition 2.4 (Tâche active/ tâche inactive)
Une tâche est active lorsqu'elle exécute le traitement d'un signal et retourne un résultat. Une tâche est inactive lorsque pour tout signal en entrée elle retourne un résultat déterministe (aucune sortie ou alarme) tout en maintenant ses variables internes à jour.

Le pilote peut choisir de désactiver une tâche lorsqu'il sait qu'aucun algorithme réalisant la tâche n'est capable d'effectuer le traitement sans provoquer un trop grand nombre d'erreurs. Cette désactivation peut avoir des effets sur les tâches en aval. C'est pourquoi la vision centralisée du pilote permet de désactiver une tâche et de modifier les autres tâches de façon à ce que l'application puisse continuer de fonctionner.

2.3 Présentation de CALICOT

2.3.1 Introduction

CALICOT (*Cardiac Arrhythmias Learning for Intelligent Classification of On-line Tracks*) (Carrault et coll., 2003; Wang, 2002) est un système de reconnaissance et d'apprentissage d'arythmies cardiaques à partir d'un électrocardiogramme (ECG). Sa principale caractéristique est de combiner des méthodes d'abstraction temporelle, d'apprentissage inductif et de supervision de processus industriel.

Ce prototype de système de monitorage se concentre surtout sur les étapes d'extraction de caractéristiques et de raisonnement pour le diagnostic médical. L'extraction de caractéristiques est appelée *abstraction temporelle*. Elle est assurée par un ensemble d'algorithmes de traitement du signal. Le diagnostic médical est réalisé par un raisonnement temporel basé sur la représentation des données sous forme de réseaux temporels. Une autre fonction de CALICOT est l'acquisition automatique de connaissances à partir d'exemples de pathologies. L'acquisition d'expertises étant un problème majeur, cette approche permet une mise à jour régulière de la base de connaissances.

Les liens entre les constituants de CALICOT sont détaillés en section 2.3.2. L'abstraction temporelle, en charge de l'analyse du signal, est présentée section 2.3.3. La section 2.3.4 se focalise sur l'apprentissage symbolique qui permet d'acquérir les motifs des arythmies cardiaques. La reconnaissance d'arythmies est assurée par un système de reconnaissance de chroniques exposé en section 2.3.5. Cette section se termine par les évolutions possibles de CALICOT en section 2.3.6.

2.3.2 Architecture de Calicot

La figure 2.10 présente l'architecture de CALICOT qui est composée de trois principaux modules :

- un module d'*abstraction temporelle* (1),
- un module de *reconnaissance de chroniques* (2),
- un module d'*apprentissage des arythmies cardiaques* (3).

L'abstraction temporelle, réalisée par des algorithmes de traitement de signal, traduit les ECG en événements symboliques. La reconnaissance de motifs temporels liés aux arythmies cardiaques s'appuie sur des structures de haut niveau : les chroniques, et les contraintes temporelles associées, apprises par programmation logique inductive (PLI) à partir d'une base d'exemples représentatifs d'arythmies. Cette architecture assure une bonne séparation entre l'extraction de caractéristiques du signal et la reconnaissance de pathologies qui nécessite des connaissances de plus haut niveau.

2.3.3 Abstraction temporelle

Pour effectuer le diagnostic médical, l'ECG est décrit selon une grammaire qui représente les objets caractéristiques à détecter. Un ECG comporte des événements structurés tels que l'onde P, le complexes QRS, les intervalles entre les ondes, qui vont constituer la base de la grammaire. L'abstraction temporelle du signal ECG représente

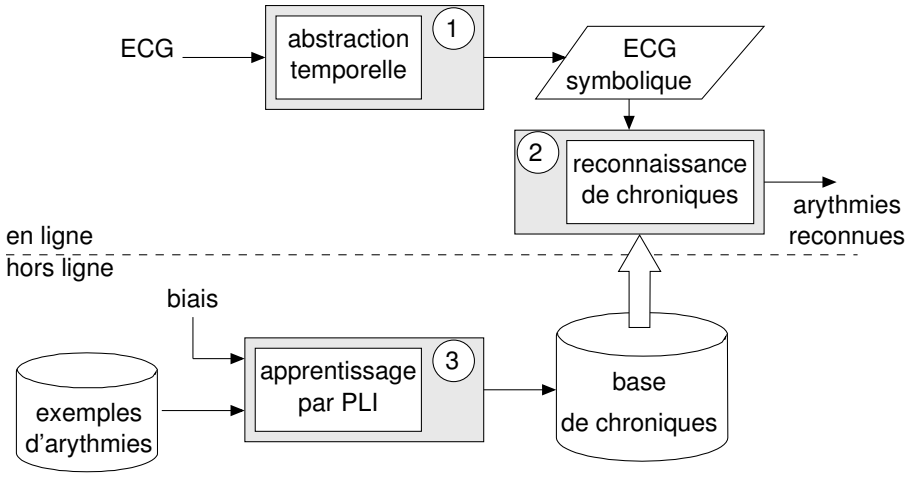


FIG. 2.10 – Architecture générale du système de monitorage CALICOT.

l'ECG numérique sous une forme symbolique. Elle détecte les événements représentatifs du signal et leur attribue des propriétés.

La figure 2.11 illustre la chaîne de traitements mise en œuvre pour réaliser cette transformation. Elle est constituée de 3 sous-modules : 1) la détection du complexe QRS, 2) sa classification et 3) la détection de l'onde P.

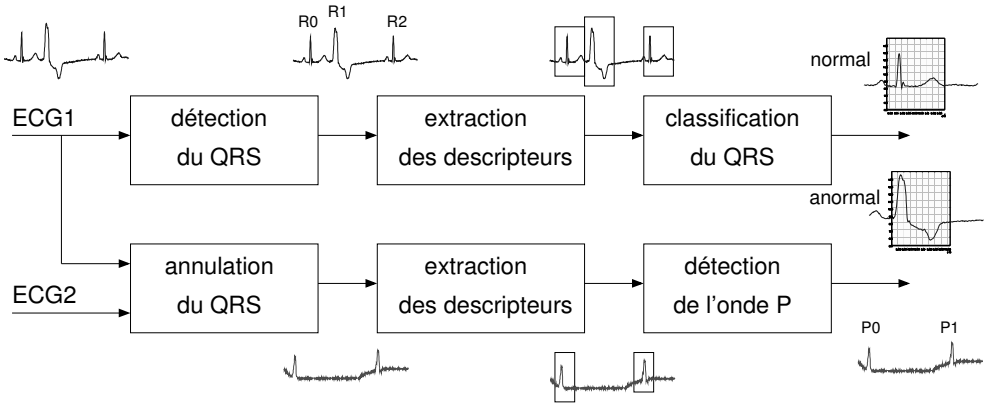


FIG. 2.11 – Schéma de l'abstraction temporelle de CALICOT.

Le module de détection estime les instants d'occurrence du complexe QRS, l'étape de classification partitionne les complexes détectés dans des classes connues. Enfin, la détection de l'onde P s'appuie sur l'association de trois niveaux. Le premier permet d'annuler le complexe QRS, le second extrait un vecteur descriptif chaque fois qu'une onde P est suspectée, le troisième enfin réalise la détection.

Détection du QRS L'algorithme choisi dans CALICOT pour la détection du complexe QRS est celui proposé dans (Gritzali, 1988). Le signal, une fois transformé, est seuillé en fonction de l'histogramme des valeurs pour trouver les dates d'occurrence des QRS.

Classification du QRS La classification du QRS comporte trois phases : le prétraitement, l'extraction de caractéristiques, et la classification. Le prétraitement filtre, aligne et normalise la forme du QRS. L'extraction de caractéristiques extrait les descripteurs du segment QRS considéré. La classification du QRS en `normal` ou `anormal` est ensuite effectuée par un réseau de neurones perceptron multicouche à deux sorties.

Détection de l'onde P La méthode de détection de l'onde P proposée est basée sur une annulation de l'ensemble QRS-T qui permet de rehausser les ondes P présentent sur l'ECG. La méthode est composée de trois étapes : l'annulation du QRS-T, la sélection des ondes P candidates, et la classification. L'annulation du QRS-T utilise deux voies d'ECG, l'une contenant l'ensemble P-QRS-T et l'autre, plus courante, contient principalement l'ensemble QRS-T. La méthode exploite le fait que les bandes spectrales des QRS-T des deux voies sont approximativement les mêmes. Les deux voies subissent une décomposition en ondelettes et après association des extrema, la suppression de ceux-ci efface du signal l'information concernant le QRS-T et la reconstruction donne un signal composé principalement de l'onde P. La décomposition en ondelettes est aussi utilisée pour éliminer le bruit. Une fois l'activité auriculaire rehaussée, toutes les ondes qui dépassent un seuil sont sélectionnées pour être candidates. La classification est ensuite effectuée par un réseau de neurones perceptron multicouche à deux sorties. L'onde P est ainsi classée comme étant `normal` ou comme étant un `artefact`.

Sorties de l'abstraction temporelle Les sorties de cette chaîne de traitements sont des flots d'événements symboliques. Deux formats sont utilisés selon que la sortie est à destination de l'apprentissage ou du module de reconnaissance. Les sorties sont codées dans le formalisme d'une base de faits PROLOG. Pour la partie apprentissage, les sorties ont la forme suivante :

- `p_wave(nom de l'onde, instant d'apparition, normal ou anormal, nom de l'onde précédente)`,
- `qrs_complex(nom de l'onde, instant d'apparition, normal ou anormal, nom de l'onde précédente).`

Ces formes de prédicat sont exploitées pour découvrir des relations cachées entre les ondes. Pour la reconnaissance d'arythmies, le flot d'événements est codé de la manière suivante :

- `instant d'apparition p_wave[type d'onde : normal ou anormal]`,
- `instant d'apparition qrs[type d'onde : normal ou anormal]`

De cette manière, les événements respectent le formalisme imposé par le reconnaisseur de chroniques.

2.3.4 Apprentissage des arythmies cardiaques

L'apprentissage des arythmies cardiaques permet d'obtenir les règles qui expriment les relations temporelles entre les ondes, autrement dit, les règles qui expriment les motifs des arythmies sous forme de chroniques. En effet, un motif d'arythmie peut être vu comme *un ensemble d'événements temporellement liés entre eux*, ce qui est la définition même d'une chronique (Cordier et Dousson, 2000). L'apprentissage des arythmies cardiaques est réalisé par programmation logique inductive (PLI), à partir d'exemples positifs et négatifs d'ECG d'arythmies annotés. L'apprentissage produit des règles de reconnaissance qui sont traduites en modèles de chroniques. Ainsi, à chaque arythmie correspond au moins un modèle de chronique. La PLI hérite des avantages respectifs de la programmation logique et de l'apprentissage inductif qui permet d'induire des hypothèses à partir d'exemples. Grâce à la grande expressivité de la logique du premier ordre, la PLI est capable d'induire des règles compréhensibles et explicables. Enfin, la PLI peut accomplir des tâches d'apprentissage complexes en employant les connaissances du domaine.

Apprentissage par PLI La PLI est fréquemment définie par :

- un ensemble E constitué d'exemples positifs E^+ et d'exemples négatifs E^-,
- une théorie initiale du domaine B (*Background Knowledge*),
- un langage de biais L qui définit les restrictions syntaxiques et sémantiques sur les hypothèses.

Ceci donné, le problème de PLI revient à chercher :

une hypothèse $H \subset L$ qui vérifie la condition de complétude : $B \wedge H \models E^+$
et la condition de consistance $B \wedge H \not\models E^-$, où $\models$ est la conséquence logique.

La condition de complétude garantit que tous les exemples positifs E^+ peuvent être déduits à partir de l'hypothèse induite H jointe à la connaissance initiale B. La condition de consistance permet de vérifier que l'on ne peut pas trouver d'exemples négatifs E^- à partir de H joint à B.

En pratique, les exemples utilisés sont souvent imparfaits et entachés de bruit. Il est alors impossible de trouver une hypothèse remplissant les deux conditions. En relâchant les conditions de complétude et de consistance, c'est à dire en faisant en sorte que certains exemples négatifs puissent être déduits de H et B et en permettant que certains exemples positifs ne soient pas déduits, la PLI peut traiter des données imparfaites. La figure 2.12 représente un exemple de la connaissance initiale B pour le logiciel d'apprentissage ICL (*Inductive Classification Logic*[4]) dans le cas d'un apprentissage d'arythmies à partir d'exemples d'ECG.

Le prédicat `previous` donne une définition récursive de la succession des éléments d'une liste chaînée. Le prédicat `qualify` définit comment une valeur symbolique peut être associée à un délai numérique selon le type d'onde impliquée. La connaissance dans B permet aussi de définir le vocabulaire des concepts et la sémantique des prédicats associés et de leurs relations aux exemples. Les prédicats `p_wav(P, Shape, R)` et `qrs(R,`

[4]http ://www.cs.kuleuven.ac.be/ wimv/ICL/

```
p_wav(P, D, Prec) :- p_wave(P, _, D, Prec).
qrs(R, D, Prec) :- qrs_complex(R, _, D, Prec).

previous(p_wave, P, P) :- p_wave(P, _, _, _).
previous(qrs, R, R) :- qrs(R, _, _, _).
previous(Type, W, R) :- ((p_wave(W, _, _, R1) ; qrs(W, _, _, R1)),
        previous(Type, R1, R).

qualify(W, D, short) :- (W = pp1 ; W = rr1), D < 700.
qualify(W, D, normal) :- (W = pp1 ; W = rr1), 700 =< D < 1100.
qualify(W, D, long) :- (W = pp1 ; W = rr1), 1100 =< D.
qualify(pr1, D, short) :- D < 100.
qualify(pr1, D, normal) :- 100 =< D < 140.
qualify(pr1, D, long) :- 140 =< D.

equal(X,Y) :- p_wav(X, _, _),qrs(Y, _, X), !, fail.
equal(X,X).
```

FIG. 2.12 – Extrait de la théorie initiale du domaine.

`Shape, P)`, donnent ainsi accès aux caractéristiques particulières des données.

Biais d'apprentissage Selon sa définition, une tâche d'apprentissage de type PLI peut être considérée comme un problème de recherche dans un espace d'hypothèses. L'efficacité de la recherche dépend fondamentalement de la construction et de la structure de l'espace des hypothèses ainsi que de l'algorithme de recherche. Pour permettre une exploration plus rapide de l'espace de recherche un biais déclaratif peut être utilisé. Celui-ci impose des contraintes ou des restrictions de type syntaxique ou sémantique. Par exemple le biais de la figure 2.13 représente la spécification d'un cycle cardiaque par le biais syntaxique DLAB.

```
 1-1:[
   len-len:[
       p_wav(P1, 1-1:[normal, abnormal], R0),
       qrs(R1, 1-1:[normal, abnormal], P1),
       0-len:[rr1(R0, R1, 1-1:[short, normal, long]),
         pr1(P1, R1, 1-1:[short, normal, long])]
   ],
   len-len:[
       p_wav(P1, 1-1:[normal, abnormal], R0),
       pp1(P0, P1, 1-1:[short, normal, long])
   ],
   len-len:[
       qrs(R1, 1-1:[normal, abnormal], R0),
       0-1:[rr1(R0, R1, 1-1:[short, normal, long])]
    ]
],
```

FIG. 2.13 – Spécification syntaxique d'un cycle cardiaque en DLAB

```
class(bigeminy) :- %[15, 0, 0, 0, 0]
   qrs(R0, abnormal, _),
   p_wav(P1, normal, R0), qrs(R1, normal, P1),
   qrs(R2, abnormal, R1), rr1(R1, R2, short).
class(bigeminy) :- %[5, 0, 0, 0, 0]
   p_wav(P0, normal, _), qrs(R0, normal, P0),
   p_wav(P1, normal, R0), qrs(R1, abnormal, P1),
   p_wav(P2, normal, R1), qrs(R2, normal, P2).
class(lbbb) :- %[0, 20, 0, 0, 0]
   p_wav(P0, normal, _), qrs(R0, abnormal, P0),
   p_wav(P1, normal, R0), qrs(R1, abnormal, P1),
   p_wav(P2, normal, R1), qrs(R2, abnormal, P2).
class(mobitzII) :- %[0, 0, 17, 0, 0]
   p_wav(P0, normal, _), qrs(R0, normal, P0),
   p_wav(P1, normal, R0), pp1(P0, P1, normal), equal(P1, R1),
   p_wav(P2, normal, R1), qrs(R2, normal, P2).
class(mobitzII) :- %[0, 0, 3, 0, 0]
   p_wav(P0, normal, _), qrs(R0, abnormal, P0),
   p_wav(P1, normal, R0), pp1(P0, P1, normal), equal(P1, R1),
   p_wav(P2, normal, R1), qrs(R2, abnormal, P2).
...
```

FIG. 2.14 – Règles apprises avec un biais imposant 3 cycles cardiaques et un cycle optionnel

La grammaire DLAB est basée sur l'expression `l-h :[el1,...,eln]` qui signifie : choisir de `l` à `h` des éléments de l'ensemble `[el1,...,eln]`. L'exemple de la figure 2.13 impose qu'un cycle cardiaque soit composé d'une des configurations suivantes.

- Une onde P suivie d'un QRS suivi des contraintes temporelles optionnelles (`pr1` et `rr1`. Cette spécification DLAB satisfait l'expression suivante : `p_wav(P1, normal, R0), qrs(R1, abnormal, P1), pr1(P1, R1, long)`.
- Une onde P seule, dans ce cas une contrainte temporelle de type `pp1` est obligatoire entre cette onde et la précédente,
- Un QRS seul, dans ce cas, une contrainte temporelle entre cette onde et la précédente est optionnelle.

Résultats d'apprentissage La figure 2.14 donne les règles obtenues avec trois arythmies et un bloc de branche : extrasystole ventriculaire (pvc), bigéminisme, mobitz de type II et bloc de branche gauche (lbbb). Une classe `normal` permet d'améliorer la séparation entre les rythmes pathologiques et les rythmes normaux.

2.3.5 Reconnaissance de chroniques

En pratique, le diagnostic clinique des troubles du rythme cardiaque s'appuie sur l'analyse de l'ECG et, plus précisément, sur l'analyse des ondes principales (onde P, complexe QRS) d'une part et les relations temporelles entre les ondes (intervalles RR, PR, PP) d'autre part. La reconnaissance des arythmies peut donc être considérée

	mobitzII	bigeminy	pvc	lbbb	normal	FP	Total
mobitzII	427	0	3	84	2	1	517
bigeminy	0	154	26	27	0	16	223
pvc	0	2	267	0	0	0	269
lbbb	0	0	1	1804	0	1124	2929
normal	0	0	2	4	2716	0	2722
NR	0	8	236	118	0	0	362
Total	427	164	535	2037	2718	1141	7022
Sensibilité	1	0.94	0.50	0.89	1.00	0	
Spécificité	0.99	0.99	1.00	0.77	1.00	1	

TAB. 2.1 – Matrice de confusion des reconnaissances d'arythmies obtenues avec le système CALICOT.

comme un problème de raisonnement temporel. La surveillance cardiaque dans un système de monitorage nécessite un diagnostic en temps réel afin de soigner les patients sans délai. C'est pourquoi le reconnaisseur de chronique CRS (*Chronicle Recognition System*) (Dousson, 1994) a été choisi pour effectuer cette reconnaissance en ligne.

CRS a été initialement développé pour le suivi et la reconnaissance d'évolutions particulières de systèmes dynamiques. En recevant en entrée un flot d'événements datés provenant des capteurs, CRS permet d'identifier, à la volée, tous les motifs s'appariant avec une situation décrite par un modèle de chronique. Un modèle de chronique contient un ensemble d'événements reliés par des contraintes temporelles qui restreignent l'intervalle des occurrences entre les dates des événements. Les règles apprises précédemment peuvent se transcrire assez facilement dans le formalisme de CRS.

Le prototype CALICOT a été testé sur quatre enregistrements appartenant à la base de données MIT-BIH Arrhythmia database (Mark et Moody, 1988). Ces enregistrements comprennent trois classes d'arythmies (PVC, mobitz de type II et bigéminisme), un bloc de branche gauche (lbbb) et la classe normale. Le tableau 2.3.5 présente les résultats obtenus dans (Carrault et coll., 2003) sous forme de matrice de confusion.

Les lignes représentent les détections et les colonnes les arythmies à reconnaître. NR donne le nombre d'arythmies non reconnues. CALICOT présente de bonnes performance de détection. On peut cependant noter un nombre important de `pvc` non reconnus et des confusions entre `bigéminisme` et `pvc` et entre `lbbb` et les exemples de bloc de branche droit non appris dans cette étude. Ces confusions sont dues à l'annotation trop simple des QRS (normal/anormal). Elles témoignent aussi du fait que l'apprentissage est effectué en monde clos. Les arythmies non apprises durant l'apprentissage sont susceptibles d'être couvertes par des règles. Il faudrait donc un nombre plus important d'arythmies pour spécialiser les règles. Enfin, les tests ont été réalisés sur des enregistrements peu ou pas bruités. Le test en situation bruité doit aussi être effectué.

2.3.6 Évolutions envisageables

Calicot présente des performances de reconnaissance satisfaisantes sur les exemples d'arythmies et de signaux traités (Carrault et coll., 2003). Cependant, le système s'avère

sensible aux erreurs de l'abstraction temporelle qui peuvent perturber la reconnaissance des arythmies et plusieurs évolutions sont envisageables.

- Le nombre d'arythmies apprises est assez restreint, le système doit être évalué sur un ensemble d'exemples d'arythmies plus complet.
- Le langage de description du signal ECG et plus particulièrement l'attribut de forme du complexe QRS doit être étendu pour améliorer la reconnaissance.
- Les performances de l'abstraction temporelle doivent être améliorées par un paramétrage adéquat de la chaîne de traitements car la reconnaissance des arythmies dépend fortement de la performance du module de traitement du signal.
- Le système doit prendre en compte l'utilisation d'ECG multivoie pour améliorer la performance du système de reconnaissance. Comme il est illusoire d'atteindre une abstraction temporelle sans aucune fausse alarme dans un environnement bruité, l'utilisation de plusieurs sources permettrait de fiabiliser la détection. Cette démarche semble logique dans le sens où les USIC possèdent plusieurs types de matériel d'acquisition de signaux physiologiques, notamment deux voies d'ECG et la mesure de la pression artérielle (cf. 1.2.2).

CALICOT se concentre surtout sur les étapes d'extraction de caractéristiques et de diagnostic médical et n'inclut pas l'acquisition des signaux et la recommandation de thérapies. Si l'acquisition n'est pas du domaine de la recherche et pourrait être incluse dans le système, la recommandation de thérapies peut être aussi envisagée. Cependant, dans le domaine des arythmies, mise à part la thérapie d'urgence telle que la défibrillation, l'élaboration d'une thérapie demande des examens complémentaires, tels que les IRM, les tests de laboratoire, etc. Ce travail se centre donc surtout sur la reconnaissance des arythmies, qui serait déjà d'une grande aide dans les USIC.

2.4 Conclusion

La lecture attentive de la littérature concernant le pilotage d'algorithmes a permis de faire ressortir les concepts et les contraintes inhérents à notre domaine d'application. Il a également été vu que CALICOT est un système dédié au monitorage cardiaque qui utilise des algorithmes de traitement du signal pour l'abstraction des données et un raisonnement temporel qui peuvent être pilotés. L'analyse a permis de montrer que, l'ensemble étant très lié, le pilotage d'algorithmes doit s'effectuer non seulement au niveau des algorithmes d'abstraction mais aussi au niveau du raisonnement temporel (diagnostic médical). Les solutions proposées en pilotage d'algorithmes permettent surtout de piloter des modules de bas niveau tels que les algorithmes de traitement du signal mais ne prennent pas en compte les modules de haut niveau tel que le raisonnement temporel. De plus, les algorithmes utilisés sont très spécifiques au domaine et leur pilotage nécessite des informations qui dépendent du domaine d'application. Le pilotage efficace de CALICOT ne pourra donc se faire qu'en fonction de connaissances dépendantes et indépendantes du domaine d'application. Le chapitre suivant présente l'intégration du pilotage dans CALICOT qui a menée au système IP-CALICOT (*Integrated Piloting and Cardiac Arrhythmias Learning for Intelligent Classification of On-line*

Tracks).

Chapitre 3

Intégration du pilotage dans CALICOT : le système IP-CALICOT

3.1 Introduction

Le système de monitorage CALICOT présente certes des résultats satisfaisants en terme de reconnaissance d'arythmies mais ceux-ci se dégradent dès lors que l'on travaille avec des données bruitées. Cette faible résistance au bruit est un handicap important car le milieu hospitalier étant hostile aux systèmes électriques et informatiques (interférences avec d'autres équipements, manipulation des câbles, etc.), les données médicales sont souvent très bruitées. Un système de monitorage intelligent réaliste doit donc pouvoir assurer une surveillance minimale même dans ces conditions hostiles. Ceci explique, comme vu au 1.3, que les systèmes de monitorage intelligent incluent une adaptation aux données à certains niveaux de la chaîne de traitements (sélection de sources, raisonnement à différents niveaux de granularité, etc.). Cependant, peu de systèmes de monitorage utilisent les diagnostics et informations courantes pour adapter les algorithmes de la chaîne de traitements.

Le pilotage d'algorithmes semble donc être une alternative pertinente pour amener une adaptation dynamique et automatique à tous les niveaux de la chaîne de traitements (*cf.* 2.2) en utilisant des informations courantes de haut et bas niveau. L'intégration d'un pilote dans CALICOT doit permettre une plus grande flexibilité et une meilleure réactivité du système pour affronter des situations difficiles. Ces objectifs ont conduit à la conception du système IP-CALICOT (*Integrated Piloting and Cardiac Arrhythmias Learning for Intelligent Classification of On-line Tracks*). La solution du pilotage, plutôt qu'une simple adaptation, a été retenue car elle apporte plus de bénéfices :

- la centralisation des actions par le pilote permet d'ajouter facilement un nouveau module, surtout s'il influence d'autres modules de la chaîne de traitements ;
- la structuration du programme imposée par le pilotage, autorise les ajouts et modifications de manière simple ;
- l'ajout aisé de modules permet de gérer des sources de données supplémentaires ;
- la séparation des règles de pilotage et des algorithmes par le système rend la

connaissance facilement accessible et modifiable ;
- l'intervention du pilote à tous les niveaux de la chaîne de traitements apporte une grande réactivité du système, notamment en modifiant les algorithmes ;
- enfin, les algorithmes deviennent facilement réutilisables pour d'autres applications une fois saisis dans la base.

Contrairement à certaines des approches présentées, le pilotage d'algorithmes réalisé ne contient pas de cycle d'évaluation des résultats (*cf.* 2.2.5). La réaction du système devant être très rapide, les algorithmes sont choisis en fonction de connaissances *a priori* stockées dans le *contexte courant*. Chaque algorithme aura préalablement été testé sur de multiples scénarios pour dériver les règles de pilotage qui s'y rapportent.

Ce chapitre présente l'architecture globale du nouveau système IP-CALICOT en section 3.2. Les différents niveaux de pilotage ainsi que les différents composants sont détaillés. Le pilotage du système de monitorage implique que les informations nécessaires au diagnostic médical sont susceptibles de varier au cours du temps. Pour assurer un diagnostic quel que soit le niveau d'abstraction des données disponibles, des motifs d'arythmies sont appris selon différents langages de description. Cet apprentissage est présenté en section 3.3

3.2 Présentation générale de IP-CALICOT

Cette section présente le nouveau système IP-CALICOT. L'architecture globale du système, exposée en section 3.2.1, est répartie pour tenir compte du cadre particulier du monitorage de patient en USIC. L'architecture détaillée du système piloté est présentée en section 3.2.2. Le pilotage s'appuie sur l'analyse du contexte courant. Ce contexte est régulièrement remis à jour par des analyseurs de contexte décrits en section 3.2.3. Grâce à ce contexte, le pilote agit à trois niveaux sur le système. L'architecture du pilote et les niveaux de pilotage sont introduits en section 3.2.4. Enfin, le système utilise différentes bases de connaissances pour effectuer les traitements. Ces bases sont présentées en section 3.2.5.

3.2.1 Architecture globale pour le monitorage en USIC

La figure 3.1 représente l'architecture globale du système IP-CALICOT pour le monitorage de patient en USIC.

Les différents modules du système sont répartis dans un réseau qui permet l'échange des données entre l'acquisition des signaux (1), le traitement des données (2) et l'affichage (3). Ce type de répartition, déjà utilisé dans (Dawant et coll., 1993) et (Moret-Bonillo et coll., 1998), permet une plus grande flexibilité d'installation et d'utilisation du système. En effet, en USIC, l'acquisition des signes vitaux est réalisée par des systèmes dédiés et l'affichage des signaux monitorés et des alarmes se fait au chevet du patient et dans l'unité centrale (*cf.* 1.2.2). Les systèmes de monitorage doivent donc s'insérer dans cette configuration en permettant un affichage multisite et une connexion souple des appareils de mesure. De plus, cette architecture permet non seulement de

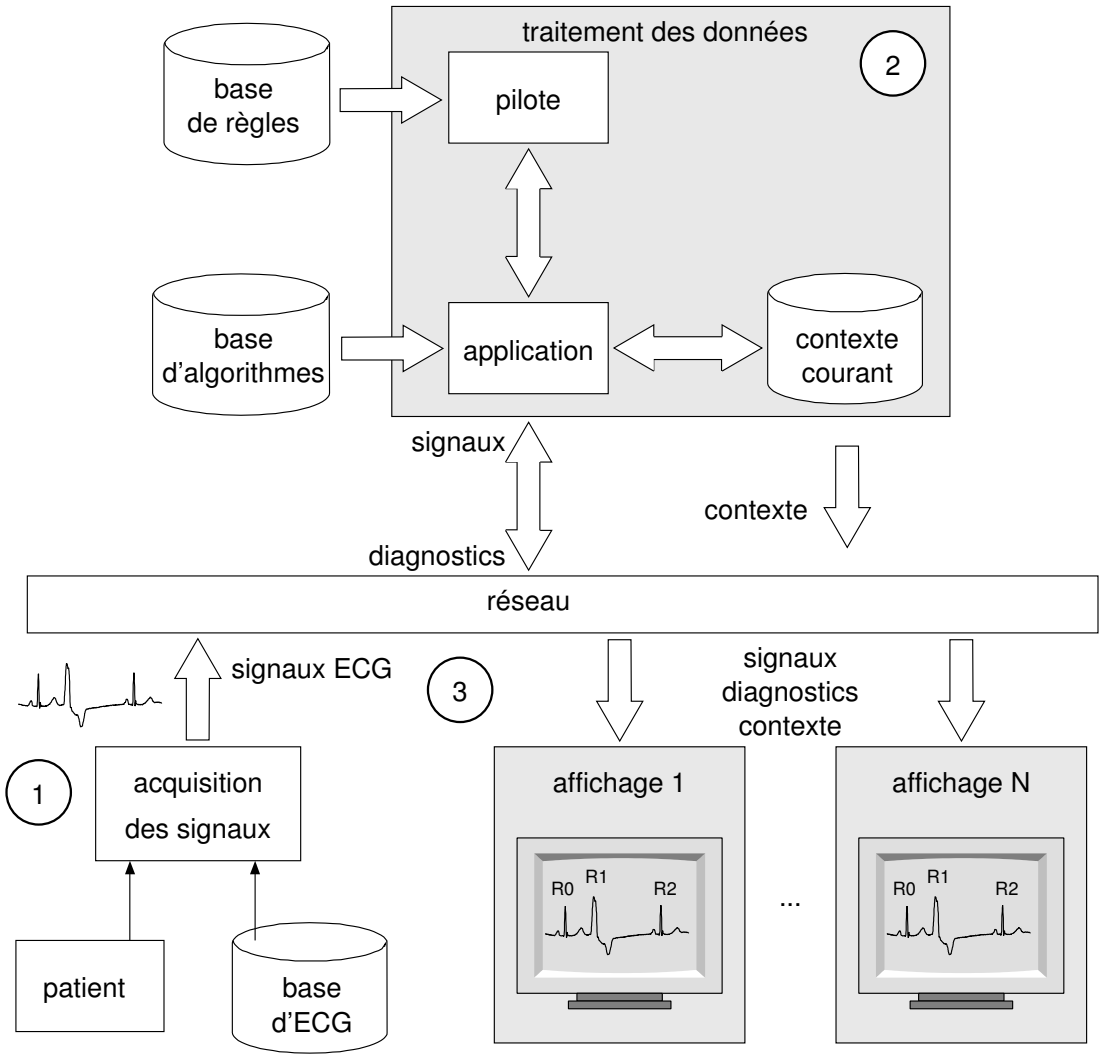


FIG. 3.1 – Architecture globale du système piloté.

fonctionner au niveau du réseau de l'hôpital, mais aussi de s'adapter à des réseaux distants, comme dans le cas du télémonitorage.

Dans IP-CALICOT, l'acquisition des signaux (1) est simulée à partir des enregistrements ECG. Elle permet d'étudier l'acquisition en temps réel (c.-à-d. à la vitesse d'échantillonnage) et en temps machine (c.-à-d. à la vitesse de calcul de la machine).

Le traitement des données (2) est effectué par le système piloté présenté dans la section suivante. Une application de traitement du signal est pilotée grâce à un système expert qui a le rôle du pilote. Les éléments de l'application s'échangent des signaux par des connexions directes et des informations ponctuelles via le contexte courant. À chaque cycle de traitement, l'application fait appel au pilote et lui envoie les informa-

tions contextuelles. Grâce à ces informations et à une base de règles de pilotage, le pilote retourne les actions à effectuer sur l'application. L'application modifie ensuite sa structure en réglant ses paramètres et en remplaçant certains algorithmes de traitement du signal. Les algorithmes sont pris dans la base d'algorithmes selon les recommandations du pilote.

L'affichage (3) peut être effectué sur N stations distantes. Chaque station récupère d'une part, les signaux envoyés par l'acquisition et, d'autre part, les diagnostics (diagnostics médicaux et alarmes) et le contexte émis par le traitement des données. La figure 3.2 représente l'affichage des signaux, des arythmies reconnues et des événements détectés sur les signaux.

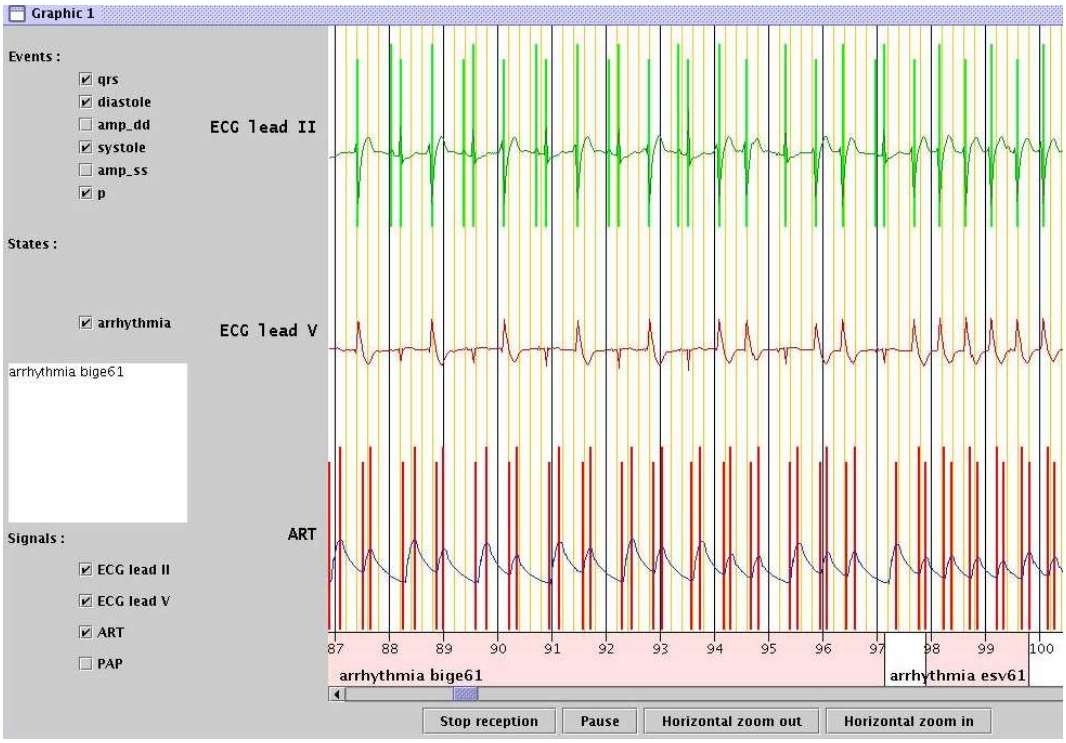


FIG. 3.2 – Affichage des signaux dans le système IP-CALICOT.

3.2.2 Architecture du système

La structure initiale de CALICOT, représentée Figure 3.3, autorise un pilotage à trois niveaux : au niveau du *diagnostic médical* en choisissant les modèles de chroniques les plus adaptés au contexte, au niveau des *tâches d'abstraction du signal* en activant ou désactivant des traitements, et au niveau des *algorithmes de traitement du signal* en choisissant les algorithmes les plus adaptés au contexte. Ces différentes manières de piloter ont conduit à l'architecture de IP-CALICOT représentée par la figure 3.4.

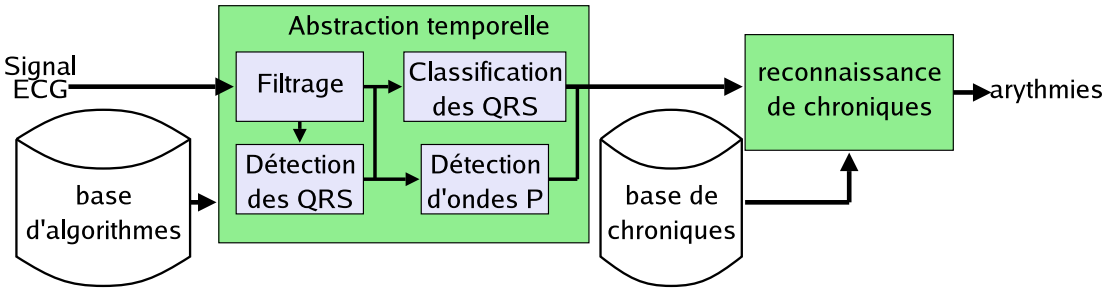


FIG. 3.3 – Architecture du système CALICOT.

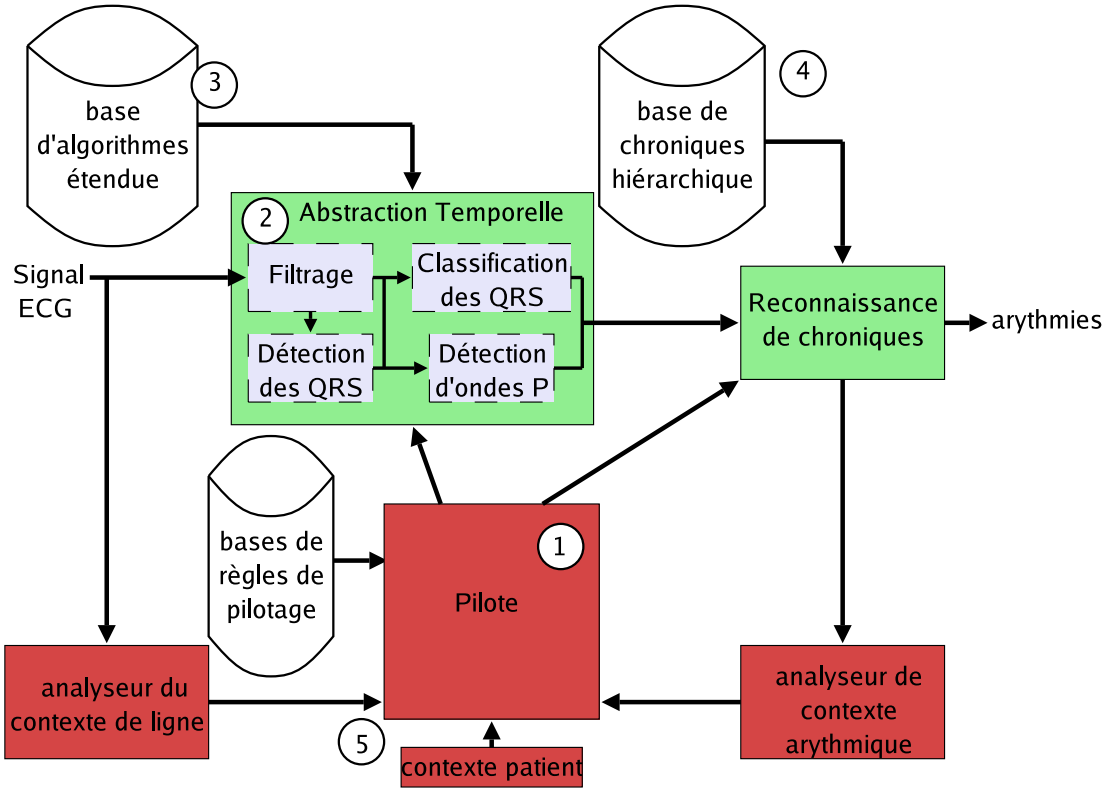


FIG. 3.4 – Architecture du système IP-CALICOT.

Cette architecture complète celle de CALICOT. Le pilote (1) et les éléments nécessaires à son fonctionnement s'insèrent dans la chaîne de traitements pour effectuer les trois types de pilotage.

Le pilotage de la reconnaissance d'arythmies Une arythmie peut être diagnostiquée à partir de différentes caractéristiques extraites de l'ECG. Dans CALICOT, toutes les caractéristiques de l'ECG sont constamment extraites (dates d'occurrence de QRS, formes de QRS, et ondes P) et envoyées à la reconnaissance de chroniques. Cependant, dans certains contextes, un nombre réduit de caractéristiques du signal peut être suffisant pour reconnaître une arythmie. Par exemple, lorsque l'on se trouve en présence d'un rythme cardiaque rapide avec des QRS rapprochés, deux arythmies sont fortement probables : une tachycardie ventriculaire (pouvant dégénérer en fibrillation mortelle) et une tachycardie supraventriculaire (moins dangereuse). Lorsque toutes les caractéristiques de l'ECG sont extraites (avec une abstraction temporelle très précise), les ondes P sont recherchées pour les discriminer. Mais, l'analyse des formes de QRS, qui est beaucoup moins coûteuse en terme de ressources de calcul, est suffisante pour discriminer les deux arythmies. Par ailleurs, certaines caractéristiques ne peuvent pas être extraites dans certains contextes. Par exemple, s'il y a présence de bruit sur la ligne, l'extraction de l'onde P peut être très perturbée et accompagnée d'erreurs. Il est alors préférable d'ignorer l'onde P pour réaliser le diagnostic jusqu'à ce que le niveau de bruit redevienne acceptable. Pour représenter ces différentes stratégies de diagnostic, un ensemble de modèles de chroniques (c.-à-d. modèles d'arythmies) est appris en fonction d'une hiérarchie de langages de description de l'ECG et stocké dans la base de modèles de chroniques hiérarchiques (4). Ainsi, le pilotage de la reconnaissance d'arythmies consiste à choisir le langage de description des modèles de chroniques le plus adapté au contexte courant.

Pilotage des tâches L'abstraction temporelle de CALICOT peut être décomposée en quatre tâches principales (2) :

- *Filtering* : la tâche de filtrage sépare au maximum l'ECG du bruit parasite,
- *QRSDetection* : la tâche de détection des QRS donne leur date d'occurrence,
- *QRSClassification* : la tâche de classification des QRS les étiquette,
- *PWaveDetection* : la tâche de détection d'ondes P fournit leur date d'occurrence.

Dans certains contextes, certaines tâches ne peuvent pas être accomplies correctement. Par exemple, lorsque la ligne est trop bruitée, la détection de l'onde P est vouée à l'échec et entraîne beaucoup d'erreurs. Dans ce contexte, cette tâche est pénalisante car elle fournit beaucoup de fausses informations à la reconnaissance de chroniques. De même, lorsque le pilotage de la reconnaissance d'arythmies choisit des modèles de chroniques qui n'ont pas besoin de certaines caractéristiques, les tâches associées aux caractéristiques inutiles peuvent être désactivées. Pour fonder le diagnostic sur des informations fiables et économiser des ressources, les tâches sont activées et désactivées en fonction du contexte.

Pilotage des algorithmes de traitement du signal Dans l'abstraction temporelle, chaque tâche est réalisée par un algorithme de traitement du signal. Or, pour réaliser chaque tâche, on peut choisir entre plusieurs algorithmes plus ou moins performants en fonction du contexte. En effet, chaque algorithme présente des performances qui varient en fonction du contexte d'utilisation. Ainsi, le pilotage de l'abstraction temporelle choisit les algorithmes dans une base (3) et leurs paramètres les plus adaptés aux tâches à accomplir en fonction du contexte courant.

3.2.3 Analyse du contexte courant

Un pilotage efficace dépend de l'analyse précise du contexte courant. Le contexte courant est composé de trois sous-contextes : le *contexte de ligne*, le *contexte arythmique*, et le *contexte patient*. Le contexte patient est considéré comme statique alors que le contexte de ligne et le contexte arythmique sont dynamiques et remis à jour régulièrement par deux analyseurs.

Contexte patient Le contexte patient est utilisé pour paramétrer certains algorithmes. Il est constitué des données patient (âge, rythme ECG de base, etc.) et pourrait intégrer les traitements suivis, l'historique ainsi que des résultats de laboratoire qui sont des informations ponctuelles considérées comme statiques.

Contexte de ligne Le contexte de ligne est constitué du niveau et du type de bruit présent sur la ligne à un instant donné. Dans cette étude, trois types de bruit typiques rencontrés sur les ECG sont considérés. Il s'agit : des ondulations de ligne de base, des bruits musculaires et des mauvais contacts peau-électrode. L'analyseur de contexte de ligne est placé au début de la chaîne, il communique ainsi rapidement au pilote le contexte de ligne courant. Le pilote peut alors modifier l'abstraction temporelle avant que l'ECG ne soit traité.

Contexte arythmique L'analyseur de contexte arythmique utilise les hypothèses du module de reconnaissance de chroniques afin d'établir une liste des arythmies les plus susceptibles d'apparaître. Cette liste constitue le contexte arythmique. Elle permet au pilote de faire des hypothèses sur les formes d'ondes de l'ECG et les arythmies que l'abstraction temporelle doit traiter.

3.2.4 Pilote

La structure du pilote est représentée par la figure 3.5.

Le pilote est composé de trois moteurs d'inférence, qui déduisent les actions à effectuer sur la chaîne de traitements pour les trois niveaux de pilotage. Le *gestionnaire de contexte* déduit les informations nécessaires aux moteurs à partir des variables du contexte.

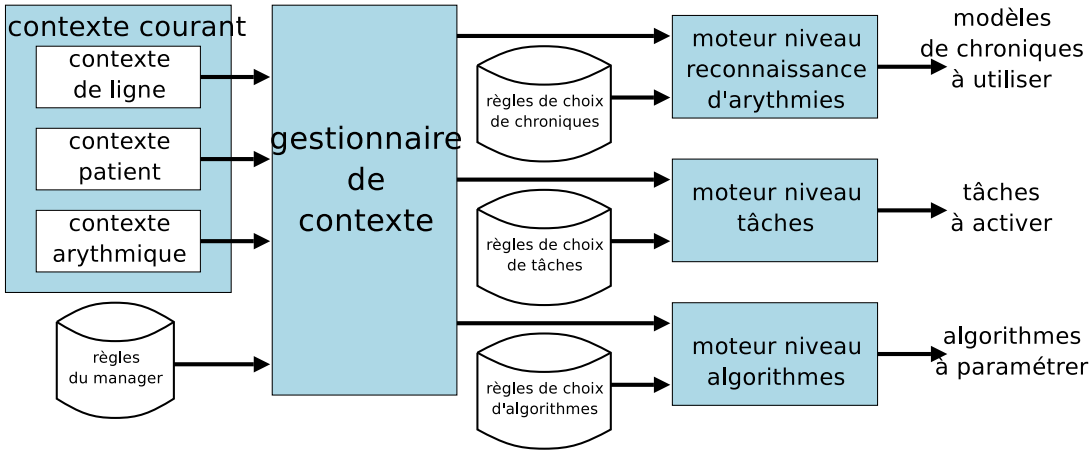


FIG. 3.5 – Architecture du pilote.

3.2.4.1 Gestionnaire de contexte

Le rôle du gestionnaire de contexte est d'instancier et de mettre à jour les variables utiles pour le pilotage à partir des informations transmises par les analyseurs de contexte. Par exemple, la tâche de classification du QRS est activable uniquement si la ligne n'est pas trop bruitée, ce qui s'exprime par la règle suivante :

Si *niveau de bruit* $\geq 0dB$ **alors**
QRSClassification est activable
Fin si

De la même manière, à partir du contexte arythmique, le gestionnaire de contexte peut déduire les principales formes de QRS et les arythmies les plus susceptibles d'être traitées par l'abstraction temporelle. En ce sens, le gestionnaire de contexte met à jour une base de faits comme dans un système expert. La connaissance du gestionnaire de contexte est représentée par des règles de production stockées dans la base de règles du gestionnaire.

3.2.4.2 Moteurs d'inférence

Le système est piloté à trois niveaux : au niveau de la reconnaissance d'arythmies, au niveau des tâches, et au niveau des algorithmes. À partir des informations transmises par le gestionnaire de contexte, les moteurs infèrent les actions à appliquer au système. Leurs règles de pilotage sont regroupées dans : les règles de choix de modèles de chroniques, les règles de choix des tâches, et les règles de choix d'algorithmes. Ces règles proviennent majoritairement d'expertises exceptées les règles de choix d'algorithmes qui proviennent d'une analyse statistique présentée au chapitre 4.

Règles de choix de modèles de chronique La reconnaissance de chroniques doit adapter le niveau d'abstraction au contexte courant. Par exemple, si toutes les tâches sont activées, alors la reconnaissance de chroniques doit utiliser la définition des modèles de chroniques *exper5* qui sont décrits dans le langage d'abstraction le plus précis. Cet exemple est représenté par la règle qui suit :

soit T l'ensemble des tâches
Si $\forall t \in T$, *t est activée* **alors**
  choisir exper5
Fin si

Règles d'activation des tâches L'abstraction temporelle doit activer les tâches selon les besoins et la possibilité de les effectuer sans erreur. Pour activer une tâche t il faut qu'elle soit nécessaire au diagnostic médical et qu'elle soit activable au vu du bruit de ligne. Ceci se traduit par la règle suivante :

soit T l'ensemble des tâches
Si $t \in T \wedge t$ *est nécessaire* $\wedge t$ *est activable* **alors**
  activer t
Fin si

Règles de choix d'algorithmes Les règles de choix d'algorithmes déterminent quel est l'algorithme le plus adapté à la tâche à accomplir. Par exemple, si la tâche *QRS-Detection* est activée, alors il est nécessaire de choisir l'algorithme le plus adapté au contexte courant. Celui-ci est choisi en fonction du contexte de ligne et du contexte arythmique comme dans la règle suivante :

Si *contexte ligne = sans bruit* $\wedge$ *contexte arythmique = pacemaker* **alors**
  choisir l'algorithme de Pan et Tompkins
Fin si

Cette règle permet de choisir l'algorithme de détection de QRS de Pan et Tompkins lorsque la ligne n'est pas bruitée et que le contexte arythmique informe que le rythme est un rythme de pacemaker. L'acquisition de ces règles est un problème particulièrement difficile et fait l'objet du chapitre 4.

3.2.5 Bases de connaissances

Le pilote intervient sur le système grâce à trois bases de connaissances : la base de règles de pilotage présentée précédemment, une base d'algorithmes et une base de modèles de chroniques.

Base d'algorithmes La base d'algorithmes (3) contient tous les algorithmes utilisés dans CALICOT pour la réalisation des tâches de l'abstraction temporelle : *Filtering*, *QRSDetection*, *QRSClassification* et *PWaveDetection*. Par exemple, pour la tâche *QRSDetection* sept algorithmes de la littérature sont pilotés. Les algorithmes utilisés sont présentés au chapitre 4.

Base de chroniques Le pilotage de la reconnaissance d'arythmies consiste à choisir les modèles de chroniques qui correspondent au langage de description courant de l'ECG. Lorsqu'une tâche de l'abstraction temporelle est désactivée, l'ECG est décrit de manière moins précise et la reconnaissance d'arythmies doit utiliser des modèles qui ne dépendent pas des informations manquantes. Par exemple, si la tâche *PWaveDetection* est inactive, la reconnaissance d'arythmies doit fonctionner avec des modèles de chroniques qui n'incluent pas l'onde P. Pour représenter les différents niveaux de détail, une hiérarchie de modèles de chroniques (4) a été apprise à partir d'exemples exprimés dans plusieurs langages de description. Cet apprentissage est détaillé en section 3.3.

3.3 Apprentissage des modèles de chroniques hiérarchiques

3.3.1 Introduction

Le pilotage permet au diagnostic de fonctionner avec plusieurs langages de description. Ainsi, si certaines tâches de l'abstraction sont désactivées par le pilote, la reconnaissance d'arythmies peut tout de même fonctionner avec un nombre réduit d'informations. Dans IP-CALICOT, ce fonctionnement est possible si la reconnaissance d'arythmies possède des modèles de chroniques décrits dans chacun de ces langages. Le but de cette section est donc de présenter l'apprentissage des arythmies cardiaques effectué par programmation logique inductive (PLI) à partir d'exemples d'électrocardiogrammes décrits dans différents langages de description.

Dans un système de monitorage, on tente généralement de caractériser au maximum le signal pour inférer un diagnostic. L'ECG est ainsi décrit dans un langage précis basé sur la caractérisation des complexes QRS et des ondes P qui constitue la base de l'interprétation des arythmies (*cf.* 1.2.5). Cependant, utiliser un système basé uniquement sur un langage riche pour l'apprentissage peut être problématique à plusieurs niveaux.

- Un langage riche nécessite de détecter beaucoup d'attributs sur le signal, or l'extraction de certains attributs peut être délicate voire pénalisante dans un contexte bruité (plus l'extraction est faussée plus le diagnostic médical est erroné).
- Plus le langage de description sera riche, moins l'apprentissage sera une généralisation des exemples, entraînant ainsi une perte de généricité.
- L'extraction de plusieurs attributs nécessite une puissance et un temps de calcul non négligeables qui peuvent rendre difficile une application temps réel.
- Enfin, certaines arythmies ne nécessitent pas un langage riche pour être caractérisées.

Pour résoudre ces problèmes, IP-Calicot est capable de modifier son langage de description en ligne selon le contexte courant. Par exemple, si on cherche à discriminer des arythmies qui ne nécessitent pas beaucoup d'attributs, alors le langage de description sera plus abstrait, ce qui entraînera moins de calcul pour la description de l'ECG. De la même façon, si l'extraction de certains attributs ne peut être effectuée (trop de bruit ou pas assez de ressources), le diagnostic médical utilisera un langage plus abstrait jusqu'au retour à une situation normale.

La section 3.3.2 présente les exemples utilisés pour l'apprentissage et les langages de description associés. La section 3.3.3 présente les résultats de l'apprentissage hiérarchique.

3.3.2 Données d'apprentissage

3.3.2.1 Base d'exemples d'arythmies pour l'apprentissage

Conformément aux évolutions à apporter à Calicot, déjà exprimées dans (Wang, 2002), le nombre d'arythmies traitées a été étendu de trois à cinq. En effet, Calicot était capable de reconnaître :

- un rythme `normal` (qui n'est pas une arythmie),
- un rythme normal accompagné d'un bloc de branche gauche (`lbbb`), qui n'est pas véritablement une arythmie mais un trouble de la conduction ventriculaire,
- un `bigéminisme` ventriculaire, qui est une succession de battements normaux et d'extrasystoles,
- un mobitz de type II (`mobitzII`), qui est un blocage intermittent de la commande de dépolarisation au niveau du nœud auriculo-ventriculaire,
- une contraction prématurée des ventricules (`pvc`), qui est représentée par une extrasystole isolée dans un rythme normal.

Dans cette étude deux nouvelles arythmies ont été ajoutées :

- la `tachycardie ventriculaire` (vt, *ventricular tachycardia*), qui est un rythme rapide à commande ventriculaire qui, si elle n'est pas traitée rapidement, peut dégénérer en fibrillation ventriculaire,
- le `doublet` ventriculaire, qui est un couple d'extrasystoles isolé dans un rythme normal.

La description de ces arythmies peut être trouvée au chapitre 1.Les six classes de rythme sont apprises par PLI sur 109 exemples extraits des enregistrements de la base d'arythmies MIT-BIH (Mark et Moody, 1988) :

- 20 exemples pour la classe `bigeminy` : 106, 119, 213, 223 ;
- 13 exemples pour la classe `doublet` : 214, 223 ;
- 20 exemples pour la classe `mobitzII` : 231 ;
- 25 exemples pour la classe `normal` : 100, 101, 117, 121, 122 ;
- 20 exemples pour la classe `pvc` : 106, 119, 214, 223 ;
- 11 exemples pour la classe `vt` : 203, 205, 223 ;

Pour chaque exemple, les ondes QRS et P sont étiquetées. La méthode d'apprentissage est la même que celle décrite dans (Wang, 2002) pour le système CALICOT.

3.3.2.2 Langages de description des modèles de chroniques

Une hiérarchie de modèles de chroniques a été apprise, sur les exemples décrits selon 6 niveaux de langage de description. Deux événements importants de l'ECG sont pris en compte : le QRS et l'onde P. La date d'occurrence de ces attributs permet d'exprimer des intervalles entre les ondes. De plus, le QRS peut être classé en fonction de la forme de l'onde. En effet, la forme du QRS visible sur l'ECG est très informative sur l'origine de la commande de contraction. Dans CALICOT, les complexes QRS étaient classés en `normal` ou `abnormal`, selon que la forme du QRS appartenait à un rythme normal ou non. Cependant, les QRS appartenant à des blocs de branche étaient classés comme `abnormal`, une extrasystole qui, elle aussi, était classée `abnormal` ne pouvait donc être distinguée au milieu de blocs de branche. De même, les confusions entre `lbbb` et `rbbb` et les erreurs de classification de `pvc` dans des rythmes de blocs de branche sont inévitables avec ce type de classification trop simple.

Pour éviter ces erreurs et prendre le contexte patient en compte, une nouvelle classification en `basic` ou `nonbasic` permet de bien faire la différence entre les rythmes établis (`normal`, `lbbb` et `rbbb`) et les extrasystoles (`pvc`). Un complexe QRS appartenant au rythme de base (normal ou bloc de branche) est classé comme `basic` tandis qu'un complexe n'appartenant pas à ce rythme est classé comme `nonbasic`. Cette approche permet d'être beaucoup précis sur les arythmies reconnues sur l'ECG du patient et surtout de prendre plus en compte le contexte patient, ce qui représente une véritable avancée dans les systèmes de monitorage intelligent. Seule la déviation par rapport au rythme de base est prise en compte pour reconnaître les arythmies. Les informations déjà connues ou indépendantes des arythmies telles que les blocs de branche, ne perturbent plus la reconnaissance. Les rythmes normaux et les blocs sont maintenant regroupés sous le même nom de rythme `normal` mais les bigéminismes et les extrasystoles peuvent maintenant être reconnus dans un rythme de blocs de branche. Le regroupement des blocs de branches déjà installés du patient et des rythmes normaux n'est pas gênant dans cette approche car les blocs ne sont pas, à proprement parler, des arythmies même s'ils ont une origine pathologique.

La figure 3.6 présente la hiérarchie de langages de description utilisée pour l'apprentissage.

Le langage $L1$ est le plus abstrait, il ne prend en compte que les occurrences de QRS. Le langage $L2$ permet d'exprimer la forme du QRS. Celle-ci peut appartenir à un rythme de base ou non. Le langage $L2'$ permet d'exprimer en plus l'extrasystole notée `pvc`. Le langage $L3$ prend en compte les occurrences de QRS et les occurrences d'onde P. Le langage $L4$ (resp. $L4'$) ajoute les occurrences d'onde P au langage $L2$ (resp. $L2'$).

L'apprentissage effectué sur les exemples décrits par ces différents langages de description a conduit aux modèles de chroniques suivants :

- exper1 décrit par $L1$,
- exper2 décrit par $L2$,

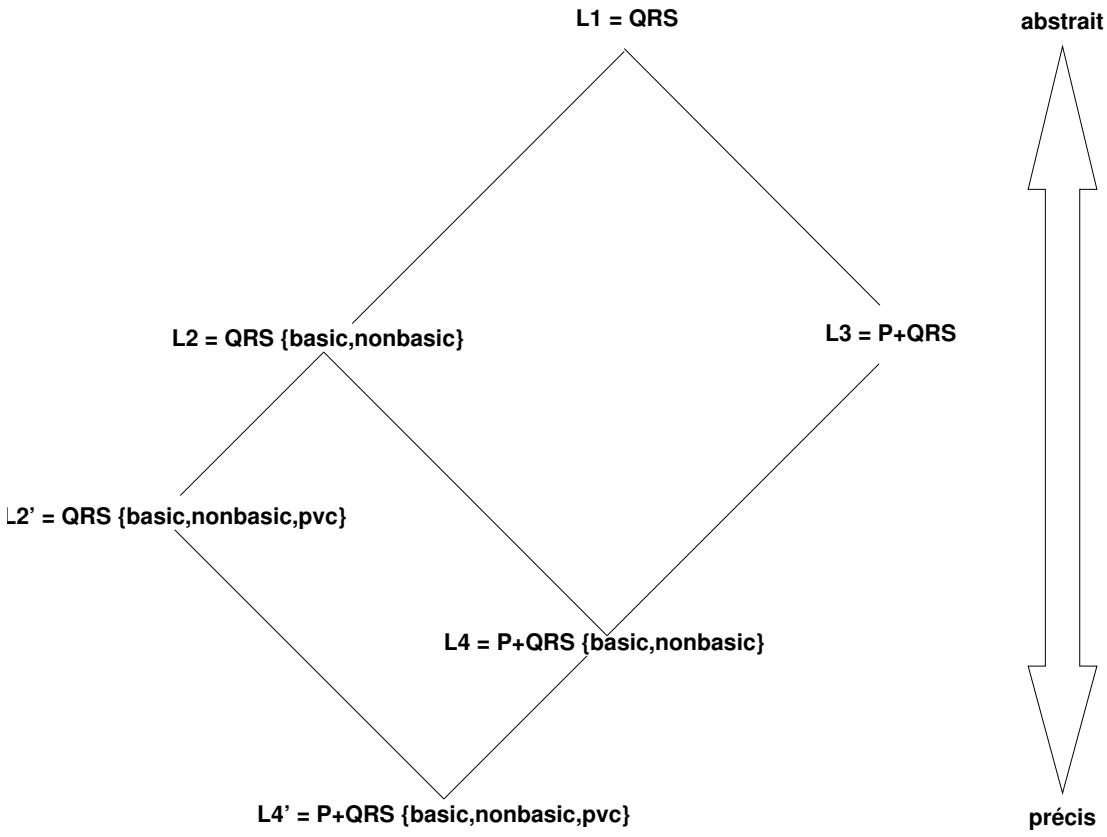


FIG. 3.6 – Langages de description de l'ECG utilisés dans IP-CALICOT

- exper3 décrit par $L2'$,
- exper4 décrit par $L3$,
- exper5 décrit par $L4$,
- exper6 décrit par $L4'$.

3.3.3 Résultats de l'apprentissage

Cette section présente les résultats d'apprentissage des règles de reconnaissance d'arythmies. Les règles sont représentées dans le formalisme d'une base de faits PROLOG. L'ensemble des règles est présenté dans l'annexe B. Chaque tête de clause représente un modèle de classe d'arythmie suivi du nombre d'exemples couverts par la règle. Dans toutes les règles suivantes, les couvertures de classes sont représentées comme suit : `[[exemples couverts],[exemples non couverts]]` en respectant l'ordre suivant `[bigeminy,mobitzII,normal,pvc,doublet,vt]`. Par exemple

```
class(bigeminy):- [[15,0,0,0,0,0], [5,20,25,20,13,11]]
```

signifie que 15 exemples de la classe `bigeminy` sont couverts par cette règle et qu'aucun exemple des autres classes n'est couvert. 5 exemples de bigéminisme ne sont pas couverts

par cette règle.

3.3.3.1 Exper1

L'apprentissage utilisant le langage le plus abstrait $L1$ conduit à des règles assez complexes représentées Figure 3.7.

- Un bigéminisme peut être caractérisé par deux règles. La première présente une séquence composée de deux paires de QRS avec un intervalle RR court (`rr1(R2, R3,short)`) séparées par une séquence avec un intervalle RR normal. La deuxième ne met en jeu que des QRS à intervalles courts et couvre 7 exemples de tachycardie qui sont aussi des rythmes rapides.
- Un `mobitzII` est caractérisé par des QRS à intervalles longs qui traduisent la commande de dépolarisation bloquée au sein du nœud AV.
- Dans un rythme normal, tous les QRS sont normaux espacés à intervalles normaux. Afin de distinguer la classe `normal` de la classe `pvc`, caractérisée par l'apparition d'une extrasystole isolée dans une séquence d'ECG normal, le nombre de cycles nécessaires est relativement grand.
- La classe `pvc` est caractérisée par deux règles qui consistent toutes deux en un intervalle RR court au milieu d'intervalles normaux.
- La classe `doublet` est caractérisée par trois règles qui consistent toutes en deux intervalles RR courts au milieu d'intervalles normaux. La dernière règle, qui ne couvre qu'un exemple, est représentative d'un sur-apprentissage.
- La classe `vt` est caractérisée par trois règles qui consistent toutes en un ensemble d'intervalles RR courts avec présence ou non d'un intervalle normal.

Les règles obtenues avec ce langage de description très abstrait où seules les informations temporelles sont présentes, permettent déjà d'obtenir de bonnes couvertures de classes. Ceci est cohérent avec le fait que les arythmies sont des désordres du rythme et qu'un indicateur temporel permet de les reconnaître. Cependant, cette information temporelle, si elle permet de reconnaître les arythmies, discrimine mal les classes. Par exemple, la deuxième règle de bigéminisme couvre aussi des exemples de tachycardie. Un fort taux de fausses alarmes est donc à prévoir d'autant plus que certaines règles se recouvrent et risquent de provoquer des reconnaissances multiples pour une même cause. Par exemple, les règles de la classe `pvc` couvrent chacune 17 exemples sur 20, il est donc clair qu'elles couvrent au moins les 14 mêmes exemples. Lors de la reconnaissance, deux chroniques de `pvc` seront souvent reconnues en même temps, ce qui doublera le nombre de fausses alarmes.

3.3.3.2 Exper2 et Exper3

Le passage à un langage de description plus riche n'a pas vraiment diminué la taille des règles représentées Figure 3.8. C'est même le contraire dans le cas du `mobitzII` où l'ajout de l'information sur le QRS a obligé l'apprentissage à inférer deux règles. En effet, les exemples de la classe `mobitzII` proviennent d'un enregistrement avec bloc de branche intermittent où le rythme de base est choisi comme étant un bloc de branche.

```
class(bigeminy):-%[[15,0,0,0,0,0],[5,20,25,20,13,11]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1,short),
   qrs(R2, R1),rr1(R1, R2,normal),rr2(R0, R2,normal),
   qrs(R3, R2),rr1(R2, R3,short).
class(bigeminy):-%[[5,0,0,0,0,7],[15,20,25,20,13,4]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1,short),qrs(R2, R1),
   rr1(R1, R2,short),qrs(R3, R2),rr1(R2, R3,short),
   qrs(R4, R3),rr1(R3, R4,short).

class(mobitzII):-%[[0,20,0,0,0,0],[20,0,25,20,13,11]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1,long),qrs(R2, R1),
   rr1(R1, R2,long),qrs(R3, R2),rr1(R2, R3,long).

class(normal):-%[[1,0,24,12,0,1],[19,20,1,8,13,10]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1,normal),qrs(R2, R1),
   rr1(R1, R2,normal),qrs(R3, R2),rr1(R2, R3,normal),qrs(R4, R3),
   rr1(R3, R4,normal),qrs(R5, R4),rr1(R4, R5,normal).

class(pvc):-%[[1,0,0,17,0,1],[19,20,25,3,13,10]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1,normal),qrs(R2, R1),
   rr1(R1, R2,normal),qrs(R3, R2),rr1(R2, R3,short),qrs(R4, R3),
   rr1(R3, R4,normal),qrs(R5, R4),rr1(R4, R5,normal).
class(pvc):-%[[4,0,0,17,0,1],[16,20,25,3,13,10]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1,normal),qrs(R2, R1),
   rr1(R1, R2,short),qrs(R3, R2),rr1(R2, R3,normal),qrs(R4, R3),
   rr1(R3, R4,normal),qrs(R5, R4),rr1(R4, R5,normal).

class(doublet):-%[[0,0,0,0,6,0],[20,20,25,20,7,11]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1,short),qrs(R2, R1),
   rr1(R1, R2,normal),rr2(R0, R2,short),qrs(R3, R2),rr1(R2, R3,normal).
class(doublet):-%[[0,0,0,0,4,0],[20,20,25,20,9,11]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1,short),qrs(R2, R1),
   rr1(R1, R2,short),qrs(R3, R2),rr1(R2, R3,normal),rr2(R1, R3,normal).
class(doublet):-%[[0,0,0,0,1,0],[20,20,25,20,12,11]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1,short),qrs(R2, R1),rr1(R1, R2,short),
   qrs(R3, R2),rr1(R2, R3,long).

class(vt):-%[[5,0,0,0,0,9],[15,20,25,20,13,2]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1,short),qrs(R2, R1),rr1(R1, R2,short),
   qrs(R3, R2),rr1(R2, R3,short).
class(vt):-%[[0,0,0,0,0,1],[20,20,25,20,13,10]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1,normal),qrs(R2, R1),rr1(R1, R2,short),
   qrs(R3, R2),rr1(R2,R3,short),qrs(R4, R3),rr1(R3, R4,normal),rr2(R2, R4,short).
class(vt):-%[[1,0,0,0,0,4],[19,20,25,20,13,7]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1,short),qrs(R2, R1),rr1(R1, R2,normal),
   rr2(R0, R2,short),qrs(R3, R2),rr1(R2, R3,short).
```

FIG. 3.7 – Règles apprises pour exper1.

```
...

class(mobitzII):-%[[0,17,0,0,0,0],[20,3,25,20,13,11]]
   qrs(R0, nonbasic, _),qrs(R1, nonbasic, R0),rr1(R0, R1, long),
   qrs(R2, nonbasic, R1), rr1(R1, R2, long),
   qrs(R3, nonbasic, R2), rr1(R2, R3, long).
class(mobitzII):-%[[0,3,0,0,0,0],[20,17,25,20,13,11]]
   qrs(R0, basic, _),qrs(R1, basic, R0),rr1(R0, R1, long),
   qrs(R2, basic, R1), rr1(R1, R2, long),
   qrs(R3, basic, R2),rr1(R2, R3, long).

class(normal):-%[[0,0,24,6,0,0],[20,20,1,14,13,11]]
   qrs(R0, basic, _),qrs(R1, basic, R0),rr1(R0, R1, normal),
   qrs(R2, basic, R1), rr1(R1, R2, normal),
   qrs(R3, basic, R2),rr1(R2, R3, normal),qrs(R4, basic, R3),
   rr1(R3, R4, normal),qrs(R5, basic, R4),rr1(R4, R5, normal).

...

class(vt):-%[[0,0,0,0,0,9],[20,20,25,20,13,2]]
   qrs(R0, nonbasic, _),qrs(R1, nonbasic, R0),rr1(R0, R1, short),
   qrs(R2, nonbasic, R1), rr1(R1, R2, short),
   qrs(R3, nonbasic, R2),rr1(R2, R3, short).
```

FIG. 3.8 – Extraits de règles apprises pour exper2.

Dans la première règle, les QRS appartiennent au bloc et sont donc classés comme `basic` tandis que dans la seconde règle les QRS appartiennent à un rythme normal et sont classés comme `nonbasic`. Il faut tout de même remarquer que l'information supplémentaire a permis d'éliminer toutes les mauvaises couvertures. Ces règles ne doivent donc théoriquement pas provoquer de fausses alarmes.

Avec `exper3` les résultats sont strictement les mêmes que dans `exper2` en remplaçant les QRS `nonbasic` par des QRS `pvc` excepté pour la classe `mobitzII`. Cette spécification de langage a donc peu d'intérêt.

3.3.3.3 Exper4

L'utilisation du langage $L3$ permet véritablement d'obtenir des règles compactes. Celles-ci sont présentées Figure 3.9. L'information sur l'onde P est très complémentaire des occurrences des QRS, voire plus que la forme des QRS. Cette information a permis de remplacer les attributs d'intervalles RR presque partout. La classe `mobitzII` est caractérisée par une absence de QRS. Cette absence est marquée dans les règles induites par le prédicat `equal(P,R)` qui signifie que P et R indiquent une onde identique. Cet artifice assure qu'aucune rupture de séquence ne se produit dans une règle. Enfin, l'onde P peut même être très informative par son absence. Ainsi la tachycardie ventriculaire

```
...

class(mobitzII):-%[[0,20,0,0,0,0],[20,0,25,20,13,11]]
   p_wav(P0, _),qrs(R0, P0),p_wav(P1, R0),equal(P1, R1),
   p_wav(P2, R1), equal(P2, R2), qrs(R3, R2).

...

class(doublet):-%[[0,0,0,0,13,0],[20,20,25,20,0,11]]
   qrs(R0, _),qrs(R1, R0),qrs(R2, R1),
   p_wav(P3, R2),qrs(R3, P3).

class(vt):-%[[0,0,0,0,0,11],[20,20,25,20,13,0]]
   qrs(R0, _),qrs(R1, R0),qrs(R2, R1),qrs(R3, R2).
```

FIG. 3.9 – Extraits de règles apprises pour exper4.

est caractérisée par des QRS sans onde P.

3.3.3.4 Exper5 et Exper6

Dans le langage très riche de `exper5` l'amélioration par rapport à `exper4` est peu visible. L'onde P semble être le principal échelon pour obtenir des règles compactes.

Avec `exper6` les résultats sont strictement les mêmes que dans `exper5` en remplaçant les QRS `nonbasic` par des QRS `pvc`. Encore une fois, cette spécification de langage a donc peu d'intérêt.

Discussion Le choix du langage de description de l'ECG est central dans la vision du pilotage. Il est intéressant de constater que les arythmies peuvent être reconnues même avec un langage abstrait basé uniquement sur des informations temporelles réduites. Lorsque le langage est plus précis, ces informations sont vite abandonnées au profit de la forme du QRS ou de l'onde P étant donné que l'intervalle RR est très fluctuant à l'intérieur d'une même classe. Pour toutes les classes, les profils de règle restent les mêmes. Ils sont dans l'ensemble composés de battements normaux et anormaux. Ces derniers sont représentés avec des intervalles RR courts ou longs dans `exper1`, un QRS nonbasic ou pvc dans `exper2` et `exper3`, une absence d'onde P dans `exper4` un mélange des deux derniers dans `exper5` et `exper6`. Dans l'ensemble, les apprentissages sont très bons dans le sens où il semble possible d'ajouter encore plus d'arythmies sans arriver aux limites du système.

```
...

class(mobitzII):-%[[0,17,0,0,0,0],[20,3,25,20,13,11]]
   p_wav(P0, normal, _),qrs(R0, nonbasic, P0),
   p_wav(P1, normal, R0),equal(P1, R1),
   p_wav(P2, normal, R1),equal(P2, R2),qrs(R3, nonbasic, R2).
class(mobitzII):-%[[0,3,0,0,0,0],[20,17,25,20,13,11]]
   p_wav(P0, normal, _),qrs(R0, basic, P0),
   p_wav(P1, normal, R0),equal(P1, R1),
   p_wav(P2, normal, R1),equal(P2, R2),qrs(R3, basic, R2).

...

class(doublet):-%[[0,0,0,0,13,0],[20,20,25,20,0,11]]
   qrs(R0, basic,_),qrs(R1, nonbasic,R0),qrs(R2, nonbasic,R1),
   p_wav(P3, normal,R2), qrs(R3, basic,P3).

class(vt):-%[[0,0,0,0,0,11],[20,20,25,20,13,0]]
   qrs(R0, nonbasic, _),qrs(R1, nonbasic, R0),
   qrs(R2, nonbasic, R1),qrs(R3, nonbasic, R2).
```

FIG. 3.10 – Extraits de règles apprises pour exper5.

3.3.3.5 Transcription des règles en modèles de chroniques

Dans CALICOT, la reconnaissance des arythmies est effectuée par le système de reconnaissance de chroniques CRS (*Chronicle recognition system*) (Dousson, 1994). Ce principe est conservé dans IP-CALICOT. Dans ce contexte, les règles apprises par PLI sont automatiquement traduites en chroniques CRS. Les deux formalismes de représentation sont assez proches et le passage d'un formalisme à l'autre n'est pas détaillé. La table 3.11 donne un exemple de règle PROLOG (à gauche sur la figure) apprise pour la classe `vt` extrait de `exper5` et sa transcription sous forme de chronique CRS (à droite sur la figure).

Une tachycardie ventriculaire se caractérise par quatre QRS successifs de forme `nonbasic`. Aucune onde P ne doit intervenir entre ces QRS. Cette information qui est sous-entendue dans la règle PROLOG, doit être écrite explicitement dans la chronique CRS. Elle correspond aux lignes `noevent(p_wave[*],(R0+1, R1-1))` qui signifient qu'aucun évènement de type `p_wave` ne doit intervenir entre l'instant `R0+1` et `R1-1`.

3.4 Conclusion

Le système de monitorage IP-CALICOT, successeur de CALICOT a été présenté. Son originalité est qu'il intègre un pilote d'algorithmes qui permet un pilotage à trois niveaux : au niveau du *diagnostic médical* qui permet de consommer moins de ressources

`class(vt):-`	`chronicle vt5[](){`

`qrs(R0, nonbasic, _)`	`noevent(qrs[*], (start+1, R0-1))`
	`noevent(p_wave[*], (start+1, R0-1))`
	`event(qrs[?w0], R0)`
	`?w0 in {nonbasic}`

`qrs(R1, nonbasic, R0)`	`noevent(qrs[*], (R0+1, R1-1))`
	`noevent(p_wave[*], (R0+1, R1-1))`
	`event(qrs[?w1], R1)`
	`R0 < R1`
	`?w1 in {nonbasic}`

`qrs(R2, nonbasic, R1)`	`noevent(qrs[*], (R1+1, R2-1))`
	`noevent(p_wave[*], (R1+1, R2-1))`
	`event(qrs[?w2], R2)`
	`R1 < R2`
	`?w2 in {nonbasic}`

`qrs(R3, nonbasic, R2)`	`noevent(qrs[*], (R2+1, R3-1))`
	`noevent(p_wave[*], (R2+1, R3-1))`
	`event(qrs[?w3], R3)`
	`R2 < R3`
	`?w3 in {nonbasic}`
	
	`end - start in nb_cycles3}`

FIG. 3.11 – Exemple de règle PROLOG pour la tachycardie ventriculaire extrait de `exper5` sur l'ECG et sa chronique correspondante

et d'assurer un diagnostic même lorsque certains attributs de l'ECG sont manquants ; au niveau des *tâches d'abstraction du signal* qui permet de désactiver les tâches d'extraction de caractéristiques qui ne peuvent être maintenues sans erreurs pour fournir les informations les plus fiables possibles ; au niveau des *algorithmes de traitement du signal* qui permet de choisir les algorithmes les plus adaptés au contexte courant.

Le contexte courant a été défini comme une composition d'informations de haut et de bas niveau telles que le bruit de ligne et les pathologies courantes du patient. Ces informations permettent une adaptation précise de la chaîne de traitements.

Le pilotage permet une extraction des caractéristiques de l'ECG selon différents langages de description. Ainsi, même si certaines tâches de l'abstraction temporelle ne peuvent être accomplies, le diagnostic médical peut être assuré avec un langage de description plus abstrait. Pour permettre ce diagnostic, une hiérarchie de modèles de chroniques a été apprise selon différents langages de description.

L'architecture proposée apporte une grande flexibilité à tous les niveaux de la chaîne de traitements ce qui permet d'envisager d'autres types d'adaptation, tels que la gestion des ressources de calcul.

L'approche proposée reste très dédiée au monitorage cardiaque mais certaines améliorations pourraient rendre ce pilotage d'algorithmes transposable à d'autres systèmes.

- Les règles qui régissent le pilotage des tâches ont été acquises uniquement sur expertises. Celles-ci stipulent surtout le niveau et le type de bruit présent sur la ligne qui empêche la réalisation satisfaisante de la tâche. Une évolution serait d'utiliser les connaissances des performances des algorithmes de chaque tâche. Les performances attendues selon le contexte présent pourraient ainsi être calculées en fonction des performances des algorithmes selon les mêmes contextes. Les règles des tâches reposeraient ainsi sur la connaissance acquise des algorithmes de traitement du signal. De plus, si les performances des algorithmes sont connues quantitativement alors les tâches pourraient être activées ou désactivées selon un seuil de performance donné à la conception de l'application.
- Une phase d'initialisation est nécessaire dans un tel système que ce soit pour le pilotage (initialisation des tâches) ou pour le monitorage (saisie des informations patient). Cette phase pourrait aussi être utilisée par les algorithmes pour l'apprentissage de certains paramètres. Par exemple, certains classificateurs de QRS sont constitués de réseaux de neurones. Le réseau de neurones pourrait ainsi apprendre les formes de QRS spécifiques du patient pendant une courte période. Ainsi, l'influence de la variabilité inter-patient sur les systèmes de monitorage serait réduite.
- Les données pourraient aussi être retraitées de manière plus précise lorsque le contexte le permet. Par exemple, lorsque le nombre de ressources est restreint, les traitements se limitent aux aspects vitaux du patient. Les données extraites de cette manière pourraient être analysées de nouveau en tâche de fond de l'activité de monitorage lorsque les ressources redeviennent libres.

Si le pilotage apporte une solution séduisante, il n'en reste pas moins qu'il est fondé sur des règles de pilotage d'algorithmes dont l'acquisition reste un problème difficile. Cette activité délicate fait l'objet du prochain chapitre.

Chapitre 4

Acquisition des règles de pilotage d'algorithmes

4.1 Introduction

Le pilotage d'algorithmes consiste à choisir un algorithme dans une base, qui parait le plus adapté pour traiter une situation donnée. Pour cela, il est nécessaire de mettre au point une méthode pour obtenir les règles qui vont permettre au pilote d'associer un contexte à un algorithme particulier.

Avant d'exposer la méthode conçue pour acquérir ces règles, les algorithmes de traitement du signal utilisés pour l'abstraction temporelle de IP-Calicot sont tout d'abord introduits en section 4.2. Ces algorithmes proviennent tous de la littérature ou de Calicot. Cette présentation des algorithmes est importante car les règles de pilotage sont très dépendantes des algorithmes utilisés et des situations considérées. Les situations représentatives des cas rencontrés en ECG sont définies par les contextes déjà abordés au chapitre 3. Ces derniers sont détaillés au cours de ce chapitre.

Enfin, l'acquisition des règles d'association entre les algorithmes et les différents contextes est présentée en section 4.3. Succinctement, elle consiste à générer un ensemble d'ECG représentatifs d'une situation clinique (un contexte) et à analyser les performances des algorithmes sur chacun de ces ECG par une méthode d'analyse factorielle. Dans ce travail, l'effort a été restreint au pilotage des algorithmes de détection de QRS qui demeure l'événement primordial à détecter pour l'analyse de l'ECG.

4.2 Présentation de la base algorithmes

Dans IP-Calicot la chaîne de traitements de l'abstraction temporelle est composée de quatre tâches : le *Filtrage*, la *détection du QRS*, la *classification du QRS* et la *détection de l'onde P*. Une tâche peut généralement être réalisée par un algorithme. Toutefois, dans la littérature, il existe plusieurs algorithmes capables de réaliser une telle tâche. Le but du pilotage est de choisir l'algorithme le plus adapté en fonction du contexte. Cette section présente les différents algorithmes retenus pour le filtrage, la

détection de QRS, la classification de QRS et la détection d'ondes P. Ils constituent la *base d'algorithmes*.

4.2.1 Filtrage de l'ECG

Le filtrage a pour but de séparer les signaux utiles des bruits indésirables. En électrocardiographie, ces bruits sont bien identifiés mais certains ont la particularité de recouvrir la bande spectrale de l'ECG, ce qui les rend difficile à filtrer. La figure 4.1 donne la densité spectrale de puissance de l'ECG et de ses différentes composantes. Sur cette figure, on voit que l'énergie de l'ECG est repartie dans la bande $[2, 40]Hz$

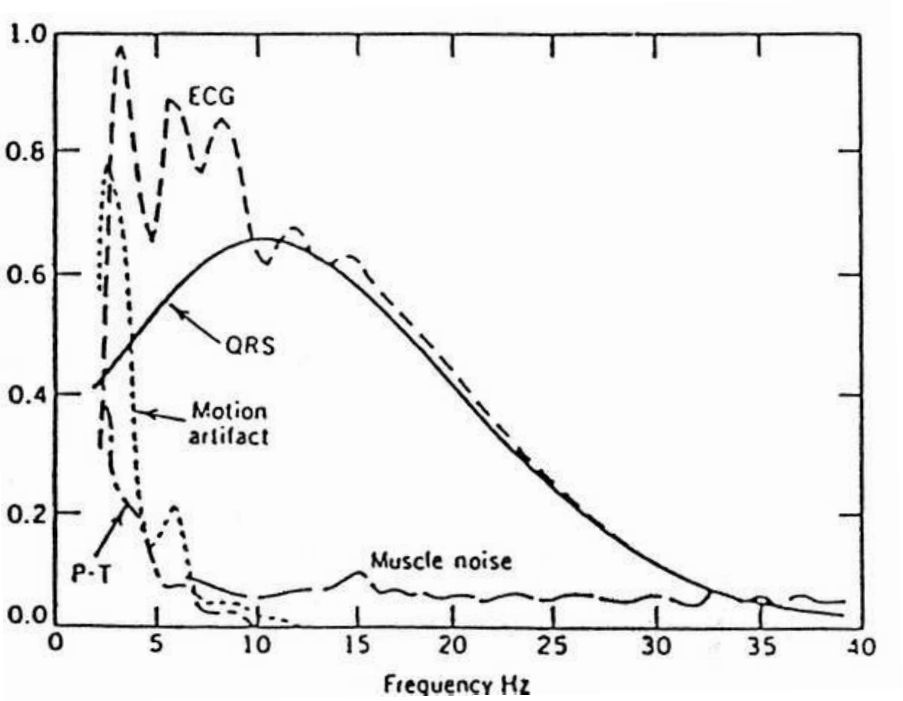


FIG. 4.1 – Densité spectrale de puissance de l'ECG et de ses composantes

qui recouvre celle des bruits cliniques typiques. Dans notre application, un seul filtre a été mis en œuvre. Il utilise une analyse multi-résolution (Mallat, 1999) fondée sur une ondelette de Daubechies (Daubechies, 1988). Il revient à un filtrage de bande passante $[1.9, 45]Hz$ qui permet de récupérer la gamme de fréquence appartenant à l'ECG. Ce choix se base sur les travaux de thèse de Senhadji (Senhadji, 1993) qui a montré les bénéfices tirés d'une telle démarche. L'autre avantage de l'analyse multi-résolution est qu'elle conduit à un développement unifié pour le filtre d'entrée, la détection de l'onde P et l'analyseur de contexte de ligne.

4.2.2 Détection du QRS

Le choix du détecteur de QRS est très important pour réaliser un système d'analyse de l'ECG. Un détecteur de QRS est généralement composé de quatre modules représentés Figure 4.2.

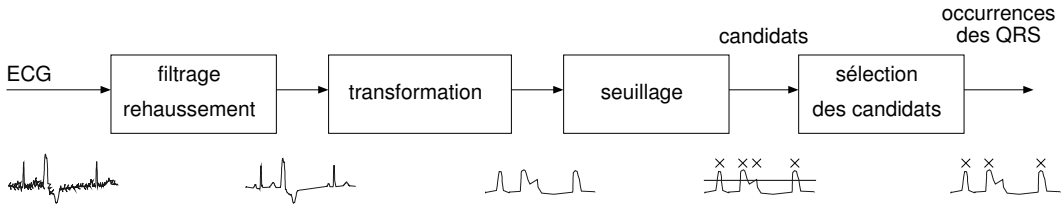


FIG. 4.2 – Schéma bloc d'un détecteur de QRS.

Le signal ECG est d'abord filtré puis transformé pour mettre en exergue les informations qui concernent le QRS. Ensuite, la transformée est seuillée puis un étage de décision sélectionne les occurrences de QRS parmi les candidats.

La détection du QRS a fait l'objet de nombreux travaux depuis une trentaine d'années et continue d'être un champ de recherche très actif. On peut ainsi trouver des algorithmes basés sur : 1) des filtres numériques (Ruha et coll., 1997; Daskalov et Christov, 1999; Wieben et coll., 1999; Nagin et Selishchev, 2001), 2) des analyses temps-fréquence et ondelettes (Li et coll., 1995; Senhadji et coll., 1995; Afonso et coll., 1999; Kadambe et coll., 1999), 3) des transformations linéaires et non-linéaires (Pan et Tompkins, 1985; Gritzali, 1988; Benitez et coll., 2001), 4) des analyses statistiques (réseaux de neurones) (Watrous et Towell, 1995; Silipo et Marchesi, 1998), 5) des grammaires de formes (*template matching*), 6) des méthodes évolutionnaires (Poli et coll., 1995), 7) de la fusion multisource (Thoraval et coll., 1997; Hernández et coll., 1999; Koulouris et coll., 2000; Wrzesniowski et Augustyniak, 2001). Cette recherche est toujours active et plusieurs nouvelles méthodes de détection ont été proposées récemment (Burke et Nasor, 2004; Dotsinsky et Stoyanov, 2004; Christov, 2004; Fernández et coll., 2005). Parmi cet ensemble de détecteurs, l'objectif a été de constituer une base de détecteurs. Comme il est impensable de tous les mettre en œuvre dans le système, seuls sept ont été sélectionnés selon les critères suivants :

- capacité à travailler en temps réel,
- facilité de mise en œuvre,
- robustesse au bruit.

Les sept détecteurs choisis sont ceux proposés par Benitez et coll. (Benitez et coll., 2001), Gritzali (Gritzali, 1988), Suppappola et Sun (Suppappola et Sun, 1994), Kadambe et coll. (Kadambe et coll., 1999), Pan et Tompkins (Pan et Tompkins, 1985), Fraden et Neuman (Fraden et Neuman, 1980) et Okada (Okada, 1979). Ces deux derniers ont été présentés dans (Friesen et coll., 1990) sous la dénomination de AF2 (*Amplitude Filter 2*) et DF2 (*Digital Filter 2*). Ils ont démontré que ces deux détecteurs sont sensibles à des bruits différents les rendant ainsi complémentaires. Les paragraphes suivants détaillent les détecteurs choisis.

AF2 : L'algorithme 2 présenté est une adaptation du détecteur analogique de QRS de Fraden et Neuman (Fraden et Neuman, 1980), il est tiré de l'article de Friesen et coll. (Friesen et coll., 1990). Il travaille sur l'amplitude et la dérivée première du signal ECG.

Algorithme 2 Algorithme de AF2.

Fonction **AF2**
Soit N *le nombre d'échantillons*
/* *Transformation : Un premier seuil d'amplitude est calculé* */
$Seuil_1 = 0.15 * max(ecg)$
/* *Les données brutes sont alors rectifiées en prenant la valeur absolue* */
Pour $n = 0$ à N **faire**
  $Y_0(n) = |x(n)|$
Fin pour
/* *L'ECG rectifié est ensuite seuillé* */
Pour chaque $n \in \{0, \ldots, N\}$ **faire**
  Si $Y_0(n) >= seuil_1$ **alors**
    $Y_1(n) = Y_0(n)$
  Sinon
    $Y_1(n) = 0$
  Fin si
Fin pour
/* *La dérivée première est calculée* */
Pour $n = 1$ à $N - 1$ **faire**
  $Y_2(n) = Y_1(n + 1) - Y_1(n - 1)$
Fin pour
renvoyer Y_2

Il n'y a pas d'étage de décision ni de filtrage dans cet algorithme. L'étude de Friesen et coll. (Friesen et coll., 1990) a montré que AF2 est très sensible au bruit basse fréquence mais particulièrement résistant au bruit électromyographique.

Benitez : Cet algorithme récent est présenté dans (Benitez et coll., 2001). Il utilise la dérivée et la transformée de Hilbert pour rehausser les QRS de l'ECG. La figure 4.3 montre les différentes étapes de ce détecteur.

Le filtre passe-bande est formé par un filtre à réponse impulsionnelle finie avec une fenêtre de Kaiser-Bessel. La dérivée est ensuite calculée puis les passages à zéro sont rehaussés par la transformée de Hilbert. Après le seuillage, une recherche de QRS concurrents dans une fenêtre est effectuée. Les QRS sont ensuite choisis en fonction de l'amplitude des points dans l'ECG original et en fonction de l'intervalle RR précédent. le seuil est calculé à partir de l'échantillon d'amplitude maximum et de la valeur efficace du signal qui représentent respectivement l'amplitude des QRS et l'amplitude du bruit. Après le seuillage, une recherche de QRS concurrents dans une fenêtre est effectuée. Les QRS concurrents sont départagés en fonction de l'amplitude des échantillons dans

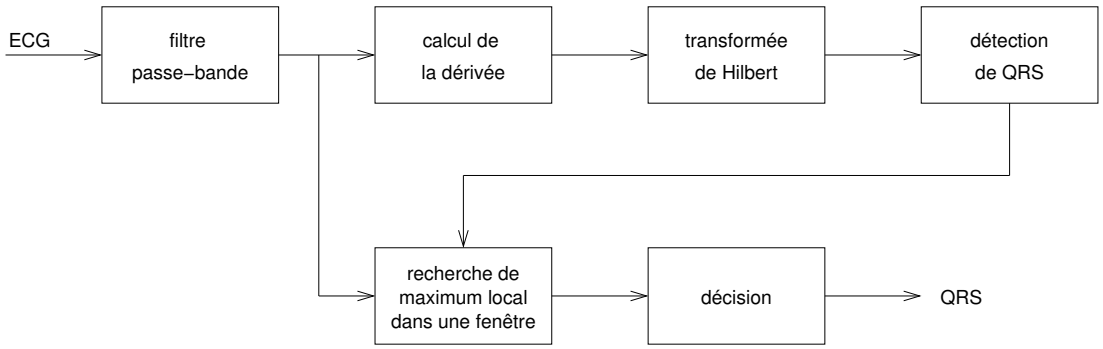


FIG. 4.3 – Diagramme des étapes de l'algorithme de Benitez et coll.

l'ECG original et en fonction de l'intervalle RR précédent.

DF2 : L'algorithme 3 présenté est une adaptation du détecteur de QRS par filtre numérique d'Okada. Il est tiré de l'article de Friesen et coll. (Friesen et coll., 1990).

La dernière partie de l'algorithme représente un étage de décision réduit qui utilise une vérification de signe du signal ECG filtré pour éliminer des candidats. L'étude de Friesen et coll. (Friesen et coll., 1990) a montré que cet algorithme est très sensible au bruit électromyographique.

Gritzali : L'algorithme de détection, développé par Gritzali (Gritzali, 1988), a permis de généraliser la plupart des détecteurs de QRS. Celui-ci est basé sur l'analyse de la longueur d'un signal ECG représenté par un ou plusieurs canaux. En pratique, pour un seul canal, on choisit une fenêtre temporelle de taille q, à l'intérieur de laquelle on somme toutes les longueurs :

$$L(q,i) = \sum_{i}^{i+q} \sqrt{(1 + (x(i + retard) - x(i))^2)}$$

Sur n canaux, la fusion est obtenue en calculant la norme des longueurs en chaque point :

$$L(n,q,i) = \sum_{i}^{i+q} \sqrt{\sum_{k=1}^{n} (1 + (x_k(i + retard) - x_k(i))^2)}$$

La méthode de seuillage proposée n'est pas valide en pratique car elle est destinée uniquement aux données utilisées dans l'expérience originale. Dans notre application, la méthode développée utilise un seuillage du même type que *benitez*.

Kadambe : L'algorithme de détection, développé par Kadambe et coll. (Kadambe et coll., 1999) s'appuie sur une transformation en ondelettes de type spline. Cette transformation a la particularité de rehausser la bande spectrale du QRS tout en filtrant le

Algorithme 3 Algorithme de DF2.

Fonction **DF2**
Soit N *le nombre d'échantillons*
/ Filtrage : L'ECG est passé à travers un filtre moyenneur */*
Pour $n = 1$ à $N - 1$ **faire**
  $Y_0(n) = [ecg(n-1) + 2ecg(n) + ecg(n+1)]/4$
Fin pour
/ Puis à un filtre passe-bas numérique */*
Pour $n = m$ à $N - m$ **faire**
  $Y_1(n) = [1/(2m+1]\Sigma_{k=n-m}^{n+m} Y_0(k)$
Fin pour
/ Transformation : La différence entre l'entrée et la sortie est ensuite calculée */*
Pour $n = m$ à $N - m$ **faire**
  $Y_2(n) = (Y_0(n) - Y_1(n))^2$
Fin pour
/ Puis cette différence carré est filtrée */*
Pour $n = m$ à $N - m$ **faire**
  $Y_3(n) = Y_2(n)\{\Sigma_{k=n-m}^{n+m} Y_2(k)\}^2$
Fin pour
/ La sortie de la transformée est ensuite calculée */*
Pour chaque $n \in \{(m, \ldots, N - m\}$ **faire**
  Si $[(Y_0(n) - Y_0(n-m)]$ **et** $[(Y_0(n) - Y_0(n+m)] > 0$ **alors**
    $Y_4(n) = Y_3(n)$
  Sinon
    $Y_4(n) = 0)$
  Fin si
Fin pour
renvoyer Y_4

signal inutile (bruit, ondes P, ondes T). Le filtrage, la transformation et la détection se font de manière itérative comme montré dans le diagramme de la figure 4.4.

Le signal ECG est analysé segment par segment. Chaque segment est multiplié par une fenêtre de Hamming puis transformé en ondelettes. La transformation est d'abord faite à l'échelle i puis à l'échelle $i+1$. Un seuillage est ensuite effectué à chaque niveau. Si le nombre de QRS est le même à chaque niveau alors la détection est considérée comme correcte sinon l'ECG est de nouveau analysé à l'échelle suivante. Une fois les QRS trouvés, l'intervalle RR est utilisé pour éliminer les dates incohérentes. Chaque segment suivant contient 75% de l'ancien segment. Autrement dit, chaque portion de l'ECG est analysée quatre fois. Cette répétition en fait une méthode très coûteuse mais permet de compenser les nombreuses non-détections dues à la contrainte stricte d'égalité du nombre de QRS trouvés à chaque échelle.

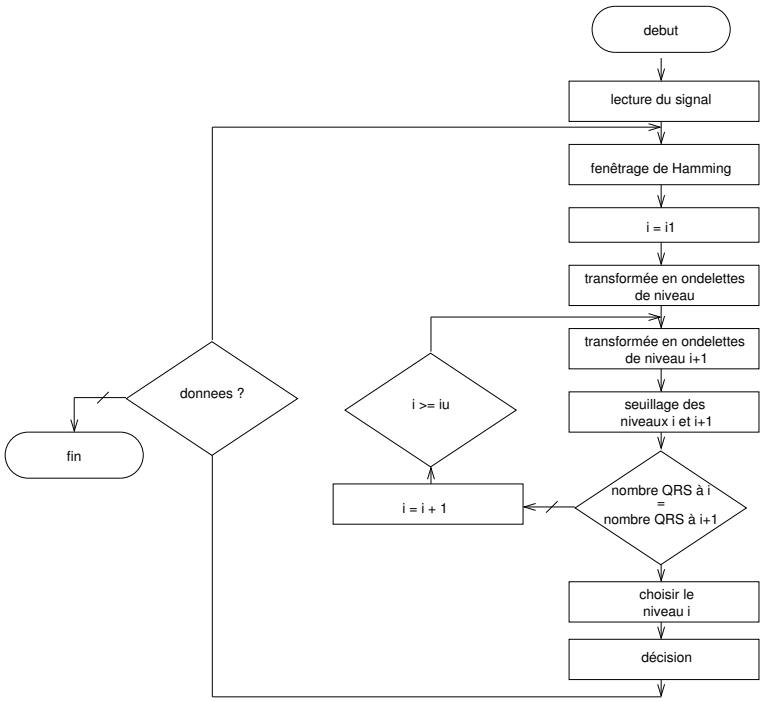


FIG. 4.4 – Diagramme des étapes de l'algorithme de Kadambe et coll.

MOBD : L'algorithme de détection, développé par Suppappola et Sun (Suppappola et Sun, 1994) est basé sur une transformation non linéaire appelée MOBD (*Multiplication of backward difference*). La différence de premier ordre est calculée comme suit :

$$x[k] = ecg[k] - ecg[k-1]$$

et la transformée MOBD d'ordre N est donnée par :

$$y[n] = \prod_{k=0}^{N-1} |x[n-k]|$$

une consistance de signe est ensuite assurée par :

Si $signe(x[n-k]) \neq signe(x[n-(k+1)]), \forall k \in \{0, \ldots, N-2\}$ **alors**
  $y[n] = 0$
Fin si

La présence ou l'absence d'un QRS est jugée en employant un seuil adaptatif calculé en fonction d'une estimation du bruit. Cet algorithme possède des performances moyennes mais il a l'avantage d'être très peu coûteux en ressources tant que l'ordre N du détecteur reste raisonnable.

Pan : L'algorithme de détection, développé par Pan et Tompkins (Pan et Tompkins, 1985), identifie les complexes QRS en se basant sur l'analyse de la pente, de l'amplitude et de la largeur des ondes. La figure 4.5 montre les différentes étapes de la détection. Le filtre passe-bande est formé par un passe-bas et un passe-haut et a pour fonction de réduire le bruit dans le signal ECG.

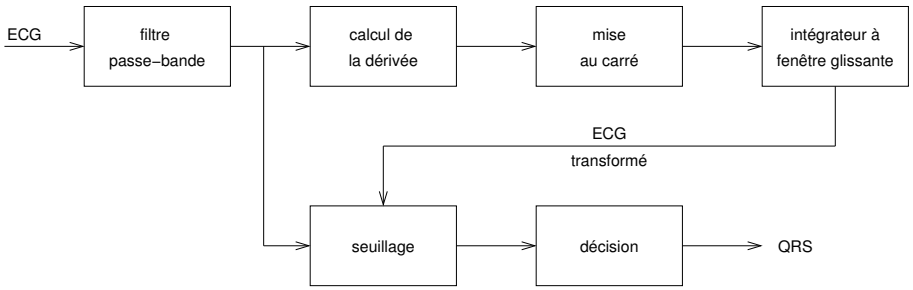


FIG. 4.5 – Diagramme des étapes de l'algorithme de Pan et Tompkins.

Après filtrage, le signal est dérivé afin de mettre en évidence les fortes pentes qui distinguent les complexes QRS des composants ECG de basse fréquence, tels que les ondes P et T. L'opération suivante est la quadrature, qui fait ressortir les valeurs les plus hautes qui sont principalement celles des complexes QRS. L'opération finale est l'intégration sur une fenêtre glissante de longueur N. La sortie de l'intégrateur à fenêtre glissante peut être utilisée pour détecter des complexes QRS, mesurer les intervalles RR ou déterminer la durée des complexes QRS.

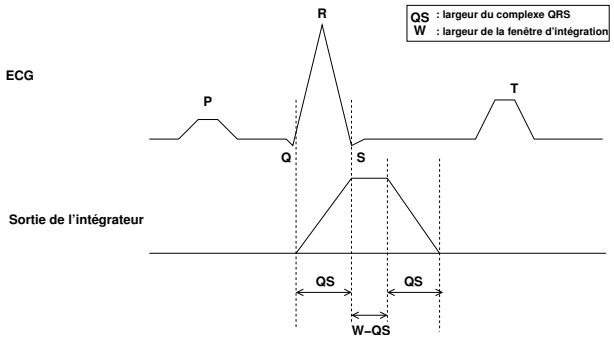


FIG. 4.6 – Sortie de l'intégrateur à fenêtre glissante du détecteur de QRS de Pan et Tompkins.

Le signal filtré et le signal transformé sont ensuite utilisés par un étage de décision pour détecter les candidats. Les règles mises en jeu dans cet étage sont assez complexes. Elles sont basées sur l'estimation du niveau de bruit et du dernier intervalle RR pour mettre à jour des seuils adaptatifs. Cet algorithme est le plus largement utilisé dans la littérature.

4.2.3 Classification du QRS

Le complexe QRS reflète l'activité ventriculaire et sa morphologie peut changer lorsque sa commande ou la conduction du front de dépolarisation ou de repolarisation devient anormale. Par exemple, une activité prématurée ventriculaire et un bloc de branche conduisent à un élargissement du complexe QRS. La classification du complexe QRS est donc nécessaire lors de l'élaboration d'un système de reconnaissance d'arythmies ou lors d'une transformation du signal numérique en flots d'événements symboliques. Le système met en œuvre les deux classificateurs de QRS à réseau de neurones de Calicot :

– le premier possède deux sorties classant le QRS en `normal` ou `abnormal`
– le deuxième possède quatre sorties classant le QRS en `normal`, `pvc`, `lbbb`, et `rbbb`.

C'est le deuxième classificateur qui est utilisé dans IP-Calicot car il permet de distinguer les battements de base du patient (`normal`, `lbbb`, et `rbbb`)[1], des battements prématurés ventriculaires (`pvc`) autrement appelés extrasystoles. De plus, il permettrait de détecter les blocs de branche intermittents tels que des alternances entre blocs de branche et rythme normal.

Le premier classificateur est tout de même conservé car il permet de classer le QRS dans un langage plus abstrait (`normal` et `abnormal`) utilisé parfois par des modèles de chroniques reposant sur ce langage.

4.2.4 Détection d'ondes P

L'onde P est le reflet de l'activité auriculaire, sa faible amplitude et la grande variabilité de sa morphologie en font une onde particulièrement difficile à détecter, ce qui explique qu'elle demeure un problème non résolu.

Les algorithmes de détection d'onde P sont généralement conçus selon deux approches. La première s'appuie sur une détection préalable du QRS et consiste à rechercher, dans une fenêtre précédant le QRS, une trace de l'onde P comme dans l'algorithme SA/RT (Agilent Technologies, 2000). La seconde est basée sur l'annulation de l'ensemble QRS-T (Stridh et Sörnmo, 2001; Vásquez et coll., 2001; Wang, 2002). Dans IP-Calicot, c'est l'algorithme proposé dans Calicot (*cf.* 2.3.3) qui à terme sera mis en œuvre. Ce dernier s'appuie sur l'annulation du QRS-T. Cette approche est, de fait, moins dépendante des dissociations P-QRS notamment lors de blocs AV.

4.3 Étude des détecteurs de QRS selon différents contextes

4.3.1 Introduction

Pour obtenir un pilotage efficace des détecteurs de QRS, il est nécessaire d'extraire les règles qui permettent au pilote d'associer un contexte à un algorithme particulier. Le pilotage proposé dans cette étude repose sur deux hypothèses préliminaires :

[1] On estime donc ici le contexte propre à chaque patient

- **Le contexte est reconnaissable en ligne :** ce qui signifie que l'on arrive toujours à estimer le contexte dans lequel on se trouve. La reconnaissance des formes de QRS et du bruit de ligne est effectuée par des analyseurs de contexte. L'analyse du contexte de ligne est actuellement à l'étude et associe une analyse multi-résolution et le calcul de l'énergie en sous-bande. Les premiers résultats enregistrés sont encourageants mais n'ont malheureusement pas pu être finalisés.
- **Le bruit et les formes de QRS influent sur les résultats :** ce qui signifie que les performances des détecteurs sont dépendantes non seulement de l'information non pertinente qui vient parasiter la ligne (bruit clinique) mais aussi des informations pertinentes concernant la forme des QRS porteuse de signes pathologiques. Alors qu'il a été démontré que le bruit clinique est très gênant pour le fonctionnement des détecteurs, la forme des QRS et les arythmies ont souvent été négligées. À ma connaissance, aucune étude n'a pris sérieusement ce facteur comme mesure de performance. L'analyse du contexte arythmique est actuellement à l'étude. Il peut être basé sur les précédents diagnostics d'arythmies pour obtenir une tendance à court terme.

Ces hypothèses ne peuvent être confirmées par les travaux de la littérature actuelle. En effet, de nombreuses études comparatives de détecteur de QRS existent mais peu d'entre elles ont généré des connaissances suffisamment précises pour améliorer la tâche de détection de cette onde importante.

État de l'art de la comparaison de détecteurs de QRS Plusieurs comparaisons quantitatives de performances ont été présentées dans la littérature (Poli et coll., 1995; Kadambe et coll., 1999; Benitez et coll., 2001; Wagner et coll., 2003). Ces évaluations sont généralement effectuées sur des enregistrements provenant de bases standard (MIT-BIH, CSE, AHA, etc.) et sont basées sur des scores de détection généralement exprimés en terme de sensibilité et de spécificité. Un score moyen calculé sur l'ensemble des résultats est sensé refléter les performances générales des détecteurs. Cependant, un ECG est composé de différents niveaux de bruit et de morphologies de QRS variées, un score moyen n'explique donc pas quelles sont les caractéristiques spécifiques du signal qui affectent les résultats des détecteurs (Kohler et coll., 2002). De plus, une comparaison fiable des résultats implique que l'évaluation des performances des détecteurs de QRS doit être faite sur la même base de données, ce qui n'est pas toujours le cas dans la littérature.

L'étude réalisée par Friesen et coll. (Friesen et coll., 1990) constitue un bon exemple de comparaison de détecteurs. Les auteurs comparent neuf algorithmes de détection de QRS par rapport à un ECG *gold standard*. Cet ECG est corrompu par cinq types de bruit artificiel représentatifs des bruits rencontrés en situation clinique. Les auteurs sélectionnent le meilleur détecteur en comparant les performances moyennes et en montrant que chaque détecteur est influencé différemment selon les contextes de bruit. Cependant cette étude ne prend ni en compte l'étape de filtrage ni les effets de la morphologie du QRS sur la détection. D'autres études ont pris en compte l'influence du bruit, tel que dans (Moody et coll., 1984) avec du bruit expérimental ou dans (Sun et coll., 2002) avec du bruit simulé. Cependant, l'influence des morphologies de QRS

n'a pas été pas étudiée. Dans (Nygårds et L. Sörnmo, 1981), les auteurs étudient la distance entre une méthode manuelle et automatique de délimitation de QRS en ajoutant du bruit gaussien à différents RSB (rapport signal-à-bruit). Mais ce travail était limité à deux formes de QRS et seulement deux méthodes (manuelle et automatique).

Ce bref état de l'art montre que les comparaisons rapportées dans la littérature n'évaluent pas la sensibilité des détecteurs de QRS par rapport à l'ensemble des contextes qui peuvent les influencer. C'est pourquoi, une nouvelle méthode de comparaison est proposée. La méthode mise au point compare un ensemble de détecteurs de QRS sous différentes conditions de bruit et de morphologie afin de déterminer le détecteur le plus adapté à un contexte donné.

La définition précise des contextes est détaillée en section 4.3.2 et les données utilisées pour les composer sont présentées en section 4.3.3. La qualité de mise en œuvre des détecteurs de QRS doit aussi être prise en compte. Elle est discutée en section 4.3.4. Puis, la méthode de comparaison est exposée en section 4.3.5. Celle-ci se base sur une analyse en composantes principales dont quelques notions sont introduites en section 4.3.6. Enfin, les résultats de comparaison et les règles de pilotage sont données en section 4.3.7.

4.3.2 Définition des contextes

Un contexte est défini ici comme étant la combinaison particulière d'un bruit et de morphologies de QRS appartenant à un rythme pathologique ou normal. Un changement de rythme ou de bruit implique un changement de contexte. Le contexte peut donc être séparé en deux sous contextes : le contexte de ligne et le contexte arythmique. Le contexte patient n'est pas considéré ici car il est indépendant des signaux analysés (*cf.* 3.2.3).

Contexte de ligne Le contexte de ligne est constitué du niveau et du type de bruit présent sur la ligne à un instant donné t. Pour rappel, les trois types de bruit considérés sont : les ondulations de ligne de base (*bw*, *baseline wander*), les bruits musculaires (*ma*, *muscle artefact*) et les mauvais contacts peau-électrode (*em*, *electrode motion artefact*). Le bruit *bw* est majoritairement basse fréquence, le bruit *ma* est majoritairement haute fréquence, et le bruit *em* a des composantes hautes et basses fréquences. Trois rapports signal sur bruit (5, -5, et -15 dB) sont considérés mais la ligne peut, bien sûr, ne présenter aucun bruit. Le contexte prend donc une valeur parmi dix :

$$\forall n \in \mathbf{N},\ ctxtline(t) \in (\{bw, ma, em\} \times \{5, -5, -15\}) \cup \{sans\ bruit\}$$

Les artefacts de mouvements d'électrode et le bruit musculaire sont généralement considérés comme étant les plus perturbateurs, puisqu'ils peuvent être pris pour des ondes ectopiques et ne peuvent pas être filtrés facilement car ils recouvrent le spectre des QRS (voir Figure 4.1).

Contexte arythmique Le contexte arythmique est composé des différents rythmes et morphologies de QRS qui composent l'ECG. Un détecteur est autant influencé par le type de QRS présent sur la ligne que par une succession de QRS de types différents. Les rythmes considérés permettent de couvrir un nombre important de configurations. Il s'agit des huit classes de contexte suivantes :

- la classe `normal` qui représente un rythme normal composé de QRS normaux,
- la classe `lbbb` qui représente un rythme normal composé de QRS de bloc de branche gauche,
- la classe `rbbb` qui représente un rythme normal composé de QRS de bloc de branche droit,
- la classe `paced` qui représente un rythme normal composé de QRS provoqués par un pacemaker,
- la classe `ventricular tachycardia` qui est une succession d'extrasystoles à un rythme rapide.
- la classe `mobitzII` qui est représentée par un ECG dans lequel un QRS sur deux n'apparaît pas,
- la classe `bigeminy` qui est la succession d'un QRS normal et d'une extrasystole répétée pendant un certain nombre de cycles,
- la classe `trigeminy` qui est la succession de deux QRS normaux et d'une extrasystole répétée pendant un certain nombre de cycles.

Les cinq premières classes permettent surtout d'étudier l'influence d'une forme particulière de QRS sur la détection. Les dernières classes apportent des informations sur la capacité des algorithmes à traiter un rythme irrégulier composé de plusieurs formes de QRS.

4.3.3 Données d'ECG

Les ECG sélectionnés proviennent de l'électrode MLII (*Modified Lead II*) des enregistrements de la base d'arythmies MIT-BIH (Mark et Moody, 1988). Chaque enregistrement a été annoté indépendamment par plusieurs cardiologues (deux au moins). L'annotation est une date avec un label. La date correspond au moment d'apparition du R du QRS et le label correspond au type du QRS (lbbb, pvc, etc.). Les enregistrements : 100, 101, 103, 106, 107, 109, 111, 113, 115, 119, 122, 123, 124, 201, 205, 231 et 233 ont été choisis pour leur qualité et la présence de nombreuses formes de QRS différentes.

Les signaux de bruit proviennent de la base *MIT-BIH Noise Stress Test Database* (Moody et coll., 1984). Les enregistrements de bruits ont été acquis sur des volontaires physiquement actifs en utilisant des enregistreurs ECG et des électrodes standard. Les électrodes ont été placées sur les membres à des endroits où les ondes ECG n'étaient pas visibles. Des extraits de types de bruit sont représentés Figure 4.7.

4.3.4 Mise en œuvre et test des détecteurs de QRS

La mise en œuvre des détecteurs suit le schéma exposé en section 4.2.2. Cependant une adaptation a due être faite pour utiliser les détecteurs de la manière la plus efficace possible. Cette section présente ces différentes adaptations et les résultats obtenus.

Constitution des détecteurs de QRS Actuellement, il serait peu réaliste de considérer un détecteur sans un filtrage préalable des données. Pour plus de cohérence, le filtre utilisé est le même que celui décrit dans la section 4.2.1. Dans cette étude, tous les signaux de contexte sont donc filtrés. Cette étape est très importante car le filtre déforme le signal ECG (Jenkins et Caswell, 1996). Les performances des détecteurs de QRS sont donc aussi très liées à la qualité du filtrage.

Méthodes de seuillage Deux méthodes de seuillage ont été développées :
- la première utilise le seuillage proposé par les auteurs,
- la deuxième utilise un seuillage global.

La comparaison de ces deux méthodes sur l'ensemble de la base d'arythmies MIT-BIH, a montré la supériorité du seuillage « global » sauf pour *pan* et *kadambe* qui dans ce cas conservent leur seuillage très spécifique. La méthode de seuillage global reprend celle qui a été proposée dans (Benitez et coll., 2001). Pour rappel, le seuil est calculé à partir de l'échantillon d'amplitude maximum et de la valeur efficace du signal qui représentent respectivement l'amplitude des QRS et l'amplitude du bruit. Après le seuillage, les QRS concurrents sont départagés en fonction de l'amplitude des échantillons dans l'ECG original et en fonction de l'intervalle RR précédent.

Calcul des scores des détecteurs Le *calcul du score* est effectué en évaluant la capacité des algorithmes à détecter correctement les QRS qui apparaissent dans le signal. Toutes les dates des véritables QRS sont repérées par des annotations. Un événement de type QRS est détecté si un segment du signal est au dessus du seuil et la date d'occurrence de l'événement correspond au maximum de ce segment. Si un événement appartient à une fenêtre d'acceptation autour d'une annotation de QRS alors il est annoté vrai positif (VP), sinon il apparaît comme un faux positif (FP). Si plus d'un événement appartiennent à la même fenêtre, un seul est étiqueté VP, les autres sont ignorés. Les faux négatifs (FN) sont les annotations de QRS qui n'ont pu être associées à aucun événement. Les détecteurs ont généralement soit un grand nombre de FP soit un grand nombre de FN ce qui rend la comparaison des performances difficile. Pour donner un critère global sur lequel fonder la comparaison, le nombre d'erreurs $Ne = FP + FN$ et le taux d'erreur $Te = \frac{Ne}{VP+FN}$, où $VP + FN$ correspond au nombre total d'annotations de QRS, sont calculés. Deux critères peuvent ensuite être considérés : la probabilité de détection (ou sensibilité) $Pd = \frac{VP}{VP+FN}$ qui donne le nombre de QRS détectés sur le nombre total de QRS présents et le taux de fausses alarmes $Tfa = \frac{FP}{VP+FP}$ qui donne le taux de fausses détections parmi l'ensemble des détections. La spécificité, définie par $Sp = \frac{VN}{VN+FP}$, souvent estimée dans la littérature, n'est en fait

pas suffisamment informative. En effet, dans le cas de la détection de QRS, le nombre de VN (Vrais Négatifs) est bien supérieur au nombre de faux positifs (de l'ordre de 1000 fois plus), ce qui implique que les variations de FP sont quasiment invisibles avec cette mesure. C'est pourquoi, Tfa a été préféré car ce critère met en jeu des valeurs du même ordre de grandeur.

Comparaison des résultats Les tables 4.1 et 4.2 montrent les différences entre les performances données dans la littérature (Table 4.1) et celles acquises dans l'expérimentation (Table 4.2).

Les performances de $df2$ et $af2$ avec une ensemble de données réelles sont inconnues. Le détecteur *benitez* est légèrement moins bon mais dans l'article original (Benitez et coll., 2001), l'algorithme n'a pas été testé sur les enregistrements les plus difficiles de la base d'arythmies MIT-BIH. Le détecteur *gritzali* présente lui aussi des performances moins bonnes mais l'ensemble de test utilisé dans l'article original (Gritzali, 1988) n'est pas standard et ne peut donc être vérifié. Qui plus est, les résultats présentés concernaient une détection à partir de plusieurs sources d'ECG. Le détecteur *kadambe* présente des performances supérieures. Cette différence peut s'expliquer par le fait que la base de données utilisée dans cette étude n'est pas la même. Le détecteur *mobd* est véritablement moins bon mais la méthode de seuillage présentée dans l'article original (Suppappola et Sun, 1994) est incohérente et la mise en œuvre directe sans adaptation conduit à des résultats extrêmement médiocres. Enfin, les résultats de *pan* présentés dans l'article original (Pan et Tompkins, 1985) sont entachés d'erreurs. En effet, les auteurs ont fourni plusieurs errata sur l'algorithme et les résultats publiés comprennent beaucoup d'erreurs de calcul. Une fois ces erreurs corrigées les performances de détection baissent sensiblement.

L'utilisation de la méthode de seuillage globale ne laisse aucun doute sur l'augmentation de performance qu'elle entraîne. Ceci montre bien que l'étage de seuillage et de décision a un rôle primordial sur la qualité du détecteur. C'est cette méthode globale qui est utilisée par défaut dans les algorithmes, excepté pour les détecteurs *pan* et *kadambe* qui utilisent un seuillage très spécifique.

4.3.5 Méthode de comparaison des algorithmes

La méthode de comparaison considère deux types de contexte présents en même temps mais de nature différente. Le bruit clinique est indésirable et, en général, on cherche à le réduire alors que le contexte arythmique contient de l'information pertinente pour l'analyse de l'ECG. La constitution des contextes est donc l'agrégation de ces deux types de contextes. La méthode de comparaison se déroule en huit phases résumées dans la Table 4.3.

Pour construire la base de signaux de test, différents types de QRS sont combinés avec du bruit additif réel. Les morphologies de QRS sont extraites des 17 enregistrements sélectionnés (*cf.* 4.3.3). Pour simuler le recueil des données par buffer des systèmes d'acquisition, les signaux sont traités par blocs de 4096 échantillons. Cette

détecteurs	n. battements	FP	FN	Ne	Te(%)	Pd(%)	Tfa(%)
benitez	45856	30	28	58	0.13	99.94	0.07
gritzali	14292	47	2	49	0.34	99.99	0.33
kadambe	8598	420	272	692	8.05	96.84	4.80
mobd	358551	11633	13253	24886	6.94	96.30	3.26
pan	116137	507	277	784	0.68	99.76	0.44
pan(*)	109494	507	277	784	0.72	99.75	0.46

TAB. 4.1 – Résultats de détection de QRS dans la littérature
(*) après corrections

détecteurs	n. battements	seuillage original			seuillage global		
		Te(%)	Pd(%)	Tfa(%)	Te(%)	Pd(%)	Tfa(%)
af2	109494	61.41	75.37	32.79	36.28	98.50	26.09
benitez	109494	1.19	99.45	0.63	1.19	99.45	0.63
df2	109494	9.78	93.54	3.43	6.66	94.26	0.97
gritzali	109494	3.46	99.25	2.66	3.06	99.61	2.61
kadambe	109494	3.23	99.21	2.40	3.23	99.21	2.40
mobd	109494	10.13	95.42	5.50	6.70	99.54	5.90
pan	109494	1.78	98.34	0.12	1.78	98.34	0.12

TAB. 4.2 – Résultats de détection de QRS selon deux méthodes de seuillage.

longueur correspond à une dizaine de QRS pour une durée moyenne de 11 secondes (pour une fréquence d'échantillonnage de 360Hz).

La première étape, appelée *rééchantillonnage*, consiste à construire les contextes appelés *morphologiques*. Huit morphologies de QRS et rythmes ont été extraits de 17 signaux dont le nombre varie entre 30 et 23000. Une méthode pseudo-bootstrapping, adaptée de Zoubir et Boashash (Zoubir et Boashash, 1998), est effectuée pour sélectionner un ensemble de contextes morphologiques plus grand à partir du nombre limité d'individus disponibles dans les enregistrements originaux. Les contextes morphologiques sont stockés dans des buffers de longueur définie. La figure 4.7 présente un exemple de signal contextuel aux multiples étapes de la constitution de contexte.

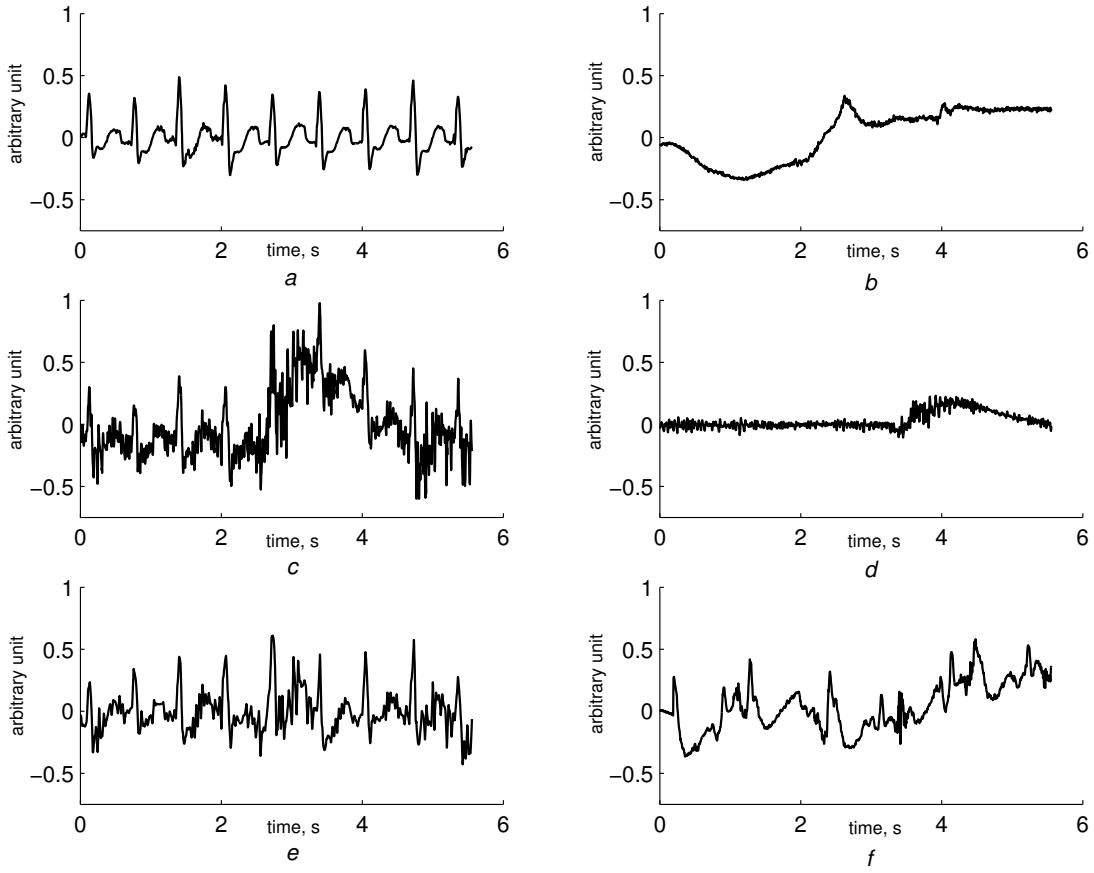


FIG. 4.7 – Exemple de contexte aux différentes étapes de l'analyse de performance de détection de QRS. a) classe de contexte `lbbb`; b) ondulations de ligne de base; c) contexte corrompu par d) du bruit musculaire; e) sortie du filtre; f) mouvements d'électrode.

Chaque buffer contient un seul contexte morphologique. Comme les données qui composent un buffer peuvent provenir de plusieurs enregistrements différents, on ef-

fectue la *liaison* des QRS en utilisant une approximation sigmoïde aux bords pour minimiser les sauts de raccord. Par exemple, un buffer de taille 10 peut contenir un segment de bigéminisme de 5 QRS qui provient d'un enregistrement particulier et un autre segment de bigéminisme de 5 QRS qui provient d'un autre enregistrement. Un raccord doit alors être établi entre ses deux segments.

Afin de réaliser l'*ajout du bruit*, chaque bruit est additionné au signal de référence suivant trois RSB : 5, -5, -15 dB. Le RSB est calculé selon les formules suivantes :

$$RSB = 10 * log_{10} \frac{P(S)}{P(N * \alpha)}$$

soit

$$\alpha = \sqrt{\frac{P(S)}{P(N)} * 10^{-\frac{RSB}{10}}}$$

où la variable α contrôle l'énergie du bruit ajouté et où $P(X)$ est l'énergie de X. Bien entendu, les contextes de morphologies sont aussi testés sans ajout de bruit.

Le *filtrage* est effectué pour retirer le plus de bruit possible. Il importe de rappeler que nous nous intéressons ici à l'influence du bruit qui ne peut pas être filtré aisément. Chaque réalisation de chaque contexte est utilisée comme entrée de chaque détecteur pour lequel le *calcul du score* est effectué.

Dans les tests $B = 10$ battements, $R = 100$ réalisations, $M = 8$ morphologies et $N = 10$ types de bruit (dont un sans bruit et *bw*, *ma*, *em* à trois RSB) pour $D = 7$ détecteurs. Ce qui conduit à $100 * 7 * 8 * 10 = 56000$ valeurs. Pour une telle masse de données, les résultats intéressants peuvent être difficiles à analyser et à extraire. C'est pourquoi, une analyse en composantes principales (ACP) a été effectuée sur les données pour faire ressortir les différentes classes de contexte intéressantes.

4.3.6 Analyse en composantes principales

Cette section présente succinctement les objectifs de l'analyse en composantes principales (ACP) et les critères importants pour l'évaluation. Une description détaillée de l'ACP peut être trouvée dans (Bertier et Bouroche, 1977).

Le but d'une ACP est de représenter un large ensemble de données multidimensionnelles dans un espace réduit en maximisant l'inertie de nouveaux axes orthogonaux. Par exemple, pour une représentation à une seule dimension, on cherchera la droite la plus « proche » des points du nuage et le nuage sera représenté par la projection des points sur cette droite ; la proximité du nuage de points à la droite sera mesurée par l'*inertie* du nuage par rapport à cette droite. L'axe d'inertie minimum est appelé *premier axe principal*. Le plan d'inertie minimum est constitué des deux axes orthogonaux d'inertie minimum (voir Figure 4.8).

Le nuage de points original est une matrice X de dimension n, p représentant les valeurs quantitatives prises par n individus décrits par p variables. Une fois les axes principaux trouvés chaque point est projeté orthogonalement sur les axes comme illustré Figure 4.8. L'axe est calculé de façon à minimiser l'inertie du nuage par rapport à l'axe.

TAB. 4.3 – Algorithme de rééchantillonnage (pseudo-bootstrap) pour calculer les scores des détecteurs de QRS

Init.	*Données* : pour un contexte donné, supposer un ensemble Q de BT battements
Étape 1	*Rééchantillonnage* : effectuer une sélection de B battements, avec retirage, à partir de Q pour obtenir la population rééchantillonnée Q^*
Étape 2	*Liaison* : concaténer les B battements de Q^* en utilisant une approximation sigmoïde aux bords pour minimiser les sauts de raccord entre deux battements
Étape 3	*Ajout du bruit* : corrompre les battements concaténés par du bruit réel additif à un RSB donné pour obtenir S. Pour la génération de contextes sans bruit, l'étape 3 est ignorée
Étape 4	*Filtrage* : filtrer S pour obtenir le signal S'
Étape 5	*Calcul du score* : utiliser S' comme entrée des D détecteurs et calculer le score
Étape 6	*Répétition* : répéter l'étape 1 à 5 pour obtenir R réalisations d'un même contexte
Étape 7	*Répétition* : répéter l'algorithme pour les M contextes morphologiques associés aux N bruits pour collecter tout les scores
Étape 8	*Présentation des résultats* : projeter tous les scores sur les axes principaux (ACP) pour analyser visuellement les dispersions

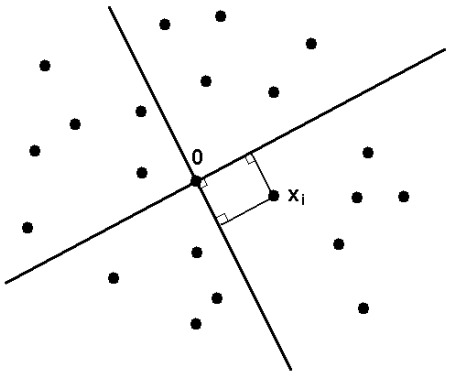


FIG. 4.8 – Nuage de points en trois dimensions d'une matrice d'individus.

Pour interpréter les résultats d'une ACP normée (centrée réduite), il est nécessaire de vérifier la qualité de représentation des variables. Cette qualité peut être estimée à partir des valeurs propres de la matrice d'inertie X^tX. Plus la valeur propre associée à un axe l (λ_l) est grande, plus l'axe associé représente de variables et plus il sera intéressant. La valeur $\lambda = 1$ est donc un point de repère dans le sens où un axe associé

à une valeur propre inférieure à 1 synthétise moins de données qu'une variable isolée. On calcule généralement l'inertie expliquée pour chaque axe qui est donnée par $\varphi_l = \frac{\lambda_l}{Tr(X)}$ où $Tr(X)$ est la trace de X (la longueur de la diagonale).

Lorsque deux axes principaux sont retenus, la contribution des variables originales à un axe particulier peut être évaluée par le cercle des corrélations. Dans le cercle de rayon 1 centré à l'origine des deux axes, les variables sont projetées. L'angle formé par la droite passant par l'origine et un point-variable et la droite d'un axe principal donne la corrélation entre la variable et l'axe. Plus l'angle sera petit plus l'axe sera corrélé avec la variable. Enfin, plus le point sera proche de l'unité plus la variable sera représentative de l'axe.

4.3.7 Résultats et règles de pilotages

Deux analyses distinctes ont été réalisées :
- La première organise les scores Ne en une matrice **M1** dont les lignes représentent les détecteurs pour les différentes réalisations et les colonnes représentent les morphologies et les bruits ;
- La seconde organise les scores Ne en une matrice **M2** dont les lignes sont composées des valeurs des réalisations pour chaque morphologie et bruit et les colonnes représentent les détecteurs.

Une ACP a été appliquée à ces matrices dans le but d'étudier :

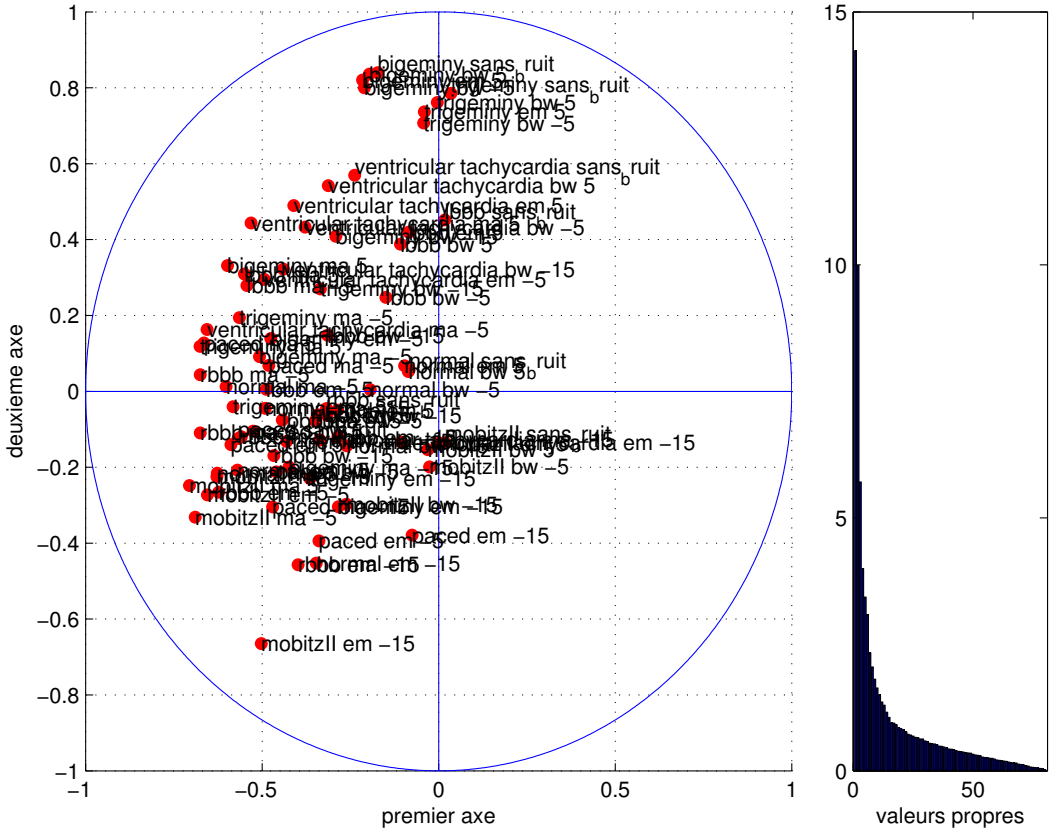
1. la dépendance des performances des détecteurs selon les contextes (ACP sur **M1**),
2. les contextes selon les performances des détecteurs (ACP sur **M2**).

Pour chaque ACP, quatre groupes de contextes particuliers sont analysés : les contextes morphologiques (sans bruit), les ondulations de ligne de base (*bw* à tous les RSB), les artefacts musculaires (*ma* à tous les RSB), les artefacts de mouvements des électrodes (*em* à tous les RSB). Le cercle des corrélations et les deux composantes principales (c.-à-d. les individus projetés) sont affichés. Chaque axe est repéré par une ligne passant par l'origine. Pour chaque groupe, différents symboles sont utilisés et deux propriétés sont analysées : le barycentre (affiché en noir), qui donne une information sur la qualité générale de détection et la dispersion, qui donne une information sur la stabilité de la détection.

4.3.7.1 Performance des détecteurs selon le contexte (M1)

Cinq ACP sont réalisées sur **M1**, une pour la matrice globale et une pour chaque groupe de contextes. Les résultats obtenus sur la matrice globale **M1** de taille 700x80, représentés Figure 4.9, montrent les performances globales des détecteurs.

La valeur propre associée au premier axe vaut 14.23 ce qui signifie que le premier axe résume environ 14 variables. Le premier axe possède ainsi une inertie expliquée $\varphi_1 = 17.8\%$ et le deuxième une inertie expliquée $\varphi_1 2 = 12.5\%$. Pour des données initiales dans $\mathbf{R}^{80}$ projetées dans un plan $\mathbf{R}^2$, 30.3% d'inertie expliquée cumulée consitute une



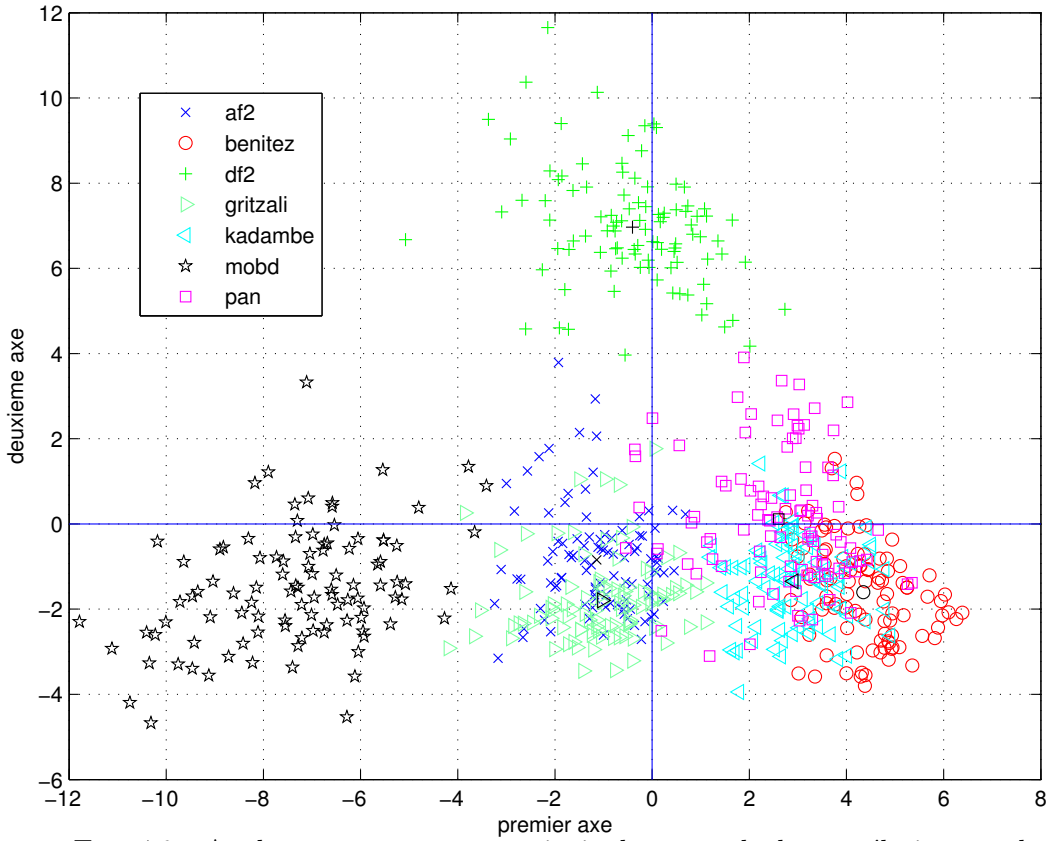


FIG. 4.9 – Analyse en composantes principales et cercle des corrélations sur les résultats globaux.

bonne qualité de représentation. Le cercle des corrélations indique que le premier axe est corrélé négativement avec l'ensemble des scores des détecteurs. Une valeur négative sur le premier axe est donc représentative d'une grande valeur de Ne. Autrement dit, le premier axe donne les performances moyennes des détecteurs. Plus les projections appartenant à un détecteur seront positives meilleur ce détecteur sera. L'analyse de l'ACP donne ainsi le détecteur *benitez* comme étant le meilleur en moyenne. Il est suivi de près de *kadambe* et de *pan* puis du groupe $df2$, *gritzali* et $af2$ et enfin du *mobd*. Le cercle des corrélations indique que le deuxième axe confronte les classes `bigeminy` et `ventricular tachycardia` pour le type de bruit *bw* et sans bruit (corrélation positive) aux classes `mobitzII`, `paced` et `rbbb` pour le type de bruit *em* à $-15dB$ (corrélation négative). Ceci implique que le détecteur $df2$ est fortement perturbé par les classes `bigeminy` et `ventricular tachycardia` tandis qu'il est relativement indépendant des classes `mobitzII`, `paced` et `rbbb` pour le type de bruit *em* à $-15dB$ alors qu'elles perturbent fortement les autres détecteurs.

Les résultats sont beaucoup plus informatifs que l'étude réalisée sur les simples enregistrements (*cf.* 4.3.4). Pour le premier axe, on voit que certains détecteurs tels que *mobd* ne résistent pas à la hausse du bruit tandis que d'autres tels que *kadambe* se montrent particulièrement résistants. Le deuxième axe indique que les contextes *em* et `mobitzII` sont particulièrement favorables au détecteur $df2$.

Ce type d'étude, qui montre déjà un degré supérieur d'interprétation que les études classiques de comparaison, peut être approfondi. Quatre études complémentaires ont été réalisées pour obtenir une comparaison sur des contextes spécifiques.

- Une ACP sur des contextes morphologiques (700x8) montre pour le premier axe ($\varphi_1 = 25.9\%$), les mauvaises performances de l'ensemble des détecteurs pour la classe `ventricular tachycardia` et `bigeminy` notamment pour le détecteur $df2$. Les détecteurs *benitez*, *gritzali* et *kadambe* sont les plus résistants. Le deuxième axe ($\varphi_2 = 16.9\%$) montre surtout les bons scores de $df2$ pour la classe `mobitzII` et `paced`.
- Une ACP sur des contextes *bw* (700x24) montre, pour le premier axe ($\varphi_1 = 19.7\%$), les faibles performances de $af2$ et $df2$. Le deuxième axe ($\varphi_2 = 14.6\%$) souligne encore une fois les bons scores de $df2$ pour la classe `mobitzII` et `paced`.
- Une ACP sur des contextes *ma* (700x24) montre pour le premier axe ($\varphi_1 = 37.6\%$) que *kadambe* est le meilleur détecteur pour ce type de bruit devant *benitez* et $af2$. *gritzali*, $df2$ et *mobd* sont fortement perturbés. Le deuxième axe ($\varphi_2 = 7.15\%$) conforte les bons scores de $df2$ pour la classe `mobitzII` et `paced` pour ce type de bruit.
- Une ACP sur des contextes *em* (700x24) est représentée Figure 4.10. Le cercle des corrélations indique que le premier axe ($\varphi_1 = 21\%$) est corrélé négativement aux contextes $RSB \leq -5dB$ tandis que le deuxième axe ($\varphi_2 = 12.3\%$) est positivement corrélé aux classes `bigeminy` et `ventricular tachycardia`. Les projections montrent que $df2$ est le détecteur le plus résistant au bruit *em* en moyenne. Le deuxième axe indique toutefois qu'il reste perturbé par les classes `trigeminy`, `bigeminy` et `ventricular tachycardia` pour un $RSB \geq -5$.

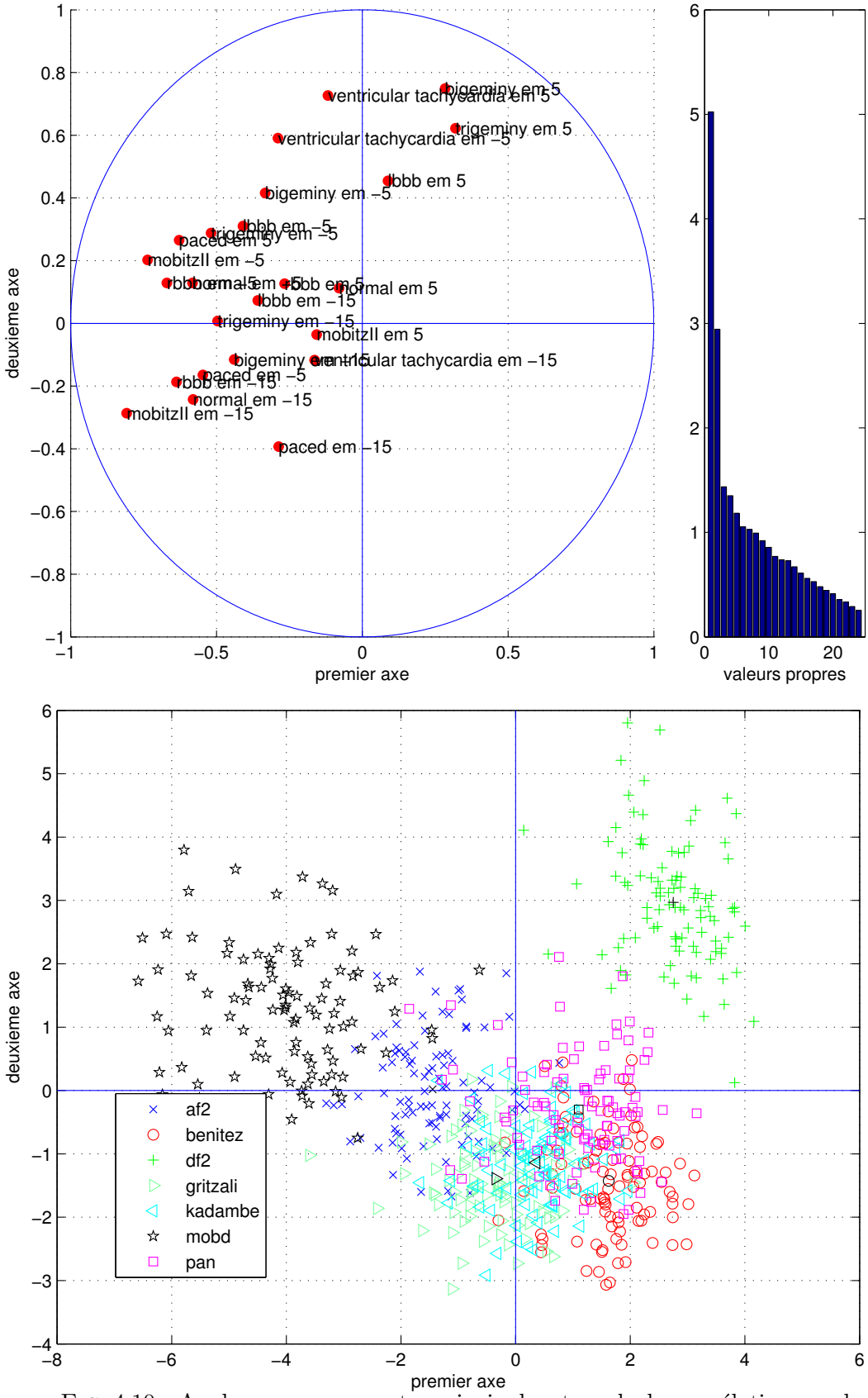


FIG. 4.10 – Analyse en composantes principales et cercle des corrélations sur les contexte *em*.

4.3.7.2 Influence des contextes sur les performances des détecteurs (M2)

L'ACP sur les contextes morphologiques (8000x7), présentée Figure 4.11 permet d'analyser les contextes arythmiques qui ont le plus d'influence sur les performances des détecteurs. Cette ACP montre les différences entre le groupe `bigeminy`, `ventricular tachycardia` `trigeminy`, la classe `paced` et le reste des contextes.

Le premier axe ($\varphi_1 = 58.07\%$) résume beaucoup de variables et montre les performances globales obtenues tandis que le deuxième axe ($\varphi_2 = 13.90\%$) se focalise surtout les performances de *gritzali* et $af2$. On peut voir que sur le premier axe, les QRS appartenant à la classe `ventricular tachycardia` sont très durs à détecter. La classe `trigeminy` est également très dure à traiter car composée d'une succession de QRS normaux et d'extrasystoles. La classe `paced` présente aussi un type de QRS difficile à extraire notamment pour *gritzali* et $af2$ si on tient compte du deuxième axe.

Des tests similaires ont été effectués pour *bw*, *ma*, et *em*. Les résultats montrent que *bw* n'affecte pas beaucoup les détecteurs alors que *ma* et *em* perturbent fortement la détection.

4.3.7.3 Résultats généraux et inférences de règles

L'ACP a permis de mettre en exergue les données intéressantes présentes dans les résultats bruts et d'obtenir, après interprétation des plans, les règles de pilotage présentées Figure 4.12. Il convient de préciser qu'il s'agit ici d'une vraie fouille de données car les Ne représentent les performances de 7 détecteurs dans 8000 situations distinctes. Ces règles ont été obtenues manuellement à partir des scores moyens mais une méthode d'apprentissage telle qu'un arbre de décision permettrait de les acquérir automatiquement.

D'une manière générale, le détecteur *benitez* est jugé le meilleur. Ceci est sûrement dû à son étage de filtrage et son système de décision particulièrement efficace qui inclut déjà de l'intelligence puisque la connaissance de l'intervalle RR est prise en compte dans la décision. Il doit donc être choisi par défaut.

Comme montré par l'ACP, le *kadambe* présente les meilleures performances dans le contexte *ma*. Ceci est sûrement dû à la décomposition en ondelettes qui permet de filtrer au maximum le bruit. L'algorithme *mobd* est inférieur à tous les autres détecteurs mais il est particulièrement peu coûteux en calcul. Le détecteur *pan* est particulièrement adapté à la classe `paced`. Ce détecteur a des performances très proches de *benitez* mais son filtrage permet de mieux gérer les cas de pacemaker. Le détecteur $af2$ a des performances toujours en dessous d'un autre détecteur cependant, d'un point de vue algorithmique, il est aussi l'un des plus simple. L'algorithme $df2$ possède de bons résultats pour le bruit *em* $RSB = -15$. Il est de plus, le meilleur détecteur pour traiter la classe `mobitzII`. Tout comme remarqué dans l'étude de Friesen et coll. (Friesen et coll., 1990), le détecteur $af2$ est très sensible au bruit basse fréquence (*bw*) tandis que $df2$ est plutôt perturbé par le bruit haute fréquence (*ma*). Enfin, le détecteur *gritzali* est particulièrement adapté aux classes `ventricular tachycardia` et `trigeminy` lorsqu'il y a peu de bruit de ligne. En effet, ce détecteur ne contient pas de filtrage pouvant déformer les extrasystoles qui composent ce type de rythme. Qui plus est, les extrasystoles

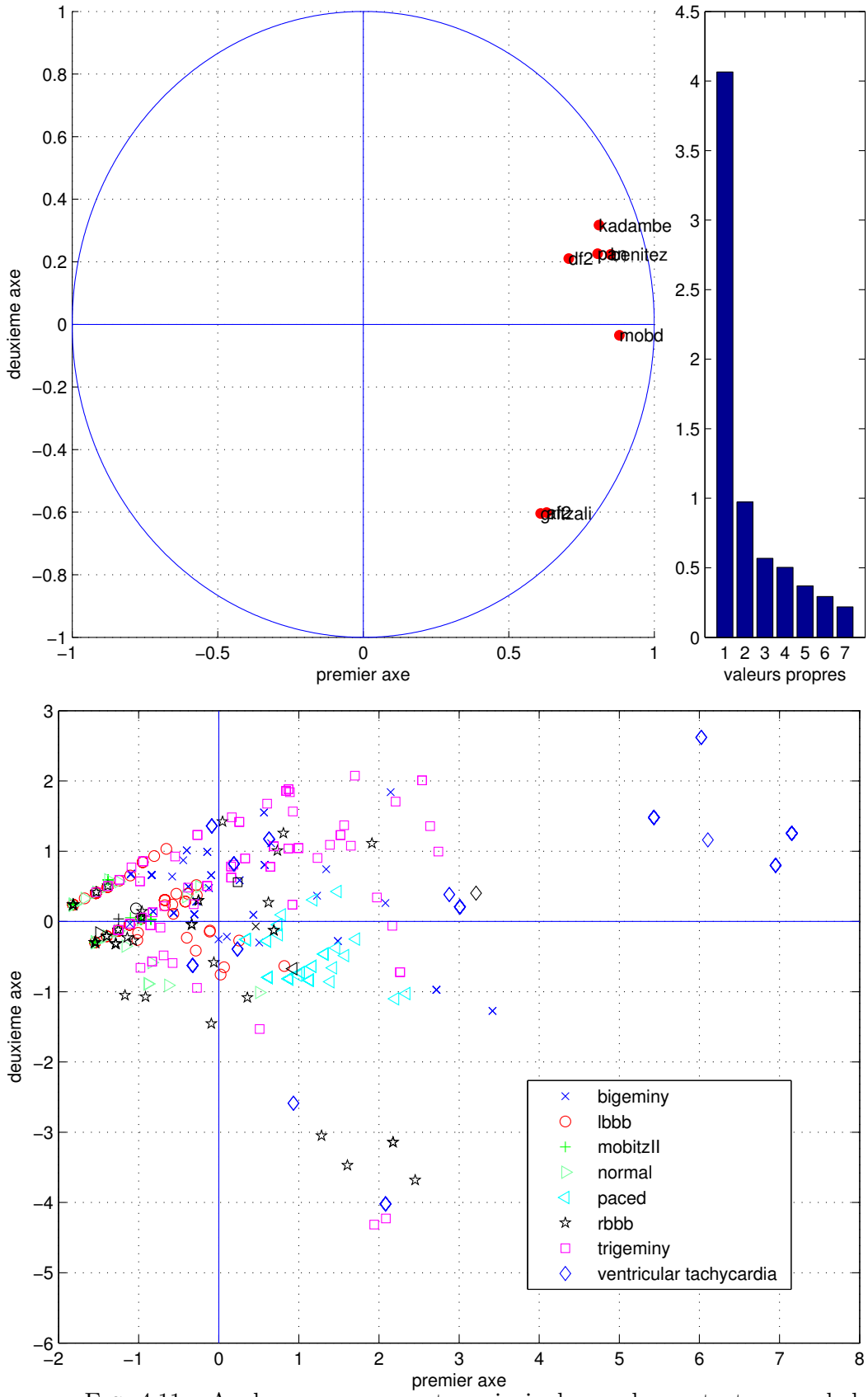


FIG. 4.11 – Analyse en composantes principales sur les contextes morphologiques.

Si `trigeminy` $\wedge$ *sans bruit* **alors**
choisir gritzali
Fin si
Si `ventricular tachycardia` $\wedge$ *sans bruit* **alors**
choisir gritzali
Fin si
Si `trigeminy` $\wedge$ *bw* $\wedge$ $RSB > -5dB$ **alors**
choisir gritzali
Fin si
Si `ventricular tachycardia` $\wedge$ *bw* $\wedge$ $RSB > -5dB$ **alors**
choisir gritzali
Fin si
Si `mobitzII` **alors**
choisir df2
Fin si
Si `normal` $\wedge$ *em* $\wedge$ $RSB = -5dB$ **alors**
choisir df2
Fin si
Si *em* $\wedge$ $RSB = -15dB$ **alors**
choisir df2
Fin si
Si `paced` $\wedge$ *sans bruit* **alors**
choisir pan
Fin si
Si `paced` $\wedge$ *bw* **alors**
choisir pan
Fin si
Si `bigeminy` $\wedge$ *ma* $\wedge$ $RSB = -5dB$ **alors**
choisir kadambe
Fin si
Si `lbbb` $\wedge$ *ma* **alors**
choisir kadambe
Fin si
Si `rbbb` $\wedge$ *ma* **alors**
choisir kadambe
Fin si
Si `paced` $\wedge$ *ma* **alors**
choisir kadambe
Fin si
Si `normal` $\wedge$ *ma* **alors**
choisir kadambe
Fin si
Si `trigeminy` $\wedge$ *ma* **alors**
choisir kadambe
Fin si
Si `lbbb` $\wedge$ *em* $\wedge$ $RSB > -15dB$ **alors**
choisir kadambe
Fin si
Si *défaut* **alors**
choisir benitez
Fin si

FIG. 4.12 – Règles de pilotages inférées à partir des résultats.

ont souvent des formes variables et une transformée basée sur l'énergie telle que celle de *gritzali* est plus stable qu'une transformée basée sur la dérivée telle que celle utilisée par *benitez*. Ces résultats sont importants dans la mesure où les rythmes pathologiques tels que la tachycardie ventriculaire, sont souvent composés d'extrasystoles. Le choix du bon détecteur est donc primordial pour la reconnaissance de ces arythmies.

4.4 Conclusion

L'un des buts du système IP-CALICOT est de réaliser le pilotage des algorithmes de l'abstraction temporelle. Pour cela, il est nécessaire d'acquérir les règles de pilotage. Même si les algorithmes utilisés dans IP-CALICOT proviennent tous de la littérature, il est important de remarquer que le manque d'études comparatives approfondies ne permet pas de dériver directement les règles qui associent un algorithme à un contexte particulier. Ceci a conduit à mettre au point une méthode d'analyse de performances, focalisée sur l'étude des détecteurs de QRS, pour acquérir les règles de pilotage de

ces algorithmes et améliorer la tâche de détection de QRS. Cette analyse a mis en évidence les caractéristiques des algorithmes d'une manière originale et a montré que les détecteurs choisis sont bien complémentaires en terme de performance et que leur pilotage a un sens. De plus, ces résultats permettent de valider la définition donnée des contextes. En effet les performances des algorithmes de détection dépendent non seulement du bruit de ligne mais aussi du contexte arythmique.

La méthode proposée n'est pas réduite à la seule détection de QRS et d'autres types d'algorithmes tel que les classificateurs de QRS et les détecteurs d'ondes P peuvent être évalués de la même manière.

- **Les filtres** peuvent être analysés pour mesurer la distorsion appliquée à l'ECG. Des algorithmes classiques peu coûteux en terme de ressources doivent être comparés aux algorithmes basés sur des ondelettes.
- **La classification de QRS** pourrait posséder plusieurs types d'algorithmes. Par exemple, il serait intéressant de mettre en œuvre un classificateur de type *pattern matching* (Agilent Technologies, 2000). L'étude de ces classificateurs selon différentes morphologies de QRS est bien évidemment courante étant donné que le but d'un classificateur est justement de réagir à plusieurs types d'ondes. Cependant, il y a peu, à ma connaissance, d'études approfondies sur la tenue des classificateurs à différents types et niveaux de bruit.
- **La détection d'onde P** pourrait aussi être effectuée par plusieurs types d'algorithmes. Par exemple, il serait intéressant de développer un détecteur avec recherche arrière dans une fenêtre qui est moins fiable mais moins coûteux que le détecteur avec annulation de QRS-T. Là aussi, la robustesse au bruit et à différentes morphologies mérite d'être étudiée.

Chapitre 5

Expérimentations

5.1 Introduction

Dans ce chapitre, le système IP-Calicot est testé avec différents signaux réalistes caractéristiques de situations cliniques. Ces tests sont autant destinés à évaluer les règles de pilotage que les algorithmes de traitement du signal, les modèles de chroniques et l'application de monitorage. Dans cette étude, nous nous plaçons volontairement dans des situations qui sortent des « cas d'école ». Des ECG sont bruités pour tester les performances du système dans des cas extrêmes. Les données sur lesquelles les tests ont été effectués sont présentées en section 5.2.

À partir de ces données, le système analyse le contexte courant pour : choisir les modèles de chroniques de la reconnaissance d'arythmies, activer et désactiver les tâches de l'abstraction temporelle et choisir les algorithmes de traitement du signal. Ainsi, l'opération de monitorage emploie toujours les traitements les plus adaptés au contexte courant. Pour faire la part des améliorations apportées par les types de pilotage, les résultats obtenus sur le même ensemble de test sont présentés selon trois axes :

1. L'apport du pilotage au niveau des algorithmes est détaillé en section 5.3. Cette analyse permet d'étudier la validité des règles de pilotage d'algorithmes et de quantifier l'accroissement de qualité de l'abstraction temporelle.
2. La hiérarchie des modèles de chroniques est évaluée en section 5.4. Les performances sont obtenues en effectuant la reconnaissance d'arythmies avec les annotations d'événement. Ces résultats permettent de quantifier la qualité des règles et de donner le seuil de reconnaissance maximal atteignable par le système de pilotage sur des données de test.
3. Les performances du système de pilotage complet (trois niveaux de pilotage) sont détaillées en section 5.5. Pour donner une échelle de comparaison, les résultats obtenus sans et avec pilotage sont discutés.

Il convient de préciser que le pilotage des tâches de l'abstraction temporelle et des modèles de chroniques est réalisé uniquement avec l'information de contexte de ligne (analyse du bruit de ligne). De même, pour focaliser l'étude des performances unique-

ment sur les trois niveaux de pilotage, les différents contextes sont étiquetés ce qui revient à utiliser des analyseurs de contexte parfaits.

5.2 Données de test

Pour réaliser les signaux de test, huit ECG ont été extraits de la base d'arythmies MIT-BIH (Mark et Moody, 1988), il s'agit des enregistrements reportés Table 5.1.

signal de test	enregistrement original	`bigeminy`	`doublet`	`mobitzII`	`normal`	`pvc`	`vt`
cont_100	100				X		
cont_101	101				X		
cont_103	103				X		
cont_106	106	X	X		X	X	X
cont_119	119	X			X	X	
cont_222	222				X		
cont_223	223	X	X		X	X	X
cont_231	231			X	X (rbbb et normal)		

TAB. 5.1 – Enregistrements utilisés pour composer les signaux de test.

Ces enregistrements ont été modifiés pour inclure les annotations des ondes P. Ils représentent au total quatre heures d'enregistrement. Pour tester les signaux dans des situations qui sortent des « cas d'école » du bruit réel est ajouté aléatoirement parmi les valeurs possibles du contexte de ligne (*cf.* 4.3.2). La figure 5.1 montre un extrait d'ECG bruité. Sur cette figure on voit bien que l'information utile est très difficile à extraire lorsque le rapport signal-à-bruit (RSB) devient très faible. L'ajout artificiel de bruit permet de connaître exactement le RSB lors de l'évaluation des performances. Les sections de bruit ont été ajoutées sur des segments variant de 10 à 50 QRS et proviennent toutes de la base de bruit clinique *MIT-BIH noise stress test database* (Moody et coll., 1984) décrite en section 4.3.3. Chaque ECG est composé d'une cinquantaine de contextes de façon à évaluer non seulement les performances des détecteurs dans ces contextes, mais aussi, le comportement du pilotage au moment des transitions entre contextes.

5.3 Résultats du pilotage des algorithmes de traitement du signal

Les règles de pilotage présentées au chapitre 4 permettent de choisir les algorithmes de détection du complexe QRS. Les contextes de ces différents signaux de test ont amené le pilote à utiliser les détecteurs *benitez*, *gritzali*, *kadambe* et *df*2.

Pour chacun des contextes des ECG de test, le pilote décide, avec les règles de pilotage, quel est l'algorithme le plus adapté. Pour chaque signal, les critères de performance décrits en section 4.3.4 (Ne, Te, Tfa et Pd) ont été calculés.

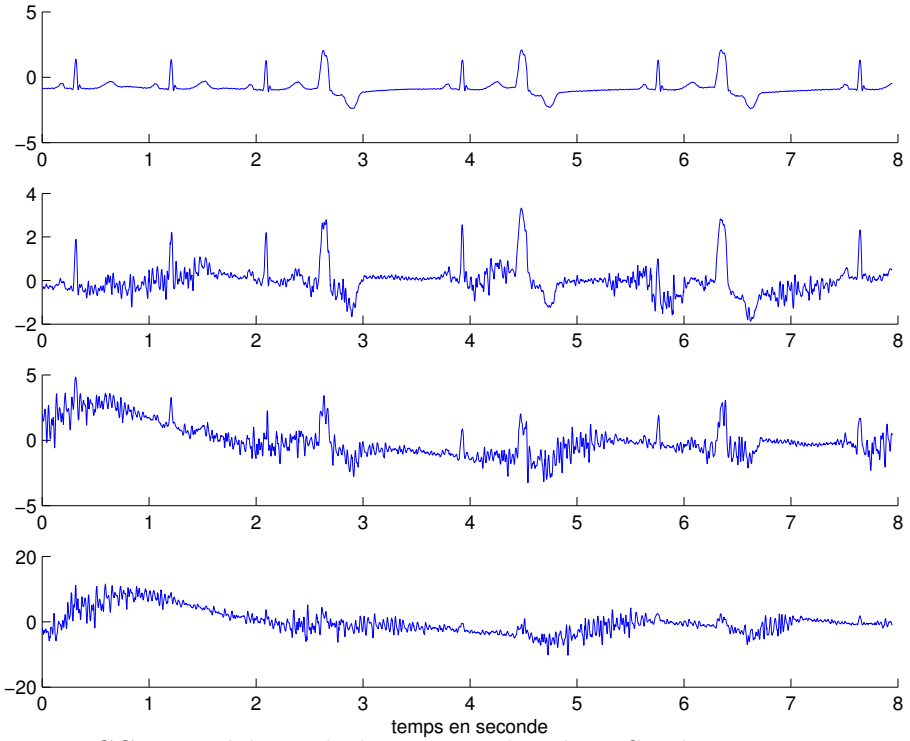


FIG. 5.1 – ECG perturbé par du bruit *ma* selon des RSB de ∞, 5, -5 et $-15dB$.

Résultats globaux du pilotage de la tâche de détection de QRS La table 5.2 donne les résultats de la détection de QRS sur l'ensemble des signaux de test en utilisant chaque détecteur séparément sans pilotage puis en activant le pilotage d'algorithmes.

détecteur	n. batts	VP	FP	FN	Ne	Te(%)	Pd(%)	Tfa(%)
$af2$	16895	15968	7974	927	8901	52.68	94.51	33.31
benitez	16895	16495	1870	400	2270	13.44	97.63	10.18
$df2$	16895	14506	1584	2389	3973	23.52	85.86	9.84
gritzali	16895	15655	4343	1240	5583	33.05	92.66	21.72
kadambe	16895	16317	2043	578	2621	15.51	96.58	11.13
mobd	16895	16411	5858	484	6342	37.54	97.14	26.31
pan	16895	14434	826	2461	3287	19.46	85.43	5.41
pilote	16895	16237	1470	658	2128	12.60	96.11	8.30

TAB. 5.2 – Synthèse des résultats de détection de QRS.

Le pilotage des algorithmes de traitement du signal mis en œuvre permet de réduire le nombre d'erreurs en apportant une détection des événements de l'ECG plus robuste au bruit. Si l'on s'intéresse au critère Te, les résultats présentés permettent de confirmer la classification globale du chapitre 4. Si l'on considère uniquement le point de vue des détecteurs, *benitez* présente les meilleures capacités de détection suivi de *kadambe*, *pan*, $df2$, *gritzali*, *mobd* et $af2$. On peut remarquer que les plus mauvais détecteurs

sont aussi ceux pour lesquels l'étage de seuillage est originalement peu performant. Si l'on compare les résultats du détecteur *benitez* avec les résultats obtenus par le pilote on constate que le pilotage permet de réduire le taux d'erreurs de 13.44% à 12.60% (2270 à 2128 erreurs). Avec le pilote, le nombre d'erreurs ($Ne = FN + FP$) est moins important. Le nombre de FP est beaucoup plus faible, le pilote permet donc de réduire le nombre de fausses alarmes. Par contre le nombre de FN est plus grand, le pilotage doit donc être amélioré pour réduire ce nombre d'événements non détectés.

Détails des résultats du pilotage de la tâche de détection de QRS La figure 5.2 présente les taux d'erreurs des détecteurs utilisés dans le pilotage pour les signaux de tests.

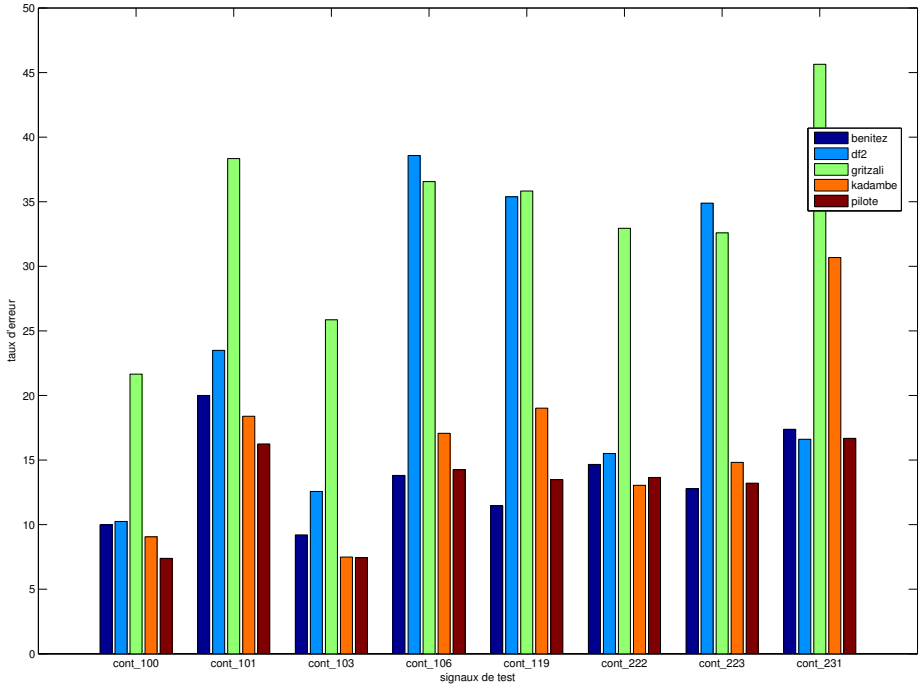


FIG. 5.2 – Taux d'erreur des détecteurs utilisés dans le pilotage sur l'ensemble des enregistrements de test.

Le pilote est le meilleur pour l'ensemble des résultats sauf dans les cas cont_106, cont_119, cont_223, où ses performances sont très proches de celles de *benitez* et dans le cas cont_222 où ses performances sont très proches de celles de *kadambe*. Les enregistrements cont_119, cont_223 comportent des phases de bigéminisme et de trigéminisme qui pourraient expliquer le fait que *benitez* est supérieur au pilote. En effet, le pilote utilise dans ces signaux de test le détecteur *kadambe* ainsi que $df2$ alors qu'utiliser uniquement *benitez* semble un meilleur choix. On peut aussi remarquer que $df2$ est le

meilleur après le pilote dans le cas cont_231. Le signal cont_231 contient beaucoup de phases de mobitz, ce qui explique que le pilote obtient de bonnes performances en utilisant $df2$ dans les contextes de mobitz et les autres détecteurs dans les autres contextes. Ces résultats montrent que dans la majorité des cas, le pilote utilise les détecteurs adéquats pour obtenir les meilleurs performances bien que les règles de pilotage puissent être améliorées notamment dans les cas de bigéminisme. Il convient de noter que le pilote utilise les informations du contexte uniquement pour choisir le détecteur. Ces informations devraient aussi être utilisées pour paramétrer les algorithmes (p. ex. seuils de détection).

Les autres algorithmes de l'abstraction temporelle ont aussi été évalués. Le classificateur de QRS présente un taux de bonnes classifications ($\frac{\text{QRS bien classés}}{VP}$) de 55% ce qui est assez médiocre. Ce résultat n'est pas surprenant car la classification des ondes est généralement peu robuste au bruit. La désactivation de cette tâche dans les cas bruités est donc une solution pour éviter la production d'erreurs de classification. La détection d'ondes P n'est pas mise en œuvre dans le système, elle est simulée à partir des annotations. Le simulateur produit les bonnes sorties lorsque le signal n'est pas bruité et simule des sorties avec 50% d'erreurs lorsque du bruit est présent sur la ligne. Cette simulation parait trop optimiste étant donné que les détecteurs d'ondes P actuels ont encore du mal à traiter tous les cas d'ECG même lorsqu'ils sont propres.

Les tests réalisés sur les algorithmes permettent de conclure que le pilotage d'algorithmes accroît la qualité de l'abstraction temporelle. Les résultats globaux de cette étude montrent surtout que la détection de QRS est améliorée par le pilotage d'un ensemble d'algorithmes selon le contexte courant. Grâce à ce pilotage, le taux d'erreur moyen de 13.44% obtenu avec un seul algorithme de détection de QRS passe à 12.60% avec le pilotage. Les résultats valident les règles de pilotage qui permettent non seulement de piloter ces algorithmes mais aussi d'apporter des informations encore mal connues sur les algorithmes eux-mêmes.

5.4 Test des règles de reconnaissance d'arythmies

5.4.1 Introduction

Afin d'évaluer les capacités intrinsèques du pilotage, il convient d'évaluer les performances maximales que peut atteindre le système de reconnaissance d'arythmies. Pour ce faire, la hiérarchie de modèles de chroniques présentée au chapitre 3 a été expérimentée sur les signaux de tests. Cette reconnaissance de chroniques a été effectuée avec les événements annotés des enregistrements originaux. Cette manière de faire revient à utiliser une abstraction temporelle parfaite. La qualité des règles peut ainsi être évaluée séparément du pilotage. Ces résultats permettent ainsi de donner le seuil de reconnaissance maximal atteignable par le système de pilotage. La méthode de calcul utilisée est tout d'abord présentée en section 5.4.2 puis les résultats sur les données de test sont discutés en section 5.4.3.

5.4.2 Méthode de calcul des performances de reconnaissance d'arythmies

5.4.2.1 Considérations sur le calcul de performance de la reconnaissance d'arythmies

Pour évaluer la qualité de reconnaissance d'arythmies, il convient de mettre au point une méthode de calcul de performances. Ce calcul des performances de reconnaissance des arythmies n'est pas trivial car les arythmies reconnues par le système et les annotations d'arythmies des enregistrements n'utilisent pas la même échelle temporelle. En effet, dans notre cas, une arythmie est reconnue si une des chroniques associées est reconnue sur le signal. On dispose ainsi d'une représentation de cette connaissance sous forme d'une instance de chronique. Le problème est alors de pouvoir associer une instance de chronique à une arythmie présente dans l'enregistrement car une instance de chronique peut chevaucher deux annotations d'arythmie, ne recouvrir qu'une partie d'une annotation, etc. Les paragraphes suivants exposent la méthode adoptée pour quantifier la qualité de reconnaissance.

Format des annotations d'arythmies Dans les annotations des enregistrements, les arythmies sont repérées par une date de début d'arythmie. Si un enregistrement contient trois cycles de bigéminisme successifs, l'annotation sera constituée d'une seule date : celle de début du premier cycle. Si un enregistrement contient seulement un rythme normal (ou bloc de branche) alors il n'y aura qu'une seule annotation pour tout l'enregistrement. La fin d'une arythmie est repérée par le début d'une autre arythmie. De cette manière, tout battement cardiaque appartient soit à une arythmie soit à un rythme normal (il ne peut y avoir simultanément 2 arythmies dans ce type d'annotation).

Le format de la reconnaissance d'arythmies Dans IP-Calicot, les arythmies sont reconnues grâce à la hiérarchie de modèles de chroniques. Chaque instance de chronique est délimitée par une fenêtre temporelle. Dans cette application, les chroniques reconnues couvrent un nombre restreint de battements (3 à 5) et peuvent se recouvrir entre elles (un battement au milieu d'une chronique `normal` peut aussi être le début d'une autre chronique `normal`)

La figure 5.3 illustre les différences de format entre les annotations et les chroniques reconnues. Les chroniques instanciées se recouvrent entre elles et ne respectent pas forcement les limites des annotations de l'enregistrement. Par exemple, une instance de chronique `bigeminy` commence lors d'un rythme annoté normal et finit dans un rythme annoté bigéminisme. Pourtant, cette reconnaissance est visiblement correcte.

Association entre la reconnaissance et l'annotation Le calcul de performance se fait généralement par le calcul d'une matrice de confusion qui donne, pour chaque arythmie, le nombre de bonnes reconnaissances et le nombre de confusions avec les

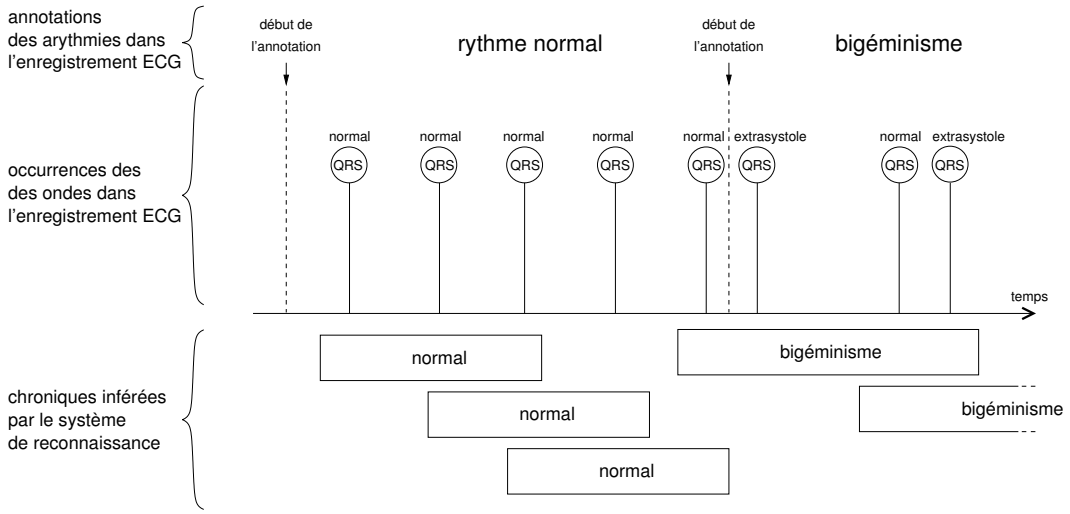


FIG. 5.3 – Exemples d'annotations et d'instances de chronique.

autres arythmies. Ce calcul nécessite de formaliser ce qu'on appellera une bonne *reconnaissance*, une *confusion* ainsi qu'une *non-reconnaissance*.

Sur un enregistrement complet normal il n'existe qu'une seule annotation, or la reconnaissance de chroniques inférera n chroniques pour une arythmie. La question est de savoir si n bonnes reconnaissances doivent être considérées ou seulement une seule. Dans le premier cas, il y aura un problème d'échelle, si on compte n bonnes reconnaissances on aurait compté seulement une seule non-reconnaissance pour la même période ce qui avantage les performances de reconnaissance et désavantage le calcul des arythmies non-reconnues. Dans le deuxième cas, une seule chronique reconnue serait valable pour un enregistrement entier, ce qui n'est pas réaliste.

D'une manière générale, utiliser le nombre de reconnaissances ne permet pas d'avoir une base de référence fiable car pour une même arythmie le nombre de reconnaissances peut varier selon le langage de description utilisé même si les performances de reconnaissance restent les mêmes. De même, on ne peut pas prendre les annotations comme référence car les reconnaissances n'utilisent pas la même échelle temporelle.

Dans cette étude c'est le nombre de battements qui est utilisé comme unité de référence. Chaque QRS sera représentatif d'un battement cardiaque. De plus, les ondes P bloquées seront aussi représentatives d'un battement pour pouvoir calculer les reconnaissances de la classe `mobitzII`. Cette référence permet de lier les deux représentations car, dans les annotations, chaque battement cardiaque est associé à une arythmie et chaque instance de chronique couvre plusieurs battements. En transformant les annotations et les instances de chronique dans une même unité de mesure, le nombre de battements couverts par la bonne instance de chronique peut être calculé. De cette manière, les performances de la reconnaissance sont analysées plus finement (au niveau battement et non au niveau arythmie).

5.4.2.2 Méthode de calcul des performances

Pour calculer les performances de reconnaissance, trois valeurs sont considérées.
- VP (Vrais Positifs) : le nombre de battements couverts par une instance de chronique qui correspond à la bonne arythmie.
- FP (Faux Positifs) : le nombre d'instances de chronique qui ne couvrent pas de battements appartenant à l'arythmie correspondante.
- FN (Faux Négatifs) ou NR (Non Reconnus) : le nombre de battements qui n'ont pas été couverts par l'instance de chronique qui correspond à l'arythmie à laquelle ils appartiennent.

Méthode de calcul des matrices de confusion Dans une matrice de confusion (exemple Table 5.3), les lignes représentent les instances de chronique et les colonnes les véritables éléments à reconnaître (annotations). Ainsi, pour chaque classe, les VP se trouvent dans la diagonale de la matrice, les FP sont les confusions et les FN sont les battements qui n'ont pas été couverts par une instance d'arythmie (les battements non reconnus). La colonne `inconnu` représente les confusions avec des arythmies dont les motifs n'ont pas été appris par le reconnaisseur de chroniques. Les confusions avec ces classes inconnues ne sont pas intégrées dans le calcul des performances par la suite.

\	A1	A2	A3	inconnu
A1	VP	FP	FP	FP
A2	FP	VP	FP	FP
A3	FP	FP	VP	FP
Non Reconnu	FN	FN	FN	\

Tab. 5.3 – Exemple de matrice de confusion

Pour chaque enregistrement, chaque QRS (représentatif d'un battement cardiaque) est étiqueté par le nom de l'arythmie à laquelle il appartient (chaque battement appartient à un rythme). Nous nous intéressons à un sous-ensemble d'arythmies (cf. 3.3.2.1) et seuls les QRS référençant une arythmie appartenant à cet ensemble sont étiquetés du nom de cette arythmie. Les autres sont annotés `inconnu`. La procédure de génération de la matrice de confusion est la suivante.

1. Pour chaque instance de chronique, la fenêtre temporelle allant du premier événement au dernier événement appartenant à l'arythmie est considérée. Pour éviter les problèmes de temps de détection d'événements (p. ex. retard de détection de QRS), chaque fenêtre est élargie d'une durée correspondant à un retard standard de détection d'événement (ici le retard correspond à la période réfractaire d'activation des cellules myocardiques du ventricule qui est égale à 200ms). Dans cette fenêtre, tous les battements étiquetés du même nom que l'arythmie correspondant à l'instance de chronique sont marqués. Comme les fenêtres de reconnaissance se chevauchent, le marquage des battements permet d'éviter d'attribuer deux fois un battement à un même type d'instance de chronique. Ainsi, dans la fenêtre

d'une instance de chronique, tous les battements étiquetés du nom de l'arythmie correspondant à la chronique et non déjà marqués comptent pour des VP. S'il n'existe aucun battement de même nom alors il y a eu confusion (FP).

2. Dans le cas de confusion (FP), il faut déterminer quelle colonne incrémenter, c'est à dire avec quelle arythmie il y a eu confusion. La procédure respecte les priorités suivantes :
   a. si, dans la fenêtre de la chronique, il existe un battement annoté `inconnu` alors la confusion a eu lieu avec une arythmie inconnue ;
   b. sinon, s'il y a dans la fenêtre des battements différents de `normal`, alors la confusion a eu lieu avec l'arythmie étiquetant le premier battement différent de `normal` ;
   c. sinon, il y a eu confusion avec la classe `normal`.
3. Lorsque, sous une même annotation, il existe plusieurs reconnaissances successives de la même arythmie, les fenêtres des instances de chronique sont agrégées pour n'en former qu'une.
4. Le nombre de non reconnus (FN) est le nombre de battements non marqués de chaque arythmie.

La figure 5.4 résume la méthode de calcul des performances.

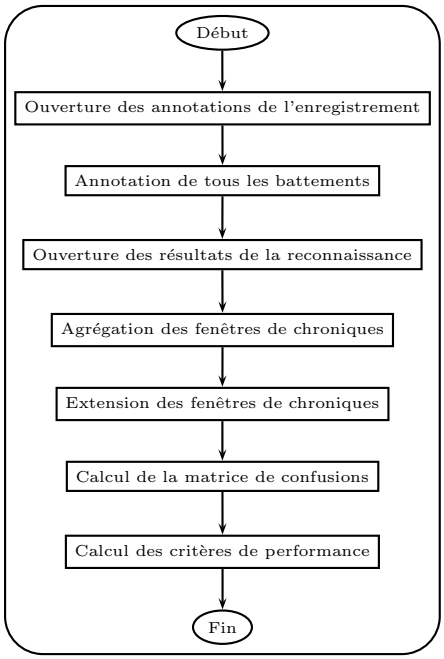


FIG. 5.4 – Diagramme de calcul de performance de reconnaissance de chroniques.

La figure 5.5 donne un exemple de calcul de performances. Une première chronique de la classe `mobitzII` a été instanciée. Sa fenêtre de reconnaissance couvrait un battement `mobitzII`. La case de la ligne `mobitzII` et de la colonne `mobitzII` a donc été

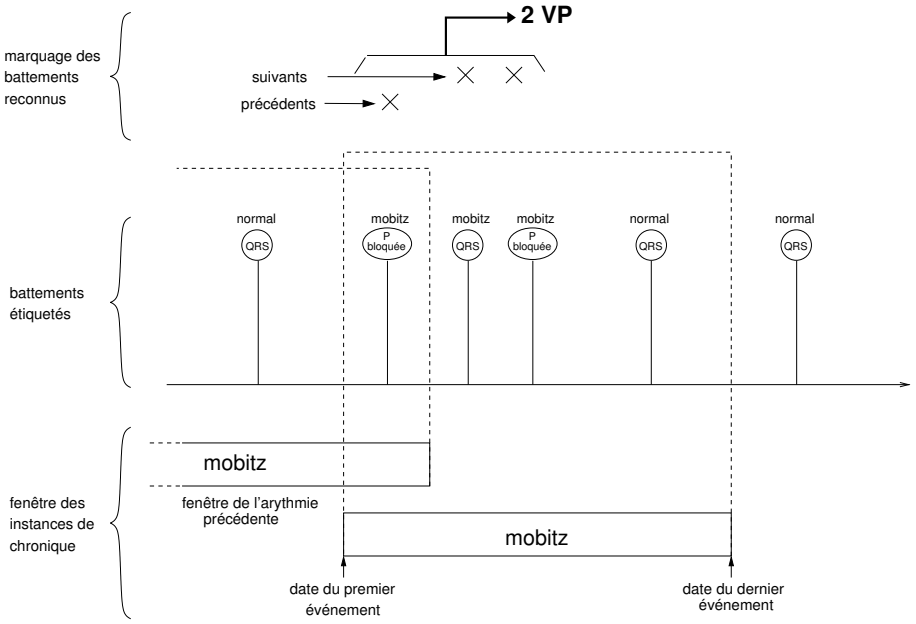


FIG. 5.5 – Exemple de reconnaissance d'arythmie.

incrémentée d'un VP. Le battement a ensuite été marqué. Une autre chronique de la classe `mobitzII` a ensuite été instanciée. Sa fenêtre couvrait quatre battements. Trois battements appartenaient effectivement à la chronique mais, le premier ayant déjà été marqué, seuls deux VP ont incrémenté la case (`mobitzII`,`mobitzII`). Les deux battements ont ensuite été marqués et n'ont plus été attribués à une autre instance de chronique.

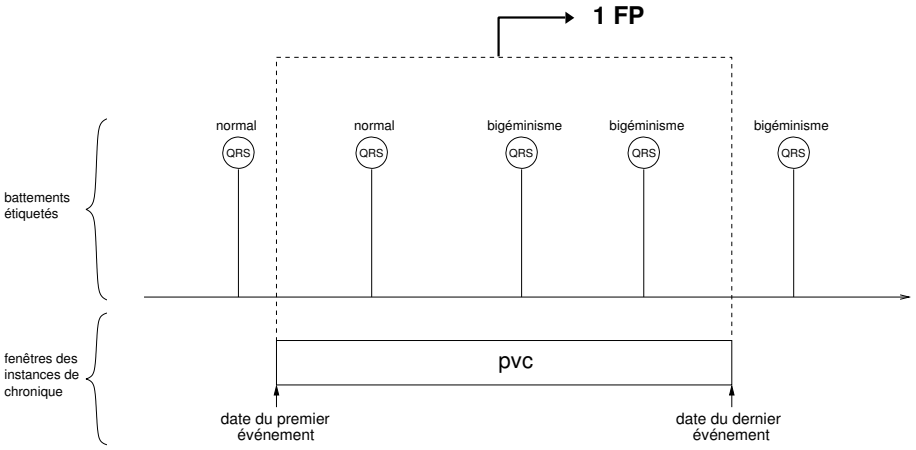


FIG. 5.6 – Exemple d'arythmie confondue.

Un exemple de confusion est représenté par la figure 5.6. La fenêtre d'une instance de chronique de la classe `pvc` recouvre trois battements qui n'appartiennent pas à la chronique reconnue, celle-ci est donc une confusion. Conformément à la procédure décrite précédemment (2.a.), cette confusion est attribuée à la première arythmie de la fenêtre, c'est à dire un bigéminisme. Les résultats de ces deux exemples sont donnés dans la table 5.4.

\	bigé.	mobitzII	normal	pvc	inc.
bigé.	0	0	0	0	0
mobitzII	0	3	0	0	0
normal	0	0	0	0	0
pvc	1	0	0	0	0
Non Reconnu	3	0	5	0	\

TAB. 5.4 – Calcul de la matrice de confusion avec les exemples des figures 5.5 et 5.6

Critères pour la comparaison des résultats : À partir des valeurs de la matrice de confusion nous pouvons calculer trois critères : la probabilité de reconnaissance, le taux de fausses alarmes et l'indice critique de succès.

La probabilité de reconnaissance, ou sensibilité, ou encore rappel, est une mesure couramment utilisée. Elle donne la fraction d'arythmies présentes dans les enregistrements qui ont été correctement reconnues. Elle s'obtient par la formule suivante :

$$Pr = \frac{VP}{VP + FN}$$

Un score parfait de 1 indique que tous les événements sont reconnus. Cette mesure est sensible aux VP mais ignore totalement les FP. On peut remarquer que la somme $VP + FN$ correspond au véritable nombre d'arythmies à reconnaître.

Le taux de fausses alarmes donne la fraction d'arythmies reconnues qui n'étaient pas vraiment présentes.

$$Tfa = \frac{FP}{VP + FP}$$

Un score parfait de 0 indique qu'il n'y a aucune confusion. Cette mesure est sensible aux FP mais ignore les FN. Le taux de fausses alarmes respecte la relation suivante avec la prédictivité positive parfois utilisée : $P+ = 1 - Tfa$

Les scores peuvent être difficiles à évaluer avec ces deux critères : des règles très bonnes en terme de reconnaissance mais générant beaucoup de fausses alarmes peuvent être difficiles à comparer à des règles peu performantes en reconnaissance mais générant peu de fausses alarmes. C'est pourquoi, pour obtenir une mesure globale des performances, l'indice critique de succès (Gerapetritis et Pelissier, 2004) (ou encore coefficient de Jaccard) est calculé comme suit :

$$ICS = \frac{1}{\frac{1}{1-Tfa} + \frac{1}{Pr} - 1} \quad \text{soit,} \quad ICS = \frac{VP}{VP + FP + FN}$$

Il prend en compte les deux types d'erreur FP et FN. Un score de 0 n'indique aucune compétence tandis qu'un score de 1 montre une reconnaissance parfaite. La précision (*accuracy*) $Acc = \frac{VP+VN}{VP+VN+FP+FN}$ parfois utilisée comme critère n'est pas utilisable dans cette étude car les vrais négatifs (VN) ne peuvent être comptabilisés.

Ces critères permettent d'avoir une vue d'ensemble des performances de reconnaissance avec le critère ICS et de spécifier ensuite grâce aux critères Pr et Tfa quelles sont les erreurs de la reconnaissance (arythmies non reconnues, trop de confusions, etc.)

5.4.2.3 Discussion

La méthode de calcul proposée permet d'obtenir une vision précise des capacités de reconnaissance des modèles de chroniques. Le choix du battement cardiaque comme unité de référence apporte une analyse fine et rigoureuse. Ce choix est aussi plus défavorable au système que de prendre l'arythmie comme unité de référence. En effet, le comptage s'effectuant au niveau du battement, le nombre d'événements à reconnaître est beaucoup plus important.

Les QRS isolés amènent des FN. Par exemple, quelques battements normaux entre deux extrasystoles seront considérés comme non reconnus car ils ne seront pas assez nombreux pour qu'une chronique de classe `normal` soit instanciée. Ceci est dû à l'apprentissage car les rythmes normaux sont ceux qui nécessitent le plus d'événements pour être reconnus. Les rythmes normaux ajouteront donc toujours des FN. Cependant, cette augmentation ne dépend pas de la méthode et la reconnaissance des rythmes normaux n'est pas un objectif majeur du système qui est plutôt consacré à la détection des rythmes pathologiques.

Enfin, lorsqu'une instance de chronique couvre deux arythmies, il est souvent difficile de décider avec quelle arythmie elle a été confondue. Les priorités définies amènent de la cohérence dans le choix. Si un battement appartient à une arythmie inconnue alors il est considéré que c'est l'élément qui a amené la confusion. Si un battement appartient à une arythmie dans l'ensemble des événements de la fenêtre de la chronique, alors il est considéré que c'est l'arythmie associée à ce battement qui a amené la confusion. S'il y a plusieurs arythmies parmi les battements couverts alors l'arythmie du premier battement est prise en compte. C'est seulement lorsque tous les battements sont étiquetés `normal` qu'il est considéré que la confusion s'est faite avec l'arythmie `normal`.

5.4.3 Résultats de la reconnaissance

Les performances de la reconnaissance d'arythmies sont calculées selon la méthode précédemment exposée. Ces résultats sont présentés sous forme de matrice de confusion pour chaque niveau hiérarchique des modèles de chroniques. Différents critères ont aussi été calculés pour comparer les performances selon les niveaux. Pour rappel, les modèles de chroniques utilisent un langage de description composé :

- pour `exper1`, des dates d'occurrence des QRS uniquement ;
- pour `exper2` (resp. `exper3`), des dates d'occurrence des QRS et la classe de leur morphologies dans l'ensemble `{basic,nonbasic}` (resp. `{basic,nonbasic,pvc}`) ;

– pour `exper4`, des dates d'occurrence des QRS et des ondes P ;
– pour `exper5` (resp. `exper6`), des dates d'occurrence des QRS plus des dates d'occurrence des ondes P plus les morphologies des QRS classées dans l'ensemble `{basic,nonbasic}` (resp. `{basic,nonbasic,pvc}`);

5.4.3.1 Matrices de confusion

Sur les matrices de confusion on trouve en colonnes les battements de référence pour chaque classe d'arythmie dans les signaux de test et en lignes le nombre de battements reconnus couverts par une instance de chronique relative à une arythmie. Au fil des résultats des différentes bases de chroniques, les lignes et colonnes « Total » ne sont jamais les mêmes. Ceci est dû au nombre de FP qui varie. Pour donner une référence constante, la ligne « VP+FN » qui représente le véritable nombre de battements par arythmie est ajoutée. Pour chaque arythmie, les performances de la reconnaissance dénotées par Pr et Tfa sont données dans deux lignes supplémentaires.

	bigeminy	doublet	mobitzII	normal	pvc	vt	inconnue	Total
bigeminy	1145	45	0	27	87	87	440	1831
doublet	10	202	0	28	3	3	119	365
mobitzII	0	0	836	0	0	0	0	836
normal	3	0	0	11910	0	0	40	11953
pvc	39	22	0	160	269	1	117	608
vt	3	8	0	3	3	175	533	725
NR	13	18	2	830	53	9	0	925
Total	1213	295	838	12958	415	275	1249	17243
VP + FN	1158	220	838	12740	322	184	0	15462
Pr	0.99	0.92	1.00	0.93	0.84	0.95		
Tfa	0.18	0.18	0.00	0.00	0.45	0.09		

TAB. 5.5 – Résultats de reconnaissance de chroniques pour la définition `exper1`

Commentaires sur exper1 Les résultats de reconnaissance d'`exper1` présentés Table 5.5 montrent que cette définition de chroniques, la plus abstraite, est presque parfaite pour la classe `mobitzII` et la meilleure pour reconnaître les rythmes normaux et les `pvc`. Par contre, elle ne parvient pas à bien discriminer les classes `bigeminy`, `pvc` et `doublet`. Ces modèles de chroniques sont aussi ceux qui génèrent le plus de fausses alarmes. En effet, beaucoup d'arythmies sont confondues avec `normal`. Ceci n'est guère surprenant car les règles apprises couvrent beaucoup de contre-exemples. On voit aussi l'effet du sur-apprentissage de `pvc` dont les deux règles, qui lors de l'apprentissage couvraient au moins 14 exemples communs, provoquent des doubles reconnaissances et amènent vers un grand nombre de fausses alarmes. Il convient de préciser que ce type de reconnaissances multiples de même classe implique une surestimation du nombre de FP. En effet, si plusieurs instances de chronique d'une seule classe sont instanciées au même moment sans qu'il y ait une arythmie de cette classe effectivement présente alors

il y aura autant de FP que d'instances de chronique. Pourtant, une seule instance devrait compter étant donné que ce sont des instances de la même chronique provoquées par le même phénomène. Ces reconnaissances multiples sont dues à l'apprentissage car les règles apprises ne sont pas discriminantes entre elles mais entre arythmies.

	bigeminy	doublet	mobitzII	normal	pvc	vt	inconnue	Total
bigeminy	945	0	0	0	67	0	6	1018
doublet	9	178	1	0	3	0	3	194
mobitzII	0	0	836	0	0	0	0	836
normal	2	0	0	11695	0	0	33	11730
pvc	2	0	0	0	123	0	65	190
vt	0	0	0	0	0	128	0	128
NR	213	42	2	1045	199	56	0	1557
Total	1171	220	839	12740	392	184	107	15653
VP + FN	1158	220	838	12740	322	184	0	15462
Pr	0.82	0.82	1.00	0.94	0.53	0.67		
Tfa	0.07	0.08	0.00	0.00	0.01	0.00		

TAB. 5.6 – Résultats de reconnaissance de chroniques pour les définitions `exper2` et `exper3`

Commentaires sur exper2 et exper3 Les résultats de reconnaissance d'`exper2` et d'`exper3` sont exactement les mêmes et sont présentés Table 5.6. La spécialisation des règles a permis d'améliorer leur taux de fausses alarmes car il y a effectivement moins de confusions, les classes sont mieux séparées. Par contre, le critère Pr indique que toutes les classes exceptées `mobitzII` et `normal`, sont devenues plus difficiles à reconnaître. La spécialisation des règles a toutefois supprimée toutes les confusions avec `normal` et `doublet`. Un nombre restreint de confusions subsiste entre les classes `bigeminy`, `pvc`, `doublet` et `vt`, ce qui s'explique assez bien par le fait que ce sont tous des rythmes composés d'une ou plusieurs extrasystoles.

Commentaires sur exper4 Les résultats de reconnaissance d'`exper4` présentés Table 5.7 montrent que cette définition de chroniques est plus abstraite que `exper2` et `exper3`. Sa probabilité de reconnaissance leur est supérieure dans toutes les classes, exceptée pour la classe `normal`. Par contre, l'utilisation de l'onde P dans le langage de description de ces modèles de chroniques est accompagnée d'un nombre de fausses alarmes qui ne pouvait pas être deviné à partir des résultats de l'apprentissage des règles de reconnaissance d'arythmies (*cf.* 3.3). D'une manière générale, l'apport de l'onde P ne semble pas décisif pour la reconnaissance. Ceci est d'autant plus intéressant que pour obtenir un système peu coûteux en ressources, la détection de l'onde P doit être employée le moins possible.

	bigeminy	doublet	mobitzII	normal	pvc	vt	inconnue	Total
bigeminy	1143	10	0	90	89	2	214	1548
doublet	9	214	1	24	2	7	160	417
mobitzII	0	0	836	7	0	0	0	843
normal	0	0	0	10255	0	0	2	10257
pvc	0	0	0	38	148	0	123	309
vt	2	1	0	23	3	179	295	503
NR	15	6	2	2485	174	5	0	2687
Total	1169	231	839	12922	416	193	794	16564
VP + FN	1158	220	838	12740	322	184	0	15462
Pr	0.99	0.97	1.00	0.86	0.65	0.97		
Tfa	0.19	0.17	0.01	0.00	0.13	0.14		

TAB. 5.7 – Résultats de reconnaissance de chroniques pour la définition `exper4`

	bigeminy	doublet	mobitzII	normal	pvc	vt	inconnue	Total
bigeminy	973	0	0	0	66	0	8	1047
doublet	5	174	1	0	0	0	1	181
mobitzII	0	0	836	7	0	0	0	843
normal	1	0	0	10823	0	0	1	10825
pvc	2	0	0	0	120	0	66	188
vt	0	0	0	0	0	163	0	163
NR	185	46	2	1917	202	21	0	2373
Total	1166	220	839	12747	388	184	76	15620
VP + FN	1158	220	838	12740	322	184	0	15462
Pr	0.84	0.79	1.00	0.85	0.37	0.89		
Tfa	0.06	0.03	0.01	0.00	0.02	0.00		

TAB. 5.8 – Résultats de reconnaissance de chroniques pour les définitions `exper5` et `exper6`

Commentaires sur exper5 et exper6 Les résultats de reconnaissance d'`exper5` et d'`exper6` sont exactement les mêmes et sont présentés Table 5.8. Dans l'ensemble, les résultats ont sensiblement les mêmes performances que ceux présentés pour `exper2` et `exper3`. L'apport de l'onde P dans le langage de description a surtout permis d'améliorer la classe `vt`. On peut par contre constater que cet apport a rendu beaucoup plus difficile la détection des arythmies de la classe `pvc`.

5.4.4 Discussion et analyse globale de la reconnaissance

D'une manière générale on ne constate pas de hiérarchie dans les résultats, exceptés pour `exper1` et `exper4` qui entraînent plus de confusions mais pour une meilleure reconnaissance. Selon l'indice critique de succès représenté Figure 5.7, les règles sont sensiblement aussi bonnes.

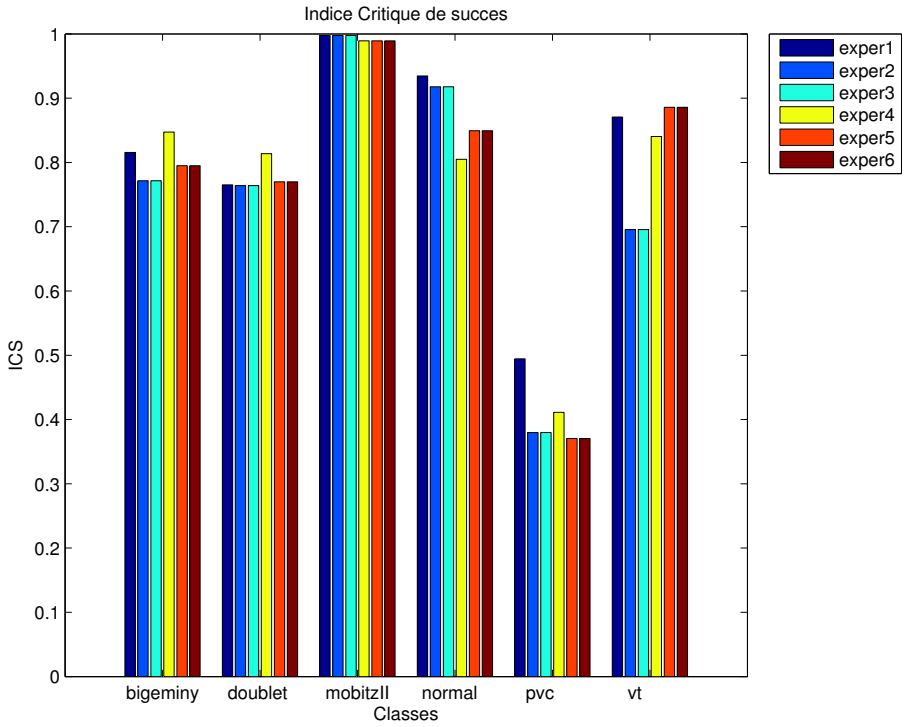


FIG. 5.7 – Indice Critique de succès pour toutes les règles de reconnaissance.

Ces résultats se confirment sur les figures 5.8 et 5.9 qui représentent respectivement la probabilité de reconnaissance et le taux de fausses alarmes pour toutes les règles sur toutes les classes.

La classe `pvc` est la plus difficile à reconnaître pour tous les niveaux de langage de description excepté pour la classe `exper1` qui possède une bonne capacité de reconnaissance mais un taux de fausses alarmes très élevé. Ces résultats sont cohérents, car les extrasystoles n'ont pas de schéma d'apparition bien précis. Les extrasystoles sont dues

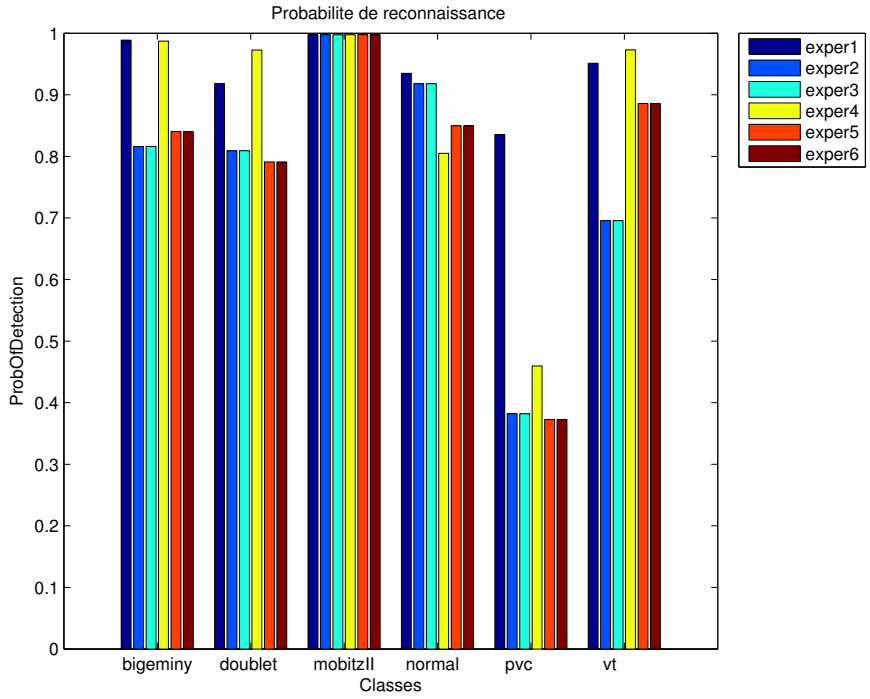


FIG. 5.8 – Probabilité de reconnaissance pour toutes les règles de reconnaissance.

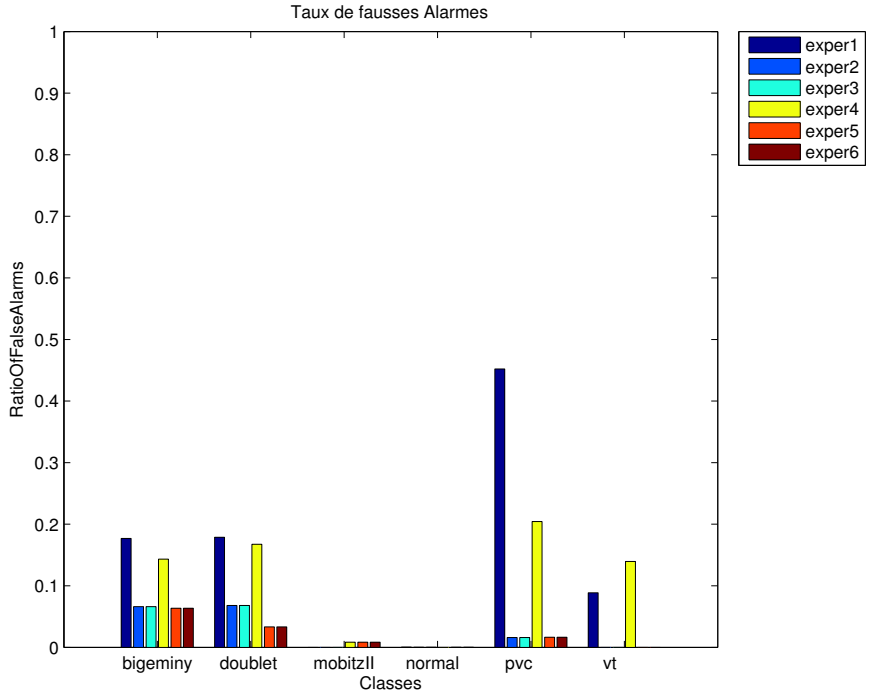


FIG. 5.9 – Taux de fausses Alarmes pour toutes les règles de reconnaissance.

à une activation spontanée des ventricules qui peuvent être activés de concert avec une dépolarisation normale. Les extrasystoles peuvent donc être précédées d'une onde P ou non et intervenir à des intervalles complètement irréguliers (extrasystole avec repos compensatoire ou non). Les autres arythmies ont, au contraire, un schéma d'apparition bien défini. Par exemple, les bigéminismes sont des successions de battements de base et d'extrasystoles (le plus souvent) intervenant à des intervalles très réguliers. Les classes `doublet`, `bigeminy`, sont plus difficiles à reconnaître pour les règles utilisant la forme du QRS. Elles provoquent cependant moins de fausses alarmes. Il est à noter que les seuils utilisés pour classer les informations temporelles sont des seuils absolus et non relatifs à un patient. Autrement dit, les intervalles considérés comme rapides pour un patient peuvent être normaux pour un autre patient. La figure 5.10 présente les différences entre les intervalles temporels d'une classe à une autre pour des QRS appartenant à un rythme de base (classes `normal, lbbb, rbbb, paced`) et les extrasystoles.

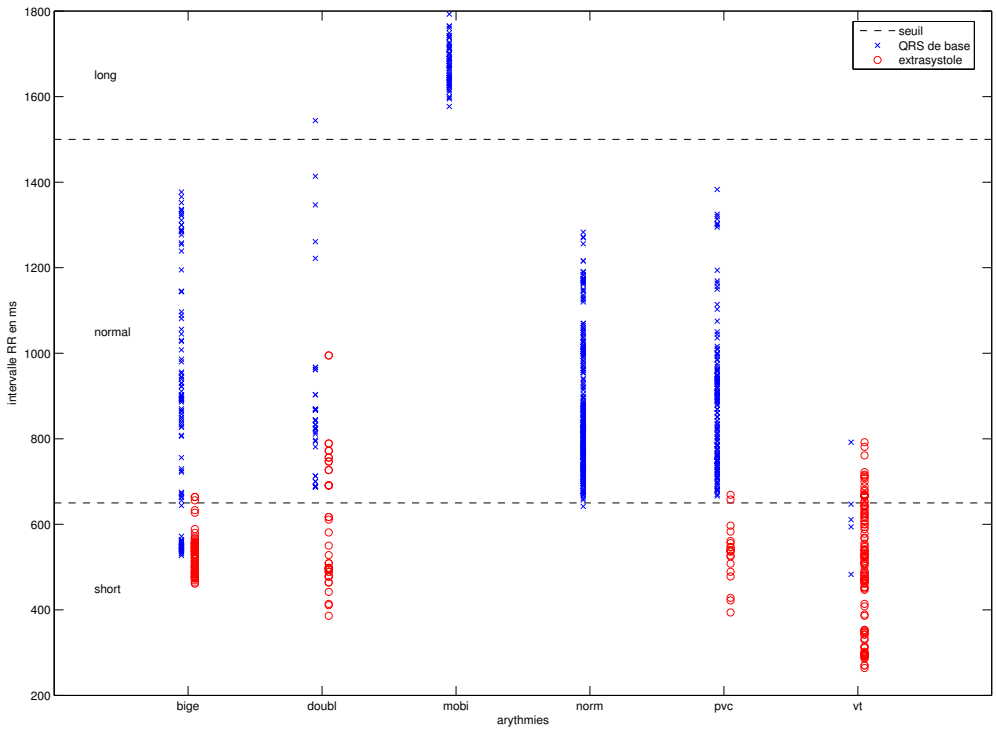


FIG. 5.10 – Intervalles RR associés à des QRS de rythme de base et des extrasystoles.

Généralement, les extrasystoles provoquent un intervalle RR[1] court (donc un rythme rapide). Cependant la séparation entre les QRS et des extrasystoles selon les intervalles

[1]distance entre les maximums de deux complexes QRS successifs

RR n'est pas très nette, surtout pour les classes `bigeminy` et `doublet`. Le seuil de séparation entre les QRS de chaque classe n'est pas le même. Or, dans l'apprentissage seul un seuil fixe de 650ms a été utilisé pour séparer les intervalles `normal` de `short` et un seuil de 1500ms pour séparer les intervalles `normal` de `long` . Des seuils différents pour chaque classe ou un ensemble de seuils flous permettraient d'augmenter les performances des règles `exper1`, `exper2` et `exper3`.

Il est à noter que l'apport de l'onde P ne semble pas décisif pour la reconnaissance notamment entre `exper2` et `exper5`. Dans cette dernière règle, la possibilité d'employer l'onde P dans le langage a juste permis d'exprimer l'absence d'onde P dans `vt`. Pour tester l'apport de l'onde P, il serait intéressant d'apprendre des rythmes lents tels que la bradycardie qui ne peut être théoriquement différenciée d'un mobitz sans l'analyse des ondes P.

Enfin, il reste à étudier l'intérêt de conserver les ensembles `{basic,nonbasic}` et `{basic,nonbasic,pvc}` étant donné qu'ils n'entraînent aucune différence de résultat et que ce dernier ensemble est plus difficile à assurer d'un point de vue traitement du signal.

Le test de la hiérarchie de définitions de modèles de chroniques permet de conclure que les arythmies cardiaques peuvent être reconnues à partir d'informations très peu précises telles que les intervalles temporels entre les complexes QRS moyennant un taux de fausses alarmes conséquent. Le nombre de fausses alarmes est réduit dès que l'information sur la forme du QRS devient disponible. Par ailleurs, le constat selon lequel l'information d'onde P n'est pas essentielle pour discriminer les arythmies est assez surprenant. Ceci peut toutefois s'expliquer car les arythmies traitées sont essentiellement à commande ventriculaire, exceptées pour les classes `normal` et `mobitzII`. La synthèse de ces résultats permet de choisir les définitions de chroniques recommandées pour le pilotage : 1) la définition `exper1` permet d'assurer un diagnostic dans les cas les plus défavorables, 2) la définition `exper2` est utilisée lorsque l'information de forme de QRS est présente et enfin 3) la définition `exper3` permet de profiter de l'information concernant l'onde P pour reconnaître de manière plus précise la classe `vt`. De plus, les résultats présentés dans cette section donnent les résultats optimaux atteignables par le système piloté. En effet, l'abstraction temporelle étant ici parfaite, les performances dans le cas piloté ne peuvent être meilleures.

5.5 Résultats du pilotage des tâches selon le contexte de ligne

Le pilotage des tâches permet d'activer ou de désactiver les tâches de l'abstraction temporelle. Ces activations vont de pair avec le pilotage de la reconnaissance d'arythmies. En effet, si une tâche de l'abstraction temporelle est désactivée, l'ECG est décrit de manière plus abstraite et la reconnaissance de chroniques doit utiliser des modèles employant le langage de description correspondant. Le pilotage des tâches permet de limiter les erreurs de diagnostic en désactivant les tâches qui ne peuvent être accomplies sans erreurs. Un autre objectif est de limiter la consommation de ressources en utilisant

les langages de description les plus abstraits qui permettent cependant de discriminer les arythmies. Ce dernier objectif dépend essentiellement du contexte arythmique tandis que le premier est très dépendant du bruit de ligne. Les résultats de pilotage des tâches présentés sont ceux effectués uniquement avec le contexte de ligne. De plus, pour limiter les effets négatifs des reconnaissances multiples en cas de fausses alarmes, les règles de reconnaissances de chroniques redondantes (p. ex. la deuxième règle de la classe `pvc` dans `expert1`) et les règles ne couvrant qu'un seul exemple ont été supprimées de l'étude. Les résultats obtenus sur les données de test sans le pilotage sont donnés en section 5.5.1 et ceux obtenus avec le système piloté sont décrits en section 5.5.2. Ces résultats sont discutés en section 5.5.3.

5.5.1 Résultats sans pilotage

La configuration sans pilotage se place dans le cas classique de monitorage. C'est-à-dire que les meilleurs algorithmes de traitement du signal sont utilisés, que toutes les tâches sont activées et que le langage de description le plus précis est choisi (ici `exper5`). Le tableau 5.9 présente les résultats de la reconnaissance d'arythmies sans pilotage. En comparant ces résultats aux performances de reconnaissance sur les anno-

	bigeminy	doublet	mobitzII	normal	pvc	vt	inconnue	Total
bigeminy	798	2	0	45	36	0	5	886
doublet	1	14	0	8	1	0	2	26
mobitzII	2	0	784	1434	0	0	50	2270
normal	0	0	0	4127	0	0	1	4128
pvc	4	4	0	16	49	0	26	99
vt	0	0	0	0	0	0	0	0
NR	360	206	54	8613	273	184	0	9690
Total	1165	226	838	14243	359	184	84	17099
VP + FN	1158	220	838	12740	322	184	0	15462
Pr	0.69	0.06	0.94	0.32	0.15	0.00		
Tfa	0.09	0.42	0.65	0.00	0.33	0.00		

TAB. 5.9 – Résultats de la reconnaissance sans le pilotage sur les signaux bruités avec le modèle exper5

tations (Table 5.8), on constate une chute importante du nombre de reconnaissances et une augmentation importante du nombre de confusions pour les classes `doublet` et `mobitzII`. Les classes `vt` et `doublet` ne sont absolument pas reconnues alors qu'il s'agit des arythmies les plus sévères. De plus, le taux de fausses alarmes indique que 42% des reconnaissances de `doublet` sont fausses. Les classes `pvc` et `normal` ne sont presque pas reconnues. La classe `mobitzII` est souvent confondue et la classe `bigeminy` est la seule à obtenir des performances correctes sur les signaux de test. Le système n'est plus en mesure de fournir un diagnostic médical pour ces signaux bruités. Cette baisse de performance est due d'une part, à l'abstraction temporelle avec une classification peu robuste et d'autre part, au langage de description trop riche des modèles de chroniques

donc trop exigeant dans cette situation.

5.5.2 Résultats avec le pilotage

Le pilotage utilise les modèles de chroniques `exper1`, `exper2`, et `exper5` selon le contexte courant. Dans les tests, le bruit de type *ma* étant très perturbateur pour les algorithmes de traitement du signal, la reconnaissance d'arythmies a été suspendue dans les contextes *ma* pour les $RSB \leq -5dB$ car elle n'entraînait que des fausses alarmes. Les résultats obtenus avec le pilotage (Table 5.10) améliorent nettement la probabilité de reconnaissance et sont plus proches du cas de référence (Table 5.8).

	bigeminy	doublet	mobitzII	normal	pvc	vt	inconnue	Total
bigeminy	838	22	0	34	62	2	52	1010
doublet	16	74	0	106	15	5	64	280
mobitzII	0	0	763	5	0	0	0	768
normal	0	0	0	10112	0	0	16	10128
pvc	2	3	0	82	78	1	36	202
vt	24	15	0	50	22	104	209	424
NR	320	146	75	2628	244	80	0	3493
Total	1200	260	838	13017	421	192	377	16305
VP + FN	1158	220	838	12740	322	184	0	15462
Pr	0.72	0.34	0.91	0.79	0.24	0.57		
Tfa	0.13	0.66	0.01	0.00	0.53	0.52		

TAB. 5.10 – Résultats de la reconnaissance sur les signaux de test avec un pilotage utilisant les modèles `exper1`, `exper2` et `exper5`.

Ceci est dû à l'amélioration apportée par le pilotage des tâches de l'abstraction temporelle et au pilotage de la reconnaissance de chroniques qui permet d'utiliser des langages de description moins riches lorsque l'abstraction temporelle ne peut fournir tous les éléments nécessaires. Cette hausse de reconnaissances s'accompagne d'une hausse de confusions pour les classes `doublet`, `pvc` et `vt`. Cependant, le nombre de confusions de la classe `mobitzII` a été grandement diminué en ne tenant pas compte des ondes P dans les cas bruités. Le nombre de fausses alarmes est donc augmenté mais pour un gain en diagnostic médical très significatif.

5.5.3 Discussion et résultats globaux de pilotage

L'analyse de l'indice critique de succès (Figure 5.11) montre la supériorité du système piloté par rapport au système non piloté sur des signaux très bruités. Le pilote apporte une meilleure probabilité de reconnaissance (Figure 5.12) mais un taux de fausses alarmes (Figure 5.13) plus élevé.

Ces résultats sont à comparer avec les figures de référence 5.7, 5.8 et 5.9 de la section 5.4.4. Concernant la classe `bigenimy`, Pr et Tfa sont très proches des résultats de références. Les modèles de chroniques utilisés sont donc très résistants au bruit et aux

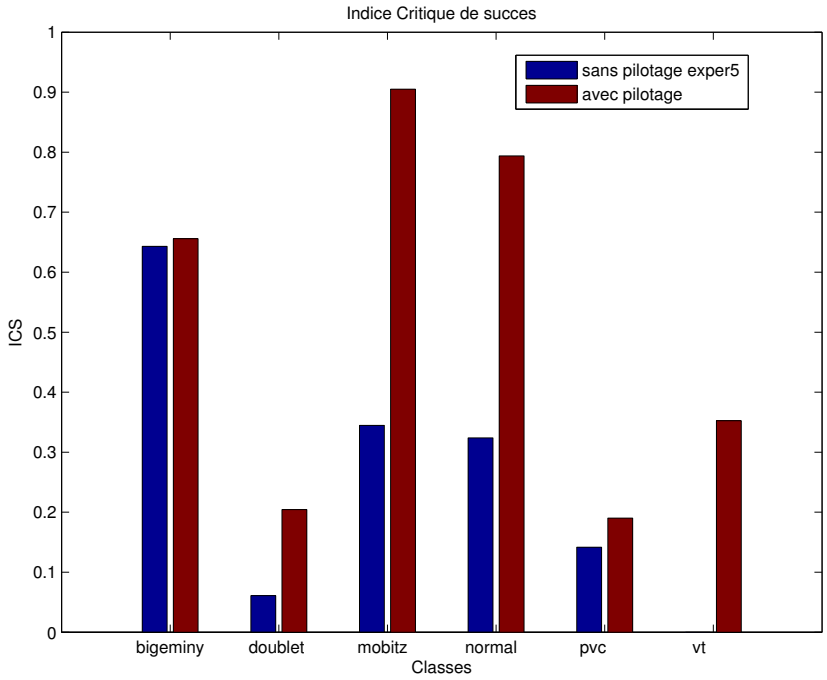


FIG. 5.11 – Indice Critique de Succès

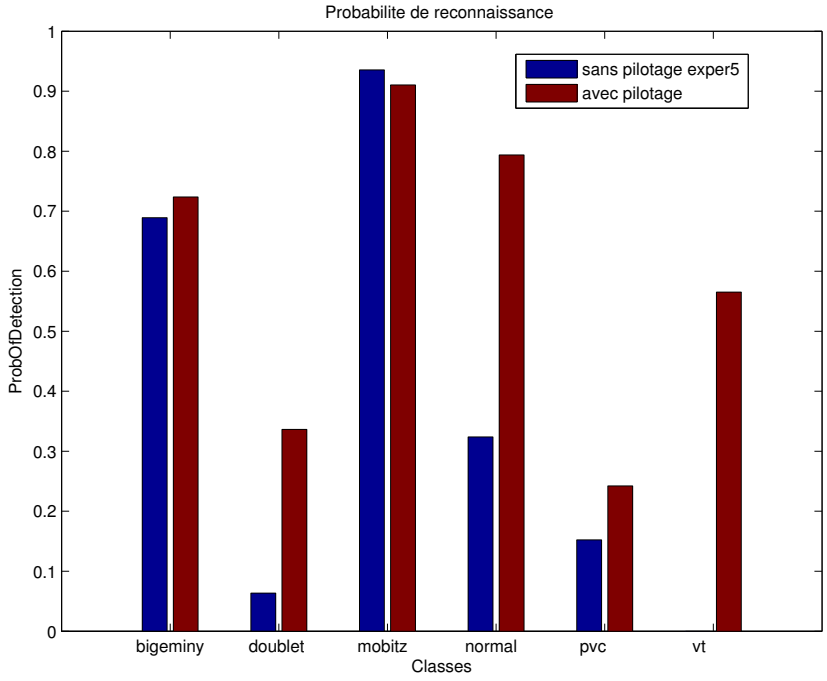


FIG. 5.12 – Probabilité de reconnaissance

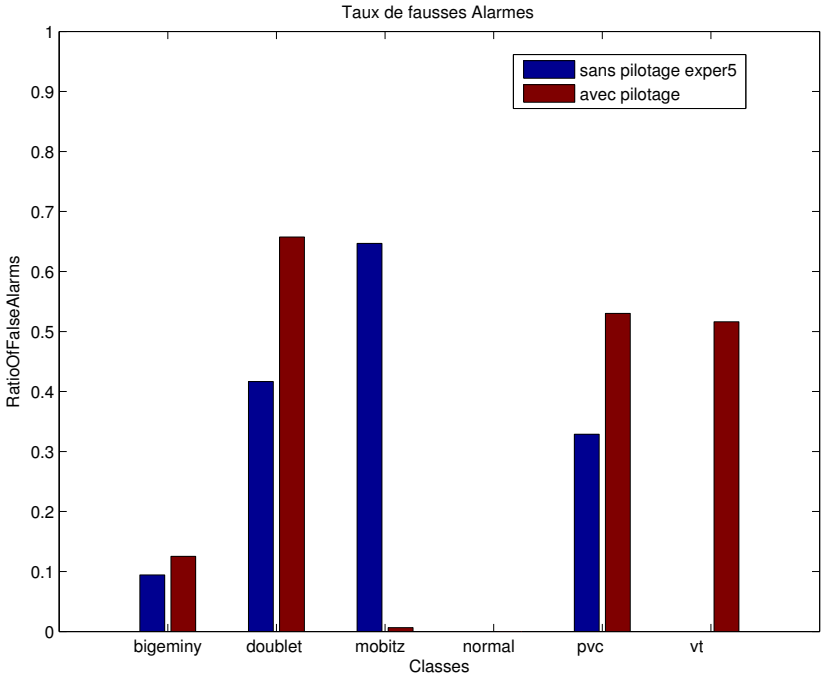


FIG. 5.13 – Taux de fausses alarmes

informations imprécises. La classe `doublet` est très difficile à reconnaître et demeure très sensible aux FP de la détection de QRS. En effet, si un FP de QRS est détecté entre deux QRS alors, pour la reconnaissance, il apparaîtra trois QRS espacés par deux intervalles courts qui déclencheront automatiquement la reconnaissance d'un doublet dans la classe `exper1`. La figure 5.14 représente un exemple de FP de QRS qui provoque la reconnaissance d'un doublet.

De même, les erreurs de classification et de détection d'ondes P sont nuisibles à la détection de doublets. C'est pourquoi, le pilotage, en utilisant `exper1`, est particulièrement indépendant des erreurs de classification et de détection d'onde P et subit surtout les erreurs de détection de QRS, ce qui explique le Tfa relativement important. Cependant, selon l'ICS, le pilotage est beaucoup plus performant. De la même manière, la relative indépendance du pilotage par rapport aux ondes P lui permet d'éviter les fausses alarmes de la classe `mobitzII` pour obtenir un ICS très proche de l'optimal. La classe `pvc` est à l'origine particulièrement difficile à reconnaître même dans les cas de référence. Cette classe est particulièrement sensible aux erreurs de détection de QRS. En effet, si un QRS est détecté en avance alors il peut provoquer une fausse reconnaissance. Enfin, on peut constater que le langage de description utilisé par `exper5` est beaucoup trop exigeant pour reconnaître la classe `vt` alors que le pilotage arrive à reconnaître cette arythmie sévère.

Enfin, il serait intéressant de comparer les résultats avec des annotations de médecins obtenues sur les signaux bruités. En effet, les annotations originales ont été effectuées

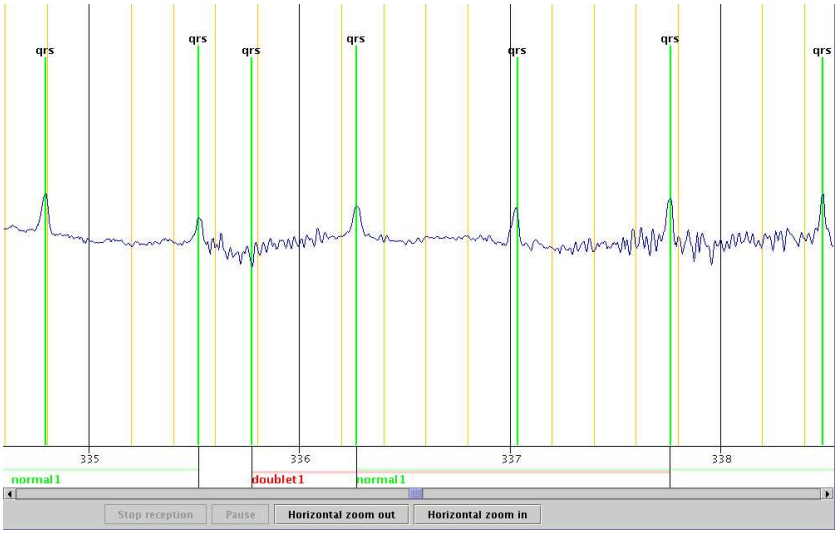


FIG. 5.14 – Fausse alarme de QRS provoquant une fausse reconnaissance de doublet

sur des signaux non bruités alors que le système traite les signaux dans des conditions plus difficiles.

5.6 Conclusion

Les résultats montrent bien que dans un environnement clinique hostile, les sources d'erreurs provoquent un nombre de fausses alarmes important et un diagnostic médical défaillant. Cette baisse de performance est provoquée par le manque de robustesse des algorithmes de traitement du signal et l'impossibilité du diagnostic médical à s'adapter au contexte. Le pilote du système IP-CALICOT permet de pallier ces deux problèmes. Plutôt que d'optimiser les algorithmes de traitement du signal, l'utilisation adéquate des algorithmes selon les contextes de bruit diminue le nombre d'erreurs engendrées. Pour l'instant, seule la tâche de détection de QRS en tire profit mais le pilotage des autres tâches pourrait en bénéficier avec une base d'algorithmes plus importante. De plus, le pilote adapte le diagnostic médical à la résolution courante de l'abstraction temporelle augmentant ainsi grandement le nombre de reconnaissances d'arythmies.

Le nombre de fausses alarmes reste important mais dans un contexte aussi bruité (mouvement des patients, décollement d'électrodes, etc.) il faut surtout assurer une bonne reconnaissance des pathologies. De plus, le nombre de fausses alarmes du pilotage est surestimé. En effet, dans des contextes bruités, le pilote utilise des règles de reconnaissance apprises avec un langage très abstrait. Cette abstraction, quoi que plus généralisante, entraîne l'apprentissage de plusieurs règles qui couvrent souvent les mêmes exemples. Ainsi, lorsqu'une reconnaissance survient, plusieurs chroniques sont instanciées. En cas de confusions (fausses alarmes) chaque chronique compte pour une confusion bien qu'il s'agisse de la même confusion. Par exemple, pour la classe `vt` de

la base `exper1` (Figure 3.7), plusieurs chroniques sont régulièrement reconnues en cas de FP ce qui multiplie d'autant le nombre de fausses alarmes. Pour supprimer les alarmes redondantes, le système de monitorage devrait intégrer un filtrage des alarmes redondantes.

Les résultats montrent que le pilotage permet d'assurer un diagnostic médical sur des électrocardiogrammes bruités. Cependant, le pilotage est surtout guidé par le contexte de bruit. La prise en compte du contexte arythmique pour piloter les tâches et la reconnaissance d'arythmies reste à accomplir. En effet, dans un contexte où il est possible de discriminer deux arythmies avec un langage de description moins précis, le pilote peut désactiver les tâches de l'abstraction temporelle non nécessaires et économiser des ressources.

L'architecture de IP-Calicot permet d'ajouter aisément les modules et la connaissance nécessaires pour gérer des données multisources. L'extension de IP-Calicot au monitorage multisource est justement l'une des évolutions envisagées pour utiliser les informations redondantes pour proposer un diagnostic médical plus robuste au bruit (Chambrin, 2001). Cet objectif a fait l'objet d'une étude préliminaire présentée en annexe C qui a montré la faisabilité d'une telle approche.

Le pilotage utilisé dans le système IP-Calicot permet l'amélioration du système Calicot. Le pilotage des algorithmes de traitement du signal réduit le nombre d'erreurs générées par l'abstraction temporelle. Les résultats présentés valident non seulement l'approche par pilotage mais aussi les connaissances concernant les algorithmes.

Chapitre 6

Mise en œuvre de IP-Calicot

6.1 Introduction

Dans IP-Calicot, deux types de représentation sont utilisés pour structurer la connaissance : la représentation par objets et la représentation par règles. On entend ici par connaissance, les techniques liées au traitement du signal (algorithmes) et la connaissance liée aux décisions du pilote (règles). La représentation par objets permet de structurer les connaissances en traitement du signal et en monitorage, ainsi que les concepts de pilotage décrits en section 2.2.5. Il s'agit principalement des signaux et événements, des algorithmes de traitement du signal, du contexte et des tâches. La mise en œuvre de ces connaissances fait l'objet de la section 6.2. La représentation par règles est utilisée pour la connaissance liée aux décisions du pilote, c.-à-d. les règles de pilotage. Elle est détaillée en section 6.3. Toutes ces connaissances sont ensuite utilisées pour concevoir une application composée de tâches qui interagissent avec un pilote. Le mécanisme de ce pilotage et la procédure employée pour développer une application sont exposés en section 6.5.

6.2 Représentation par objets

La connaissance représentée par des objets est développée en langage Java[1]. Ce langage de programmation a été choisi car c'est une programmation orientée objet idéale pour représenter les algorithmes et les tâches de traitement. Les avantages de Java par rapport à d'autres langages sont nombreux, parmi ceux-ci on peut citer :

- l'indépendance du système d'exploitation, idéale pour une architecture repartie,
- la notion de *thread* qui permet de lancer des processus autonomes avec partage de données,
- la possibilité d'interfacer d'autres langages, notamment Prolog et C,
- les nombreuses API qui permettent des communications réseaux et la réalisation d'interfaces utilisateur,

[1] http ://java.sun.com/

– enfin, la diffusion et la popularité de ce langage qui facilite la prise en main des réalisations par d'autre programmeurs.

Java a donc été utilisé pour mettre en œuvre les objets tels que les signaux et événements, les algorithmes de traitement du signal, le contexte et les tâches du système détaillés dans les sections suivantes. Dans la suite de ce chapitre des diagrammes UML (*Unified Modeling Language*) sont utilisés exprimer la représentation des objets.

6.2.1 Représentation des données

Les tâches de traitement d'une application s'échangent des informations via des données représentées selon des types de base tels que les signaux et différentes informations temporelles structurées.

Comme annoncé dans la section 2.2.2, un signal est représenté par un objet de type `Signal` dont les valeurs sont stockées dans un tableau de nombres (seuls les signaux temporels à une dimension sont traités). La représentation par objets permet de mettre en œuvre directement dans la classe `Signal` des méthodes de base qui peuvent le transcrire dans ses différentes représentations (p. ex. transformée de Fourier) ou appliquer des transformations tel qu'un sous-échantillonnage.

Si la notion fondamentale de représentation des données en traitement du signal est le signal, il faut rappeler que l'activité de monitorage consiste à surveiller un système évoluant dans le temps. L'évolution d'un système peut être généralement décrite par une représentation points-intervalles (Shahar et Musen, 1993) ou événements-états (Dojat et Sayettat, 1996). Dans IP-CALICOT, une ontologie des concepts temporels est considérée, telle que dans (Dojat et Sayettat, 1996). L'entité de base de l'ontologie est l'objet appelé `TemporalObject` qui est ensuite raffiné en deux autres types d'objets représentés par le diagramme de la figure 6.1 : les *événements* (objets de type `Event`) et les *états* (objets de type `State`).

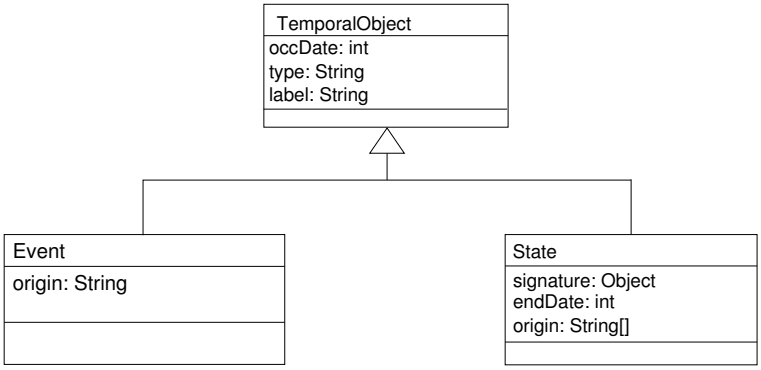


FIG. 6.1 – Diagramme UML des objets temporels.

Par exemple, les événements sont générés en sortie d'une tâche de détection sous forme de liste. Ils contiennent une date d'occurrence, un type, un attribut et une origine. Les états sont utilisés pour représenter les diagnostics de sortie d'une tâche de diagnostic

médical ou les intervalles de bruit de ligne en sortie d'un analyseur de ligne. Ainsi, un diagnostic de pathologie sera représenté par un objet de type `State` qui contient une date d'occurrence, un type, un attribut, une origine (qui peut être sous forme de liste), une signature (quels événements ou états ont généré le diagnostic) et une date de fin.

6.2.2 Algorithmes de traitement du signal

Chaque algorithme est représenté par un objet dédié à la réalisation d'un traitement de signal. Les algorithmes de traitement du signal sont organisés dans une hiérarchie selon des relations de généralisation et de spécification. Par exemple, pour chaque type de traitement (p. ex. filtrage, détection de QRS, etc.) il existe un objet parent duquel tous les algorithmes fils héritent. Le diagramme UML de la figure 6.2 représente cet héritage pour la classe *QRSDetection*.

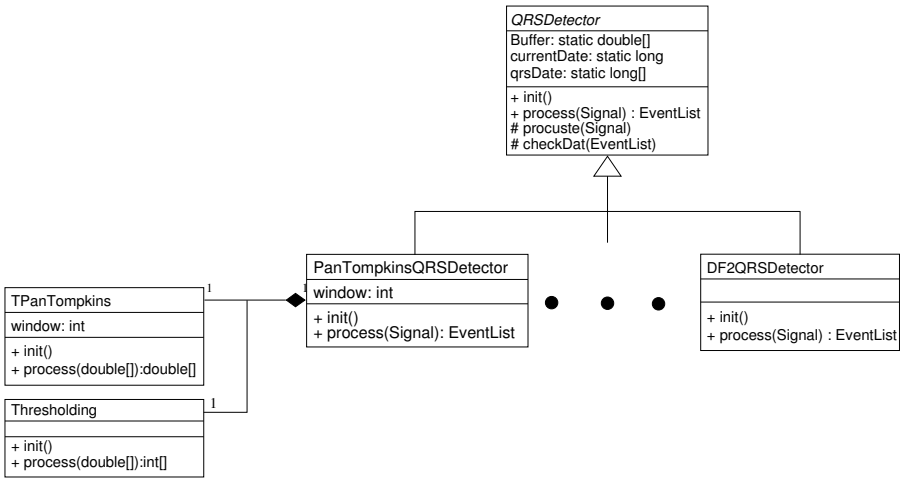


FIG. 6.2 – Diagramme UML des objets héritant de `QRSDetection`.

L'objet `QRSDetector` représente l'abstraction d'un algorithme de détection de QRS. Le triangle vide symbolise une relation d'héritage. Ainsi, l'objet `DF2QRSDetector` *est un type de* `QRSDetection`. Le losange noir traduit une relation de composition. L'objet `PanTompkinsQRSDetector` *est composé d'*un objet de type `TPanTompkins` et d'un objet de type `Thresholding`.

Comme la plupart des abstractions d'algorithmes, l'objet `QRSDetector` contient un `Buffer` pour stocker les variables d'entrée, la date courante `currentDate`, et un tableau destiné à contenir les occurrences de QRS `qrsDate`. Ses variables sont statiques pour permettre le partage des valeurs entre les différents détecteurs héritiers. Ainsi, lorsque le pilote décide de changer d'algorithme, les données sont conservées dans la classe mère et le nouvel algorithme peut directement les utiliser. Par exemple, certains algorithmes peuvent avoir besoin de la dernière valeur de l'intervalle entre deux QRS pour estimer un seuil. En cas de changement d'algorithme, celui qui a été sélectionné peut utiliser

directement la valeur plutôt que prendre une valeur par défaut. Les transitions d'algorithmes au sein d'un même type s'effectuent donc de manière transparente dans le système. La classe abstraite du type d'algorithme contient aussi quatre méthodes :

- `init` qui effectue l'initialisation de base,
- `process`, qui exécute le traitement,
- `procuste`, qui gère la taille du buffer interne,
- `checkDate`, qui élimine les dates aberrantes.

Les classes héritières, telle que la classe `PanTompkinsQRSDetector`, surchargent les méthodes `process` et `init`. Chaque algorithme peut être composé d'autres algorithmes pour réaliser les différentes étapes de leur traitement.

6.2.3 Contexte

Le contexte est l'ensemble des informations qui influent sur l'application (*cf.* 2.2). Il contient les variables courantes qui décrivent l'état actuel du patient à différents niveaux. Le contexte courant est composé de trois sous-contextes (*cf.* 3.2.3) : le *contexte de ligne*, le *contexte arythmique*, et le *contexte patient*.

Le contexte est développé sous forme de table de hachage qui permet d'associer à un attribut une valeur. Un extrait de contexte courant est donné Figure 6.3.

```
// contexte patient
basicBeat: N

// contexte de ligne
noiseType_1: bw
noiseLevel_1: 5

// contexte arythmique
qrsWaveform: N
mainArrhythmia: normal
```

Fig. 6.3 – Extrait de contexte courant.

Le contexte patient contient la description du battement de base du patient. Cette information est utile, par exemple, pour savoir si le patient présente un bloc de branche qui déforme le QRS. Le contexte de ligne contient des informations sur le type et le niveau de bruit présent sur chaque ligne (ici seule la ligne 1 est présentée). Le contexte arythmique indique quelle est la principale arythmie courante.

6.2.4 Tâches

D'après la définition 2.1, une tâche est l'exécution d'un traitement réalisé par un ou plusieurs algorithmes. Les tâches sont des classes spéciales mise en œuvre par le concepteur. Elles raffinent toutes la classe abstraite `Task` représentée Figure 6.4.

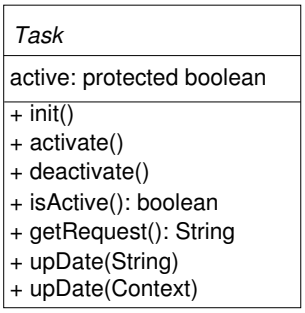


FIG. 6.4 – Diagramme UML de la classe `Task`.

Chaque tâche peut être *active* ou *inactive* selon que le pilote ait décidé de l'activer. Il s'ensuit que chaque objet de tâche doit contenir les méthodes suivantes :

- `activate` qui permet d'activer la tâche ;
- `deactivate` qui permet de désactiver la tâche ;
- `init` qui permet d'initialiser la tâche au début du lancement de l'application ;
- `getRequest` qui selon le contexte courant pris en entrée retourne la requête à envoyer au pilote pour mettre à jour la tâche ;
- `upDate` qui selon le contexte courant pris en entrée met à jour les variables internes qui dépendent du contexte ;
- `upDate` qui prend en entrée le résultat de la requête envoyée au pilote pour effectuer les modifications nécessaires dans la tâche,

De plus, chaque tâche doit contenir les deux méthodes suivantes :

- `<T> process(...)` qui effectue le traitement de la tâche et retourne le résultat de type `<T>` ;
- `<T> unactiveProcess(...)` qui prend les mêmes arguments mais retourne un résultat déterministe, par exemple un tableau de type `<T>` vide ;

Le traitement qui correspond à la tâche est réalisé par l'appel de la méthode `process`. Généralement, à chaque tâche est associée un type d'algorithme, le rôle de l'objet tâche est alors d'assurer les changements d'algorithmes demandés par le pilote et de mettre en forme les entrées et les sorties entre l'application et l'algorithme. Par exemple l'application peut solliciter le traitement d'un objet de type `Signal` alors que la tâche utilisera un algorithme qui n'accepte que des tableaux de flottants. Lorsque la tâche est inactive, la méthode `process` fait appel à la méthode `unactiveProcess`. Dans cette méthode les valeurs de retour sont spécifiées lors de la conception de l'application. Par exemple, lorsque la tâche de classification de QRS est inactive on peut choisir d'attribuer à toutes les occurrences de QRS la valeur `normal`, `anormal` ou `inconnue`. Les changements d'algorithmes s'effectuent lorsque la méthode `upDate` est appelée. Cette méthode récupère les actions à effectuer sur la tâche (activation, nom de l'algorithme à utiliser). La figure 6.5 donne des exemples d'entrées-sorties des méthodes de la tâche `QRSDetection`.

La méthode `getRequest` prend en entrée le contexte courant et récupère la requête

```
méthode : getRequest
     entrée : contexte: bruit = ma; niveau = -5
     sortie : activateTask(qrsDetection,-5,ma,Active),
              which_detector(NextDetector,-5,ma,N,gritzali)

méthode : process
     entrée : Signal[](...)
     sortie : EventList(Event(120, qrs, ?, voieII),
                        Event(240, qrs, ?, voieII),... )

méthode : unactiveProcess
     entrée : Signal[](...)
     sortie : EventList(vide)
```

FIG. 6.5 – Exemples d'entrées-sorties de la tâche `QRSDetection`.

à exécuter. Dans cette exemple, la requête `activateTask` demande au pilote si la tâche doit être activée et la requête `which_detector` demande quel algorithme doit être choisi. La méthode `process` prend en entrée un objet de type signal, applique un algorithme dessus puis retourne une liste d'événements donnant la date d'occurrence des complexes QRS. Enfin, la méthode `unactiveProcess` retourne toujours une liste d'événements vide.

6.3 Représentation par règles

La représentation par règles permet de structurer la connaissance du pilote pour la prise de décision. Le pilote est le cœur du système, c'est lui qui prend les décisions d'activer ou non les tâches et de choisir les algorithmes. Son fonctionnement est celui d'un système expert. Sa connaissance est représentée par des règles qu'il active pour prendre des décisions à partir des faits qui lui sont transmis (contenus dans le contexte).

Le pilote est un interpréteur PROLOG (SICStus PROLOG[2]) qui est lié au reste de l'application Java par une interface. Cette interface permet la communication de requêtes PROLOG vers le pilote et de retourner les résultats de l'évaluation de la requête au reste de l'application. L'intégration d'un interpréteur PROLOG dans le système permet par ailleurs de mettre à disposition toute la puissance de la programmation logique pour développer des tâches de haut niveau.

Les décisions du pilote sont calculées par l'activation de règles de production. Avant de prendre une décision, les requêtes sont traitées par des règles internes qui représentent les décisions du gestionnaire de contexte. Ces dernières sont exposées en section 6.3.1. Elles permettent de mettre à jours des informations pour prendre les trois types de décision finale du pilote : choisir le niveau d'abstraction du diagnostic médical, choisir

[2] www.sics.se/sicstus/

les tâches à activer et choisir les algorithmes qui sont respectivement présentés en section 6.3.2, 6.3.3 et 6.3.4. La plupart des règles employées proviennent d'expertises exceptées les règles de pilotage d'algorithmes qui sont tirées d'analyses statistiques. Toutes les règles sont exprimées dans le formalisme PROLOG. Les principaux éléments du langage sont donnés en annexe A.

6.3.1 Règles du gestionnaire de contexte

Le gestionnaire de contexte met à jour des variables utiles pour les étages de décisions en aval. La figure 6.6, présente un extrait des règles du gestionnaire de contexte.

```
/* determiner si une tâche est activable
   activableTask(Task,noiseLevel,noiseType) */

activableTask(_,_,no_noise).
activableTask(qrsDetection,NoiseLevel,_NoiseType):-
        -15 =< NoiseLevel.
activableTask(qrsClassification,NoiseLevel,_NoiseType):-
        0 =< NoiseLevel.
activableTask(arrhythmiaRecognition,_NoiseLevel,_NoiseType).
```

FIG. 6.6 – Exemple de règles du gestionnaire de contexte.

La tête de clause `activableTask` est appelée avec trois arguments qui sont : le nom de la tâche testée, le niveau de bruit et le type de bruit. Ces règles servent à déterminer si le niveau et le type de bruit permettent d'activer la tâche ou non. La première règle spécifie que toutes les tâches sont activables tant qu'il n'y a pas de bruit (`no_noise`). La deuxième clause spécifie que la tâche de détection de QRS est activable tant que le niveau de bruit reste supérieur ou égal à $-15dB$.

6.3.2 Règles de choix des modèles de chroniques pour la reconnaissance d'arythmies

Les modèles de chroniques doivent être adaptés à la précision de l'abstraction temporelle. Par exemple, si les tâches *QRSDetection*, *QRSClassification* sont actives et *PWaveDetection* est inactive, alors le reconnaisseur de chroniques doit utiliser la base de chroniques `exper2`. La figure 6.7, présente les règles de choix de modèles de chroniques sous forme de clauses PROLOG.

Le prédicat `chronicleResolution` permet de vérifier le choix d'un modèle de chroniques. Il existe une clause pour chaque modèle de chroniques `exper1, exper2, exper5`. Ces modèles de connaissances sont hiérarchiques (*cf.* 3.3) et la relation de généralité est mise en œuvre par une notion de priorité entre les règles qui s'exprime par l'ordre des clauses associées. Par exemple le placement de la clause `chronicleResolution(exper5)` devant la clause `chronicleResolution(exper2)` traduit le fait que `exper5` est moins

```
/* déterminer quelle est la résolution à avoir
   en fonction des besoins et des tâches
   chronicleResolution(ChronicleModel,NoiseLevel,NoiseType) */

chronicleResolution(exper5):-
        needTask(qrsDetection),bb_get(qrsDetection,activated),
        needTask(qrsClassification),bb_get(qrsClassification,activated),
        needTask(pWaveDetection),bb_get(pWaveDetection,activated).
chronicleResolution(exper2):-
        needTask(qrsDetection),bb_get(qrsDetection,activated),
        needTask(qrsClassification),bb_get(qrsClassification,activated),
        bb_get(pWaveDetection,deactivated).
chronicleResolution(exper1):-
        needTask(qrsDetection),bb_get(qrsDetection,activated),
        bb_get(qrsClassification,deactivated),
        bb_get(pWaveDetection,deactivated).
```

FIG. 6.7 – Règles de choix de modèles de chroniques.

général que `exper2`. Le prédicat `bb_get(X,Y)` est vrai si la variable interne `X` a la valeur `Y` ainsi le pilote peut vérifier que les tâches ont été préalablement activées ou non par le pilote.

6.3.3 Règles d'activation des tâches

Les tâches sont activées principalement selon le contexte de ligne et selon les besoins de la reconnaissance d'arythmies. La figure 6.8 présente la règle d'activation des tâches.

```
/* une tâche est active si elle est nécessaire et activable
   activateTask(Task,NoiseLevel,_NoiseType) */

activateTask(Task,NoiseLevel,NoiseType):-
        needTask(Task),
        activableTask(Task,NoiseLevel,NoiseType).
```

FIG. 6.8 – Règle d'activation de tâches.

Cette règle exprime que pour toute tâche `Task`, niveau de bruit `NoiseLevel`, et type de bruit `NoiseType`, `activateTask` est vrai si la tâche `Task` est nécessaire et qu'elle est activable.

6.3.4 Règles de pilotage d'algorithmes de traitement du signal

Les règles de pilotage d'algorithmes permettent de choisir et de paramétrer les algorithmes les plus adaptés à une tâche. La figure 6.9, présente des extraits de ces règles.

```
/* règles de pilotage d'algorithmes de QRS
   which_detector(NextDetector,NoiseType,NoiseLevel,
                  MainArrhythmia,QRSWaveform,CurrentDetector)*/

which_detector(no_change,no_noise,_,"(VT",_,gritzali).
which_detector(gritzali,no_noise,_,"(VT",_,D):-
        detector(D),
        D\=gritzali.
...

which_detector(no_change,_,_,"(BII",_,df2).
which_detector(df2,_,_,"(BII",_,D):-
        detector(D),
        D\=df2.

which_detector(no_change,em,-15,_,_,df2).
which_detector(df2,em,-15,_,_,D):-
        detector(D),
        D\=df2.
...

which_detector(benitez,_,_,_,_,X):-
        detector(X),
        X\= benitez.
which_detector(no_change,_,_,_,_,_).
```

FIG. 6.9 – Extrait de règles de choix d'algorithmes.

Le prédicat `which_detector` permet de choisir le détecteur à utiliser en fonction du type de bruit, du niveau de bruit, de l'arythmie la plus probable, de la principale forme de QRS et de l'algorithme courant employé. La première clause spécifie que s'il n'y a pas de bruit présent sur la ligne et que l'arythmie principale est une tachycardie ventriculaire ("(VT"), alors il est préférable de choisir le détecteur `gritzali`. La dernière clause spécifie que si aucune des règles précédentes n'est satisfaite alors il faut utiliser le détecteur `benitez`. Ces règles sont dérivées d'une analyse statistique présentée au chapitre 4.

6.4 Connaissances indépendantes et dépendantes du domaine

Deux types de connaissances cohabitent dans le système : les connaissances liées aux algorithmes de traitement du signal et les connaissances liées à l'application. Pour reprendre la définition donnée dans (Shekhar et coll., 1994), ce sont les *connaissances indépendantes* du domaine d'application et les *connaissances dépendantes* du domaine.

Les connaissances indépendantes du domaine sont celles qui ne nécessitent pas de connaissances *a priori* sur le domaine d'application. Dans le système présenté, il s'agit des connaissances sur les *algorithmes*. Ceux-ci contiennent uniquement les traitements et la procédure d'initialisation. Contrairement à (Shekhar et coll., 1994), les algorithmes ne contiennent pas de méthodes d'ajustement ceci pour deux raisons. Premièrement, les objectifs temps réel de l'application visée (monitorage cardiaque) et l'impossibilité de contrôler les résultats ne permettent pas de phase de réajustement des paramètres. Deuxièmement, comme il n'existe pas de phase d'ajustement, le meilleur ajustement des algorithmes est fixé par des connaissances *a priori* qui dépendent du domaine d'application (p. ex. le contexte arythmique). Par exemple, un filtre d'entrée de ligne sélectionne la bande spectrale utile grâce à une connaissance *a priori*. Il paraît difficile de calculer automatiquement les coefficients du filtre par un mécanisme d'*essai-erreur*. C'est pourquoi le paramétrage des algorithmes est effectué par des connaissances dépendantes et indépendantes du domaine.

Les connaissances dépendantes du domaine représentent les connaissances *a priori* disponibles et utiles sur le domaine d'application. Dans le système présenté, les connaissances dépendantes du domaine sont fixées dans l'*application*, les *tâches* et les *règles de pilotage*. L'application nécessite de connaître la succession des tâches nécessaires à la réalisation de l'application et les connexions de signaux, d'événements et d'états entre les tâches. Les tâches sont mises en œuvre pour utiliser les algorithmes et faire l'interface des entrées-sorties entre l'application et les algorithmes. Par exemple, c'est la tâche `QRSDetection` qui étiquette les sorties des algorithmes de détection de QRS comme étant des événements QRS. Les règles de pilotage quant à elles, sont constituées de connaissances dépendantes et indépendantes. Elles sont donc utilisables uniquement pour l'application donnée.

6.5 Détails du mécanisme de pilotage

Cette section présente le fonctionnement général de l'application de monitorage. Tout d'abord la réalisation d'une application est détaillée puis l'exécution du pilotage est décrite.

6.5.1 Conception d'une application

La réalisation d'une application passe tout d'abord par la définition de la succession des tâches à accomplir et la mise en place de l'infrastructure qui permet la transmission des données entre les tâches. Les échanges entre tâches sont réalisés par des signaux, des

listes d'événements ou d'états. Dans IP-Calicot, les signaux sont traités par blocs. Il s'ensuit que, les tâches sont exécutées séquentiellement et sont comprises dans une boucle qui traite les données bloc par bloc. La figure 6.10 représente le diagramme de l'application de monitorage cardiaque.

À chaque pas de traitement, l'application attend les données provenant de l'acquisition de données. Lorsque les données sont en nombre satisfaisant, l'application est lancée. Le contexte est tout d'abord mis à jour puis la tâche de filtrage ôte le bruit parasite et envoie le signal traité à la tâche de détection de QRS. Celle-ci envoie les occurrences de QRS détectés à la tâche de classification qui, à l'aide du signal filtré, affecte un attribut au QRS. Dans notre cas, il s'agit des étiquettes : `basic` et `nonbasic`. La détection d'ondes P utilise ensuite le signal filtré pour estimer les occurrences d'ondes P. Les événements de QRS et d'onde P sont utilisés par la reconnaissance d'arythmies qui retourne les diagnostics médicaux sous forme d'états. Les événements et les états sont ensuite envoyés au module d'affichage via le réseau. Chaque tâche peut être désactivée en fonction du contexte. Lorsqu'une tâche est désactivée elle devient transparente dans la boucle de traitement.

Par ailleurs, il faut aussi définir quelles informations vont servir à piloter le système. La définition du contexte et des analyseurs de contexte est donc une opération importante. L'analyse du contexte est perçue comme une tâche à part entière qui pourrait elle-même être pilotée.

L'étape suivante consiste à définir toutes les tâches. Chaque tâche hérite de la tâche abstraite `Task`. Le concepteur d'une application doit développer les méthodes qui génèrent les requêtes en direction du pilote et récupèrent les résultats. La tâche a souvent le rôle d'assurer un formatage des données de sorties. Par exemple, la tâche `QRSClassification` peut utiliser un algorithme de classification à deux sorties de type `normal`, `abnormal` alors que les attributs utilisés dans la tâche de reconnaissance d'arythmies sont `basic`, `nonbasic` (QRS de base, ou non de base). La tâche devra donc faire la traduction entre les différentes représentations.

L'étape suivante est la constitution de la base d'algorithmes. La création d'un nouveau type d'algorithme doit passer par la définition de la classe abstraite décrivant les attributs et méthodes de base du type d'algorithme. Le nouvel algorithme doit ensuite hériter de cette classe et doit mettre en œuvre son traitement et son initialisation.

L'étape terminale est la saisie des règles de pilotage à partir d'expertises ou de méthodes d'acquisition de règles.

6.5.2 Exécution du pilotage

Comme dit précédemment, l'application est exécutée pas à pas en fonction des données qui sont traitées bloc par bloc. Le pilotage, quant à lui, est exécuté pour chaque tâche. Avant chaque exécution d'une tâche, celle-ci envoie une requête au pilote. La figure 6.11 illustre cet échange.

La requête consiste à demander quelles actions la tâche doit effectuer (p. ex. désactivation, changement d'algorithmes). Le pilote utilise ses règles qu'il active avec les données envoyées pour répondre aux requêtes. Le pilote retourne ainsi les actions à

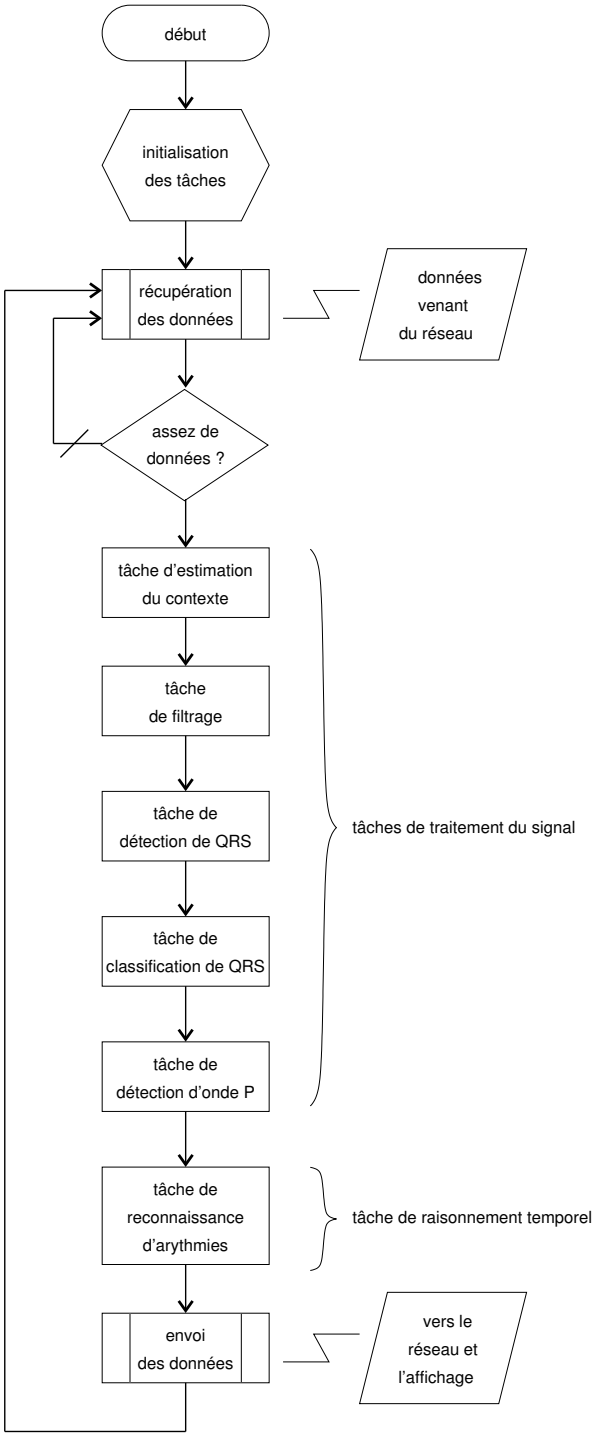


FIG. 6.10 – Diagramme de l'application de monitorage cardiaque.

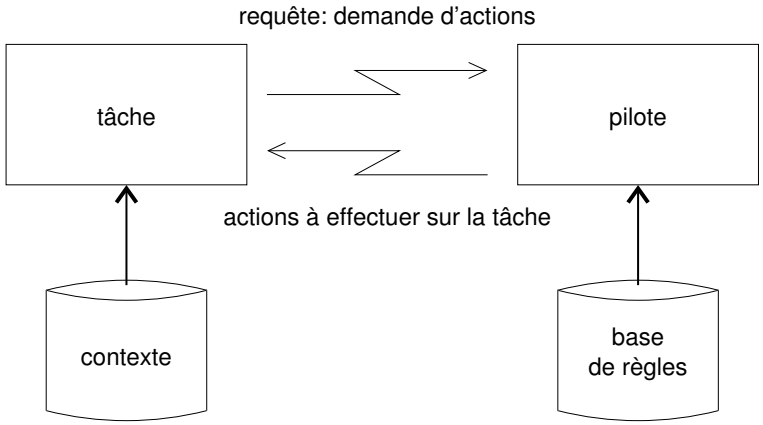


FIG. 6.11 – Exemple de communication entre une tâche et le pilote.

effectuer ou `no_change` pour spécifier qu'aucun changement ne doit être entrepris.

Cette approche permet de gérer les échanges de données de manière efficace et centralisée. Les gros volumes de données tels que les signaux sont envoyés directement aux tâches qui doivent les traiter tandis que seules les données de pilotage, qui sont d'un faible volume, sont échangées avec le pilote. Il n'y a donc pas de risque d'engorgement tel qu'il peut y en avoir avec une architecture multiagent communiquant par messages. De plus, les faibles volumes échangés permettent de respecter les contraintes temps réel. L'architecture proposée permet aussi l'ajout aisé de nouvelles tâches dans une application.

6.6 Conclusion

L'approche proposée permet la conception simple d'une application. Les représentations par objets et par règles ont déjà montré leur efficacité (Clément, 1990; Moisan, 1998; Shekhar et coll., 1999). Cette approche reste très dédiée au monitorage cardiaque mais certaines améliorations pourraient rendre ce pilotage d'algorithmes plus générique.

- Le développement de l'application nécessite de coder directement les tâches. Ceci rend la conception d'une application très dépendante de la connaissance que peut avoir l'utilisateur du langage de programmation utilisé. Une évolution consisterait à procurer une aide à la conception pour représenter ensuite la chaîne de traitements sous forme d'un graphe comme dans (Clément et Thonnat, 1993) ou (Sztipanovits et coll., 1998), ou sous forme de schéma XML. Cette évolution permettrait aussi de stocker les chaînes de traitements pour pouvoir les réutiliser.
- Les classes abstraites des algorithmes permettent le partage des variables entre les différents héritiers sous forme de variables `static`. Cette mise en œuvre n'est pas satisfaisante dans le cas où plusieurs tâches utilisent les mêmes types d'al-

gorithmes, comme par exemple dans le cas de détection de QRS sur plusieurs voies.

- La connaissance du pilote est regroupée dans le même fichier PROLOG ce qui ne facilite pas le maintien de la base lorsque le nombre de règles devient important. Une manière de mieux structurer la connaissance serait soit de la stocker dans les tâches et algorithmes, soit de mettre au point un éditeur permettant de manipuler aisément les règles stockées dans un même fichier. La première solution qui se rapproche des travaux de (Clément et Thonnat, 1993; Moisan, 1998; Shekhar et coll., 1999) (règles d'initialisation, règles d'ajustement, etc.) est applicable dans le cas ou il existe une bonne séparation des connaissances qui permette la réutilisation des algorithmes. Dans notre cas, les types de connaissance sont mélangés pour prendre les décisions et seule une approche centralisée le permet. C'est pourquoi la solution de l'éditeur semble la plus adéquate.
- Les liaisons des modules via le réseau permettent la communication à distance, mais une solution doit aussi être proposée pour permettre une communication plus directe (et plus rapide) notamment entre l'acquisition de signaux et le traitement des données.
- Une évolution dans le traitement des données serait de permettre de retraiter les données de manière plus précise à la manière de (Lesser et coll., 1993b). Par exemple, lorsque le nombre de ressources est limité, les traitements peuvent se limiter aux traitements vitaux. Les données traitées de cette manière pourraient être retraitées en tâche de fond de l'activité de monitorage lorsque les ressources redeviennent libres.
- Pendant la phase d'initialisation certains paramètres d'algorithmes de traitement du signal pourraient être appris. Par exemple, les classificateurs de QRS pourraient apprendre les propriétés des complexes QRS durant une phase dédiée pour être plus spécifiques au patient.

Conclusion

Ce mémoire décrit l'étude et la réalisation d'un système de monitorage de patient capable de fonctionner et de s'adapter aux multiples situations rencontrées en Unité de Soins Intensifs pour Coronariens (USIC). Dans ce type de monitorage médical, les fonctions physiologiques (c.-à-d. les signes vitaux) d'un patient sont mesurées pour détecter des situations anormales, les caractériser (diagnostic médical) et solliciter, par des alarmes, une intervention du personnel clinique. Ces systèmes permettent ainsi la surveillance 24 heures sur 24 des patients nécessitant des soins intensifs et apportent une assistance au personnel médical. Cependant, dans l'environnement hostile que constitue l'USIC, le matériel médical informatique et électronique subit l'influence de nombreuses perturbations extérieures (interférences, mouvements des patients, mouvements de câbles, etc.). Ces bruits limitent les performances des systèmes car ils entraînent le déclenchement d'un grand nombre d'alarmes intempestives sans signification médicale et, surtout, conduisent à un diagnostic médical erroné. La réalisation de systèmes de monitorage pouvant assurer une surveillance fiable dans un contexte bruité en milieu USIC est donc un objectif majeur.

Depuis la fin des années 80, cet objectif est devenu de plus en plus accessible grâce à l'émergence des *systèmes de monitorage intelligent*. Ces systèmes intègrent des algorithmes de traitement du signal pour analyser les signes physiologiques et des outils d'intelligence artificielle pour réaliser un diagnostic en ligne. Lorsque les signaux analysés deviennent bruités, certains de ces systèmes adaptent le monitorage aux données traitées de façon à limiter les effets néfastes du bruit de ligne. Toutefois, ces systèmes ont, en général, une approche qui leur est spécifique et peu d'entre elles proposent des adaptations génériques à tous les niveaux de la chaîne de traitement : de l'analyse du signal au diagnostic médical. De plus, dans ces systèmes, le problème de la réduction des effets du bruit est vu comme étant uniquement du domaine du traitement du signal ou du domaine de l'intelligence artificielle et peu de recherches ont tiré parti du diagnostic médical et de l'analyse du bruit de ligne pour optimiser la chaîne de traitement.

Dans notre approche, pour obtenir un diagnostic médical fiable même dans des conditions d'utilisations extrêmes, une technique de pilotage d'algorithmes est définie afin de modifier dynamiquement la chaîne de traitement et adapter le diagnostic médical en fonction d'informations telles que le bruit de ligne et les pathologies suspectées. Ce pilotage est intégré au système de monitorage des arythmies cardiaques Calicot, développé au laboratoire, pour réaliser un nouveau système de monitorage, appelé IP-Calicot (*Integrated Piloting and Cardiac Arrhythmias Learning for Intelli-*

gent Classification of On-line Tracks). Le pilote d'algorithmes mis en œuvre s'inspire de différents travaux présentés dans la littérature. La flexibilité du système de monitorage est réalisée de manière originale en intégrant un pilotage à trois niveaux : 1) au niveau de la *reconnaissance des arythmies cardiaques*, 2) au niveau des *tâches d'abstraction du signal* et 3) au niveau des *algorithmes de traitement du signal*.

Le pilotage de la reconnaissance d'arythmies cardiaques permet de modifier en ligne le langage de description des signaux physiologiques, qui sont ici des électrocardiogrammes (ECG) en fonction du bruit de ligne. Généralement une pathologie est diagnostiquée à partir de caractéristiques qui doivent être constamment extraites des signaux (dans notre cas, pour l'ECG il s'agit de la date d'occurrence des QRS, des formes de QRS, et des ondes P). L'extraction de toutes ces caractéristiques correspond à un langage de description très précis qui peut être trop exigeant dans des situations où le signal n'est pas de bonne qualité. Dans IP-Calicot, le pilote est un système expert dont les décisions de pilotage sont représentées par des règles de production. Il sélectionne dynamiquement, dans une hiérarchie de langages de description partant du plus abstrait (occurrences de QRS seules) au plus précis, le langage de description de la reconnaissance d'arythmies pour assurer un diagnostic fiable. Par exemple, s'il y a présence de bruit sur la ligne, l'extraction de l'onde P peut être très perturbée et accompagnée d'erreurs. Il est alors préférable d'ignorer l'information d'onde P pour réaliser le diagnostic jusqu'à ce que le niveau de bruit redevienne acceptable. De plus, l'extraction de certaines caractéristiques du signal pourrait être désactivée de façon à économiser des ressources de calcul. Pour représenter ces différentes stratégies de diagnostic, un ensemble de modèles de chroniques (c.-à-d. modèles d'arythmies) est appris en fonction d'une hiérarchie de langages de description de l'ECG. Les modèles de chroniques obtenus par apprentissage artificiel ont ensuite été testés sur des enregistrements réels.

Le pilotage de tâches d'abstraction du signal permet de désactiver certaines tâches d'extraction de caractéristiques du signal. Par exemple, lorsque la ligne est trop bruitée, la classification du QRS est vouée à l'échec et entraîne beaucoup d'erreurs. Dans ce contexte, cette tâche est pénalisante car elle fournit beaucoup de fausses informations à la reconnaissance de chroniques. De même, lorsque le pilotage de la reconnaissance d'arythmies choisit des modèles de chroniques qui n'ont pas besoin de certaines caractéristiques, les tâches associées aux caractéristiques inutiles peuvent être désactivées. Pour fonder le diagnostic sur des informations fiables et économiser des ressources, les tâches sont activées et désactivées en fonction du bruit de ligne. Ce pilotage des tâches a été expérimenté avec des enregistrements de patients auxquels a été ajouté du bruit clinique réel. Les résultats montrent que ce pilotage permet d'assurer un diagnostic médical quel que soit la quantité de bruit présent alors que le même système non piloté n'est plus capable d'assurer sa tâche de monitorage. Ces résultats ont conduit à une publication (Portet et coll., 2006).

Le pilotage des algorithmes de traitement du signal consiste à choisir les algorithmes les plus adaptés à une situation donnée. Chaque tâche d'extraction de caractéristiques est réalisée par un algorithme de traitement du signal. Or, une tâche peut être réalisée par plusieurs algorithmes, plus ou moins performants selon le contexte. Un contexte est défini ici comme étant la combinaison particulière d'un type de bruit et d'arythmie. Pour réaliser ce pilotage, les règles qui permettent au pilote d'associer un contexte à un algorithme particulier ont été extraites par une méthode originale basée sur une analyse factorielle. Cette méthode a permis d'obtenir des règles qui ont non seulement rendu possible le pilotage mais ont aussi apporté une connaissance plus profonde des algorithmes utilisés. Cette méthode a fait l'objet de deux publications (Portet et coll., 2005c; Portet et Carrault, 2005). Ce niveau de pilotage a été testé sur des enregistrements bruités. Ces tests ont montrés que le pilotage apporte une amélioration significative de la tâche de détection de complexes QRS.

Les résultats du pilotage obtenus lors des tests en situations cliniques sortant des « cas d'école » ont permis de valider l'approche proposée. La définition du contexte donnée dans ce mémoire a été montrée comme pertinente pour les différents niveaux de pilotage. L'intégration du pilotage permet donc une amélioration générale de la robustesse du diagnostic médical, ce qui est un objectif majeur des systèmes de monitorage.

La mise en place du pilotage dans le système Calicot a par ailleurs amené d'autres contributions tout aussi importantes :

- Le travail présenté a permis d'améliorer notablement le système de monitorage cardiaque initial Calicot. La première phase de cette amélioration a été de mettre en œuvre la partie extraction de caractéristiques et reconnaissance d'arythmie dans un même système. La structuration apportée par les concepts de pilotage dans le système IP-Calicot offre une architecture flexible qui permet d'ajouter aisément de nouveaux traitements. Ce système a été présenté dans deux publications (Portet et coll., 2005a; Portet et coll., 2005b).
- La capacité de *reconnaissance des arythmies* de Calicot a aussi été augmentée. À titre d'exemple, Calicot ne permettait pas de détecter des extrasystoles et des bigéminismes dans des rythmes avec blocs de branche. Une nouvelle description du QRS tenant compte du contexte patient permet d'être plus générique.
- Un plus grand nombre d'arythmies a été appris augmentant ainsi les capacités de reconnaissances initiales.
- L'architecture proposée permet un développement logiciel particulièrement adapté à l'USIC. En effet, les différentes fonctionnalités (acquisition, traitement et affichage) peuvent être distribuées à travers le réseau d'un hôpital pour permettre un suivi du patient dans des lieux différents. De ce fait, IP-Calicot pourrait être utilisé dans un cadre de télémonitorage
- Enfin, la gestion des données multisources a aussi été abordée. Les résultats préliminaires obtenus par le système en reconnaissance d'arythmies à partir de données d'ECG et de pression artérielle sont encourageants. Cette étude a conduit à une

publication (Fromont et Portet, 2005). Cette expérience a, de plus, montré la facilité avec laquelle de nouveaux modules peuvent être intégrés au système.

Ce travail ouvre la voie à de futures évolutions qui augmenteront encore les fonctionnalités du système de monitorage.

- La détection de contexte n'a été que partiellement abordée dans ce travail et reste un point à développer et à intégrer à IP-Calicot. Elle pourrait être effectuée par l'association d'une analyse multirésolution et le calcul de l'énergie en sous-bande. Les premiers résultats enregistrés sont encourageants mais doivent être finalisés.
- Les rythmes cardiaques normaux sont variables d'un patient à un autre et l'utilisation de seuils fixes amène des erreurs de classification. Une courte période d'apprentissage au début de la phase de monitorage pourrait être une solution pour adapter les seuils utilisés pour classer les rythmes cardiaques. Cette phase d'apprentissage pourrait aussi être utilisée par les algorithmes de l'abstraction temporelle tels que la classification des QRS afin d'apprendre les formes de QRS spécifiques au patient.
- La représentation des chroniques par des langages de description très abstraits amène à un nombre de fausses alarmes encore trop important en raison d'une mauvaise discrimination. Une solution serait de regrouper les arythmies apprises en familles d'arythmies appartenant au même rythme. Par exemple, les tachycardies ventriculaires et supra-ventriculaires pourraient être regroupées sous la même famille de rythmes rapides. Cette classification garderait une grande partie de sa signification clinique car les rythmes rapides impliquent une intervention médicale urgente même si la thérapie nécessitée n'est pas la même.
- Les règles, apprises lors de l'apprentissage des arythmies, peuvent couvrir plusieurs fois les mêmes exemples entraînant de fait plusieurs alarmes d'une même arythmie ou d'arythmies différentes pour une même cause. Une tâche de filtrage des alarmes intégrée au système pourrait limiter ces erreurs. En effet, un simple filtrage avec gestion de priorités entre les arythmies ou agrégation d'alarmes de même type permettrait de réduire le nombre d'alarmes non pertinentes. Par exemple, lorsqu'une arythmie mineure telle qu'une contraction prématurée du ventricule est reconnue en même temps qu'une arythmie sévère telle que la tachycardie ventriculaire alors l'arythmie mineure peut être abandonnée car elle est moins prioritaire. Ainsi une seule alarme serait considérée (la plus sérieuse) au lieu de deux ce qui réduirait le nombre de fausses alarmes et les alarmes redondantes. Ce filtrage d'alarmes pourrait entrer dans le cadre plus général d'aide à la décision. Par, exemple, l'utilisateur pourrait choisir de désactiver les alarmes des arythmies mineurs, d'afficher les causes qui ont déclenché les alarmes, de demander des suggestions de thérapies, etc.

L'analyse et l'interprétation de signaux bruités est une tâche très difficile étant donné que les informations utiles sont masquées par des bruits parasites. Nous avons vu que l'extraction des caractéristiques pertinentes à partir de ces signaux peut être améliorée

par le pilotage de méthodes adaptées au contexte. IP-Calicot apporte ainsi des solutions originales qui améliorent nettement la surveillance en milieu bruité, cependant, dans des cas extrêmes, les informations extraites par les algorithmes de traitement du signal restent erronées et provoquent des erreurs de diagnostic. Dans ce cas, la solution la plus intéressante pour assurer la tâche de monitorage serait d'utiliser plusieurs sources redondantes afin de proposer un diagnostic médical plus robuste au bruit. Cet objectif est cohérent avec l'application du monitorage de patient car les USIC sont équipées de matériel d'acquisition de signaux physiologiques multisources. L'extension de IP-Calicot à la gestion de données multisources doit donc être considérée comme une évolution de ce travail. La solution de système de monitorage présentée peut s'appliquer à de nombreux systèmes de surveillance dans le domaine médical tel que pour les unités de soins intensifs ou l'anesthésie ou dans le domaine industriel tel que dans le cas de surveillance de systèmes dynamiques.

Annexe A

Formalisme PROLOG

Cette annexe a pour but d'introduire succinctement le formalisme de PROLOG afin de pouvoir comprendre les règles présentées dans ce mémoire. Une présentation complète de PROLOG peut être trouvée dans (Sterling et Shapiro, 1986). PROLOG est un langage de programmation qui repose sur les principes de la programmation en logique. Une variable est représentée par une chaîne alphanumérique commençant par une majuscule (p. ex. `QRS`, `NoiseLevel`) ou par un souligné (p. ex. `_NoiseLevel`). Le caractère « `_` ■ représente une variable anonyme. Une constante, une fonction et un prédicat sont exprimés par une chaîne alphanumérique commençant par une minuscule (p. ex. `activableTask`, `qrsDetection`). Un atome logique de la forme `symbole_de_prédicat(t1, ..., tn)` exprime une relation entre les termes $t1$ à tn. Une clause est de la forme $\mathtt{A}_0$ `:-` $\mathtt{A}_1$`, ...,`$\mathtt{A}_n$ où $\mathtt{A}_0$`,` $\mathtt{A}_1$`, ...,`$\mathtt{A}_n$ sont des atomes logiques. Une clause signifie que la relation $\mathtt{A}_0$ est vraie si les relations $\mathtt{A}_1\wedge$`, ...,`$\wedge\mathtt{A}_n$ sont vraies. $\mathtt{A}_0$ est appelé tête de clause et l'ensemble $\mathtt{A}_1$`, ...,`$\mathtt{A}_n$ est appelé corps de clause. Une variable apparaissant dans la tête de clause est quantifiée universellement. Une variable apparaissant dans le corps d'une clause est quantifiée existentiellement. Par exemple, la clause `même_père(X,Y) :- père(P,X),père(P,Y).` se lit : « pour tout `X` et pour tout `Y`, `même_père(X,Y)` est vrai s'il existe un `P` tel que `père(P,X)` et `père(P,Y)` soient vrais. ■

En PROLOG, la mise en œuvre d'un prédicat est un programme logique constitué d'une succession de clauses de même tête et de même nombre d'arguments. Lors de la résolution d'une requête, les clauses sont évaluées dans l'ordre de leur apparition dans le programme jusqu'à ce que l'une des clauses soit vraie. Ainsi, le programme suivant :

```
parent(X,Y):- mère(X,Y).
parent(X,Y):- père(X,Y).
```

est traduit par « pour tout `X` et pour tout `Y`, `parent(X,Y)` est vrai si `mère(X,Y)` est vrai **ou** `père(X,Y)` est vrai. »

Annexe B

Règles hiérarchiques de reconnaissance d'arythmies

Cette annexe présente l'ensemble des règles obtenues par programmation logique inductive (PLI) sur les exemples d'arythmies décrits par la hiérarchie de langages de description présentée en section 3.3.2.2.

Les règles sont représentées dans le formalisme d'une base de faits PROLOG. Chaque tête de clause représente un modèle de classe d'arythmie suivi du nombre d'exemples couverts par la règle. Dans toutes les règles suivantes, les couvertures de classes sont représentées comme suit : `[[exemples couverts],[exemples non couverts]]` en respectant l'ordre suivant :

```
[bigeminy, mobitzII, normal, pvc, doublet, vt]
```

Par exemple,

```
class(bigeminy):- %[[15,0,0,0,0,0],[5,20,25,20,13,11]]
```

signifie que 15 exemples de la classe `bigeminy` sont couverts par cette règle et qu'aucun exemple des autres classes n'est couvert. 5 exemples de bigéminisme ne sont pas couverts par cette règle.

B.1 Exper1

```
class(bigeminy):- %[[15,0,0,0,0,0],[5,20,25,20,13,11]]
   qrs(R0, _),qrs(R1, R0), rr1(R0, R1, short),
   qrs(R2, R1),rr1(R1, R2, normal),rr2(R0, R2, normal),
   qrs(R3, R2),rr1(R2, R3, short).
class(bigeminy):- %[[5,0,0,0,0,7],[15,20,25,20,13,4]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1, short),qrs(R2, R1),
   rr1(R1, R2, short),qrs(R3, R2),rr1(R2, R3, short),
   qrs(R4, R3),rr1(R3, R4, short).

class(mobitzII):- %[[0,20,0,0,0,0],[20,0,25,20,13,11]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1, long),qrs(R2, R1),
   rr1(R1, R2, long),qrs(R3, R2),rr1(R2, R3, long).

class(normal):- %[[1,0,24,12,0,1],[19,20,1,8,13,10]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1, normal),qrs(R2, R1),
   rr1(R1, R2, normal),qrs(R3, R2),rr1(R2, R3, normal),qrs(R4, R3),
   rr1(R3, R4, normal),qrs(R5, R4),rr1(R4, R5, normal).

class(pvc):-%[[1,0,0,17,0,1],[19,20,25,3,13,10]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1, normal),qrs(R2, R1),
   rr1(R1, R2, normal),qrs(R3, R2),rr1(R2, R3, short),qrs(R4, R3),
   rr1(R3, R4, normal),qrs(R5, R4),rr1(R4, R5, normal).
class(pvc):-%[[4,0,0,17,0,1],[16,20,25,3,13,10]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1, normal),qrs(R2, R1),
   rr1(R1, R2, short),qrs(R3, R2),rr1(R2, R3, normal),qrs(R4, R3),
   rr1(R3, R4, normal),qrs(R5, R4),rr1(R4, R5, normal).

class(doublet):-%[[0,0,0,0,6,0],[20,20,25,20,7,11]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1, short),qrs(R2, R1),
   rr1(R1, R2, normal),rr2(R0, R2, short),qrs(R3, R2),rr1(R2, R3, normal).
class(doublet):-%[[0,0,0,0,4,0],[20,20,25,20,9,11]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1, short),qrs(R2, R1),
   rr1(R1, R2, short),qrs(R3, R2),rr1(R2, R3, normal),rr2(R1, R3, normal).
class(doublet):-%[[0,0,0,0,1,0],[20,20,25,20,12,11]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1, short),qrs(R2, R1),rr1(R1, R2, short),
   qrs(R3, R2),rr1(R2, R3, long).

class(vt):-%[[5,0,0,0,0,9],[15,20,25,20,13,2]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1,short),qrs(R2, R1),rr1(R1, R2,short),
   qrs(R3, R2),rr1(R2, R3,short).
class(vt):-%[[0,0,0,0,0,1],[20,20,25,20,13,10]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1,normal),qrs(R2, R1),rr1(R1, R2,short),
   qrs(R3, R2),rr1(R2,R3,short),qrs(R4, R3),rr1(R3, R4,normal),rr2(R2, R4,short).
class(vt):-%[[1,0,0,0,0,4],[19,20,25,20,13,7]]
   qrs(R0, _),qrs(R1, R0),rr1(R0, R1,short),qrs(R2, R1),rr1(R1, R2,normal),
   rr2(R0, R2,short),qrs(R3, R2),rr1(R2, R3,short).
```

FIG. B.1 – Ensemble complet de règles pour exper1.

B.2 Exper2

```
class(bigeminy):-%[[16,0,0,0,0,0],[4,20,25,20,13,11]]
   qrs(R0,basic, _),qrs(R1,nonbasic, R0),rr1(R0, R1,short),qrs(R2,basic, R1),
   rr1(R1, R2,normal),qrs(R3,nonbasic, R2),rr1(R2, R3,short).
class(bigeminy):-%[[5,0,0,0,0,0],[15,20,25,20,13,11]]
   qrs(R0,basic, _),qrs(R1,nonbasic, R0),rr1(R0, R1,short),qrs(R2,basic, R1),
   rr1(R1, R2,short),qrs(R3,nonbasic, R2),rr1(R2, R3,short).

class(mobitzII):-%[[0,17,0,0,0,0],[20,3,25,20,13,11]]
   qrs(R0,nonbasic, _),qrs(R1,nonbasic, R0),rr1(R0, R1,long), qrs(R2,nonbasic, R1),
   rr1(R1, R2,long),qrs(R3,nonbasic, R2),rr1(R2, R3,long).
class(mobitzII):-%[[0,3,0,0,0,0],[20,17,25,20,13,11]]
   qrs(R0,basic, _),qrs(R1,basic, R0),rr1(R0, R1,long),qrs(R2,basic, R1),
   rr1(R1, R2,long),qrs(R3,basic, R2),rr1(R2, R3,long).

class(normal):-%[[0,0,24,6,0,0],[20,20,1,14,13,11]]
   qrs(R0,basic, _),qrs(R1,basic, R0),rr1(R0, R1,normal),qrs(R2,basic, R1),
   rr1(R1, R2,normal),qrs(R3,basic, R2),rr1(R2, R3,normal),qrs(R4,basic, R3),
   rr1(R3, R4,normal),qrs(R5,basic, R4),rr1(R4, R5,normal).

class(pvc):-%[[0,0,0,18,0,0],[20,20,25,2,13,11]]
   qrs(R0,basic, _),qrs(R1,basic, R0),rr1(R0, R1,normal),qrs(R2,nonbasic, R1),
   rr1(R1, R2,short),qrs(R3,basic, R2),rr1(R2, R3,normal),qrs(R4,basic, R3),
   rr1(R3, R4,normal).
class(pvc):-%[[0,0,0,2,0,0],[20,20,25,18,13,11]]
   qrs(R0,basic, _),qrs(R1,basic, R0),rr1(R0, R1,normal),qrs(R2,basic, R1),
   rr1(R1, R2,normal),qrs(R3,nonbasic, R2),rr1(R2, R3,normal).

class(doublet):-%[[0,0,0,0,7,0],[20,20,25,20,6,11]]
   qrs(R0,basic, _),qrs(R1,nonbasic, R0),rr1(R0, R1,short),qrs(R2,nonbasic, R1),
   rr1(R1, R2,normal),qrs(R3,basic, R2),rr1(R2, R3,normal).
class(doublet):-%[[0,0,0,0,4,0],[20,20,25,20,9,11]]
   qrs(R0,basic, _),qrs(R1,basic, R0),rr1(R0, R1,normal),qrs(R2,nonbasic, R1),
   rr1(R1, R2,short),qrs(R3,nonbasic, R2),rr1(R2, R3,short).
class(doublet):-%[[0,0,0,0,4,0],[20,20,25,20,9,11]]
   qrs(R0,basic, _),qrs(R1,nonbasic, R0),rr1(R0, R1,short),qrs(R2,nonbasic, R1),
   rr1(R1, R2,short),qrs(R3,basic, R2),rr1(R2, R3,normal).
class(doublet):-%[[0,0,0,0,1,0],[20,20,25,20,12,11]]
   qrs(R0,basic, _),qrs(R1,nonbasic, R0),rr1(R0, R1,normal),qrs(R2,nonbasic, R1),
   rr1(R1, R2,short),qrs(R3,basic, R2),rr1(R2, R3,normal).

class(vt):-%[[0,0,0,0,0,9],[20,20,25,20,13,2]]
   qrs(R0,nonbasic, _),qrs(R1,nonbasic, R0),rr1(R0, R1,short),qrs(R2,nonbasic, R1),
   rr1(R1, R2,short),qrs(R3,nonbasic, R2),rr1(R2, R3,short).
class(vt):-%[[0,0,0,0,0,5],[20,20,25,20,13,6]]
   qrs(R0,nonbasic, _),qrs(R1,nonbasic, R0),rr1(R0, R1,normal),qrs(R2,nonbasic, R1),
   rr1(R1, R2,short),qrs(R3,nonbasic, R2),rr1(R2, R3,short).
```

FIG. B.2 – Ensemble complet de règles pour exper2.

B.3 Exper3

```
class(bigeminy):-%[[16,0,0,0,0,0],[4,20,25,20,13,11]]
   qrs(R0,basic, _),qrs(R1,pvc, R0),rr1(R0, R1,short),qrs(R2,basic, R1),
   rr1(R1, R2,normal),qrs(R3,pvc, R2),rr1(R2, R3,short).
class(bigeminy):-%[[5,0,0,0,0,0],[15,20,25,20,13,11]]
   qrs(R0,basic, _),qrs(R1,pvc, R0),rr1(R0, R1,short),qrs(R2,basic, R1),
   rr1(R1, R2,short),qrs(R3,pvc, R2),rr1(R2, R3,short).

class(mobitzII):-%[[0,17,0,0,0,0],[20,3,25,20,13,11]]
   qrs(R0,nonbasic, _),qrs(R1,nonbasic, R0),rr1(R0, R1,long),
   qrs(R2,nonbasic, R1),rr1(R1, R2,long),qrs(R3,nonbasic, R2),
   rr1(R2, R3,long).
class(mobitzII):-%[[0,3,0,0,0,0],[20,17,25,20,13,11]]
   qrs(R0,basic, _),qrs(R1,basic, R0),rr1(R0, R1,long),
   qrs(R2,basic, R1),rr1(R1, R2,long),qrs(R3,basic, R2),rr1(R2, R3,long).

class(normal):-%[[0,0,24,6,0,0],[20,20,1,14,13,11]]
   qrs(R0,basic, _),qrs(R1,basic, R0),rr1(R0, R1,normal),qrs(R2,basic, R1),
   rr1(R1, R2,normal),qrs(R3,basic, R2),rr1(R2, R3,normal),qrs(R4,basic, R3),
   rr1(R3, R4,normal),qrs(R5,basic, R4),rr1(R4, R5,normal).

class(pvc):-%[[0,0,0,18,0,0],[20,20,25,2,13,11]]
   qrs(R0,basic, _),qrs(R1,basic, R0),rr1(R0, R1,normal),qrs(R2,pvc, R1),
   rr1(R1, R2,short),qrs(R3,basic, R2),rr1(R2, R3,normal),qrs(R4,basic, R3),
   rr1(R3, R4,normal).
class(pvc):-%[[0,0,0,2,0,0],[20,20,25,18,13,11]]
   qrs(R0,basic, _),qrs(R1,basic, R0),rr1(R0, R1,normal),qrs(R2,basic, R1),
   rr1(R1, R2,normal),qrs(R3,pvc, R2),rr1(R2, R3,normal).

class(doublet):-%[[0,0,0,0,7,0],[20,20,25,20,6,11]]
   qrs(R0,basic, _),qrs(R1,pvc, R0),rr1(R0, R1,short),qrs(R2,pvc, R1),rr1(R1, R2,normal),
   qrs(R3,basic, R2),rr1(R2, R3,normal).
class(doublet):-%[[0,0,0,0,4,0],[20,20,25,20,9,11]]
   qrs(R0,basic, _),qrs(R1,basic, R0),rr1(R0, R1,normal),qrs(R2,pvc, R1),rr1(R1, R2,short),
   qrs(R3,pvc, R2),rr1(R2, R3,short).
class(doublet):-%[[0,0,0,0,4,0],[20,20,25,20,9,11]]
   qrs(R0,basic, _),qrs(R1,pvc, R0),rr1(R0, R1,short),qrs(R2,pvc, R1),rr1(R1, R2,short),
   qrs(R3,basic, R2),rr1(R2, R3,normal).
class(doublet):-%[[0,0,0,0,1,0],[20,20,25,20,12,11]]
   qrs(R0,basic, _),qrs(R1,pvc, R0),rr1(R0, R1,normal),qrs(R2,pvc, R1),rr1(R1, R2,short),
   qrs(R3,basic, R2),rr1(R2, R3,normal).
class(vt):-%[[0,0,0,0,0,9],[20,20,25,20,13,2]]
   qrs(R0,pvc, _),qrs(R1,pvc, R0),rr1(R0, R1,short),qrs(R2,pvc, R1),rr1(R1, R2,short),
   qrs(R3,pvc,R2),rr1(R2, R3,short).
class(vt):-%[[0,0,0,0,0,5],[20,20,25,20,13,6]]
   qrs(R0,pvc, _),qrs(R1,pvc, R0),rr1(R0, R1,normal),
   qrs(R2,pvc, R1),rr1(R1, R2,short),qrs(R3,pvc,R2),rr1(R2, R3,short).
```

FIG. B.3 – Ensemble complet de règles pour exper3.

B.4 Exper4

```
class(bigeminy):-%[[15,0,0,0,0,0],[5,20,25,20,13,11]]
   p_wav(P0, _),qrs(R0, P0),qrs(R1, R0),p_wav(P2, R1),qrs(R2, P2),qrs(R3, R2).
class(bigeminy):-%[[6,0,0,0,0,0],[14,20,25,20,13,11]]
   qrs(R0, _),p_wav(P1, R0),qrs(R1, P1),rr1(R0, R1,short),p_wav(P2, R1),
   qrs(R2, P2), p_wav(P3, R2),qrs(R3, P3).

class(mobitzII):-%[[0,20,0,0,0,0],[20,0,25,20,13,11]]
   p_wav(P0, _),qrs(R0, P0),p_wav(P1, R0),equal(P1, R1),p_wav(P2, R1),
   equal(P2, R2), qrs(R3, R2).

class(normal):-%[[0,0,24,5,0,0],[20,20,1,15,13,11]]
   p_wav(P0, _),qrs(R0, P0),p_wav(P1, R0),qrs(R1, P1),pr1(P1, R1,normal),
   p_wav(P2, R1), qrs(R2, P2),pr1(P2, R2,normal),p_wav(P3, R2),qrs(R3, P3),
   rr1(R2, R3,normal), pr1(P3, R3,normal),p_wav(P4, R3),qrs(R4, P4),
   rr1(R3, R4,normal),p_wav(P5, R4),qrs(R5, P5).

class(pvc):-%[[0,0,0,20,0,0],[20,20,25,0,13,11]]
   p_wav(P0, _),qrs(R0, P0),p_wav(P1, R0),qrs(R1, P1),qrs(R2, R1),
   p_wav(P3, R2), qrs(R3, P3),p_wav(P4, R3),qrs(R4, P4).

class(doublet):-%[[0,0,0,0,13,0],[20,20,25,20,0,11]]
   qrs(R0, _),qrs(R1, R0),qrs(R2, R1),p_wav(P3, R2),qrs(R3, P3).

class(vt):-%[[0,0,0,0,0,11],[20,20,25,20,13,0]]
   qrs(R0, _),qrs(R1, R0),qrs(R2, R1),qrs(R3, R2).
```

FIG. B.4 – Ensemble complet de règles pour exper4.

B.5 Exper5

```
class(bigeminy):-%[[15,0,0,0,0,0],[5,20,25,20,13,11]]
   p_wav(P0,normal, _),qrs(R0,basic, P0),qrs(R1,nonbasic, R0),p_wav(P2,normal, R1),
   qrs(R2,basic, P2),qrs(R3,nonbasic, R2).
class(bigeminy):-%[[5,0,0,0,0,0],[15,20,25,20,13,11]]
   p_wav(P0,normal, _),qrs(R0,basic, P0),p_wav(P1,normal, R0),qrs(R1,nonbasic, P1),
   p_wav(P2,normal, R1),qrs(R2,basic, P2),p_wav(P3,normal, R2),qrs(R3,nonbasic, P3).

class(mobitzII):-%[[0,17,0,0,0,0],[20,3,25,20,13,11]]
   p_wav(P0,normal, _),qrs(R0,nonbasic, P0),p_wav(P1,normal, R0),equal(P1, R1),
   p_wav(P2,normal, R1),equal(P2, R2),qrs(R3,nonbasic, R2).
class(mobitzII):-%[[0,3,0,0,0,0],[20,17,25,20,13,11]]
   p_wav(P0,normal, _),qrs(R0,basic, P0),p_wav(P1,normal, R0),equal(P1, R1),
   p_wav(P2,normal, R1),equal(P2, R2),qrs(R3,basic, R2).

class(normal):-%[[0,0,24,5,0,0],[20,20,1,15,13,11]]
   p_wav(P0,normal, _),qrs(R0,basic, P0),p_wav(P1,normal, R0),qrs(R1,basic, P1),
   p_wav(P2,normal,R1),qrs(R2,basic, P2),p_wav(P3,normal, R2),qrs(R3,basic, P3),
   rr1(R2, R3,normal),p_wav(P4,normal, R3),qrs(R4,basic, P4),rr1(R3, R4,normal),
   pr1(P4, R4,normal),p_wav(P5,normal, R4),qrs(R5,basic, P5).

class(pvc):-%[[0,0,0,20,0,0],[20,20,25,0,13,11]]
   qrs(R0,basic, _),p_wav(P1,normal, R0),qrs(R1,basic, P1),qrs(R2,nonbasic, R1),
   p_wav(P3,normal, R2),qrs(R3,basic, P3),p_wav(P4,normal, R3),qrs(R4,basic, P4).

class(doublet):-%[[0,0,0,0,13,0],[20,20,25,20,0,11]]
   qrs(R0,basic,_),qrs(R1,nonbasic,R0),qrs(R2,nonbasic,R1),p_wav(P3,normal,R2),
   qrs(R3,basic,P3).

class(vt):-%[[0,0,0,0,0,11],[20,20,25,20,13,0]]
   qrs(R0,nonbasic,_),qrs(R1,nonbasic,R0),qrs(R2,nonbasic,R1),qrs(R3,nonbasic,R2).
```

FIG. B.5 – Ensemble complet de règles pour exper5.

B.6 Exper6

```
class(bigeminy):-%[[15,0,0,0,0,0],[5,20,25,20,13,11]]
   p_wav(P0,normal, _),qrs(R0,basic, P0),qrs(R1,pvc, R0),p_wav(P2,normal, R1),
   qrs(R2,basic, P2),qrs(R3,pvc, R2).
class(bigeminy):-%[[5,0,0,0,0,0],[15,20,25,20,13,11]]
   p_wav(P0,normal, _),qrs(R0,basic, P0),p_wav(P1,normal, R0),qrs(R1,pvc, P1),
   p_wav(P2,normal,R1),qrs(R2,basic, P2),p_wav(P3,normal, R2),qrs(R3,pvc, P3).

class(mobitzII):-%[[0,17,0,0,0,0],[20,3,25,20,13,11]]
   p_wav(P0,normal, _),qrs(R0,nonbasic, P0),p_wav(P1,normal, R0),equal(P1, R1),
   p_wav(P2,normal,R1),equal(P2, R2),qrs(R3,nonbasic, R2).
class(mobitzII):-%[[0,3,0,0,0,0],[20,17,25,20,13,11]]
   p_wav(P0,normal, _),qrs(R0,basic, P0),p_wav(P1,normal, R0),equal(P1, R1),
   p_wav(P2,normal, R1),equal(P2, R2),qrs(R3,basic, R2).

class(normal):-%[[0,0,24,5,0,0],[20,20,1,15,13,11]]
   p_wav(P0,normal, _),qrs(R0,basic, P0),p_wav(P1,normal, R0),qrs(R1,basic, P1),
   p_wav(P2,normal,R1),qrs(R2,basic, P2),p_wav(P3,normal, R2),qrs(R3,basic, P3),
   rr1(R2, R3,normal),p_wav(P4,normal, R3),qrs(R4,basic, P4),rr1(R3, R4,normal),
   pr1(P4, R4,normal),p_wav(P5,normal, R4),qrs(R5,basic, P5).

class(pvc):-%[[0,0,0,20,0,0],[20,20,25,0,13,11]]
   qrs(R0,basic, _),p_wav(P1,normal, R0),qrs(R1,basic, P1),qrs(R2,pvc, R1),
   p_wav(P3,normal, R2),qrs(R3,basic, P3),p_wav(P4,normal, R3),qrs(R4,basic, P4).

class(doublet):-%[[0,0,0,0,13,0],[20,20,25,20,0,11]]
   qrs(R0,basic, _),qrs(R1,pvc, R0),qrs(R2,pvc, R1),p_wav(P3,normal, R2),
   qrs(R3,basic, P3).
class(vt):-%[[0,0,0,0,0,11],[20,20,25,20,13,0]]
   qrs(R0,pvc, _),qrs(R1,pvc, R0),qrs(R2,pvc, R1),qrs(R3,pvc, R2).
```

FIG. B.6 – Ensemble complet de règles pour exper6.

Annexe C

Extension de IP-Calicot au multisource

C.1 Introduction

L'une des évolutions de Calicot est l'extension du monitorage à plusieurs sources de signaux physiologiques. Le passage au multisource permet d'utiliser les informations redondantes pour proposer un diagnostic médical robuste au bruit (Chambrin, 2001). Dans le cas où les données sont redondantes et bruitées, il est nécessaire de choisir la source la plus porteuse d'information pour le diagnostic médicale (c.-à-d. la reconnaissance d'arythmies). Cette sélection peut être faite avec les informations d'un analyseur de contexte de ligne. Dans le cas d'informations complémentaires, l'utilisation conjointe de plusieurs sources permet un diagnostic des arythmies plus fiable et plus précis. Le diagnostic doit alors se baser sur des connaissances décrites dans un langage prenant en compte toutes les sources à la fois.

La présence de bruit, en variant d'une source à l'autre, interdit l'utilisation d'un système statique utilisant toutes les sources simultanément. Des méthodes telles que (Thoraval et coll., 1997; Hernández et coll., 1999) permettent de tirer parti de la redondance des sources pour améliorer les détections des événements caractéristiques sur chacune des sources, mais elles nécessitent des connaissances préalables sur les liens inter-sources entre ces événements. Les sources utiles peuvent également être sélectionnées en fonction d'une détection de bruit sur le signal, du contexte médical du patient (Dojat et coll., 1997), des contraintes temps réel et des ressources de calcul (Larsson et Hayes-Roth, 1998) ou de la complémentarité des sources pour détecter certains événements.

En ligne, l'utilisation de plusieurs sources nécessite une réactivité et une adaptabilité très importante du système. Ces qualités peuvent être apportées par le module de pilotage.

C.2 Données multisources

Deux types de source sont pris en compte : l'électrocardiogramme multivoie et la mesure de la pression artérielle. Cette sélection est particulièrement adaptée au contexte USIC car ces unités de soins possèdent les appareils de mesure capables de recueillir au moins deux voies d'ECG et une voie de pression artérielle (Beaufils et coll., 1999). La figure C.1 présente les signaux utilisés.

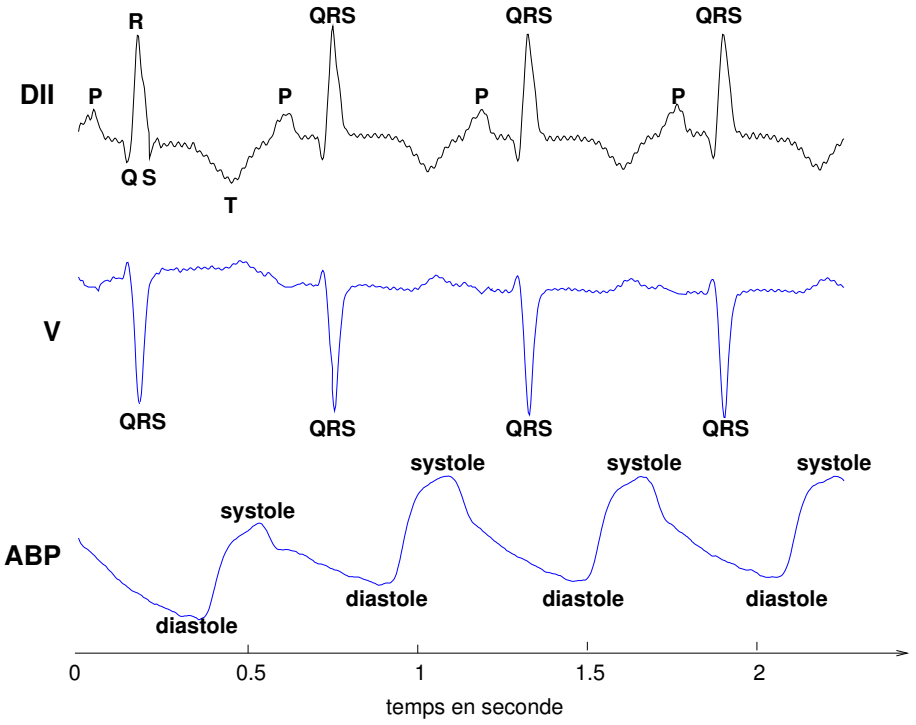


Fig. C.1 – Exemple de signaux étudiés : voies DII et V d'un ECG, voie de pression (ABP)

L'activité cardiaque se reflète non seulement sur l'ECG mais aussi sur la courbe de pression artérielle. L'ABP permet de détecter les instants d'occurrence de la *systole ventriculaire* et de la *diastole ventriculaire*.

C.3 Pilotage multisource

Grâce à l'analyse du bruit sur les sources, le pilote sélectionne la ou les sources intéressantes et les chroniques correspondantes. À chaque ensemble de sources correspond un langage propre permettant de décrire les événements caractéristiques se produisant sur les sources et leurs relations. La figure C.2 présente les différents langages de description possibles.

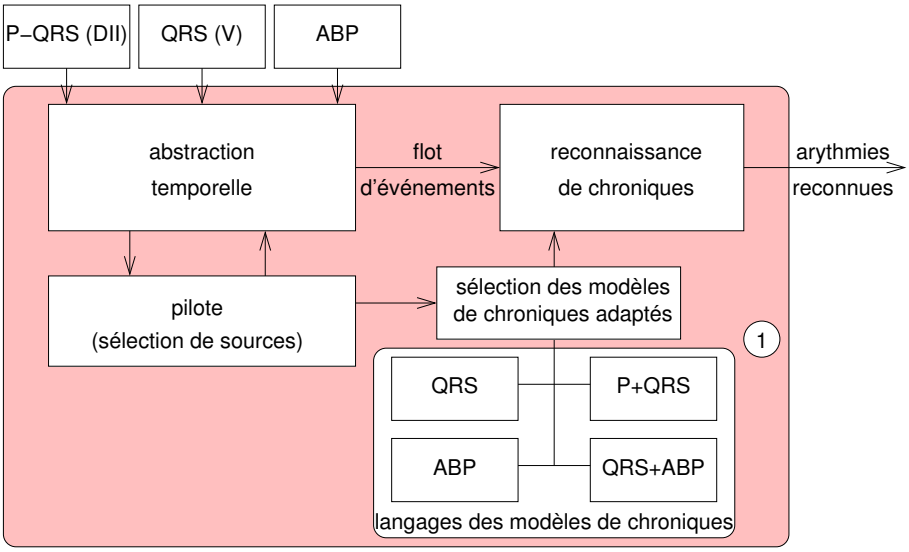


FIG. C.2 – Principe du pilotage des sources

Une fois les sources choisies, le pilote sélectionne les modèles de chroniques dont le langage de description est adapté au langage des événements fournis par l'abstraction temporelle. Par exemple, si la voie DII est bruitée, il n'est pas possible de détecter l'onde P. Dans ce cas, si les voies V et ABP ne sont pas bruitées, le pilote va permettre au module d'abstraction temporelle de traiter seulement ces deux sources et sélectionner les modèles de chroniques dont le langage est spécifique aux sources choisies. Dans l'exemple, les modèles de chroniques choisis ont été appris sans prendre en compte l'onde P. Le langage choisi était donc *QRS+ABP*.

Pour traiter les signaux de pression, de nouveaux algorithmes de détection des diastoles et des systoles doivent être inclus dans la base d'algorithmes. Il existe dans la littérature, plusieurs algorithmes pour effectuer cette tâche, comme celui de Hoeksel et coll. (Hoeksel et coll., 1997) à base de filtres adaptés pour les valeurs de pression systolique et diastolique. Cet algorithme offre de bonnes performances mais n'a pas encore été inclus dans le système. Pour l'étude, le détecteur d'événements diastoliques et systoliques a été simulé à partir des annotations du signal fournies par la base de données.

C.4 Acquisition de la base de chroniques multisources

La base de chroniques apprise avec différents langages de description provient des travaux de Fromont et coll. (Fromont et coll., 2005; Fromont, 2005). Elle contient : des modèles de chroniques monosources pour l'ECG complet (avec ondes P et QRS), pour la voie V (sans onde P et sans la forme du complexe QRS), et pour la source ABP et des modèles de chroniques multisources combinant les événements des différentes sources.

Ces différents ensembles de modèles de chroniques sont représentés au point (1) de la figure C.2.

Les signaux utilisés pour l'apprentissage proviennent de la base de données MIMIC (*Multi-parameter Intelligent Monitorage for Intensive Care*). Les arythmies et les événements P-QRS-Diastole-Systole des exemples ont tous été annotés par des experts. L'apprentissage a été effectué sur cinquante exemples décrivant six arythmies différentes : le bigéminisme (`bigeminy`), le doublet ventriculaire (`doublet`), l'extrasystole ventriculaire (`pvc`), la tachycardie ventriculaire (`vt`), la fibrillation auriculaire (`af`) et le rythme sinusal normal (`normal`) qui permet de différencier une arythmie d'un rythme non pathologique.

C.5 Résultats préliminaires

Deux façons possibles de tirer parti de l'utilisation de plusieurs sources sont présentées. Dans le cas de sources bruitées, le pilotage choisit là ou les sources dont l'information est la plus satisfaisante. Si les sources ne sont pas bruitées, deux cas de figure sont pris en compte.

- si les sources sont complémentaires, les chroniques qui incluent des événements provenant des différentes sources combinées sont utilisées pour augmenter la fiabilité du diagnostic.
- si les sources sont redondantes, la source qui apporte le plus d'informations est retenue.

Les résultats des reconnaissances des arythmies sont donnés dans des matrices de confusion. La méthode de calcul de ces matrices diffère de celle présentée précédemment. Le principe de construction de ces matrices est donné dans (Carrault et coll., 2003). Pour chaque expérience, la probabilité de reconnaissance (Pr) et le taux de fausses alarmes (Tfa) du système de monitorage sont évalués.

C.5.1 Monitorage en milieu bruité

Plusieurs expériences ont été faites en bruitant successivement les différentes sources de données pour chacun des cas suivants : systèmes de monitorage monosource et multisource non pilotés et système de monitorage multisource avec pilotage. Deux ECG provenant de la base MGH/MF (*Massachusetts General Hospital/Marquette Foundation*) ont été utilisés pour composer un ECG de test. Celui-ci a ensuite été bruité avec les bruits cliniques réels, présentés au chapitre 4. Pour le signal de pression, le bruit a été modélisé par une perte totale de ligne (ce cas extrême n'est pas rare en milieu clinique). Un exemple de lignes bruitées est donné Figure C.3. Sur le premier signal, les artefacts rendent l'onde P très difficile à détecter sans erreurs. Le second signal est propre. Le dernier signal montre une perte de ligne sur la source ABP.

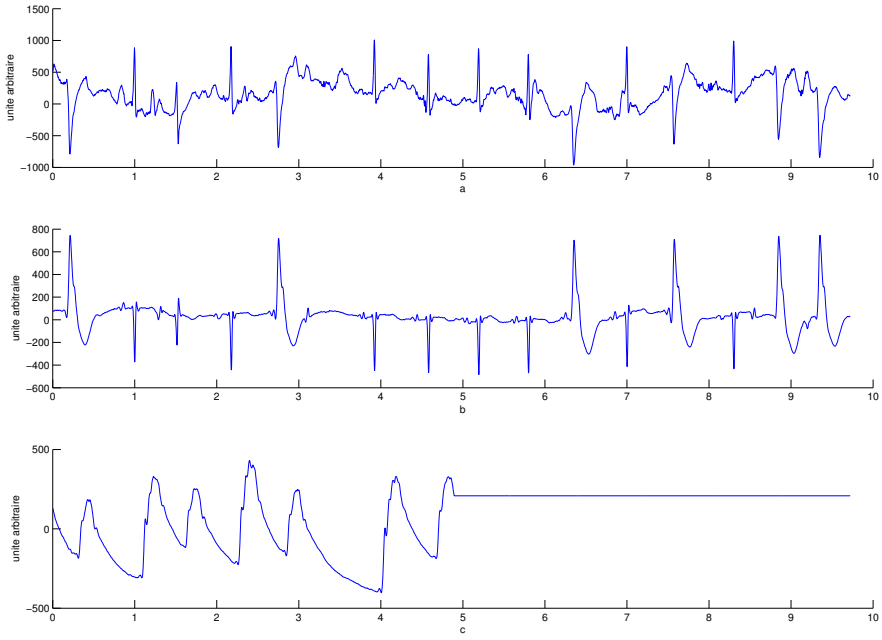


FIG. C.3 – Exemple de bruits sur les sources de données

C.5.2 Reconnaissance d'arythmies sans pilotage

Les résultats de l'apprentissage sur des sources uniques bruitées sans pilotage sont présentés dans les tables C.1, C.2, C.3 et C.4.

La table C.1 montre des résultats de précision correcte pour les arythmies tachycardies ventriculaires (`vt`) et fibrillation auriculaire (`af`) sur un ECG bruité mais le taux de fausses alarmes est élevé (0.31). La reconnaissance des bigéminismes (`bigeminy`) et des doublets ventriculaires (`doublet`) est meilleure que sur les autres voies. Par contre, aucun rythme normal et aucune extrasystole ventriculaire n'a été reconnu sur cette voie. Ceci s'explique par le fait que les chroniques décrivant ces arythmies occupent une fenêtre temporelle beaucoup plus large que celle des chroniques décrivant les autres arythmies. Ces arythmies sont caractérisées par un seul ou aucun (dans le cas `normal`) battement anormal et sont beaucoup plus difficiles à discriminer que les arythmies caractérisées par plusieurs battements anormaux. Il y a donc très peu de chance que la chronique soit reconnue dans sa totalité avant que le signal soit à nouveau bruité.

La matrice de confusion présentée Table C.2 montre que la voie V offre des performances moyennes pour les arythmies `normal, pvc, doublet` mais un taux de fausses alarmes très faible pour le `normal` en particulier (0.03). Cette source est moins sensible au bruit que la voie DII puisqu'elle ne prend pas en compte les mauvaises détections

	normal	pvc	bigeminy	doublet	vt	af	Total
normal	0	0	0	0	0	0	0
pvc	0	0	0	0	0	0	0
bigeminy	75	16	55	0	0	0	146
doublet	46	2	3	26	0	0	77
vt	48	5	15	26	270	25	389
af	123	4	29	5	20	412	593
NR	564	20	87	26	13	64	774
Total	856	47	189	83	303	501	1979
Pr	0	0	0.29	0.31	0.89	0.82	
Tfa	0	0	0.62	0.66	0.31	0.31	

Tab. C.1 – Matrice de confusion à partir de l'ECG bruité seul

	normal	pvc	bigeminy	doublet	vt	af	Total
normal	348	0	9	0	0	0	357
pvc	0	22	23	0	1	1	47
bigeminy	0	0	0	0	0	0	0
doublet	44	12	16	18	1	4	95
vt	21	2	0	7	296	485	811
af	35	5	5	16	4	0	65
NR	405	6	133	42	1	11	598
Total	853	47	186	83	303	501	1973
Pr	0.41	0.47	0	0.22	0.98	0	
Tfa	0.03	0.53	0	0.81	0.64	1	

Tab. C.2 – Matrice de confusion pour la voie V bruitée seule

	normal	pvc	bigeminy	doublet	vt	af	Total
normal	436	0	14	5	0	4	459
pvc	72	19	53	5	0	1	150
bigeminy	0	0	0	0	0	0	0
doublet	7	0	1	6	0	18	32
vt	14	1	1	8	178	73	275
af	2	0	5	1	0	115	123
NR	326	27	115	58	125	291	942
Total	857	47	189	83	303	502	1981
Pr	0.51	0.40	0	0.07	0.59	0.23	
Tfa	0.05	0.87	0	0.81	0.35	0.07	

TAB. C.3 – Matrice de confusion pour la mesure de pression bruitée (avec la systole seule)

de l'onde P et les mauvaises classifications de la forme du QRS. Ce constat explique la meilleure probabilité de reconnaissance obtenue pour les arythmies ayant des fenêtres temporelles larges comme le rythme normal ou les extrasystoles. Aucune chronique n'a été détectée pour `bigeminy` car la reconnaissance de cette arythmie repose, pour la voie V, uniquement sur les intervalles de temps entre les QRS. Il semble que les seuils utilisés pour l'apprentissage soient trop spécifiques aux exemples sur lesquels ils ont été appris. Notons que l'on ne peut pas détecter une fibrillation auriculaire (`af`) avec des informations sur le QRS seul puisque ce qui différencie cette arythmie des autres arythmies est justement l'absence de l'onde P.

La matrice de confusion correspondant à la pression avec systole seule (cf. Table C.3) donne les meilleurs résultats des trois sources testées seules pour le rythme normal. Les résultats pour `bigeminy, doublet` et `af` sont en revanche très faibles. Ceci est dû au grand nombre de contraintes temporelles entre les événements caractéristiques de ces arythmies. Les seuils choisis pour passer de la représentation numérique des données à la représentation symbolique fournie par le module d'abstraction temporelle sont spécifiques à chaque patient. Ils ont été choisis à partir de la référence du rythme normal et ne sont donc probablement pas assez adaptés aux autres arythmies. Ce problème ne se pose pas pour l'électrocardiogramme pour lequel des seuils standards sont disponibles dans les ouvrages médicaux.

La table C.4 correspond à la matrice de confusion pour la fusion de deux sources : la voie V de l'électrocardiogramme et la voie de pression. Les résultats sont globalement meilleurs que pour les deux sources prises séparément. Ces meilleurs résultats tendent à montrer que ces deux sources sont complémentaires puisqu'une utilisation conjointe des informations permet d'augmenter la probabilité de reconnaissance.

C.5.3 Utilisation du pilotage

Dans cette expérience, le module de pilotage multisource complet est utilisé. Les résultats sont donnés Table C.5. En plus de sélectionner là ou les sources les plus

	normal	pvc	bigeminy	doublet	vt	af	Total
normal	427	0	2	1	1	6	437
pvc	99	24	29	27	8	25	212
bigeminy	22	0	38	12	0	0	72
doublet	0	0	0	0	0	6	6
vt	8	0	2	9	188	81	288
af	8	0	0	1	0	215	224
NR	292	23	117	33	106	169	740
Total	856	47	188	83	303	502	1979
Pr	0.50	0.51	0.20	0	0.62	0.43	
Tfa+	0.02	0.89	0.47	1	0.35	0.04	

TAB. C.4 – Matrice de confusion pour la fusion de la voie V et de la pression

	normal	pvc	bigeminy	doublet	vt	af	Total
normal	498	0	30	2	0	0	530
pvc	25	25	25	3	0	1	79
bigeminy	24	5	23	4	0	0	56
doublet	28	1	8	33	2	5	77
vt	14	0	1	9	278	289	591
af	26	0	23	11	3	142	205
NR	241	16	79	21	20	65	442
Total	856	47	189	83	303	502	1980
Pr	0.58	0.53	0.12	0.40	0.92	0.28	
Tfa	0.06	0.68	0.59	0.57	0.53	0.31	

TAB. C.5 – Matrice de confusion la reconnaissance utilisant toutes les sources avec pilotage

intéressantes, le module de pilotage sélectionne également la chronique la plus adaptée au langage de description des événements fourni par le module d'abstraction temporelle.

Les résultats du pilotage sont, en moyenne, meilleurs que sur chacune des autres sources prise séparément. La probabilité de reconnaissance est meilleure pour les classes `pvc`, `doublet` et `normal` en utilisant le pilotage que pour toutes les autres expériences. Cette probabilité est moins bonne que pour la voie DII seule pour `bigeminy` et `af` mais le taux de fausses alarmes est égal ou meilleur. La probabilité de reconnaissance pour la classe `af` et la classe `bigeminy` pâtie des très mauvais résultats de la pression et de la voie V seule. En effet, le pilote choisit la ou les sources les moins bruitées même si celles-ci ne sont pas les plus adaptées pour reconnaître un type d'arythmie particulier. Cependant, une liste des chroniques en cours de reconnaissances à un instant donné pourrait servir dans de futures expériences de pilotage à guider le choix de la source quand toutes les sources ne sont pas bruitées. Le taux de fausses alarmes pour `vt` a également augmenté par rapport à la voie DII seule mais les confusions sont principalement faites avec la classe `af` (289 arythmies reconnues comme `vt` confondues avec `af`). Ces confusions ne sont pas les plus graves puisqu'elles ont tout de même une signification médicale (au contraire de confusion avec le rythme normal).

C.6 Conclusion

Les résultats préliminaires montrent, pour l'instant, un taux de reconnaissance moyen pour la plupart des expériences. En effet, dans un contexte bruité, la capacité des algorithmes de l'abstraction temporelle à extraire de l'information des sources est très limitée et les algorithmes n'ont pas été optimisés pour les types de signaux utilisés. Cependant, le pilotage par le contexte signal entraîne une amélioration de la sensibilité en moyenne et réduit nettement le nombre de fausses alarmes par rapport à un système de monitorage fonctionnant sans pilotage.

Ces améliorations pourraient être accentuées en permettant au pilote de prendre en compte la liste des arythmies en cours de reconnaissance . Le pilote pourrait ainsi choisir la source la plus appropriée à la reconnaissance d'une arythmie, en particulier lorsque plusieurs sources sont non bruitées. Une seconde amélioration possible serait de fixer les seuils (spécifiques au patient) des intervalles relatifs au événements apparaissant sur la pression par une phase d'initialisation du système de monitorage. Cette phase consisterait à apprendre automatiquement les seuils spécifiques à chaque patient sur une courte période lorsque le patient à un rythme normal avant de pouvoir utiliser le système de monitorage dans sa globalité.

Bibliographie

Aaron, J., Carlisle, C., Carskadon, M., Meyer, T., Hill, N. S. et Millman, R. (1996). Environmental noise as a cause of sleep disruption in an intermediate respiratory care unit. *Sleep*, 19(9):707–710.

Adamec, J. et Adamec, R. (2000). *ECG Holter, manuel d'interprétation électrocardiographique*. Editions Médecine & Hygiène.

Afonso, V., Tompkins, W., Nguyen, T. et Shen, L. (1999). ECG beat detection using filter banks. *IEEE Transactions on Biomedical Engineering*, 46:192–202.

Agilent Technologies (2000). *ST/AR Arrhythmia Algorithm*. Agilent Technologies.

Anliker, U., Ward, J. A., Lukowicz, P., Tröster, G., Dolveck, F., Baer, M., Keita, F., Schenker, E. B., Catarsi, F., Coluccini, L., Belardinelli, A., Shklarski, D., Alon, M., Hirt, E., Schmid, R. et Vuskovic, M. (2004). AMON : a wearable multiparameter medical monitoring and alert system. *IEEE Transaction on Information Technology in Biomedicine*, 8(4):415–427.

Beaufils, P. et coll. (1999). Recommandations de la société francaise de cardiologie pour la prise en charge des urgences cardiologiques. *Archives des maladies du cœur et des vaisseaux*, 92(3):337–344.

Bellazzi, R., Siviero, C., Stefanelli, M. et Nicolao, G. D. (1995). Adaptive controllers for intelligent monitoring. *Artificial Intelligence in Medicine*, 7:515–540.

Benitez, D., Gaydecki, P., Zaidi, A. et Fitzpatrick, A. (2001). The use of the Hilbert transform in ECG signal analysis. *Computers in Biology and Medecine*, 31:399–406.

Bertier, P. et Bouroche, J. (1977). *L'analyse en composantes principales*, pages 109–119. Presses Universitaires de France.

Biot, L., Holzapfel, L., Becq, G., Melot, C. et Baconnier, P. (2003). Do we need a systematic activation of alarm soundings for blood pressure monitoring for the safety of ICU patients ? *Journal of Critical Care*, 18(4):212–216.

Blitt, C. (1990). *Monitoring in the Critical Care Unit*, chapitre A philosophy of monitoring, pages 3–8. Churchill Livingstone, New York, 2^e édition.

Blondeau, M. et Hiltgen, M. (1980). *Électrocardiographie clinique*. Masson.

Bratko, I., Mozetič, I. et Lavrač, N. (1989). *KARDIO : a study in deep and qualitative knowledge for expert systems*. MIT Press, Cambridge, MA.

Burke, M. et Nasor, M. (2004). Wavelet based analysis and characterization of the ECG signal. *Journal of Medical Engineering & Technology*, 28(2):47–55.

Calvelo, D., Chambrin, M.-C., Pomorski, D. et Ravaux, P. (2000). Towards symbolization using data-driven extraction of local trends for ICU monitoring. *Artifical Intelligence in Medicine*, 19:203–223.

Carrault, G., Cordier, M., Quiniou, R., Garreau, M., Bellanger, J.-J. et Bardou, A. (1999). A model-based approach for learning to identify cardiac arrhythmias. Dans *Artificial Intelligence in Medicine and Medical Decision Making (AIMDM'1999)*, volume 1620, pages 165–174, Springer-Verlag, Aalborg, Denmark.

Carrault, G., Cordier, M., Quiniou, R. et Wang, F. (2003). Temporal abstraction and inductive logic programming for arrhyhtmia recognition from electrocardiograms. *Artificial Intelligence in Medicine*, 28:231–263.

Carver, N. et Lesser, V. (1991). A new framework for sensor interpretation : planning to resolve source of uncertainty. Dans *9th National Conference on Artificial Intelligence (AAAI-91)*, pages 724–731, Anaheim, California.

Cauvin, S., Cordier, M.-O., Dousson, C., Laborie, P., Lévy, F., Montmain, J., Porcheron, M., Servet, I. et Travé-Massuyès, L. (1998). The ALARM research group, monitoring and alarm interpretation in industrial environments. *AI Communications*, 11(3-4):139–173.

Chambrin, M. (2001). Alarms in the intensive care unit : how can the number of false alarms be reduced ? *Critical Care*, 5(4):184–188.

Christov, I. (2004). Real time electrocardiogram QRS detection using combined adaptive threshold. *Biomedical Engineering Online*, 3(28):1–9.

Clouard, R., Elmoataz, A. et Revenu., M. (1998). Une modélisation explicite et opérationnelle de la connaissance de traitement d'images. Dans *11^e édition du congrès Reconnaissance des Formes et Intelligence Artificielle (RFIA 1998)*, pages 65–74, Clermont-Ferrand.

Clément, V. (1990). *Raisonnements cognitifs appliqués au pilotage d'algorithmes de traitement d'images*. Thèse de doctorat, Université de Nice-Sophia Antipolis.

Clément, V. et Thonnat, M. (1993). A knowledge-based approach to integration of image procedures processing. *Computer Vision Graphic and Image Processing (CVGIP) : Image Understanding*, 57(2):166–184.

Coiera, E. (1993). Intelligent monitoring and control of dynamic physiological systems. *Artificial Intelligence in Medicine*, 5:1–8.

Cordier, M.-O. et Dousson, C. (2000). Alarm driven monitoring based on chronicles. Dans *SAFEPROCESS 2000*, pages 286–291, Budapest.

Cropp, A., Woods, L., Raney, D. et Bredle, D. (1994). Name that tone. the proliferation of alarms in the intensive care unit. *Chest*, 105(4):1217–1220.

Crubézy, M. (1999). *Pilotage de programmes pour le traitement d'images médicales*. Thèse de doctorat, Université de Nice Sophia-Antipolis.

Dalle, P. et Dejean, P. (1998). Planification en traitement d'images : Approche basée sur les données. Dans *11^e édition du congrès Reconnaissance des Formes et Intelligence Artificielle (RFIA 1998)*, pages 75–84, Clermont-Ferrand.

Daskalov, I. et Christov, I. (1999). Electrocardiogram signal preprocessing for automatic detection of QRS boundaries. *Medical Engineering & Physics*, 21:37–44.

Daubechies, I. (1988). Orthonormal bases of compactly supported wavelets. *Communications on Pure and Applied Mathematics*, XLI:909–996.

Dawant, B., Uckun, S., Manders, E. et Lindstrom, D. (1993). The SIMON project : model based signal acquisition, analysis and interpretation in intelligent patient monitoring. *IEEE Engineering in Medicine and Biology*, 12(4):82–91.

Dechter, R., Meiri, I. et Pearl, J. (1991). Temporal constraint networks. *Artificial Intelligence*, 49:61–95.

Dojat, M. et Chittaro, L. (1997). Using a general theory of time and change in patient monitoring : experiment and evaluation. *Computers in Biology and Medecine*, 27(5):435–452.

Dojat, M., Pachet, F., Guessoum, Z., Touchard, D., Harf, A. et Brochard, L. (1997). NéoGanesh : a working system for the automated control of assisted ventilation in ICUs. *Artificial Intelligence in Medicine*, 11:97–117.

Dojat, M. et Sayettat, C. (1996). A realistic model for temporal reasoning in real-time patient monitoring. *Applied Artificial Intelligence*, 10:121–143.

Dotsinsky, I. et Stoyanov, T. (2004). Ventricular beat detection in single channel electrocardiograms. *Biomedical Engineering Online*, 3(1).

Dousson, C. (1994). *Suivi d'évolutions et reconnaissance de chroniques.* Thèse de doctorat, Université Paul Sabatier.

Duchêne, F., Garbay, C. et Rialle, V. (2005). Apprentissage non supervisé de motifs temporels, multidimensionnels et hétérogènes. application à la télésurveillance médicale. Dans *7e conférence francophone sur l'apprentissage automatique (CAp 2005)*, pages 181–182, Nice.

Einthoven, W. (1906). Le télécardiogramme. *Archives Internationales de Physiologie*, 4:132–164.

Fernández, J., Harris, M. et Meyer, C. (2005). Combining algorithms in automatic detection of R-peaks in ECG signals. Dans *18th IEEE Symposium on Computer-Based Medical Systems (CBMS2005)*, pages 297–302, Dublin.

Fraden, J. et Neuman, M. (1980). QRS wave detection. *Medical & Biological Engineering & Computing*, 18:125–132.

Friesen, G. M., Jannett, T. C., Jadallah, M. A., Yates, S. L., Quint, S. R. et Nagle, H. T. (1990). A comparison of the noise sensitivity of nine QRS detection algorithms. *IEEE Transactions on Biomedical Engineering*, 37:85–98.

FROMONT, E. (2005). *Apprentissage multisource par programmation logique inductive : application à la caractérisation d'arythmies cardiaques*. Thèse, Université de Rennes 1.

FROMONT, E., CORDIER, M.-O., QUINIOU, R. et HERNÁNDEZ, A. (2003). KARDIO and CALICOT : a comparison of two cardiac arrhythmia classifiers. Dans *9th Conference on Artificial Intelligence in Medicine (AIME 2003) Workshop : Qualitative and Model-based Reasoning in Biomedicine*, Protaras.

FROMONT, E. et PORTET, F. (2005). Pilotage d'un système de monitoring cardiaque multisource. Dans 3^e *MAnifestation des JEunes Cherchers du domaine des Sciences et Technologies de l'Information et de la Communication (MAJECSTIC 2005)*, Rennes.

FROMONT, E., QUINIOU, R. et CORDIER, M.-O. (2005). Learning rules from multisource data for cardiac monitoring. Dans *10th Conference on Artificial Intelligence in Medicine (AIME 2005)*, pages 484–493, Aberdeen.

FUCHSBERGER, C., HUNTER, J. et MCCUE, P. (2005). Testing Asbru guidelines and protocol for neonatal intensive care. Dans *10th Conference on Artificial Intelligence in Medicine (AIME 2005)*, pages 101–110, Aberdeen.

GAMPER, J. et NEJDL, W. (1997). Abstract temporal diagnosis in medical domains. *Artificial Intelligence in Medicine*, 10:209–234.

GERAPETRITIS, H. et PELISSIER, J. M. (2004). The critical success index and warning strategy. Dans *17th Conference on Probablity and Statistics in the Atmospheric Sciences*, Seattle.

GOLDBERGER, E. (1942). A simple electrocardiographic electrode of zero potential and a technic of obtaining augmented unipolar extremity leads. *American Heart Journal*, 23:483–492.

GRITZALI, F. (1988). Towards a generalized scheme for QRS detection in ECG waveforms. *Signal Processing*, 15:183–192.

GUERTIN, M. (1996). Abductive inference of events : diagnosing cardiac arrhythmias. Dans *9th Florida Artificial Intelligence Research Symposium*, pages 106–111.

GUYET, T., GARBAY, C. et DOJAT, M. (2005). Human/computer interaction to learn scenarios from ICU multivariate time series. Dans *10th Conference on Artificial Intelligence in Medicine (AIME 2005)*, pages 424–428, Aberdeen.

HAIMOWITZ, I. et KOHANE, I. (1996). Managing temporal worlds for medical trend diagnosis. *Artificial Intelligence in Medicine*, 8:299–321.

HAYES-ROTH, B., PFELGER, K., LALANDA, P., MORIGNOT, P. et BALABANOVIC, M. (1995). A domain-specific software architecture for adaptative intelligent systems. *IEEE Transaction on Software Engineering*, 21(4):288–301.

HERNÁNDEZ, A. (2000). *Fusion de signaux et de modèles pour la caractérisation d'arythmies cardiaques*. Thèse de doctorat, Université de Rennes 1.

Hernández, A. et Carrault, G. (2004). Intégration modèle-observation pour l'interprétation des battements cardiaques. Dans *14^e édition du congrès Reconnaissance de Formes et Intelligence Artificielle (RFIA 2004)*, pages 447–456, Toulouse.

Hernández, A., Carrault, G., Mora, F., Thoraval, L., Passariello, G. et Schleich, J.-M. (1999). Multisensor fusion for atrial and ventricular activity detection in coronary care monitoring. *IEEE Transactions on Biomedical Engineering*, 46(10):1186–1190.

Hess, D. (1990). Noninvasive monitoring in respiratory care - present, past, and future : an overview. *Respiratory Care*, 35:482–498.

Hoeksel, S., Jansen, J., Blom, J. et Schreuder, J. (1997). Detection of dicrotic notch in arterial pressure signals. *Journal of Clinical Monitoring and Computing*, 13:309–316.

Holter, N. (1961). New method for heart studies. *Science*, 134:1214–1229.

Horn, W. (2001). AI in medicine on its way from knowledge-intensive to data intensive systems. *Artificial Intelligence in Medicine*, 23:5–12.

Hunter, J., Ewing, G., Freer, Y., Logie, R., McCue, P. et McIntosh, N. (2003). NEONATE : decision support in the neonatal intensive care unit - a preliminary report. Dans *9th Conference on Artificial Intelligence in Medicine (AIME 2003)*, pages 41–45, Protaras.

Hunter, J. et Kirby, I. (1995). Ticker : A qualitative model of the electrical system of the heart. Dans *Research and Development in Expert Systems XII*, pages 293–307.

Jenkins, J. et Caswell, S. (1996). Detection algorithms in implantable cardioverter defibrillators. *Proceedings of the IEEE*, 84(3):428–445.

Kadambe, S., Murray, R. et Boudreaux-Bartels, F. (1999). Wavelet transform-based QRS complex detector. *IEEE Transactions on Biomedical Engineering*, 47(7):838–848.

Karsai, G. et Sztipanovits, J. (1999). A model-based approach to self-adaptive software. *IEEE Intelligent Systems*, 14:46–53.

Keith, A. et Flack, M. (1906). The auriculo-ventricular bundle of the human heart. *Lancet*, 2:359–364.

Kent, S. (1914). A conducting path between the right auricle and exterior wall of the right ventricle in the heart of the mammal. *Journal of Physiology*, 48:57.

Klassner, F., Lesser, V. et Nawab, H. (1998). The IPUS blackboard architecture as a framework for computational auditory scene analysis. *Computational Auditory Scene Analysis*, pages 105–114.

Kohler, B.-U., Hennig, C. et Orglmeister, R. (2002). The principles of software QRS detection. *IEEE Engineering in Medicine and Biology Magazine*, 21(1):42–57.

Koulouris, A., Papakonstantinou, G. et Tsanakas, P. (2000). A decentralized multichannel length transformation algorithm and its parallel implementation for real-time ECG monitoring. *Computers and Biomedical Research*, 33:227–244.

Lake, C., éditeur (1990). *Clinical Monitoring.* Saunders, Philadelphia.

Larsson, J. et Hayes-Roth, B. (1998). GUARDIAN : An intelligent autonomous agent for medical monitoring and diagnosis. *IEEE Intelligent Systems*, 13:58–64.

Lawless, S. T. (1994). Crying wolf : false alarms in a pediatric intensive care unit. *Critical Care Medicine*, 22(6):981–985.

Le Beux, P., Burgun, A. et Cléret, M. (1994). De la méconnaissance à l'expertise. *Informatique et Santé*, 7:125–146.

Le Certen, G., Soulas, T., Carrault, G. et Le Pichon, J.-P. (1998). SIMBAD : Intelligent ECG monitoring system (a real time software implementation). Dans *8th International IMEKO Conference on Measurement in Clinical Medicine Biomedical measurement and instrumentation & 12th International Symposium on Biomedical Engineering*, pages 3.11–3.14, Dubrovnik.

Le Moulec, F. (1991). *Étude et réalisation d'un modèle qualitatif profond de l'activité électrique du cœur pour un système de monitoring intelligent en unité de soins intensifs pour coronariens.* Thèse de doctorat, Université de Rennes 1.

Lesser, V., Nawab, H., Gallastegi, I. et Klassner, F. (1993a). IPUS : an architecture for the integrated processing and understanding of signals. Dans *11th National Conference on Artificial Intelligence (AAAI-93)*, pages 249–255, Washington, D.C.

Lesser, V., Nawab, H. et Klassner, F. (1993b). IPUS : an architecture for the integrated processing and understanding of signals. Computer Science Technical Report 93-29, University of Massachusetts.

Lesser, V., Nawab, S. et Klassner, F. (1995). IPUS : an architecture for the integrated processing and understanding of signals. *Artificial Intelligence*, 77:129–171.

Li, C., Zheng, C. et Tai, C. (1995). Detection of ECG characterictic points using wavelet transforms. *IEEE Transactions on Biomedical Engineering*, 42:21–28.

Liszka, K., Mackin, M., Lichter, M., York, D., Pillai, D. et Rosenbaum, D. (2004). Keeping a beat on the heart. *Pervasive Computing*, 3(4):42–49.

Lowe, A., Jones, R. et Harrison, M. (2001). The graphical presentation of decision support information in an intelligent anaesthesia monitor. *Artificial Intelligence in Medicine*, 22:173–191.

Mabry, S., Schneringer, T., Etters, T. et Edwards, N. (2003). Intelligent agents for patient monitoring and diagnostics. Dans *ACM Symposium on Applied Computing (SAC 2003)*, pages 257–262, Melbourne.

Mackay, J. et Mensah, G. (2004). *The Atlas of Heart Disease and Stroke.* Organisation mondiale de la santé.

Mallat, S. (1999). *A Wavelet Tour of Signal Processing.* Academic Press, 2^e édition.

Mark, R. et Moody, G. (1988). MIT-BIH arrhythmia data base directory. Massachusetts Institute of Technology.

Marsh, H. (1990). *Monitoring in the Critical Care Unit*, chapitre Monitoring in Anesthesia and Critical Care Medicine, pages 815–827. Churchill Livingstone, New York, 2^e édition.

MATOUSEK, M. et POSNER, E. (1969). Purkinje's muscle fibers in the heart. *British Heart Journal*, 31:718–721.

MEYSTRE, S. (2005). The current state of telemonitoring : a comment on the literature. *Telemedicine and e-Health*, 11(1):63–69.

MIKSCH, S., HORN, W., POPOW, C. et PAKY, F. (1996). Utilizing temporal data abstraction for data validation and therapy planning for artificially ventilated newborn infants. *Artificial Intelligence in Medicine*, 8:543–576.

MOISAN, S. (1998). *Une plate-forme pour une programmation par composants de systèmes à base de connaissances.* Habilitation à diriger les recherches, Université de Nice.

MOISAN, S. et THONNAT, M. (1995). Knowledge-based systems for program supervision. Dans *1st International Workshop on Knowledge Based systems for the (re)Use of Program Libraries*, pages 4–8, Sophia Antipolis, France.

MOISAN, S., VINCENT, R. et THONNAT, M. (1997). Program supervision : from knowledge modeling to dedicated engines. Rapport de recherche 3324, INRIA.

MOISAN, S. et ZIÉBELIN, D. (2000). Résolution de problèmes en pilotage de programmes. Dans 12^e *édition du congrès Reconnaissance des Formes et Intelligence Artificielle (RFIA 2000)*, Paris.

MOODY, G., MULDROW, W. et MARK, R. (1984). A noise stress test for arrhythmia detectors. *Computers in Cardiology*, 11:381–384.

MORA, A., PASSARIELLO, G., CARRAULT, G. et LE PICHON, J.-P. (1993). Intelligent patient monitoring and management systems : a review. *IEEE Engineering in Medicine and Biology*, 12(4):23–33.

MORET-BONILLO, V., ALONSO-BETANZOS, A., MARTÍN, E. G., CANOSA, M. C. et BERDIÑAS, B. G. (1993). The PATRICIA project : A semantic-based methodology for intelligent monitoring in the ICU. *IEEE Engineering in Medicine and Biology*, 12(4):59–68.

MORET-BONILLO, V., CABRERO-CANOSA, M. et HERNANDEZ-PEREIRA, E. (1998). Integration of data, information and knowledge in intelligent patient monitoring. *Expert Systems with Applications*, 15:155–163.

MORIK, K., IMBOFF, M., BROCKHAUSEN, P., JOACHIMS, T. et GATHER, U. (2000). Knowledge discovery and knowledge validation in intensive care. *Artificial Intelligence in Medicine*, 19(3):225–249.

NAGIN, V. et SELISHCHEV, S. (2001). Implementation of algorithms for identification of QRS-complexes in real time ECG systems. *Biomedical Engineering*, 35(6):304–309.

NYGÅRDS, M.-E. et L. SÖRNMO (1981). A QRS delineation algorithm with low sensitivity to noise and morphology changes. Dans *Computer in Cardiology*, pages 347–350, Florence.

OKADA, M. (1979). A digital filter for the QRS complex detection. *IEEE Transactions on Biomedical Engineering*, BME-26:700–703.

Pan, J. et Tompkins, W. J. (1985). A real-time QRS detection algorithm. *IEEE Transactions on Biomedical Engineering*, 32(3):230–236.

Pierson, D. J. (1998). *Principles and Practice of Intensive Care Monitoring*, chapitre Goals and indications for monitoring, pages 33–44. McGraw-Hill, Health Professions Division, New York.

Poli, R., Cagnoni, S. et Valli, G. (1995). Genetic design of optimum linear and nonlinear QRS detectors. *IEEE Transactions on Biomedical Engineering*, 42(11): 1137–1141.

Popow, C., Horn, W., Rami, B. et Schober, E. (2003). VIE-DIAB : a support program for telemedical glycaemic control. Dans *9th conference on artificial intelligence in medicine (AIME 2003)*, pages 350–354.

Portet, F. et Carrault, G. (2005). Piloting real-time QRS detection algorithms in variable contexts. Dans *3rd European Medical and Biological Engineering Conference (EMBEC'05)*, Prague.

Portet, F., Carrault, G., Cordier, M.-O. et Quiniou, R. (2005a). Pilotage en ligne d'algorithmes de traitement du signal guidé par le contexte courant. Dans *7e Rencontres des Jeunes Chercheurs en Intelligence Artificielle (RJCIA'05)*, pages 1–14, Nice.

Portet, F., Carrault, G., Cordier, M.-O. et Quiniou, R. (2005b). Signal processing algorithms in a cardiac monitoring context. Dans *First Doctoral Consortium of the 10th Conference on Artificial Intelligence in Medicine (AIME 2005)*, Aberdeen.

Portet, F., Carrault, G., Cordier, M.-O. et Quiniou, R. (2006). Pilotage d algorithmes pour un diagnostic médical robuste en cardiologie. Dans *15^e édition du congrès Reconnaissance de Formes et Intelligence Artificielle (RFIA 2006)*, Tours.

Portet, F., Hernández, A. et Carrault, G. (2005c). Evaluation of real-time QRS detection algorithms in variable contexts. *Medical & Biological Engineering & Computing*, 43(3):381–387.

Purkinje, P. (1845). Mikroskopisch-neurologische beobachtungen. *Archiv für Anatomie, Physiologie und Wissenschaftliche Medizin*, 12:281–295.

Quiniou, R., Cordier, M.-O., Carrault, G. et Wang, F. (2001). Application of ILP to cardiac arrhythmia characterization for chronicle recognition. *Lecture Notes in Computer Science*, 2157:220–227.

Rubel, P., Fayn, J., Sinon-Chautemps, L., Atoui, H., Ohlsson, M., Telisson, D., Adami, S., Arod, S., Forlini, M. C., Malossi, C., Placide, J., Ziliani, G. L., Assanelli, D. et Chevalier, P. (2004). New paradigms in telemedicine : ambient intelligence, wearable, pervasive and personalized. *Studies in Health Technology and Informatics*, 108:123–132.

Ruha, A., Sallinen, S. et Nissilä, S. (1997). A real-time microprocessor QRS detector system with a 1-ms timing accuracy for the measurement of ambulatory HRV. *IEEE Transactions on Biomedical Engineering*, 44(3):159–167.

Senhadji, L. (1993). *Approche multirésolution pour l'analyse des signaux non stationnaires*. Thèse de doctorat, Université de Rennes 1, France.

Senhadji, L., Carrault, G., Bellanger, J.-J. et Passariello, G. (1995). Comparing wavelet transforms for recognizing cardiac patterns. *IEEE Engineering in Medicine and Biology*, pages 167–173.

Shahar, Y. et Musen, M. (1993). RÉSUMÉ : a temporal-abstraction system for patient monitoring. *Computers and Biomedical Research*, 26:255–273.

Sharshar, S., Allart, L. et Chambrin, M.-C. (2005). A new approach to the abstraction of monitoring data in intensive care. Dans *10th Conference on Artificial intelligence in medicine (AIME 2005)*, pages 13–22, Aberdeen.

Shekhar, C., Burlina, P. et Moisan, S. (1997). Design of self-tuning IU systems. Dans *Defense Advanced Research Projects Agency (DARPA) : Image Understanding Workshop*, volume 1, pages 529–536, New Orleans.

Shekhar, C., Moisan, S. et Thonnat, M. (1994). Towards an intelligent problem-solving environment for signal processing. *Mathematics and Computers in Simulation*, 36:347–359.

Shekhar, C., Moisan, S., Vincent, R., Burlina, P. et Chellappa, R. (1999). Knowledge-based control of vision systems. *Image and Vision Computing*, 17:667–683.

Shortliffe, E. (1976). *Computer-Based Medical Consultations : MYCIN*. Elsevier, New York.

Silipo, R. et Marchesi, C. (1998). Artificial neural networks for automatic ECG analysis. *IEEE Transactions on Signal Processing*, 46(5):1417–1425.

Siregar, P., Chahine, M., Lemoulec, F. et Le Beux, P. (1995). An interactive qualitative model in cardiology. *Computer and Biomedical Research*, 28(6):443–478.

Siregar, P., Coatrieux, J. et Le Beux, P. (1989). KISS : Knowledge based interactive signal monitoring system. AIM A10006 1.2.A-F-R-890717.

Soulas, T. (2001). *Une architecture de monitoring intelligent pour le domaine médical (Approche méthodologique - Application en cardiologie)*. Thèse de doctorat, Université de Rennes 1.

Soulas, T., Le Certen, G., Le Pichon, J.-P. et Carrault, G. (1998). Algorithm switching in real time monitoring. Dans *Symposium on Electronics and Telecommunications (ETC)*, pages 145–149, Timisoara.

Sterling, L. et Shapiro, E. (1986). *The art of PROLOG*. Logic Programming. MIT press.

Stridh, M. et Sörnmo, L. (2001). Spatiotemporal QRST cancellation techniques for analysis of atrial fibrillation. *IEEE Transactions on Biomedical Engineering*, 48(1):105–111.

Sukuvaara, T., Sydänmaa, M., Nieminen, H., Heikelä, A. et Koski, E. (1993). Object oriented implementation of an architecture for patient monitoring. *IEEE Engineering in Medicine and Biology*, 12(4):69–81.

Sun, Y., K. Chan et Krishnan, S. (2002). ECG signal conditioning by morphological filtering. *Biomedical Engineering*, 32:465–479.

Suppappola, S. et Sun, Y. (1994). Nonlinear transforms of ECG signals for digital QRS detection : a quantitative analysis. *IEEE Transaction on Biomedical Engineering*, 41(4):397–400.

Sztipanovits, J., Karsai, G. et Bapty, T. (1998). Self-adaptive software for signal processing. *Communications of the ACM*, 41(5):55–65.

Sztipanovits, J., Karsai, G., Biegi, C., Bapty, T., Ledeczi, A. et Misra, A. (1995). MULTIGRAPH : An architecture for model-integrated computing. Dans *1st International Conference on Engineering of Complex Computer Systems (ICECCS'95)*, pages 361–368.

Tawara, S. (1906). *Das reizleitungsysem des säugetierherzens : eine anatomish-histologische studie über das atrioventrikularbündel und die purkinjeschen fäden.* Verlag Von Gustav Fisher.

Tawara, S. (2000). *The conduction System of the Mammalian Heart : an anatomico-histological study of the atrioventricular bundle and the purkinje fibers.* imperial college press.

Thonnat, M. et Moisan, S. (1995). Knowledge-based systems for program supervision. Dans *Knowledge Based (re)Use of Program Libraries (KBUP'95)*, pages 3–8, Sophia Antipolis.

Thoraval, L., Carrault, G., Schleich, J.-M., Summers, R., Van de Velde, M. et Diaz, J. (1997). Data fusion of electrophysiological and haemodynamic signals for ventricular rhythm tracking. *IEEE Engineering in Medicine and Biology*, pages 48–55.

Tsien, C. et Falcker, J. (1997). Poor prognosis for existing monitors in the intensive care unit. *Critical Care in Medicine*, 25(4):614–619.

Uckun, S. (1993). Intelligent systems in patient monitoring and therapy management : a survey of research projects. Rapport technique KSL-TR-93-32, Knowledge Systems Laboratory, Stanford University.

Vila, J., Presedo, J., Delgado, M., Barro, S., Ruiz, R. et Palacios, F. (1997). SUTIL : intelligent ischemia monitoring system. *International Journal of Medical Informatics*, 47:193–214.

Vilhelm, C. (1998). *Conception et développement d'un outils de traitement de connaissances, combinant les approches objet et connexionnistes. Application au domaine des soins intensifs.* Thèse de doctorat, Université de Technologie de Compiègne.

Vilhelm, C., Jaborska, A., Chambrin, M.-C. et Ravaux, P. (1996). A software architecture for a medical data acquisition, report and interpretation system in

intensive care unit. Dans *18th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, Amsterdam.

Vilhelm, C., Ravaux, P., Calvelo, D., Jaborska, A., Chambrin, M. et Boniface, M. (2000). Think ! : a unified numerical - symbolic knowledge representation scheme and reasoning system. *Artificial Intelligence*, 116:67–85.

Vásquez, C., Hernández, A., Mora, F., Carrault, G. et Passariello, G. (2001). Atrial activity enhancement by wiener filtering using an artificial neural network. *IEEE Transactions on Biomedical Engineering*, 48(8):940–944.

Wagner, G., Stork, W. et Müller-Glaser, K. (2003). Model-based evaluation of different QRS detection algorithms for use in mobile systems. *Biomedizinische Technik*, 48(1):210–211.

Wang, F. (2002). *Abstraction temporelle de signal ECG, apprentissage inductif de contraintes temporelles et reconnaissance des arythmies cardiaques*. Thèse, Université de Rennes 1.

Wang, F., Quiniou, R., Carrault, G. et Cordier, M. (2001). Learning structural knowledge from the ECG. Dans *2nd International Symposium on Medical Data Analysis*, pages 288–294.

Watrous, R. et Towell, G. (1995). A patient-adaptive neural network ECG patient monitoring algorithm. Dans *Proceedings Computers in Cardiology*, Vienna.

Wieben, O., Afonso, V. et Tompkins, W. (1999). Classification of premature ventricular complexes using filter bank features, induction of decision trees and a fuzzy rule-based system. *Medical & Biological Engineering & Computing*, 37(5):560–565.

Wilson, F., Johnston, F., McLeod, A. et Barker, P. (1934). Electrocardiograms that represent the potential variations of a single electrode. *American Heart Journal*, 9:447–471.

Wilson, F., Johnston, F., Rosenbaum, F., Erlanger, H., Kossmann, C., Hecht, H., Cotrim, N., de Olivieira, R. M., Scarsi, R. et Barker, P. (1944). The precordial electrocardiogram. *American Heart Journal*, 27:19–85.

Wrzesniowski, A. et Augustyniak, P. (2001). Adaptive channels weighting for the QRS detection in long-term electrocardiograms. Dans *6th International Conference SYMBIOSIS 2001*, pages 27–32, Szczyrk.

Zoubir, A. et Boashash, B. (1998). The bootstrap and its application in signal processing. *IEEE Signal Processing Magazine*, 15(1):56–76.

Table des figures

Résumé

L'objectif de cette thèse est la réalisation du système de monitorage cardiaque intelligent IP-Calicot capable, grâce à un module de pilotage d'algorithmes, d'utiliser les informations du contexte courant pour modifier sa chaîne de traitements afin d'obtenir un diagnostic médical fiable même en milieu bruité. À partir d'un électrocardiogramme (ECG), le système extrait en ligne, par traitement du signal, les informations qui vont permettre d'établir un diagnostic d'arythmie cardiaque modélisé par un réseau temporel (chronique). En utilisant le contexte courant, constitué du bruit de ligne et du diagnostic médical, le module de pilotage agit dynamiquement à trois niveaux : il sélectionne et paramètre les algorithmes de traitement du signal, il choisit les éléments à extraire du signal, décrivant ainsi l'ECG dans un langage plus ou moins précis, et sélectionne le langage de description à utiliser pour établir le diagnostic en ligne. Le pilote est représenté par un système expert qui agit sur la chaîne de traitements grâce à des règles de pilotage acquises par expertises et déduites d'études statistiques. Le système a été validé sur des ECG bruités typiques de situations cliniques. Les résultats démontrent l'intérêt et la faisabilité du pilotage proposé.

Mots-clés

Pilotage d'algorithmes, système de monitorage intelligent, traitement du signal biomédical, système expert, système adaptatif, électrocardiographie, acquisition de connaissance, apprentissage artificiel.

Abstract

The aim of this work is the realization of the cardiac intelligent monitoring system IP-Calicot which is able, thanks to a piloting algorithms module, to use the current context information to modify dynamically its data processing chain to obtain a reliable medical diagnosis even in noisy situations. From an electrocardiogram (ECG), the system extracts on line, by signal processing, information which are used to establish a cardiac arrhythmia diagnosis modeled by a temporal network (chronicle). Using the current context, composed of the line noise and the medical diagnosis, the piloting module acts dynamically at three levels : it selects and tunes signal processing algorithms, it chooses the elements to be extracted from the ECG, which is thus described in a more or less precise language, and selects the description language to be used to establish the diagnosis on line. The pilot is represented by an expert system which acts on the data processing chain thanks to piloting rules acquired by expertise and deduced from statistical studies. The system has been validated on noisy ECG typical of clinical situations. The results show the interest and the feasibility of the proposed approach.

Keywords

Piloting algorithms, intelligent monitoring system, biomedical signal processing, adaptive system, knowledge acquisition, electrocardiography, expert system, machine learning.