Approches méthodologiques et logicielles pour la prise de décision et la conception paramétrique optimisée de bâtiments modulaires

Abdulaziz Afandi

THÈSE PRÉSENTÉE
POUR OBTENIR LE GRADE DE

DOCTEUR DE L'UNIVERSITÉ DE BORDEAUX

ÉCOLE DOCTORALE SCIENCES PHYSIQUES ET DE L'INGENIEUR

SPÉCIALITÉ : MÉCANIQUE

Par

Abdulaziz Naji Zaki AFANDI

Approches Méthodologiques et Logicielles Pour la Prise de Décision et la Conception Paramétrique Optimisée de Bâtiments Modulaires

Co-direction : Denis BRUNEAU, Patrick SÉBASTIAN
Co-encadrement : Aline BARLET, Xavier MARSAULT

Soutenue le 11 septembre 2020

Membres du jury :

M. TERRACOL, Pascal	(PR-HDR, ECVAU, ENSA ParisVal de Seine)	Président
M. BIGEON, Jean	(DR CNRS, G-SCOP, INP Grenoble)	Rapporteur
M. LESCOP, Laurent	(PR-HDR, AAU, ENSA Nantes)	Rapporteur
Mme. ZARCONE, Roberta	(MCF, GSA, ENSA Paris-Malaquais)	Examinatrice
Mme. BARLET, Aline	(MCF, I2M, ENSAP Bordeaux)	Co-encadrante
M. BRUNEAU, Denis	(PR-HDR, I2M, ENSAP Bordeaux)	Co-directeur
M. MARSAULT, Xavier	(Dr en Informatique, MAP-Aria, ENSA Lyon)	Co-encadrant
M. SÉBASTIAN, Patrick	(MCF-HDR, I2M, Université de Bordeaux)	Co-directeur

Invités :

M. KHAN, Ahmed	(Assistant Professor, King Abdulaziz University, Arabie Saoudite)
M. YUSTA GARCIA, Ferran	(MCF, I2M, ENSAP Bordeaux)

Approches Méthodologiques et Logicielles Pour la Prise de Décision et la Conception Paramétrique Optimisée de Bâtiments Modulaires

Par Abdulaziz AFANDI

Pour obtenir le titre de

Docteur

Sciences Physiques et de l'Ingenieur (SPI)
Spécialité : Mécanique

Résumé

L'optimisation est un processus intéressant pour la conception en général et la conception architecturale en particulier. Il existe de nombreux outils d'optimisation de conception générative en architecture. Cependant, ces outils ne sont pas très utilisés par les architectes. La conception architecturale pose des problèmes mal structurés; la créativité et l'interprétation des concepteurs sont essentielles pour résoudre ces problèmes. Donc en architecture, l'acceptabilité des solutions par les concepteurs est aussi importante que l'optimalité numérique de leurs performances. Or, bien que les préférences des concepteurs sont cruciales pour l'acceptabilité, les outils existants ne les intègrent pas dans le processus d'optimisation. Le maque d'implication possible pour le concepteur lors de l'utilisation des outils est une cause majeure de la réticence des architectes à utiliser ces outils. Cette thèse vise à définir un ensemble de recommandations qui aident les développeurs à proposer des systèmes d'aide à la décision plus attractifs pour les architectes, car permettant une plus grande intégration du concepteur dans le processus d'optimisation et donc une plus grande implication lors de l'utilisation de ces outils.

Pour définir l'ensemble des recommandations, la recherche a commencé par explorer différents processus de conception. A partir de cette exploration, un cadre de conception basé sur quatre modèles Morphogenèse, Observation, Interprétation, Agrégation, MOIA est défini. Ensuite, les typologies d'outils utilisées par les architectes sont explorées. En outre, les "workflows" d'optimisation de conception générative les plus connus sont étudiés, en utilisant MOIA comme référence. Ensuite, la recherche adopte une approche expérimentale portant sur l'acceptabilité des concepteurs. Cinq expériences différentes sont réalisées. Deux des expériences comparent différents "workflows" d'optimisation de conception générative existants en utilisant l'acceptabilité des concepteurs comme référence. Les trois autres expériences comparent les différentes fonctions d'agrégation en utilisant le jugement des concepteurs comme référence. Ces fonctions sont la fonction de Pareto, Maximin, Derringer & Suich.

Les résultats de ces expériences peuvent être résumés en quatre points. Premièrement, la programmation visuelle est recommandée pour les futurs outils d'optimisation générative. La programmation visuelle aide l'architecte à décrire des modèles paramétriques sophistiqués sans codage. En effet, les concepteurs, en général, ne sont pas formés à coder. Deuxièmement, l'aspect graphique de l'outil peut fortement influencer la décision du concepteur. Les performances des solutions doivent être présentées graphiquement aux concepteurs et la méthode de représentation doit dépendre du nombre d'objectifs. Troisièmement, l'utilisation d'un algorithme d'optimisation interactif qui permet aux concepteurs de sélectionner la solution en fonction de leur jugement subjectif de la forme peut augmenter l'acceptabilité des "workflows". Quatrièmement, la disponibilité des informations est la clé pour définir la fonction d'agrégation adaptée. L'interprétation requise pour les différentes fonctions d'agrégation n'est pas toujours la même. Lorsque les informations nécessaires sont disponibles, les fonctions cardinales à forte néguentropie sont préférées aux fonctions ordinales à faible néguentropie.

les outils d'optimisation de conception générative existants doivent être davantage attractifs pour les architectes. La recherche adopte une approche expérimentale basée sur l'acceptabilité des concepteurs. La méthodologie développée dans cette recherche a permis de définir un ensemble de recommandations visant à réaliser des outils plus attractifs pour les concepteurs, favorisant ainsi la pratique de l'optimisation lors des processus de conception. La recommandation se concentre sur le fait d'offrir la possibilité aux concepteurs d'être plus impliqués dans le processus.

Mots-clés : acceptabilité, prise de décision, désirabilité, conception générative, optimisation de la conception multi-objectifs

Résumé substantiel

Nous passons la majeure partie de notre vie dans nos bâtiments. Dans tout bâtiment, le confort des utilisateurs est donc prioritaire. Le secteur du bâtiment consomme une quantité considérable d'énergie et de ressources. L'optimisation des performances de nos bâtiments est alors cruciale. Bien que l'optimisation soit populaire parmi les ingénieurs, elle l'est moins chez les concepteurs en général, et plus spécifiquement, les architectes. De nos jours, de nombreux systèmes pouvant aider les concepteurs à optimiser existent. Cependant, ces outils sont faits pour observer le comportement des solutions candidates sans pouvoir les générer. Dernièrement, quelques outils numériques d'optimisation de la conception dotés de capacités génératives ont été introduits. Cependant, ces outils ne sont pas très utilisés par les architectes, car ils ne se sentent pas impliqués dans le processus d'optimisation. La collaboration faible ou déséquilibrée entre l'architecte et l'ordinateur dans ces outils est un problème important.

Pour résoudre ce problème, il était essentiel de revoir le processus de conception du point de vue des architectes. L'analyse des principes et des phases du processus de conception a permis de développer un cadre pour l'optimisation générative de la conception. Le cadre proposé est itératif et basé sur quatre modèles : les modèles de Morphogenèse, d' Observation, d' Interprétation et d'Agrégation (MOIA). Le cadre MOIA intègre l'évaluation par les concepteurs dans le processus d'optimisation. Dans un premier temps, le modèle de morphogenèse utilise un ensemble aléatoire de variables de conception pour définir un ensemble aléatoire de solutions candidates. Ensuite, le modèle d'observation observe le comportement de ces solutions candidates, ce qui se traduit par un ensemble de variables d'observation. Ensuite, le modèle d'interprétation interprète les variables d'observation en variables d'interprétation. Plus tard, le modèle d'agrégation classe les solutions en fonction des variables d'interprétation. Enfin, le modèle de morphogenèse utilise un algorithme d'optimisation globale pour faire évoluer un nouvel ensemble de variables de conception. Un seuil peut être utilisé pour définir la fin de l'itération.

En MOIA, les concepteurs peuvent exprimer leurs préférences concernant les critères et les objectifs à l'intérieur des modèles d'interprétation et d'agrégation. De nombreuses interprétations et fonctions d'agrégation existent. En fonction des types d'informations qu'elles utilisent, ces fonctions sont divisées en deux catégories : ordinales et cardinales. Les informations ordinales sont basées sur les rangs. Les informations cardinales sont basées sur des valeurs. À partir des informations cardinales, nous pouvons déduire les informations ordinales, mais l'inverse est impossible. Pour évaluer la valeur d'une information, nous utilisons le concept de néguentropie, opposé de l'entropie, qui correspond au caractère aléatoire de l'information. En diminuant le caractère aléatoire de l'information, nous augmentons sa néguentropie et donc sa valeur. Les fonctions cardinales sont plus élevées que les fonctions ordinales en néguentropie, et donc elles ont plus de valeur. En effet, les informations ordinales peuvent être trompeuses.

La caractéristique commune à la plupart des outils d'optimisation multi-objectifs contemporains est qu'ils utilisent la fonction de Pareto pour améliorer les compromis entre les objectifs. Cette fonction est ordinale et transfère les variables d'observation cardinales en informations ordinales. Parce que nous ne pouvons pas appliquer d'opérations mathématiques aux informations ordinales, nous ne pouvons donc pas calculer un objectif global à partir de nombreux critères et objectifs. En conséquence, nous nous retrouvons avec de nombreux objectifs et la fonction peut considérer différentes solutions avec des variables d'observation

distinctes comme également optimales. De plus, il peut considérer comme optimale une solution irrationnelle pour les experts en conception. La fonction de Pareto classe les solutions en deux catégories, non dominées (optimales) et dominées (non optimales). Par conséquent, il peut être considéré comme un filtre à faible efficacité.

Par rapport à la fonction de Pareto, les fonctions cardinales riches en néguentropie sont reconnues comme un filtre puissant. Les fonctions d'agrégation de Maximin et Derringer & Suich ont un potentiel élevé pour remplacer la fonction de Pareto. Cependant, ces fonctions nécessitent l'utilisation d'une fonction d'interprétation de l'opportunité : une fonction de désirabilité. Il s'agit d'une fonction de valeur qui transfère les variables d'observation basées sur différentes échelles en variables d'interprétation basées sur une échelle de satisfaction unifiée comprise entre zéro et une Les variables d'interprétation représentent le niveau de satisfaction des objectifs en fonction des préférences des concepteurs. La fonction d'agrégation proposée classe les solutions en fonction de ces variables.

La fonction d'agrégation Maximin sous-estime les solutions qui atteignent de très faibles niveaux de satisfaction d'au moins un objectif. Il peut être considéré comme un principe de précaution qui évite les solutions extrêmement dangereuses. Cependant, Maximin est une fonction non compensatoire. En utilisant Maximin, les concepteurs peuvent exprimer leurs préférences des critères dans le modèle d'interprétation en utilisant une fonction de désirabilité. La fonction d'agrégation de Derringer & Suich est supérieure à Maximin en néguentropie, et elle est compensatoire. Dans cette fonction, les concepteurs peuvent exprimer leurs préférences pour les critères du modèle d'interprétation en utilisant une fonction de désirabilité. En outre, ils peuvent exprimer leurs préférences en termes d'objectifs en leur permettant d'attribuer des poids différents à ces derniers.

Les problèmes de conception sont des problèmes mal structurés; leur structure manque de définition. La résolution de ces problèmes implique un jugement subjectif, difficile à traiter et non réductible à la logique mathématique pure. D'après l'évaluation faite pardes concepteurs, les solutions optimales ne sont pas nécessairement acceptables, du point de vue des perceptions humaines. Pour évaluer l'acceptabilité, les concepteurs doivent être au centre du raisonnement et de l'évaluation. A l'opposé, l'optimalité indique les mesures de performance et utilise une logique mathématique pour calculer les objectifs numériques. Dans la conception, l'acceptabilité et l'optimalité sont essentielles. En permettant aux concepteurs d'exprimer leurs préférences à l'intérieur du processus d'optimisation, MOIA peut approcher l'acceptimalité (acceptabilité et optimalité). Le développement de systèmes d'aide à la décision basés sur l'acceptimalité peut attirer plus de concepteurs car l'acceptabilité des solutions est accrue.

Il est crucial d'explorer les outils populaires parmi les architectes en tenant compte du MOIA. Sur la base de cette exploration, quatre typologies d'outils différents pouvant aider les modèles de MOIA ont été définies. La première typologie est celle des outils d'observation. Ces outils sont essentiels pour le modèle d'observation car ils permettent d'évaluer le comportement des solutions candidates. La deuxième typologie est celle des outils de modélisation paramétrique. Ces outils nous permettent de définir un modèle avec des contraintes. Les contraintes connectent les pièces du modèle. Ainsi, nous pouvons changer le modèle entier en changeant une partie de celui-ci. La troisième typologie est la modélisation algorithmique, qui peut être considérée comme une modélisation paramétrique avancée. Dans cette typologie, nous définissons un modèle paramétrique à l'aide d'algorithmes. Étant donné que la plupart des concepteurs ne sont pas formés pour utiliser la programmation textuelle, de nombreux outils de modélisation algorithmique pour les concepteurs utilisent la programmation visuelle, qui est relativement facile à utiliser et offre toutefois des capacités élevées. La quatrième typologie correspond aux outils de conception générative, qui manipulent généralement le modèle paramétrique pour générer des solutions. En conception générative, les outils génératifs sont connectés à un modèle paramétrique, à des outils d'observation et à des algorithmes d'optimisation. De cette exploration, nous pouvons conclure que les systèmes basés sur la conception générative sont les plus proches du cadre MOIA.

L'exploration des “workflows” de conception générative contemporaine est vitale car elle permet une meilleure compréhension des outils. Six “workflows” de conception générative différents ont été largement explorés. Alors que certains de ces “workflows” sont basés sur un seul outil, d'autres consistent à utiliser plusieurs outils. Sur la base de cette exploration, quatre “workflows” différents ont été définis comme les plus proches de MOIA. Ces “workflows” partagent un cadre similaire. Ils commencent par proposer un ensemble aléatoire de solutions candidates. Ensuite, ils observent le comportement des solutions candidates. Puis, ils les classent en utilisant la fonction de Pareto. Enfin, ils utilisent un algorithme évolutif pour faire évoluer un nouvel ensemble de solutions optimisées. Cependant, ces “workflows” peuvent approcher l'optimalité mais pas l'acceptimalité.

La thèse entend proposer un ensemble de recommandations pouvant accroître l'interaction entre les concepteurs et les machines. En développant un système d'aide à la décision basé sur ces recommandations, nous pouvons approcher l'acceptimalité. Il est souhaité que ces systèmes puissent encourager un plus grand nombre de concepteurs à adopter l'optimisation, procédure particulièrement éfficace. La recherche adopte une approche expérimentale pour étudier l'acceptabilité des concepteurs, en mettant en œuvre cinq expériences différentes. Les deux premières expériences se concentrent sur la comparaison de l'acceptabilité des différents outils de conception générative et des flux de travail à partir des évaluations faites par les concepteurs. Les trois autres expériences se concentrent sur l'acceptabilité des solutions, en étudiant trois fonctions d'agrégation différentes également à partir des évaluations faites par les concepteurs.

La première expérience compare l'acceptabilité pour les utilisateurs de deux “workflows” d'optimisation de conception générative différents. Le premier “workflow” est basé sur l'utilisation de la programmation visuelle. Le “workflow” observe les solutions candidates (des solutions aléatoires sont utilisées pour la première itération), puis il utilise un algorithme évolutif pour faire évoluer un nouvel ensemble de solutions candidates basé sur la classification des solutions candidates par fonction de Pareto. L’outil génératif de ce “workflow” présente graphiquement les solutions basées sur la classification des fonctions de Pareto. Le deuxième “workflow” est basé sur l'utilisation d'EcoGen ©, un outil de programmation non visuel pour générer des formes modulaires optimisées. Cet outil utilise un algorithme évolutif interactif. Il permet à l'utilisateur de sélectionner des solutions candidates en fonction de son évaluation subjective. Les résultats montrent que, d'une manière générale, les concepteurs trouvent le “workflow” qui utilise la programmation visuelle plus acceptable. Cependant, en ce qui concerne l'interface, les participants préfèrent EcoGen ©, qui présente les solutions regroupant les modèles dans une grille, et qui leur permet de choisir parmi ces solutions.

Pour compléter la première expérience, la deuxième expérience se concentre sur la programmation visuelle. Cette expérience compare deux outils génératifs différents pour la programmation visuelle. Le premier outil est basé sur la représentation graphique des solutions basées sur la fonction de Pareto, qui a été utilisée dans le test précédent. Le second est basé sur l'utilisation d'un algorithme évolutif interactif avec une interface similaire à EcoGen ©. Les résultats montrent que l'outil qui utilise un algorithme interactif est plus acceptable. Ce résultat est donc opposé à ceux du test précédent. Ainsi, lors du premier test, les participants ont sous-estimé l'algorithme interactif car il n'utilise pas de programmation visuelle. Cependant, les participants estiment que l'outil qui présente la fonction de Pareto peut produire de meilleurs résultats. En conclusion, la combinaison des deux approches dans la programmation visuelle peut augmenter l'acceptabilité des outils d'optimisation générative.

La troisième expérience étudie les deux fonctions d'agrégation qui peuvent remplacer la fonction de Pareto. Pour la première fonction, qui est la fonction de Derringer & Suich, nous avons tester si le concept d'attribution de poids convient aux concepteurs et s'ils sont capables d'attribuer des poids cohérents. Dans ce test, les participants ont été invités à attribuer des poids à cinq objectifs dans quatre scénarios. Les réponses ont montré une grande variété de pondérations entre les différents scénarios et l'objectif. Cela reflète le besoin élevé de pondérer les objectifs. Dans un second temps, les participants ont été invités à attribuer des poids à cinq objectifs par paires pour un seul scénario afin d'évaluer leur cohérence. Les résultats montrent que, en général, les concepteurs étaient incohérents. Pour la deuxième fonction, Maximin, nous avons tester si son principe convient aux concepteurs. Tout d'abord, un ensemble de dix formes analysées sur la base de deux objectifs contraires est présenté. Les concepteurs ont dû sélectionner une forme qui optimise les objectifs. Ensuite, une représentation graphique des solutions basées sur la fonction de Pareto est présentée, et ils ont dû sélectionner à nouveau une forme. La majorité des participants a changé de sélection entre les deux phases de test. Ils ont décalé leur sélection pour correspondre au classement Maximin.

La quatrième expérience vise à comparer la fonction de Pareto et la fonction de Maximin à partir des évaluations faites par les concepteurs. L'expérience observe également l'influence de la disponibilité des informations sur l'évaluation des concepteurs. Six tests ont été effectués dans cette expérience permettant aux participants de classer un ensemble de sept solutions en fonction de deux objectifs. Ce sont deux ensembles de solutions présentés selon trois niveaux d'information qui ont été ainsi évalués. Le niveau d'information le plus bas correspondà la présentation d'un nuage de points, représentant sur un graphique les performances des solutions en fonction des objectifs. Le deuxième niveau présente en plus du nuage de points des informations numériques qui décrivent les bâtiments et leur contexte,. Enfin, le troisième présente les informations des niveau précédents, ainsi que des modèles 3D des solutions accompagnés de leurs performances en fonction des objectifs. Les résultats montrent que Maximin peut entraîner une classification qui est plus acceptable pour les concepteurs si elle est comparée à la classification dérivée de la fonction de Pareto. Ils montrent également que les informations supplémentaires autres que le nuage de points n'ont pas influencé de manière significative le jugement des concepteurs.

L'ensemble des solutions utilisées dans l'expérience précédente conduit à une classification similaire si nous utilisons Maximin ou Derringer & Suich (poids égaux). La cinquième expérience vise à comparer les fonctions de Maximin et de Derringer & Suich à partir des évaluations faites par les concepteurs. Les participants à cette expérience ont été invités à classer deux ensembles différents de solutions en fonction de deux objectifs pour le premier ensemble et de cinq objectifs pour le second. Pour le premier ensemble, un nuage de points, les valeurs numériques, comprises entre zéro et un, du niveau de satisfaction des objectives et un diagramme de coordonnées parallèles représentant les performances des solutions ont été présentés. De plus, l'emplacement et le type de bâtiment ont été spécifiés. Pour le second ensemble, seules les valeurs numériques qui représentent le niveau de satisfaction des objectifs et un diagramme de coordonnées qui représentent les performances de la solution ont été présentés. Il ressort des résultats que l'augmentation du nombre d'objectifs incite les concepteurs à compenser. L'écart entre la classification des participants et Derringer & Suich (poids égaux) était inférieur à l'écart entre la classification de Maximin et les participants. Cependant, lorsque seulement deux objectifs étaient visés et que le nuage de points était présenté, la classification des participants était plus proche de celle de Maximin que de celle de Derringer & Suich (poids égaux).

Sur la base de ces cinq expériences, un ensemble de quatre recommandations pouvant augmenter l'interaction entre le concepteur et l'outil numérique dans les systèmes d'aide à la décision pour la conception est proposé. D'abord, l'utilisation d'une conception générative basée sur une programmation visuelle est fortement recommandée. Deuxièmement, l'utilisation d'un algorithme interactif peut augmenter l'acceptabilité des outils car elle augmente l'interaction avec les utilisateurs. Troisièmement, l'aspect visuel de l'interface est importante. Cela comprend la présentation des modèles volumétriques de la solution et la représentation graphique des performances des solutions (par exemple, nuage de points ou coordonnées parallèles). Enfin, les systèmes doivent pouvoir utiliser différentes fonctions d'agrégation en fonction de la disponibilité des informations. Si les informations nécessaires pour utiliser une fonction de désirabilité sont disponibles et que les pondérations des objectifs sont décidables, nous pouvons utiliser la fonction de Derringer & Suich. Si les poids des objectifs ne sont pas décidables, mais que les informations nécessaires pour utiliser une fonction de désirabilité sont disponibles, nous pouvons utiliser Maximin ou Derringer & Suich (poids égaux). Dans le cas où le concepteur dispose de peu d'informations, nous pouvons utiliser la fonction de Pareto, et contrairement aux deux autres fonctions, nous ne pouvons pas atteindre l'acceptimalité. À l'avenir, nous pouvons relier les systèmes qui adoptent ces recommandations à l'apprentissage automatique.

Methodological and Software Approaches for Decision-Making and Optimized Parametric Design of Modular Buildings

By Abdulaziz Afandi

In partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Ph.D)

Physical Science and Engineering
Specialty: Mechanics

Abstract

Optimization is a profitable behavior for design in general and architectural design in specific. Many generative design optimization tools do exist. However, these tools are not widely used among architects. Design problems are ill-structured problems; designers' creativity and interpretation are essential for solving these problems. In design, designers' acceptability of the solutions is as important as the numerical optimality of their performance. The existing tools do not integrate designers' preferences inside the optimization process; designers' preferences are crucial for acceptability. The unbalance collaboration between the tools, and the designer is a major cause of the reluctance of architects from using these tools. The dissertation aims to define a set of recommendations that helps developers to introduce decision support systems that attract more architects by improving the collaboration between the designers and the tools.

To define the set of recommendations, the research started by exploring different design processes. Based on this exploration, a design framework based on four models Morphogenesis, Observation, Interpretation, Aggregation (MOIA) is defined. Next, the tool typologies the architects use are explored. Additionally, the popular generative design optimization workflows are investigated by using MOIA as a reference. Then, the research adopts an experimental approach based on designers' acceptability. Five different experiments are performed. Two of the experiments compare different existing generative design optimization workflows by using designers' acceptability as a reference. The other three experiments compare different aggregation functions by using designers' judgment as a benchmark. These functions are Pareto's function, Maximin, and Derringer & Suich's.

The results of these experiments can be concluded in four points. First, visual programming is recommended for future generative optimization tools. Visual programming helps the architect describe sophisticated parametric models without coding; designers, in general, are not trained to code. Second, the graphical aspect of the tool can immensely influence the decision of the designer. The performance of the solutions must be graphically presented to the designers; the representation method must respond to the number of objectives. Third, using an interactive optimization algorithm that allows the designers to select the solution based on their subjective judgment of the form can increase the acceptability of the workflows. Fourth, the availability of information is the key to define the accessible aggregation function. We usually use the aggregation function that integrates more of the available information; this information includes designers' preferences, which help to approach acceptability.

The existing generative design optimization tools need to attract more architects. The research adopts an experimental approach based on designers' acceptability. The methodology helped the research define a set of recommendations that can help future tools attract more designers to optimize. The recommendation mainly focuses on enhancing the collaboration between the tools and the designers.

Keywords: acceptability, decision-making, design optimization, desirability, Generative Design.

Substantial summary

We spend most of our lives in our buildings. In any building, users' comfort is in high priority. The building sector consumes a considerable amount of energy, and it significantly consumes our resources. Optimizing the performance of our buildings is crucial. While optimization is popular among engineers, it is less popular among designers in general, and more specifically, architects. Nowadays, many systems that can help designers to optimize do exist. However, these tools are made for observing the behavior of the candidate solutions without being able to generate any solutions. Lately, few design optimization digital tools with generative capabilities were introduced. However, these tools are not widely used among architects. These tools do not engage them in the optimization process. The weak or unbalanced collaboration between the architect and the computer in these tools is a significant problem.

To solve this problem, it is was essential to review the design process from the architects' point of view. Reviewing the principles and activities of the design process helped to develop a framework for generative design optimization. The proposed framework is iterative and based on four models Morphogenesis, Observation, Interpretation, Aggregation (MOIA). MOIA framework integrates designers' judgment inside the optimization process. At first, the Morphogenesis model uses a set of random values of design variables to define a set of candidate solutions. Next, the Observation model observes the behavior of these candidate solutions, which results in a set of observation variables. Then, the Interpretation model interprets the observation variables into interpretation variables. Later on, the Aggregation model classifies the solutions based on the interpretation variables. Finally, the Morphogenesis model uses a global optimization algorithm to evolve a new set of design variables. A threshold can be used to define the end of the iteration.

In MOIA, the designers can express their preferences of the criteria and the objectives inside the Interpretation and Aggregation models. Many interpretations and aggregation functions do exist. Based on the types of information they use, these functions are divided into two categories ordinal and cardinal. The ordinal information is based on ranks. The cardinal information is based on values. From the cardinal information, we can infer the ordinal information. However, it is not possible to infer the carinal information from the ordinal information. The cardinal information is more valuable than the ordinal information. To assess the worthiness of information, we use the concept of negentropy. The information negentropy is the opposite of information entropy, which corresponds to the randomness of information. By decreasing the randomness of information, we increase its negentropy and thus its value. The cardinal functions are higher than the ordinal functions in negentropy, and thus they are more valuable. The ordinal information can be misleading.

The shared characteristic among most of the contemporary multi-objective optimization tools is that they use Pareto's function to improve tradeoffs among the objectives. This function is ordinal, and it transfers the valuable cardinal observation variables into ordinal information. Because we cannot apply mathematical operations to ordinal information, thus we cannot compute a global objective from many criteria and objectives. As a result, we end up with many objectives. Consequently, the function may consider different solutions with distinct observation variables as equally optimum. Moreover, it can consider a solution that is irrational for design experts as optimum. Pareto's function classifies the solutions into two categories, non-dominated (optimum) and dominated (not optimum). Hence, it can be regarded as a low-efficiency filter.

In comparison to Pareto's function, the high in negentropy cardinal functions are recognized as a strong filter. Maximin and Derringer & Suich's aggregation functions have a high potential to replace Pareto's function. However, these functions require the use of a desirability interpretation function. A desirability function is a value function that transfers the observation variables based on different scales into interpretation variables based on a unified scale of satisfaction ranging between zero and one. The interpretation variables represent the level of satisfaction on the objectives based on designers' preferences. The proposed aggregation function classifies the solutions based on the interpretation variables.

Maximin aggregation function underestimates the solutions which attain very low levels of satisfaction of at least one objective. It can be regarded as a precautionary principle that avoids extreme unsafe solutions. However, Maximin is a non-compensatory function. By using Maximin, the designers can express his preferences of the criteria in the Interpretation model by using a desirability function. Derringer & Suich's aggregation function is higher than Maximin in negentropy, and it is compensatory. In this function, the designers can express their preference of the criteria within the Interpretation model by using a desirability function. Also, designers can express their preferences of the objectives by using allowing them to assign different weights to the objectives.

Design problems are ill-structured problems; their structure lacks some definition. Solving these problems involves subjective judgment, which is difficult to process and non-reducible to pure mathematical logic. Based on the designers' judgment, the optimum solutions are not necessarily acceptable. The acceptability concerns human perceptions. To assess acceptability, the designers must be the center of reasoning and judgment. Optimality indicates the performance measurements, and it uses mathematical logic to compute the numerical objectives. In design, both acceptability and optimality are essential. By allowing the designers to express their preferences inside the optimization process, MOIA can approach *acceptimality* (acceptability and optimality). Developing decision support systems based on *acceptimality* can attract more designers as it can increase the acceptability of the solutions.

Exploring popular tools among architects by taking MOIA into consideration is crucial. Based on this exploration, four different tools typology that can help the models of MOIA were defined. The first typology is observation tools. These tools are essential for the Observation model as they can assess the behavior of the candidate solutions. The second typology is parametric modeling tools. These tools allow us to define a model with constraints. The constraints connect the model parts. Thus, we can change the whole model by changing part of it. The third typology is algorithmic modeling. This can be regarded as advanced parametric modeling. In this typology, we define a parametric model by using algorithms. Because most of the designers are not trained to use textual programming, many of the algorithmic modeling tools for designers use visual programming, which is relatively easy to use and still provides high capabilities. The fourth typology is a generative design tool. These tools usually manipulate the parametric model to generate solutions. In generative design, we connected the generative tools to a parametric model, observation tools, and optimization algorithms. From this exploration, we can conclude that the systems based on generative design are the closest to MOIA.

Exploring the contemporary generative design workflows is vital as it allows for a better understanding of the tools. Six different generative design workflows were extensively explored. While some of these workflows are based on one tool, some others consist of using multiple tools. Based on this exploration, four different workflows were defined as the closest to MOIA. These workflows share a similar framework. They start by proposing a set of candidate solutions. Then, they observe the behavior of the candidate solutions. Next, they use Pareto's function to classify the solutions. Finally, they use an evolutionary algorithm to evolve a new set of optimized solutions. However, these workflows can approach optimality but not acceptimality.

The dissertation intends to propose a set of recommendations that can increase the interaction between the designers and the machines. By developing a decision support system based on these recommendations, we can approach *accptimality*. It is expected that these systems can attract more designers to adopt the profitable behavior of optimization. The research adopts an experimental approach to investigate designers' acceptability. Five different experiments were performed. The first two experiments focus on comparing designers' acceptability of different generative design tools and workflows by using designers' judgment as a reference. The other three experiments focus on the acceptability of the solutions. It investigates three different aggregation functions by using designers' judgment as a benchmark.

The first experiment compares users' acceptability of two different generative design optimization workflows. The first workflow is based on using visual programming. The workflow observes the candidate solutions (random values are used for the first iteration), then it uses an evolutionary algorithm to evolve a new set of candidate solutions based on classifying the candidate solutions by Pareto's function. The generative tool of this workflow graphically represents the solutions based on Pareto's function classification. The second workflow is based on using EcoGen©, a non-visual programming tool for generating optimized modular forms. This tool uses an interactive evolutionary algorithm. It allows the user to select candidate solutions based on their subjective judgment. The results show that, in general, the designers' acceptability is leaning toward the workflow that uses visual programming. However, when it comes to the interface, the participants prefer EcoGen©, which represents the solutions massing models in a grid, and it allows them to select from these solutions.

To complement the first experiment, the second experiment focuses on visual programming. This experiment compares two different generative tools for visual programming. The first tool is based on graphically representing the solutions based on Pareto's function, which was used in the previous test. The other is based on using an interactive evolutionary algorithm with an interface similar to EcoGen©. The results show that the tool that uses an interactive algorithm is more acceptable than the other tool. This result is the opposite of the previous test. Thus, in the first test, the participants underestimated the interactive algorithm because it does not use visual programming. However, the participants of the experiment think the tool that presents the Pareto's' function can produce better results. In conclusion, combining both approaches in visual programming can increase the acceptability of the generative optimization tools.

The third experiment investigates the two aggregation functions that can alternate Pareto's function. For the first function, which is Derringer & Suich's function, we need to test if the concept of assigning weights appeals to the designers' and if they are capable of assigning consistent weights. In this test, the participants were asked to assign weights for five objectives in four scenarios. The answers showed a wide variety of weights between the different scenarios and the objective. This reflects the high need for weighting the objectives. After, the participants were asked to assign weights for five objectives in pairwise for one scenario to evaluate their consistency. The results show that, in general, the designers' were inconsistent. For the second function, Maximin, we need to test if its principle appeals to the designers. First, a set of ten forms analyzed based on two contrary objectives is presented. The designers have to select one form that optimizes the objectives. Then, a graphical representation of the solutions based on Pareto's' function is presented, and they have to select one form again. The majority of the participants changed their selection. They shifted their selection to match Maximin classification.

The fourth experiment aims to compare Pareto's function and Maximin function by using the designers' judgment as a benchmark. The experiment also observes the influence of information availability on designers' judgment. Six tests were performed in this experiment. In each test, the participants must classify a set of seven solutions to satisfy two objectives. The first three tests use one set of design solutions. The other test uses another set. Each of the three tests for each set of solutions provides the participants with different information. The first provide them with a scatterplot that represents the performance of the solutions graphically based on the objectives. The second present numerical information that describes the designs and its context, in addition to the scatterplot. The third presents all the previous information, plus analyzed massing models that represent the solutions and their performance based on the objectives. The results show that Maximin can result in a classification that is more acceptable by the designers' if compared to the classification derived from Pareto's function. It also shows that the additional information other than the scatterplot did not significantly influence designers' judgment.

The set of solutions used in the previous experiment result in a similar classification if we used Maximin or Derringer & Suich's (equal weights). The fifth experiment aims to compare Maximin and Derringer & Suich's (equal weights) functions by using designers' judgment as a benchmark. The participants of this experiment were asked to classify two different sets of solutions. The classification must satisfy two objectives for the first set and five objectives for the second set. For the first set, the numerical values of the satisfaction level ranging between zero and one were presented. Also, a scatterplot and a parallel coordinate chart that represents the performance of the solutions were presented. Additionally, the location and type of building were specified. For the second set, only the numerical values that represent the satisfaction level of the objectives and a parallel coordinate that represent the performance of the solution were presented. It seems from the results that increasing the number of objectives encourages the designers to compensate. The deviation between the classification of the participants and Derringer & Suich's (equal weights) was lower than the deviation between the classification of Maximin and the participants. However, when only two objectives were involved, and the scatterplot was presented, the deviation between the classification of the participants' and Maximin was lower than the deviation between the classification of Derringer & Suich's (equal weights) and the participants.

Based on these five experiments, a set of four recommendations that can increase the interaction between the designer and the computer in the design decision support systems are proposed. At first, using a generative design based on visual programming is highly recommended. Secondly, using an interactive algorithm can increase the acceptability of the tools as it increases the interaction with the users. Third, the visual aspect of the interface is critical. This includes presenting the massing models of the solution and the graphical representation of the performance of the solutions (e.x. scatterplot or parallel coordinate). Finally, the systems must be able to use different aggregation functions based on information availability. If the necessary information to use a desirability function is available, and objectives weights are decidable, we can use Derringer & Suich's function. If the weights of the objectives are non-decidable, but the necessary information to use a desirability function is available, we can use Maximin or Derringer & Suich's (equal weights). In the case of information scarcity, we can use Pareto’s function, and in contrast to the two other functions, we cannot achieve *acceptimality*. In the future, we can link the systems that adopt these recommendations to machine learning.

This thesis has been prepared at:

Université de Bordeaux
L'Institut de Mécanique et Ingénierie de Bordeaux, Département TREFLE
(UMR CNRS 5295)

ENSAM
Esplanade des Arts et Métiers
33405 Talence Cedex

T + (33) (0) 5 56 84 54 19
https://www.i2m.u-bordeaux.fr/

École Nationale Supérieure d'Architecture et de Paysage de Bordeaux
Groupe Recherche Environnement, Confort, Conception Architecturale et Urbaine
(EA MCC 7482)

ENSAP Bordeaux
740 Cours de la Libération - CS 70109
33405 Talence Cedex

T + (33) (0) 5 57 35 11 00
http://www.bordeaux.archi.fr/

École Nationale Supérieure d'Architecture de Lyon
Laboratoire MAP-Aria
(UMR CNRS-MCC 3495)

École Nationale Supérieure d'Architecture de Lyon
3 rue Maurice-Audin
BP 170 F
69512 VAULX-EN-VELIN CEDEX

T +(33) (0) 4 78 79 50 85/86/87/88
http://www.lyon.archi.fr/

Dedication

To my father, who taught me the importance of knowledge. To my mother, who taught me the importance of perseverance. To my wife, Rotaila, my source of inspiration. To my son, Amr, my ray of sunshine. To my siblings, Hosam and Ghada, my comrades in growing up.

Acknowledgments

Throughout this research journey, I have received plenty of support and assistance that I want to acknowledge.

I want to thank my director, Prof. Patrick Sebastian, for being patient, informative, and understanding. His guidance was instrumental in defining the direction of this research. I would also like to thank my co-director, Prof. Denis Bruneau, for helping me to overcome many administrative obstacles. He was always available to answer my questions and concerns. Also, I would like to thank my co-supervisor, Dr. Aline Barlet, for her significant role in the scientific, administrative, and personal levels. She played a significant role in structuring the research and preparing the experiments. I also want to thank my co-supervisor, Dr. Xavier Marsault, for his great effort during the different stages of this research. He provided the research with significant input.

I would also like to thank the members of GRECCAU for being welcoming and supportive. Specifically, I would like to acknwoledge Prof. Catherine Semidor, Dr. Ferran Yusta Garcia, and Dr. Ahmed Khan for their essential role in connecting me to the research team. Additionally, I want to acknowledge my collegue, Arch. Assia Talhi, for her help during my study. Likewise, I would like to thank the members of I2M, Treffle department. As well, I want to thank the members of MAP-Aria for their scientific input and for their hospitality during my visits to their laboratory. Here, I want to thank Arch. Florent Torres for the great scientific discussions. Similarly, I want to thank Nobatek for supporting the research. In particular, I would like to acknowledge Dr.Antoine Dugue, Arch.Ilona Darmon and Ing.Tessa Hubert for the great scientific discussions.

I would like to thank the government of Saudi Arabia represented by the ministry of education and King Abdulaziz University for funding my studies. Also, I want to acknowledge the great effort of the administration and the Information technology team in EnsapBx for helping me in using the different facilities. Specifically, I want to acknowledge Mr.Sylvain Barlet for his technical support in programming. Additionally, I want to thank Baitykool team for allowing me to join the marvelous experience of the Solar Decathlon competition; It was a great opportunity to test and develop a significant part of this dissertation. I also want to thank the participants of the experiments performed in this research; without their participation, it was not possible to complete this work.

Finally, I want to thank my wife for being supportive, understanding, and reliable. Likewise, I want to thank my son for giving me the energy to complete this work. I would also like to thank my parents, siblings, mother in law, father in law, brother in law, as well as my nieces and nephew for their great support and love. Similarly, I want to thank my family and friends for their encouragement.

Contents

List of figures

List of tables

List of equations

List of symbols and abbreviations

AI	Artificial Intelligence
IA	Intelligence Augmentation
FBS	Function, Behaviour, Structure
OIA	Observation, Interpretation, Aggregation
$\mathbf{x}$	Design variables
$\mathbf{y}$	Observation variables
$\mathbf{z}$	Interpretation variables
μ	Observation model
δ	Interpretation model
ζ	Aggregation model
DOI	Design Objective Indices
GDI	Global Desirability Index
MOIA	Morphogenesis, Observation, Interpretation, Aggregation design framework
Ω	Design space
y_i	Observation variables value
r_i	Observation variables value rank
z_i	Interpretation variables value
SL	Soft Limit
AC	Absolute Constraint
d^H	Harrington's Desirability
AC_L	Lower Absolute Constraint
SL_L	Lower Soft Limit
SL_U	Upper Soft Limit
AC_U	Absolute Constraint
d^D	Derringer & Suich's Desirability
U	Upper bound
L	Lower bound
T	Targeted value
d^{arctan}	Arctan Desirability
ω_i	Objective/Criteria weight
GOA	Global Optimization Algorithm
NIA	Nature-Inspired Algorithm
EA	Evolutionary Algorithm
GA	Genetic Algorithm
EGA	Epigenetic Algorithm
PSO	Particle Swarm Optimization
VR	Virtual Reality
NURBS	Non-Uniform Rational B-Spline
AEC	Architecture, Engineering, Construction
BIM	Building Information Modeling
VP	Visual Programming
IEA	Interactive Evolutionary Algorithm
IGA	Interactive Genetic Algorithm
AHP	Analytic Hierarchy Process

Introduction

"We must develop as quickly as possible technologies that make possible a direct connection between brain and computer, so that artificial brains contribute to human intelligence rather than opposing it." (Hawking, 2001)

Since the beginning of humanity, many machines were created to help humans. Tracking the history of these machines can show us how much they affected human's lifestyle. Archeologists found many pieces of evidence of tools created and used by prehistoric hunter-gatherer humans.

The agricultural revolution changed the human model of life. In contrast to a hunter-gatherer life that relays on foraging, the farmers settled on their farms to grow the crops. This revolution led to a surplus of food, allowing part of the community to focus on new professions. Many of these professions introduced new technologies and created many machines that optimized agricultural production and gradually fulfilled human ambitions and needs in all domains of life.

In the 18th century, the industrial revolution began; it was a big turn in machine development. In the beginning, the machines gradually replaced the animals in transportation and then in many hardworking tasks. Eventually, the machines started to replace unskilled laborers pushing the new generation to improve their qualifications. Later, the improvement of the machines led to partially replacing skilled laborers; many professions have disappeared while new professions were introduced.

Eventually, the invention of the computer was a big move that changed the history of the machine. The concept of Artificial Intelligence (AI) and information technology started growing very fast since then, allowing the machine to replace more jobs done by humans. In his book "Deep Thinking", Kasparov said: *"The machines have finally come for the white collared, the college graduates, the decision makers"*(Kasparov & Greengard, 2017). Markoff supports Kasparov's observation; he stated, *"We have centuries of experience with machines such as the backhoe and steam shovel, both of which replace physical labor. Smart machines that displace white-collar workers and intellectual labor, however, are a new phenomenon."*(Markoff, 2015). Some people are afraid of this replacement.

However, another group of people believes that this is a positive change, as it gives humans more time and reasons to upgrade their life. Kasparov explained that *"Machines that replace physical labor have allowed us to focus more on what makes us human: our minds. Intelligent machines will continue that process, taking over the more menial aspects of cognition and elevating our mental lives toward creativity, curiosity, beauty, and joy. These are what truly make us human, not any particular activity or skill like swinging a hammer or even playing chess"*(Kasparov & Greengard, 2017). Indeed, we should understand that we are not in competition versus our machines, we compete and challenge ourselves. We must always remember that we made them, and their success is our success.

Kasparov assigns a chapter of his book that focuses on the collaboration between the human and the machines called *"Human Plus Machine."*(Kasparov & Greengard, 2017). In this chapter, he said, *"As the curtain fell on decade of human versus machine competition, it* *was time for human plus machine collaboration to take center stage. To put it more succinctly, if you can't beat 'em, join 'em…."* and continuing *"The phrase "human plus machine" can apply to any use of technology since early man bashed something with a rock"* (Kasparov & Greengard, 2017). The points of strength and weaknesses between humans and machines are different *"it was easier to build a robot to go to the moon than to build one that could drive by itself in rush-hour traffic."*(Markoff, 2015). The human can benefit from the machines, while the machines can learn from humans. As time goes by, these machines will perform more tasks based on that learning, allowing humans to explore new things.

Today, many companies have started to focus on Intelligence Augmentation (IA), which was introduced by Engelbart (Engelbart, 1962). For example, *"The Toyota shift toward a more cooperative relationship between human and robot might alternatively suggest a new focus on technology for augmenting humans rather than displacing them."*(Markoff, 2015). In contrast to the Artificial Intelligence (AI) which was introduced by McCarthy in 1955 (McCarthy, J., Minsky, M., Rochester, N., Shannon, 1955; Nilsson, 2010), which is an independent technology, IA aims to augment the human's intelligence toward improving human's decision-making capability.

After decades of testing and comparing decision-making capabilities of both humans and machines Kasparov made a conclusion called "Kasparov's Law" which states *"weak human + machine + better process was superior to a strong computer alone and, more remarkably, superior to a strong human + machine + inferior process"*(Kasparov & Greengard, 2017). Developing a better process that focuses on the integration between the machine and humans is highly recommended to improve decision-making. The process is a primary key to consider for developing decision support systems.

Research problem

As Churchill once said, *"We shape our buildings; thereafter they shape us."*(Churchill, 1943). There are no doubts that the design of our buildings have a significant impact on us, our resources, and the environment. The International Energy Agency highlights that *"the global buildings sector is responsible for 30% of final energy consumption and more than 55% of global electricity demand. Progress towards sustainable buildings is advancing, but improvements are still not keeping up with a growing buildings sector and rising demand for energy services. The buildings and buildings construction sectors combined are responsible for 36% of global final energy consumption and nearly 40% of total direct and indirect* CO_2 *emissions. Energy demand from buildings and buildings construction continues to rise, driven by improved access to energy in developing countries, greater ownership and use of energy-consuming devices, and rapid growth in global buildings floor area, at nearly 3% per year. This growth overwhelms the improvements in global buildings final energy intensity per unit of floor area, which has only fallen by 1.3% per year. As a result, in recent years the global use of energy in buildings has grown by 1% per year, the global use of electricity in buildings has grown by 2.5% per year, and the global buildings-related* CO_2 *emissions continue to rise by nearly 1% per year."* (IEA, 2017). All these facts made it imperative to improve the performance of our buildings. At the same time, we must maintain users' comfort and the functionality of the design.

Designing a building is a complex task to accomplish with multiple dimensions and different objectives. Hence, it is essential to provide the designers with a robust decision support system. Implementing an optimization approach in the architectural design process can significantly improve design outcomes.

Today, many tools and workflows are developed to help the designers in optimizing their designs. However, most of these tools are only able to evaluate the solutions made by the designers without suggesting any reliable solutions or improvement that optimizes the design; Lawson highlights *"Modern Building science techniques have generally only provided methods of predicting how well a design solution will work. They are simply tools of evaluation and give no help at all with synthesis. Daylight protractors, heat loss or solar gain calculations do not tell the architect how to design the window but simply how to assess the performance of an already designed window."*(Lawson, 2005). Only a few tools that can help to generate optimized solutions exist; these tools are relatively new and not mature enough.

The existed generative design optimization digital tools for designers are still not widely accepted and consequently not commonly used among the architects. Howarth states, *"There's a risk that designers will feel threatened by removing a large part of the creative process from their work – simply left to choose between a set of options each time."*(Howarth, 2017). These tools are not engaging the designer in the optimization process. The weak or unbalanced collaboration between the designer and the computer in design optimization generative tools is a real problem that results in designers' reluctance to use these workflows.

Research objectives

Users' acceptability is a significant factor in the success of almost any product or service. Since the computer was introduced, software developers were aware of this importance. There is a need for developing a new model of design optimization generative workflow based on the designer's acceptability to attract more architects willing to optimize their designs. This research focuses on two different aspects of acceptability, one related to the tools and the other to the solutions. The acceptability of the tools refers to the workflow, user interface, and computation time. The acceptability of the solutions refers to the results and, therefore, to the design solutions on their own.

The lack of balanced interaction between the designers and the computer in the existing design optimization generative tools, and thus the workflows, motivated this research. In order to overcome this problem, the dissertation aims to help software developers produce a new generation of tools that attract more building designers to the world of optimization by increasing the acceptability of the optimized design solutions generated by the digital workflows.

The research must provide software developers and designers with a set of recommendations that can increase the acceptability of the optimized solutions. These recommendations must include an algorithm that helps the designers to produce acceptable optimized solutions taking into account the designer's logic. Three objectives are defined to guide the research:

- Investigating literature related to design processes, design optimization, and existing relative tools and workflows to improve the understanding of the existing design optimization systems.

- Define a reliable design framework that can integrate the designers' preferences inside the process of optimization to produce acceptable optimized solutions.
- Provide the software developers with a set of recommendations that can increase designers' acceptability by providing a balanced relationship between the designers and the tools in early stages of the design process (conceptual design); *"In the early stages of the design process, decisions taken can impact up to 70% of the life-cycle costs"* (Quirante, Sebastian, & Ledoux, 2013; Zablit & Zimmer, 2001).

Research methodology

Behind each successful software, usually, there are a vast number of programmers, scientists, and researchers working to improve users' acceptability. Investigating user's acceptability is essential during and before the software development. It helps to outline the model of the software to match users' preferences based on their feedback. *"It would be highly beneficial if information systems developers could verify requirements by predicting workplace acceptance of a new system based on user evaluations of its specifications measured during the earliest stages of the development project, ideally before building a working prototype."*(Davis & Venkatesh, 2004).

The research approach is mainly experimental based on reviewing the literature of design process, optimization methods, tools available on the market, and different workflows based on these tools and methods. Two methods are proposed to investigate the user's acceptability. Each method consists of a series of experiments.

The first method focuses on the acceptability of design optimization generative tools. The method tests and compares different design optimization generative workflows by inviting a panel of experts to try these tools and then collecting their feedback. The second method focuses on the acceptability of the solutions by testing different aggregation functions on a panel of experts. The aggregation functions are of great importance in the field of optimization. They classify the solutions based on their satisfaction with the design objectives. The results of these methods are then used to describe a new framework able to supervise the definition of acceptable design optimization generative tools for architects.

Research significance

The research develops an original ontological framework of design optimization tools for architects. An ontology is *"A set of concepts and categories in a subject area or domain that shows their properties and the relations between them."* (Lexico, 2020b), and a framework is *"A basic structure underlying a system, concept, or text"* (Lexico, 2020a). An ontological framework is a basic structure underlying a system or concept that defines the properties of its parts or subconcepts and relations between them.

The proposed framework and the applied methodology defines a set of recommendations. The recommendations improve the interaction between the designers and optimization tools, which likely increases designers' acceptability of optimization tools. Consequently, it is expected that more architects will adopt optimization in their work. Using optimization tools are incredibly profitable, as design optimization can decrease the negative impact on the environment and enhance the economic efficiency of the different structures. The computer-based optimization tools can enhance human capabilities.

Overview of dissertation

The dissertation consists of seven chapters. Figure 1 explains the structure of the research. In the following, a brief of each chapter is presented:

- Chapter 1 (Design Process): This chapter focuses on the design process by studying its principles. Then, it investigates the different activities that make it up by reviewing different models.
- Chapter 2 (Optimality and acceptability): This chapter introduces an ontological framework for design optimization. Many approaches for the framework models are investigated and criticized. Finally, the chapter explains the concepts of optimality and acceptability and why they are essential in design.
- Chapter 3 (Software typologies): This chapter explores the different software typologies in the market that commonly used by architects. This chapter helps to determine the typologies that are more suitable for generative design optimization.
- Chapter 4 (Decision support workflows): In this chapter, different design optimization generative workflows are reviewed.
- Chapter 5 (Tools and workflows): This chapter investigates designers' acceptability of different tools and workflows. The adopted methodology consists of different experiments that compare different workflows and use a variety of software.
- Chapter 6 (Aggregation for *acceptimality*): This chapter investigates designers' acceptability of the solutions. The adopted methodology consists of different experiments that test different aggregation functions.
- Chapter 7 (Conclusion): This chapter provides the designers and the developer with a list of recommendations. The proposed recommendations aim to improve designers' acceptability generative design optimization tools. Also, it concludes the main ideas and the results of the dissertation. Additionally, it defines the planned future work.

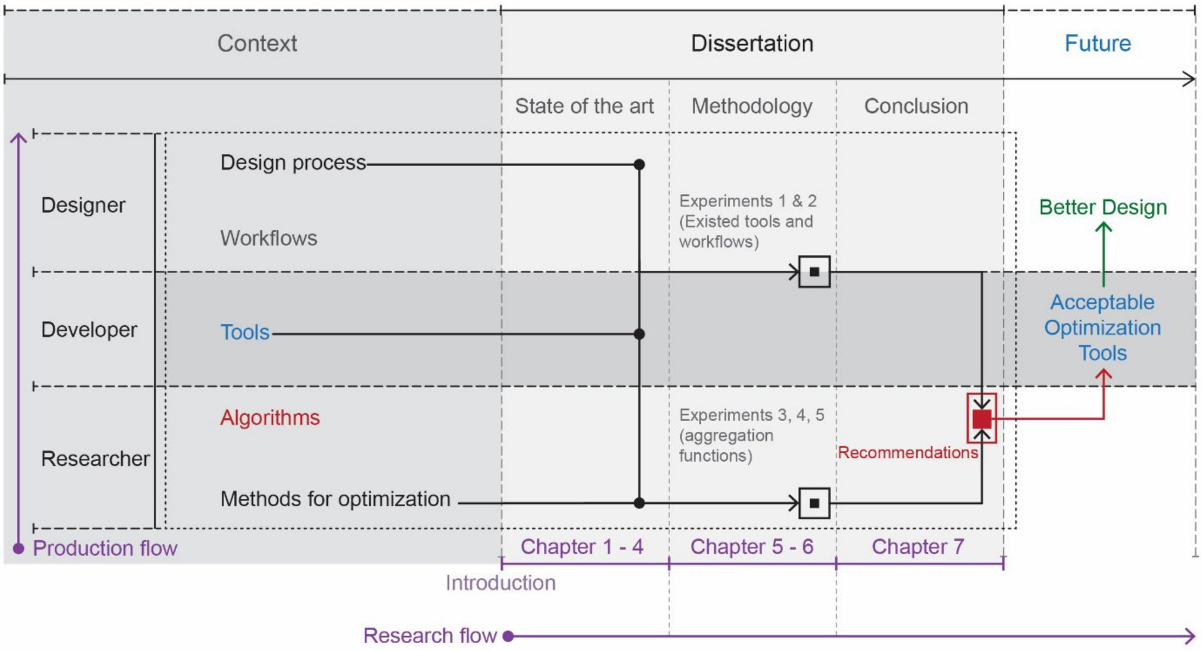


Figure 1: *The structure of the research*

CHAPTER 1 Design Process

"A profound design process eventually makes the patron, the architect, and every occasional visitor in the building a slightly better human being" Juhani Pallasmaa, 1936 (S.Dushkes, 2012).

In some contexts, the term design can represent the actions of design. In other situations, it can represent the result of the design actions (the design itself). Ching defines design as *"To conceive, contrive, or devise the form and structure of a building or other construction."*(Ching, 1995) This definition shows that together, design as actions and, as results, explain the word design; *"To conceive, contrive, devise"* represents the actions of design. The rest represents the results. Gero and Kannengiesser clarify *"the term "designing" is used to signify the act and the term "design" is used to signify the result of designing."*(J. S. Gero & Kannengiesser, 2014)

The term "design process" refers to the design as actions; it explains how we design. According to Ching, the design process is *"A purposeful activity aimed at devising a plan for changing an existing situation into a future preferred state."*(Ching, 1995) This definition is a detailed version of the part of Ching's definition of design that represents design as actions. It is essential to understand that *"Design is fundamentally about problem-solving"*(Makstutis, 2018). In design, we solve problems by using a design process.

"At an off-site for Apple Computer's Creative Services department, Tim Brennan began a presentation of his group's work by showing this model. "Here's how we work", he said, "Somebody calls up with a project; we do some stuff; and the money follows." (see Figure 2) (Dubberly, 2004). Brennan's model is a good example that can represent that *"Creativity can appear mysterious to the uninitiated, and is not easily explained or taught"*(Smith, Albert C, Smith, 2015). Design activities involve high creativity.

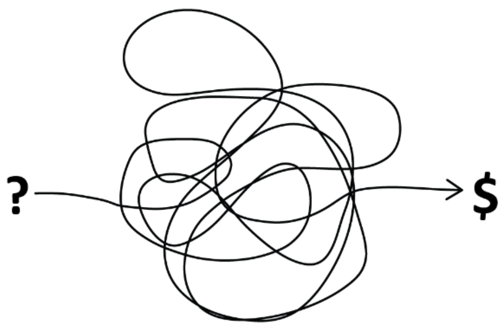


Figure 2: *Brennan's working diagram* (Dubberly, 2004)

Niemeyer (1907-2012) describe, *"I pick up my pen. It flows. A building appears. There it is. There is nothing more to say."* (Niemeyer, n.d.) While the previous phrase might appear ambiguous, especially for non-designers. It can be much clear if we consider it as a process; *"I pick up my pen"* represents design inputs, this also can include all the previous activities which include gathering the required data, *"It flows"* represents design process, *"A building appears"* represents design output. The same logic can interpret Brennan's model; *"Somebody calls up* *with a project"* represents design input, *"we do some stuff"* represents design process, *"and the money follows."* represents design output.

Whether it is the diagram of Brennan or the quote of Niemeyer, it is essential to clarify the design process. *"because of its non-linear process, design can be considered ambiguous. Assignments will be ambiguous. Advice will be ambiguous. Professors will be ambiguous. While a designer can use ambiguity to help with the development of conceptual ideas, design also implies a certain precision, so that it is not removed from the realities of life. In other words, it is your responsibility to engage this ambiguity and make the project clear"*(Smith, Albert C, Smith, 2015)

According to Ching, a process is *"A systematic series of actions or operations leading or directed to a particular end."*(Ching, 1995) the design process uses inputs and produces output. Figure 3, which is inspired by Brennan's model, adapts it to the previous analysis vocabularies. Figure 4, concludes the previous arguments and can be used as a foundation that we can proceed from to build a clear understanding of the design process.

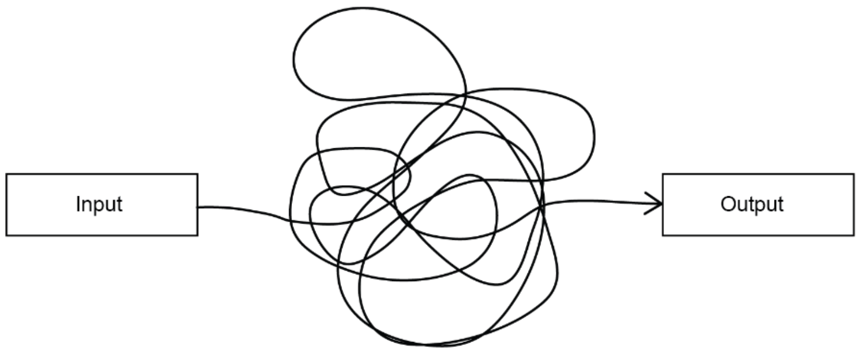


Figure 3: *Input and output in the design*

Figure 4: *Design process, input, and output* (Dubberly, 2004)

This chapter investigates the design process at two different levels. Initially, it discusses its principles. Next, it discusses its different activities. Understanding the design process is essential as it explains the broad context of the research.

1.1 Design Process principles

In order to understand the design process, it is essential to outline its principles. Lawson (Lawson, 2005) defined six principles that describe the attributes of any design process. This part intends to present and discuss these principles.

1.1.1 *"The design process is endless":*

"Since design problem defy comprehensive description and offer an inexhaustible number of solutions the design process cannot have a finite and identifiable end. The designer's job is never really done and it is probably always possible to do better. In this sense designing is quite unlike puzzling. The solver of puzzles such as crosswords or mathematical problems can often recognize a correct answer and knows when the task is complete, but not so the designer. Identifying the end of design process requires experience and judgment." (Lawson, 2005). This argument illustrates how design is an endless process. However, the definition of the term process suggests that any process has a particular end (Ching, 1995). The only way to interpret this conflict is to admit that the design process is iterative; it is a process that repeats itself until the result is satisfying, which defines the end (see Figure 5).

Many designers support this argument. Parker asserts, *Design is an iterative process. One idea often builds on another."* (Parker, n.d.) This explains not only that the design is an iterative process, but also it is evolutionary. In his book "Designing Architecture: the elements of the process" , Pressman supports Parker's assertation, *"Forced to find the objective in process, I would characterize it as iterative, requiring successive loops, each of which produces more information and resolution than the previous one."*(Pressman, 2012). Jabi also agreed that the design process is iterative; he says: *"The architectural design process is almost always iterative."* (Jabi, 2013)

Makstutis believes *"perhaps the most important characteristic of design is that it is iterative. This means that design does not happen once, and then we move on to something else. Rather design is a cyclical activity that takes place again and again."*(Makstutis, 2018). He also points out that *"The iterative nature of design means that it can be used to revise and improve the outcome. If design in architecture happened only once, there would be no opportunity to adjust and enhance a proposition before construction began. Without the iterative process, our world would be much less enjoyable and less efficient."*(Makstutis, 2018). Smith and Smith explain, *"Design concerns imagining the future and visualizing things that have not been seen before. It is a non-linear process, as it tends to move forward and backward between topics."*(Smith, Albert C, Smith, 2015)

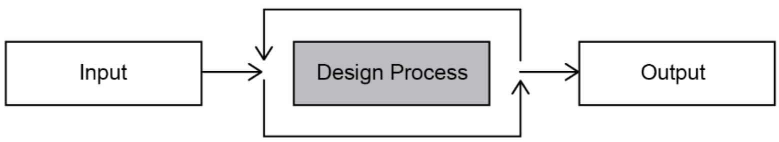


Figure 5: *Iterative design process*

In the design process, the word end is not exact. It is a decision that involves subjective judgment influenced by many factors. *"It no longer seems worth the effort of going further because the chances of significantly improving on the solution seem small. This does not mean that the designer is necessarily pleased with the solution, but perhaps unsatisfactory as it might be it represents the best that can be done. Time, money and information are often major limiting factors in design and a shortage of any of these essential resources can result in what the designer may feel to be frustratingly early end to the design process"*(Lawson, 2005). To end the iteration of a design process, we can define a threshold.

1.1.2 *"There is no infallibly correct process":*

Lawson affirms, *"there is no absolute correct process"*(Lawson, 2005). He states, *"Much though some early writers on design methodology may have wished it, there is no infallibly good way of designing."* (Lawson, 2005) Makstutis states that *"Every architect or designer will have a different way of generating ideas, but the stages of the design process generally follow a similar pattern"*(Makstutis, 2018). He affirms, *"There is no right way to design. There is no single process that will lead to a successful project. Each individual, and each team, involved in a project will have a different way of working, a different way of designing"* (Makstutis, 2018) Hence, we can infer that in design we need a flexible model that can adapt to different problems *"Controlling and varying the design process is one of the most important skills a designer must develop."*(Lawson, 2005)

However, Gero and Kannengiesser propose the following axiom *"The foundations of designing are independent of the designer, their situation and what is being designed."*(J. S. Gero & Kannengiesser, 2014). Based on this axiom, they also proposed two hypotheses about representing design and designing, which says, *"All the designs could be represented in a uniform way, and all designing could be represented in a uniform way."*(J. S. Gero & Kannengiesser, 2014). The design framework (design process model) must be broad and flexible to adapt to different contexts. Based on one framework, we can define many design processes.

1.1.3 *"The process involves finding as well as solving problems":*

The design process links the problem and the solution through the analysis and the synthesis. *"It is central to modern thinking about design that problems and solutions are seen as emerging together, rather than one following logically upon the other."*(Lawson, 2005). According to Alexander, *"The form is the solution to the problem; the context defines the problem."*(Alexander, 1964). Understanding the relationship between the problem and the solution is essential. Piotrowski explains, *"Your work as an interior designer involves defining and analyzing the problem regardless of its scope. Complex projects naturally involve a greater list of tasks. Your design work also involves solving the problem by determining a course of action to complete the project. In design problem solving, these two critical activities are called analysis and synthesis."*(Piotrowski, 2011).

Lawson notes, *"often the problem may not even be fully understood without some acceptable solution to illustrate it. In fact, clients often find it easier to describe their problems by referring to existing solutions which they know of"* (Lawson, 2005). Analyzing the problem and the solution together helps to redefine the problem, the solution, or both.

The designers typically have no or limited control over the context. However, the designers' capabilities to understand and analyze the context are crucial. On the other hand, the designers have high control over the form. The context adds restrictions on the form and thus the designers *"The context is that part of the world which puts demands on this form; anything in the world that makes demands of the form is context. Fitness is a relation of mutual acceptability between these two. In a problem of design we want to satisfy the mutual demands which the two make on one another. We want to put the context and the form into effortless contact or frictionless coexistence."*(Alexander, 1964). The relation between the problem and the solution is commingled, the iterative nature of the design process increases this commingling. The design process model should be flexible to work with different design problems, which are defined by different contexts to find design solutions.

1.1.4 *"Design inevitably involves subjective value judgment":*

The design often tackles many problems at the same time. However, *"Questions about which are the most important problems, and which solutions most successfully resolve those problems are often value-laden. Answers to such questions, which designers must give, are therefore frequently subjective."*(Lawson, 2005). Subjective judgments are always part of the design. Makstutis asserts, *"The notion of 'good design' is very subjective: what one person thinks of as 'good,' another may find disappointing. For this reason, rather than considering how design maybe 'good,' we may consider how design creates 'value.'"*(Makstutis, 2018).

However, *"Unlike the artist, the designer is not free to concentrate exclusively on those issues which seem most interesting. Clearly one of the central skills in design is the ability rapidly to become fascinated by problems previously unheard of."*(Lawson, 2005) In contrast to pure art, which is mainly subjective, the design includes objectivity. The design process is combinatorial; it involves subjective judgment and objective modeling.

1.1.5 *"Design is a prescriptive activity":*

In design, we can prescribe many design solutions that solve one problem. *"One of the popular models for the design process to be found in the literature on design methodology is that of scientific method. Problems of science however do not fit the description of design problems outlined above and, consequently, the process of science and design cannot usefully be considered as analogous. The most important, obvious and fundamental difference is that design is essentially prescriptive whereas science is predominantly descriptive. Designers do not aim to deal with questions of what is, how and why but, rather, with what might be, could be, and should be. While scientists may help us to understand the present and predict the future, designers may be seen to prescribe and create the future, and thus their process deserves not just ethical but also moral scrutiny"*(Lawson, 2005)

Gero and Kannengiesser find that *"Design appeared to present problems for scientific research in that the results of the acts of designing were always unique and therefore there would be no regularity."*(J. S. Gero & Kannengiesser, 2014) In contrast to scientific methodologies, the design process is uncertain; thus, not descriptive; instead, it is uncertain and prescriptive.

1.1.6 *"Designers work in the context of a need for action":*

In design, decision-making is central. *"Design is not an end itself. The whole point of the design process is that it will result in some action to change the environment in some way, whether by the formulation of policies or the construction of buildings. Decisions cannot be avoided or even delayed without the likelihood of unfortunate consequences."*(Lawson, 2005)

In design, important decisions must be taken during a relatively short time. *"Not only must designers face up all the problems which emerge they must also do so in limited time."*(Lawson, 2005) Time is a significant limitation for the designer, especially if we consider the responsibilities that face architects and the enormous consequences of the design decisions. *"Architects are charged with great responsibility. The work they undertake leads to results that are complex and expensive. Even a small project may require the coordination of many different people, materials and processes, and take a long time. A large building project may take years to complete. The amount of money required may run into many millions (or even billions). Small projects (such as houses or residential extensions) may cost less but their importance, for the clients, will be of the highest order. For these reasons, it is impossible for a builder or team of contractors simply to start work without some kind of plan that provides a clear direction for the project and the various people involved."*(Makstutis, 2018)

The efficient design process usually integrates decision support systems; the design process model should have the ability to adopt these systems. *"Design is often a matter of compromise decisions made on the basis of inadequate information. Unfortunately for the designer such decisions often appear in concrete form for all to see and few critics are likely to excuse mistakes or failures on the grounds of insufficient information. Designers, unlike scientists, do not seem to have the right to be wrong. While we accept that disproved theory may have helped science to advance, we rarely acknowledge the similar contribution made by mistake design."* (Lawson, 2005)

1.2 Design Process activities

The design process consists of activities usually ruled by these principles (see 1.1). These activities are the steps of the design process. This section intends to investigate the activities of the design process by exploring different design process models. From understanding these models, it is possible to develop essential knowledge of the design process.

1.2.1 Koberg and Bagnall model

In 1976 Koberg and Bagnall compared many approaches for solving different problems (Koberg, Don, Bagnall, 1972). The aim was to find a basic abstraction or common dominators. They found two primary stages are necessary, analysis and synthesis (Dubberly, 2004). Applying their conclusion to the previous model (see Figure 5) (1.1.1) results in the process presented in Figure 6. This model is fundamental for problem-solving in general. However, we need a more specific model that adapts this model to real design problems.

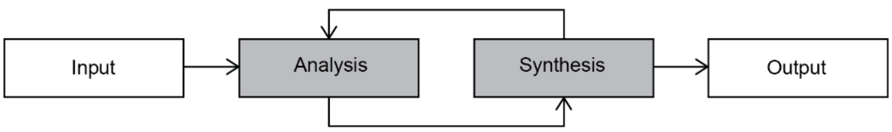


***Figure 6:** A design process model based on Koberg and Bagnall* (Dubberly, 2004; Koberg, Don, Bagnall, 1972)

1.2.2 Lawson's model

Lawson presented a template to explain the design process (see Figure 7). In his version, the design is about solving problems. Consequently, his model is similar to the model of Koberg and Bagnall (Koberg, Don, Bagnall, 1972) (1.2.1). The model uses "problem" and "solution" to describe the input and output. However, the model introduces a new step between "analysis" and "synthesis", which is "evaluation." Figure 8 link lawson's model to the previous model.

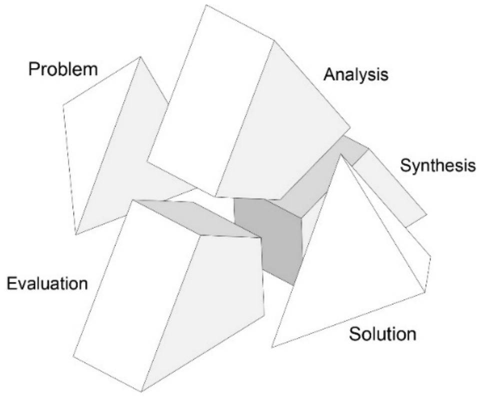


Figure 7: *Lawson's model of the design process* (Lawson, 2005)

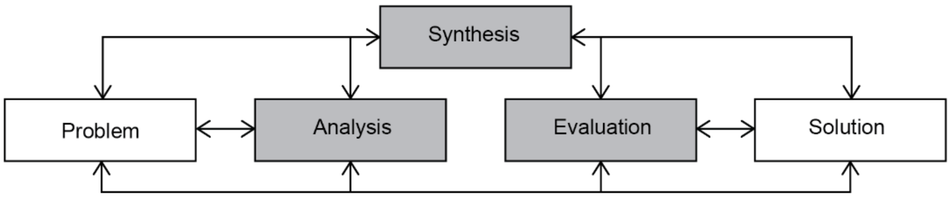


Figure 8: *The design process based according to Lawson* (Lawson, 2005)

According to Lawson, the Design Process demonstrates a negotiation between design problems and design solutions; they reflect each other (Lawson, 2005). He highlights that the analysis, synthesis, and evaluation are involved in this negotiation, but between these three activities, no start or endpoint can be defined, and no direction of flow is specified (Lawson, 2005). The process can start with a problem or a solution. We cannot assume that the process always starts from a problem; sometimes, the problem is determined from an existing solution.

1.2.3 Generative design model

Vernacular architecture evolved over centuries, which may be considered a cumulative optimization process. Lawson wonders, *"How could a few hours or days of effort on the part of a designer place the results of centuries of adaptation and evolution embodied in the vernacular product?"*(Lawson, 2005). This evolutionary quality is crucial for the design. The design optimization process is a series of actions that aim to find the best possible design. Generative design, based on the coupling of evaluation and evolution, it is widely used in the domain of design. Based on this evaluation, the process then evolves towards better solutions. Generative design is fundamental for design optimization.

"Various generative form-finding techniques existed in architecture long before the digital revolution. At the start of the twentieth century, many visionary architects, engineers and designers, such as Frederick Kiesler and Frei Otto, were applying design methods that were very similar to today's new computational approach."(Agkathidis, 2015). While the generative design is not necessarily based on digital computational design, many designers link them. Because this digital approach makes the process efficient and effective, nowadays, almost every designer who adopts the generative design uses this computational approach.

Based on this digital approach, Walmsley and Villaggi define, *"Generative Design is a framework for combining digital computation and human creativity to achieve results that would not otherwise be possible."*(Villaggi & Walmsley, 2018). They described a design process model based on the generative design approach that starts with gathering data and ends by selecting the solution (see Figure 9) (Villaggi & Walmsley, 2018). Between the start and the end, they described an iterative process. This iterative process begins by generating solutions, then, evaluating them. After, the process evolves towards new design solutions. The new generations resulting from this evolution should result in better values in the evaluation. They described this framework as *"a flexible and scalable framework. It can be applied to a wide range of design problems and scales: from industrial components all the way to buildings and cities."*(Villaggi & Walmsley, 2018).

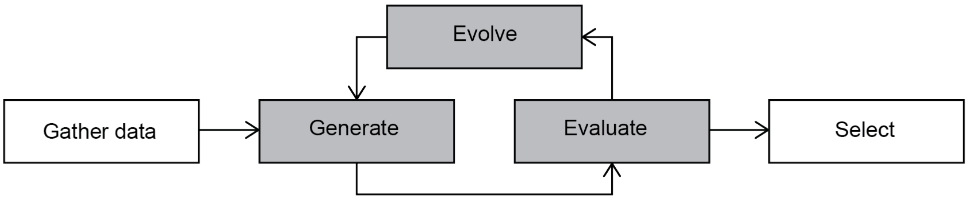


Figure 9: *The generative design model* (Villaggi & Walmsley, 2018)

A simple comparison between this model and Lawson's model (see Figure 8), show that they are somehow related. However, the flow in the generative design model is defined in a way that facilitates digital automation; it always iterates toward one direction.

The first step in the Walmsley and Villaggi iterative process "generate" is related to the design variables, which are values that must be determined at every iteration. The design variables define the information necessary to characterize a design solution. In the second step, "evaluate", we must observe the behavior of the solutions, and, as a result, we get a set of observation variables that describes the performance of the solutions. Each set of observation variables represents the performance of one generated solution characterized by a set of design variables. During the third step, "evolve", the process uses all the available data to evolve a new set of design variables.

1.2.4 Function-Behaviour-Structure (FBS) Ontology

The Function-Behaviour-Structure (FBS) is a design ontology developed between 1984-1986 by Gero and published in a paper (J. Gero, 1990). Before FBS in the 1980s, many approaches based on the division of the design and the way it worked were proposed; *"Structure (S) for the design and Behaviour (B) for how it worked or performed. Many of these approaches used the term Function (F) to mean the intended behaviour of the design and as a consequence conflated Function and Behaviour and failed the no-overlap requirement."*(J. S. Gero & Kannengiesser, 2014). In 1993 the modern idea of an ontology developed by Gruber (Gruber, 1993). The concept of FBS framework becomes an ontology because *"The notion of a foundational framework for the field of design mapped well onto the notion of an ontology since they both referred to the meta-level knowledge of a field."*(J. S. Gero & Kannengiesser, 2014).

The FBS ontology defined as *"a design ontology that describes all designed things, or artefacts, irrespective of the specific discipline of designing. Its three fundamental constructs –*

Function (F), Behaviour (B) and Structure (S)"(J. S. Gero & Kannengiesser, 2014). It is essential to understand these three constructs, *"The Function is the teleology of the artefact"* for example, to provide safety, comfort, and affordability (J. S. Gero & Kannengiesser, 2014). *"Behaviour is defined as the artefact's attributes that can be derived from its structure"* for example, strength & weight, heat absorption, and cost (J. S. Gero & Kannengiesser, 2014). *"Structure is defined as its components and their relationships"* for example, geometrically interconnected walls, floors, roof, windows, doors, pipes, and electrical systems (J. S. Gero & Kannengiesser, 2014). *"The FBS Framework (J. Gero, 1990) is an extension of the FBS ontology to represent the process of designing as a set of transformations between function, behaviour and structure."*(J. S. Gero & Kannengiesser, 2014). In the FBS framework, there are eight fundamental transformations presented in Figure 10 and listed as:

1. Formulation (R → F, and F → Be)
2. Synthesis (Be → S)
3. Analysis (S → Bs)
4. Evaluation (Be ↔ Bs)
5. Documentation (S → D)
6. Reformulation type 1 (S → S')
7. Reformulation type 2 (S → Be)
8. Reformulation type 3 (S → F (via Be))

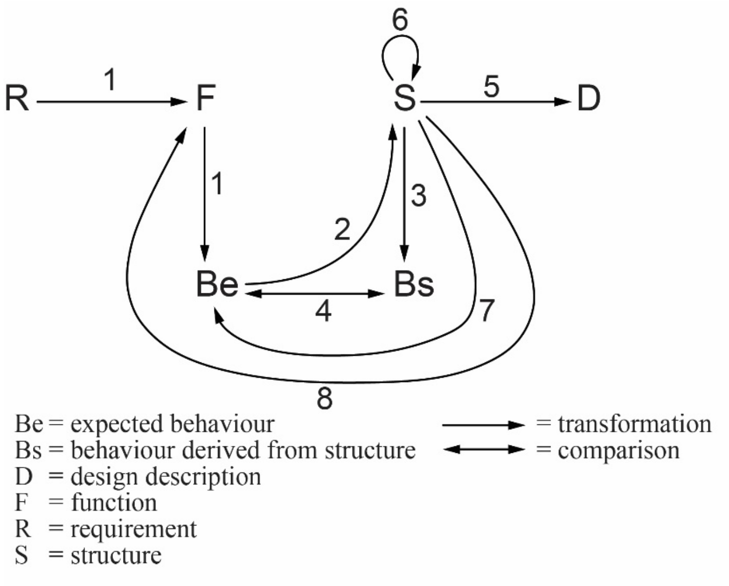


Figure 10: *FBS Ontology framework* (J. S. Gero & Kannengiesser, 2004, 2014)

"The FBS framework represents the beginnings of a theory of designing, through its ability to describe any instance of designing irrespectively of the specific domain of design or the specific methods used."(J. S. Gero & Kannengiesser, 2014). FBS helps to understand the components of the design process models presented earlier in this section. FBS ontology and framework has a strong ability to explain the terminologies used in the design process such as synthesis, analysis, evaluation. Below is a list of definition for the terminologies used in FBS (see Figure 10) defined by Gero and Kannengiesser (J. S. Gero & Kannengiesser, 2004):

- *Formulation (process 1) transforms the design requirements, expressed in function (F), into behaviour (Be) that is expected to enable this function.*

- *Synthesis (process 2) transforms the expected behaviour (Be) into a solution structure (S) that is intended to exhibit this desired behaviour.*
- *Analysis (process 3) derives the 'actual' behaviour (Bs) from the synthesized structure (S).*
- *Evaluation (process 4) compares the behaviour derived from structure (Bs) with the expected behaviour to prepare the decision if the design solution is to be accepted.*
- *Documentation (process 5) produces the design description (D) for constructing or manufacturing the product.*
- *Reformulation type 1 (process 6) addresses changes in the design state space in terms of structure variables or ranges of values for them if the actual behaviour is evaluated to be unsatisfactory.*
- *Reformulation type 2 (process 7) addresses changes in the design state space in terms of behaviour variables or ranges of values for them if the actual behaviour is evaluated to be unsatisfactory.*
- *Reformulation type 3 (process 8) addresses changes in the design state space in terms of function variables or ranges of values for them if the actual behaviour is evaluated to be unsatisfactory.*

1.2.5 Observation, Interpretation, and Aggregation (OIA)

Design optimization seeks to compute the solutions with the most favorable design objectives values. Therefore, it is vital to use a mathematical structure that links the design variables **x**, which characterizes the solutions to the design objectives. Collignan (Collignan, 2011) and Quirante (Quirante, 2012) presented a framework for design optimization based on the combination of three models, which are, Observation model μ, Interpretation model δ, and Aggregation model ζ (OIA). The purpose of OIA is *"to support the decision-making process to guide designer toward the selection of the best design solutions."*(Quirante, 2012). OIA combines all of the design objectives into a Global Desirability Index (GDI), which is then linked to **x** (see Figure 11).

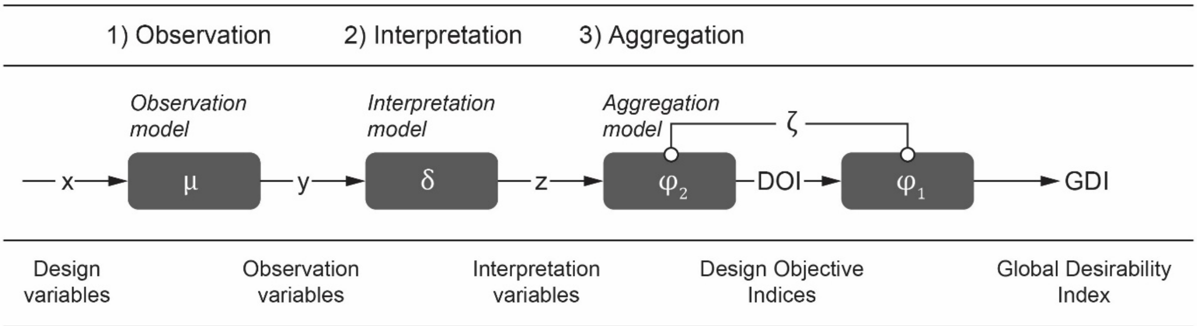


***Figure 11:** OIA model* (Quirante, 2012)

In OIA, the Observation model μ observes the behavior of each candidate solution characterized by a unique set of design variables **x**. The observation results are sets of observation variables **y**. According to Quirante, *"Observation variables **y** are quantitative measures of system effectiveness, performance or technical attributes (mass, cost, efficiency, temperature)."*(Quirante, 2012).

Next, the Interpretation model δ transfers the set of observation variables **y**, which is based on different scales, into a set of interpretation variables **z** *"which can be regarded as individual preferences set on the design criteria."*(Quirante, 2012) that uses a unified scale. *"Design criteria are physical or technical requirements that design solutions must satisfy to be considered as acceptable. They are equality or inequality relations between "observation variables" and a set of threshold values."* (Quirante, 2012)

The Aggregation model ζ aggregates **z** according to the Design Objectives into multiple Design Objective Indices **DOI**; *"Design objectives (or goals) are task specific requirements, or desired performance characteristics, that the system should meet."*(Quirante, 2012). Finally, the model aggregates the **DOI** into a Global Objective Index (GDI). Using OIA, we search for the design variables **x** values that maximize the GDI; the maximum GDI represents the most favorable design objectives (Eq. 1) (see Figure 11).

$$\mathbf{y} = \mu(\mathbf{x})$$
$$\mathbf{z} = \delta(\mathbf{y})$$
$$\mathbf{DOI} = \varphi_2(\mathbf{z})$$
$$\varphi_1 o \varphi_2 = \zeta \qquad \textit{(Eq. 1)}$$
$$\mathrm{GDI} = \zeta \circ \delta \circ \mu\ (\mathbf{x})$$

OIA is a framework to clarify the design optimization process; it helps us avoid confusion. It is noticeable that this framework can be easily mapped on to the FBS ontology of (see Figure 12).

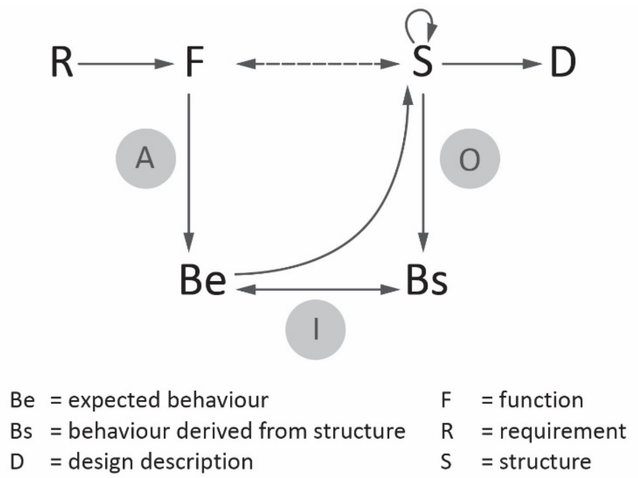


Figure 12: *The connection between OIA* (Collignan, 2011; Quirante, 2012) *and FBS ontology* (J. Gero, 1990; J. S. Gero & Kannengiesser, 2004, 2014)

1.3 Towards a new framework

By reviewing the principles of the design process and after investigating its activities by reviewing different processes, we can conclude that the generative design model provides a simple system. It can adapt to automate digital tools to generate design solutions. However, this model lacks detailed definitions. FBS and OIA provide detailed definitions of design activities. Combining FBS and OIA with the concept of generative design can emerge a well defined generative design framework. The emerged framework can be a good base for developing new systems for design optimization.

CHAPTER 2 Optimality & Acceptability

"True optimization is the revolutionary contribution of modern research to decision processes." (Dantzig, n.d.)

Optimization is *"an act, process, or methodology of making something (such as a design, system, or decision) as fully perfect, functional, or effective as possible."* (Merriam-Webster, 2020). The process of design optimization is a series of actions that aim to find the best possible design solution. From a biological evolution point of view, the best possible solution is called the fittest, and in optimization, it is called the optimum. To achieve optimality in design, we use different mathematical approaches based on numerical computing of different variables.

Lawson highlights, *"Design problems are often both multi-dimensional and highly interactive."*(Lawson, 2005). Nagy explains, *"we can think of the dimensions of any system as its 'degrees of freedom', which define the realm of possibilities within the system. Similarly we can think of any design as a complex system delineated by every decision or choice that must be made during its design."*(Nagy, 2017). These dimensions, which are regarded as the degree of freedom, are the design variables **x**. However, the increasing dimensions are known as the *"curse of dimensionality"* (Bellman, 1961).

The ultimate goal of design optimization is to maximize the satisfaction of many criteria and the objectives together. *"Very rarely does any part of a designed thing serve only one purpose."*(Lawson, 2005). Optimizing a physical system such as a building consists of computing the optimum of a complex system that usually takes into account many different physical behaviors. In the design optimization process, we search for the values of **x** that characterize the solution, which performs the observation variables **y** that maximize the satisfaction of the criteria and the objectives.

In contrast to a local optimum, the global optimum represents the true optimum in the particular context of most design activity. Figure 13 shows that local minima can be closer to the global maximum than some local maxima and vice versa. Computing local optima in this context have no significance, *"Finding with certainty the global optimum is a laudable goal for any design study."*(Papalambros & Douglass, 2017). To compute the global optimum, we can use Global Optimization Algorithms (GOA).

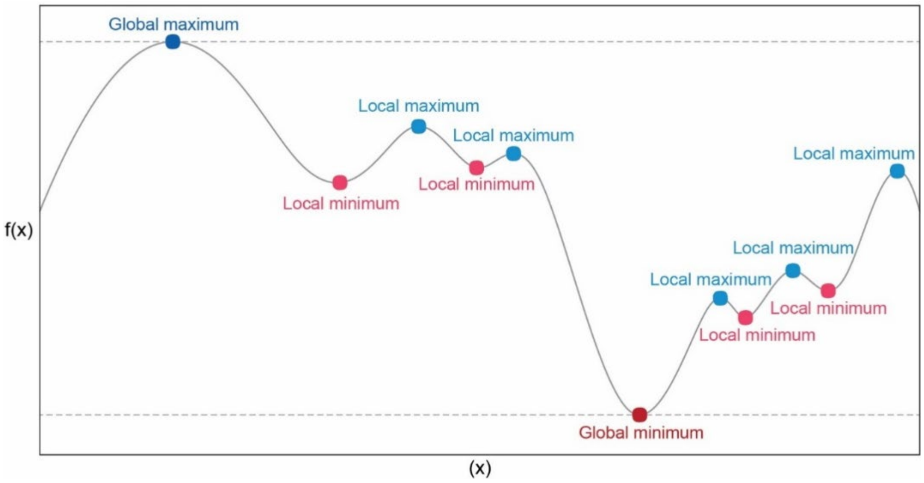


Figure 13: *Global optimum and local optimum*

FBS ontology, along with its framework (J. Gero, 1990; J. S. Gero & Kannengiesser, 2004, 2014) (see 1.2.4), and OIA framework (Collignan, 2011; Quirante, 2012) (see 1.2.5), which is an implementation of FBS, helps to expand our understanding of the design optimization process. Using these frameworks to describe the design process, clarifies the link between **x** on one side and the criteria and the objectives on the other side.

In OIA, the criteria and objectives are defined inside the process. OIA (Collignan, 2011; Quirante, 2012) (see 1.2.4) describes the design criteria within the Interpretation model (δ) and the design objectives in the Aggregation model (ζ) (Collignan, 2011; Quirante, 2012). The Interpretation model (δ) interprets the observation variables **y**, which are computed from **x** into interpretation variables **z** by allowing the designers to express their preferences of the criteria. The Aggregation model (ζ) aggregate **z** into multiple Design Objective Indices **DOI**. The model then uses these **DOI** to compute a Global desirability Index GDI; this model also allows the designers to express their preferences. By linking **x** and GDI, OIA evolves design variables values with observation variables values that are better according to the criteria and the objectives.

Based on OIA, Figure 14 proposes a design framework that is linked to FBS ontology. The proposed framework consists of design inputs, iterative design optimization, and design output. The iterative design optimization is the core of the proposed framework, and it consists of four models Morphogenesis, Observation, Interpretation, and Aggregation; this can be regarded as Morphogenesis plus OIA (MOIA). Generation and evaluation are the two main activities that describe MOIA; the morphogenesis model performs the generation, while the other models (OIA) perform the evaluation. In MOIA, the iterative design optimization initially starts by using a set of random values for **x** variables. Then as by using the models of OIA, it computes the GDI. Based on MOIA, the Interpretation model and the Aggregation model can take into account human preferences (see Figure 14). Finally, the Morphogenesis model links the values of **x** to GDI to evolve new **x** values by using GOA. The process is iterative. Hence, a threshold must be used to end it, which results in design output.

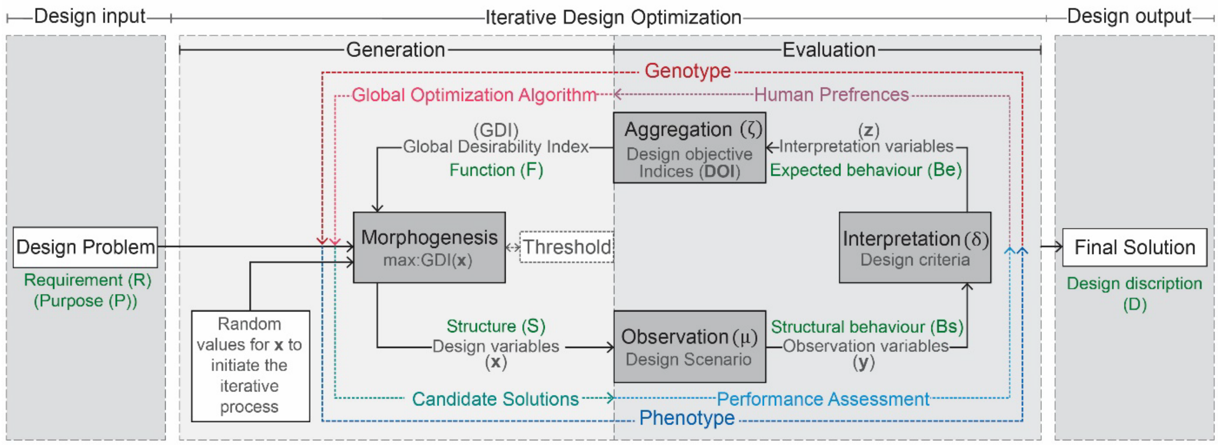


***Figure 14:** MOIA design framework*

Different classifications are used to describe MOIA (see Figure 14). The first classification includes "Generation" and "Evaluation", which is mentioned earlier. The second classification is the "Genotype", which represents the genes and the "Phenotype", which represents the physical world. Nagy explains, *"An organism's genotype is it's DNA, which encodes all the information that makes the organism unique, and guides it's development over the course of it's life. The phenotype is the physical expression of the organism, and is influenced both by its genotype, as well as interaction with the environment over the course of its life."*(Nagy, 2017).

This chapter intends to investigate MOIA by explaining its models, exploring, and criticizing different approaches for its models. Initially, the observation model (μ) is explained. Next, the Interpretation model (δ) is investigated, which is then followed by an investigation of the Aggregation model (ζ). Finally, the Morphogenesis model is investigated.

2.1 Observation

In MOIA, each candidate solution is characterized by different values of the design variables **x**. To initiate the MOIA process, first, we use a set of random values for **x**, later the GOA evolves **x**. The Observation model (μ) uses a design scenario to compute the observation variables **y** of the candidate solutions (Quirante, 2012). The design scenario includes all the information related to the context of the design, such as the environmental data, urban form, building code, etc. On the other hand, **y** describes how the candidate solutions perform in the scenario, the physical and the economic performances are two different examples of the observation variables. *"A set of observation variables* (**y**) *are computed from a set of design variables values* (x)*. The union of every design variable value domain forms the design space Ω to be explored."*(Quirante et al., 2013) (see (Eq. 2)).

$$\mathbf{y} = \mu(x), \qquad x \in \Omega \qquad \textbf{\textit{(Eq. 2)}}$$

2.2 Interpretation

In MOIA, the Interpretation model (δ) transfers the observation variables **y**, which uses different scales, into interpretation variables **z** that uses a unified scale. Lawson states, *"If we imagine that we want to assess a number of design solutions so that we can put them in order of preference we would need to begin by assessing each design against each of the criteria and then somehow combining these assessments."*(Lawson, 2005). He affirms, *"Because in design there are often so many variables which cannot be measured on the same scale, values judgments seem inescapable"*(Lawson, 2005). A value judgment is *"An assessment of something as good or bad in terms of one's standards or priorities."*(Lexico Dictionary, 2020) An interpretation variable **z** represents the degree of satisfaction for each criterion in the range [0 (non-satisfaction) to 1 (ideal satisfaction). To compute an interpretation variable values z_i, we assess the relative observation variable values y_i against the relative design criterion through an interpretation function. These interpretation variables **z** are essential to aggregate the criteria and the objectives later in the Aggregation model (ζ).

Starting from a Pseudo-function, we can describe many interpretation functions; these functions are ordinal functions or cardinal functions (see Figure 17). A Pseudo-function indicates the required maximization of y_i. Indeed the Pseudo-function is not a real function because it does not contain quantitative information; there is no need for control points to map its curve on the space (y_i, z_i).

Understanding the difference between cardinal and ordinal information is critical to understand these functions. The cardinal information is based on real values, whereas the ordinal information is based on ranking. It is noticeable that ordinal ranking can be derived from cardinal evaluations and not the opposite.

The cardinal interpretation functions are derived from the values of **y**, whereas the ordinal interpretation functions are derived from the ranking of the **y** values. Table 1 shows an example of a comparison between the two types of information. Based on the cardinal information, solution C can be regarded as a poor solution. However, based on the ordinal information, solution C may appear as a good solution, since it seems to be a good compromise between solution A and solution B.

	Cardinal (Values)		Ordinal (Ranking)	
	y_1	y_2	y_1	y_2
Solution A	1.00	0.50	1st	3rd
Solution B	0.50	1.00	3rd	1st
Solution C	0.51	0.51	2nd	2nd

Table 1: *Comparison between the ordinal and the cardinal information*

The ordinal information is less valuable than the cardinal information, it can be misleading. To assess the worthiness of information, we use the negentropy concept; the more valuable the information, the higher the negentropy is. The information negentropy is the opposite of information entropy, which corresponds to the randomness of information (Shannon, 1948). Figure 15 helps to simplify the concept of negentropy. The figure three bytes each consists of eight bits. Each bit can either equal to one or zero. However, if the information of one bit is not defined, a random value "R" which equals to one or zero is used. Increasing the bits with defined values leads to increasing the negentropy of the information.

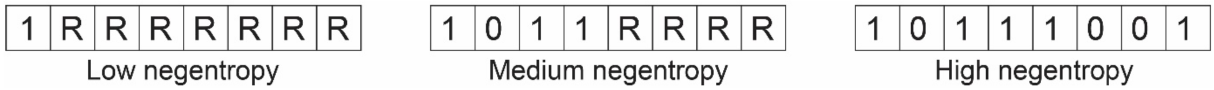


Figure 15: *The concept of information negentropy*

Therefore, the ordinal functions are considered as lower in negentropy in comparison to the cardinal functions; the lowest cardinal function in negentropy is higher than the highest ordinal function in negentropy. The number of control points or parameters used to parameterize a function determines the amount of information required to define it. This amount of information determines the level of negentropy. The higher the amount of information, the higher the level of negentropy.

In ordinal functions, each solution is linked to its rank (r_i), which ranged between 1 (high satisfaction), and "n" (low satisfaction). The Linear-rank function is the most basic ordinal function. It assigns the value one to the rank one and the value zero to the rank "n." This function is very basic, and it does not require control points or parameters to be defined in the space (r_i, z_i); r_i is the solution's rank, and z_i is the level of satisfaction. The Power-rank function is the most complex ordinal function. It defines a power curve from three control points defined by rank 1, r_i^{mid}, and "n" plus their corresponding values, which are z_i^-, $z_i^{mid,}$ and z_i^+. Therefore, this function requires the definition of four parameters.

In comparison to the ordinal functions, the cardinal functions such as the Simon's satisficing (satisfying & sufficient) (H. A. Simon, 1956), Derringer & Suich (Derringer & Suich, 1980), Harrington's (Harrington, 1965) and "arctan soft" functions help to save the valuable information contained in the observed variables **y**. These functions use the cardinal values of **y**, and no ranking is required. In the following, the cardinal functions are discussed.

From a mathematical point of view, a criterion is a condition used in a mathematical operation to test the observation variable value (y_i) and assess a satisfaction level (z_i). In pure mathematics, the criteria are strict; the level of satisfaction is only 1 (satisfied) or 0 (not satisfied). For example, to determine even numbers, the criterion divides the number by two if the remainder = 0 then the number is even, and the criterion is satisfied (the satisfaction level=1) if the remainder ≠ 0 then the number is odd and the criterion is not satisfied (the satisfaction level=0). Classical mathematics is not ambiguous; it is not possible to have satisfaction levels between 0 and 1.

Figure 16 demonstrates the criteria strictness in mathematics. The figure illustrates the mathematical criterion (cost ≤ 100 $). According to this criterion, if (cost ≤ 100 $), then the criterion is satisfied, and the satisfaction level is equal to 1, but if (cost > 100 $), then the criterion is not satisfied, and the satisfaction level is equal to 0. As a consequence, all the y_i that satisfy the criterion are equal, it has the same z_i, and all the solutions with y_i that do not satisfy the criteria have the same z_i. The mathematical function represented in Figure 16 can be referenced to the Satisficing interpretation function, which is defined by Simon (H. A. Simon, 1956). Simon's function is the simplest interpretation cardinal function; it uses a fixed Satisficing (H. A. Simon, 1956) parameter (see Figure 16 and Figure 17).

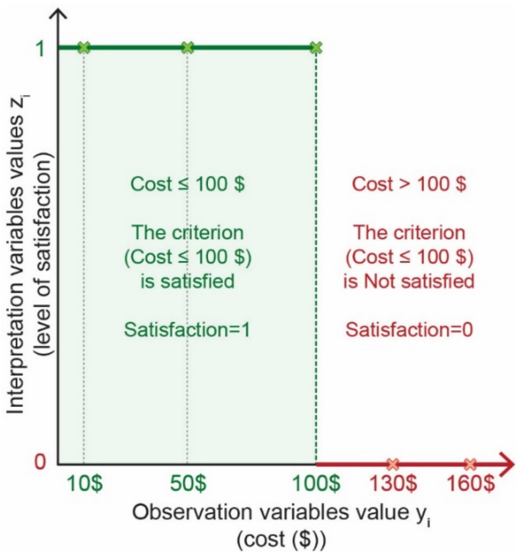


Figure 16: *Example of a criterion in mathematics (Simon's Satisficing function)* (H. A. Simon, 1956)

Using a strict function with only one control point for defining a criterion, such as Simon's function, is suitable for the purely mathematical problems. In contrast, by its very nature, design, problems are soft, and a minor violation of the criteria and objectives is acceptable. *"What a designer really needs is to have some feel for the meaning behind the numbers rather than precise methods of calculating them. As a designer you need to know the kind of changes can be made to the design which are most likely to improve it when measured against the criteria. It is thus more a matter of strategic decisions rather than careful calculations."*(Lawson, 2005)

The word desirability consists of two parts desir-ability, which means the ability to desire or *"the quality, fact, or degree of being desirable"*(Merriam-Webster, 2019b). The desirability function is a mathematical value function that aims to interpret the design criteria, which is expressed by the designer and represents his preferences of the values of **y** into the mathematical field. These functions are cardinal. Quirante defines *"Desirability functions are value functions which express the level of satisfaction of designers for attributes values according to the design requirements and his expectations."*(Quirante, 2012), he explains, *"Desirability is a preference measurement which reflects the level of satisfaction achieved by design alternatives' properties according to designers' point of view."*(Quirante, 2012) Lee, Geong, and Kim explained, *"The desirability function approach converts each response variable to an individual desirability function. The individual desirability function can be viewed as the decision maker's utility function ranging from 0 to 1."*(Lee, Jeong, & Kim, 2018). Harrington's function (Harrington, 1965), Derringer & Suich's desirability function (Derringer & Suich, 1980), and "arctan" soft function are three different types of desirability function.

Figure 17 graphically represents the different aggregation functions discussed previously. The figure classifies the functions based on the negentropy of information. It also classifies them based on the type of information (ordinal, cardinal). For each function, the figure specifies the number of control points and the number of parameters it uses.

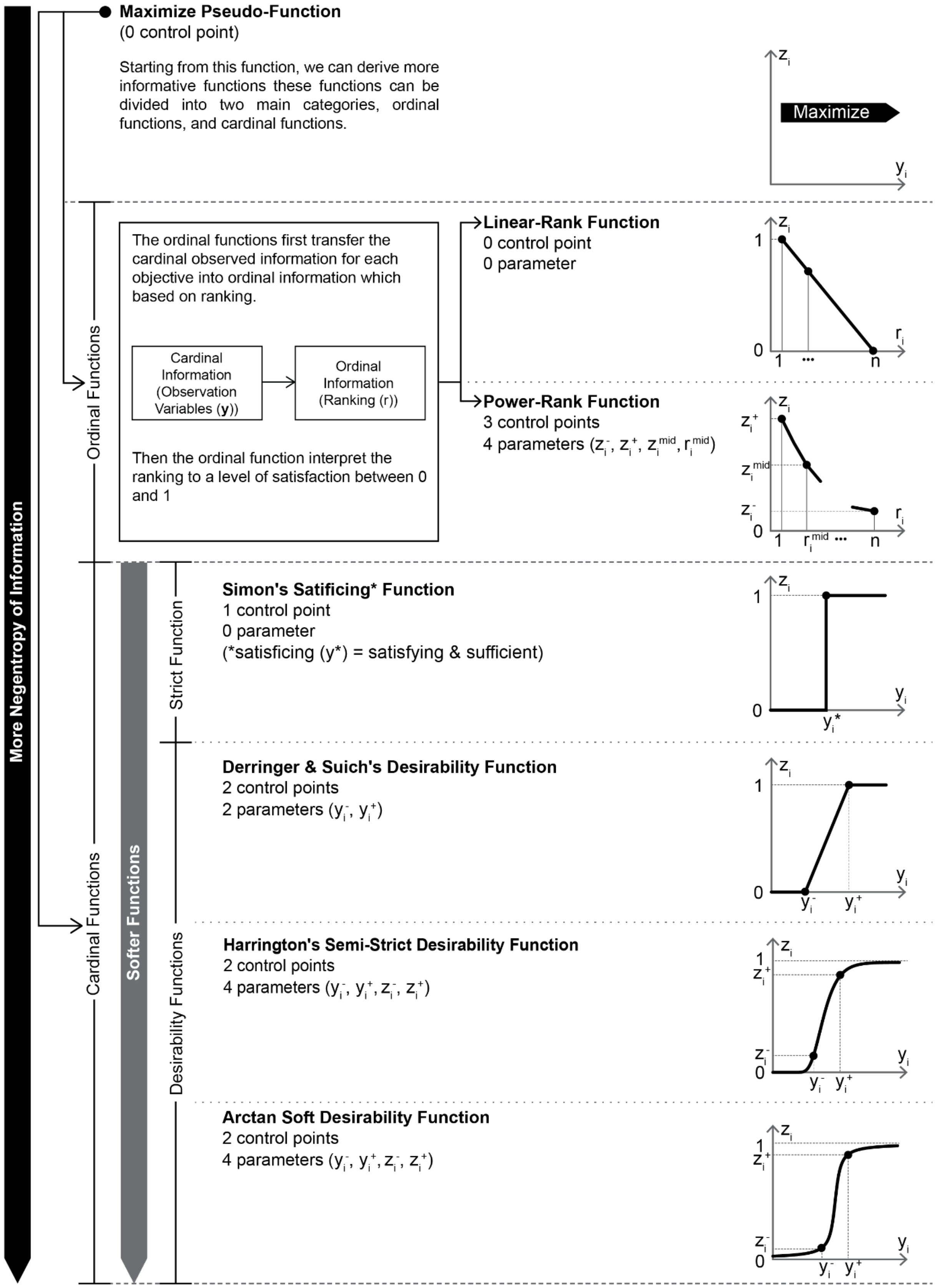


***Figure 17:** Graphical representation of some of the most significant interpretation functions of satisfaction*

2.2.1 Harrington's desirability functions

In 1965, Harrington presented the desirability functions for the first time (Harrington, 1965). Two different categories of Harrington's functions are available one-sided and two-sided, which are both continuous.

The one-sided formulation offers two different versions. Each version serves a different purpose (Eq. 3). The first version is for minimization; the lower the observed value y_i, the higher the level of desirability z_i. The other version is for maximization; the higher the observed value y_i, the higher the level of desirability z_i.

$$d^H(y_i) = \exp\left(-\exp(\beta + \alpha . y_i)\right)$$

Where:

$$\text{if } (AC > SL \text{ (minimization)}) \begin{cases} \alpha = \dfrac{\ln\left(\ln\left(d^H(AC)\right)/\ln\left(d^H(SL)\right)\right)}{AC - SL} \\ \beta = \ln\left(-\ln\left(d^H(SL)\right)\right) - \alpha \times SL \end{cases} \qquad \textbf{\textit{(Eq. 3)}}$$

$$\text{if } (SL > AC \text{ (maximization)}) \begin{cases} \alpha = \dfrac{\ln\left(\ln\left(d^H(SL)\right)/\ln\left(d^H(AC)\right)\right)}{SL - AC} \\ \beta = \ln\left(-\ln\left(d^H(SL)\right)\right) - \alpha \times SL \end{cases}$$

Where the Absolute Constraint (AC) is *"bounds correspond to strict satisfaction of design criteria"*(Quirante, 2012), and the Soft Limit (SL) is *"bounds related to the flexibility of design requirements"*(Quirante, 2012). By comparing AC and SL on the observation axis, in minimization problems, AC > SL, while in maximization AC<SL (see Figure 18). By comparing AC and SL on the scale of desirability $d^H(AC) < d^H(SL)$ for both versions. For instance, $d^H(AC) = 0.01$, $d^H(SL) = 0.99$, the designers can use any values between zero and one to according thier intentions and design requirements.

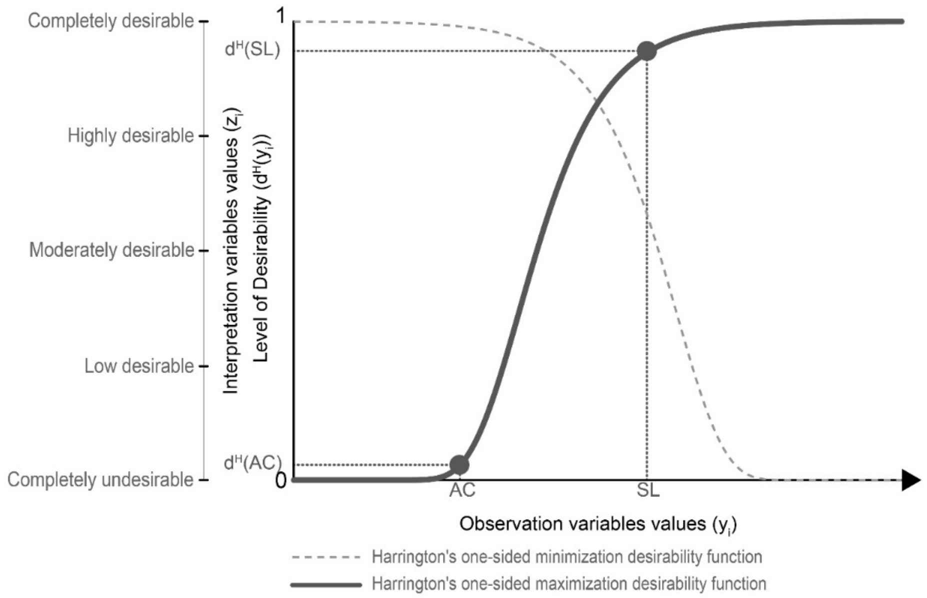


Figure 18: *Graphical representation Harrington's one-sided desirability functions*

The two-sided formulation targets particular values y_i; *"closer to a particular target value is better"* (Quirante, 2012) This formulation requires four parameters (AC_L) lower Absolute Constraint, (SL_L) lower Soft Limit, (SL_U) upper Soft Limit and (AC_U) upper absolute constraint. Generally, we express Harrington's two-sided as (Eq. 4) (see Figure 19):

$$d^H(y_i) = \exp\left(-\left|\frac{2y_i - (U+L)}{U-L}\right|^n\right)$$

Where:

$$n = \frac{\ln\left(-\ln\left(d^H(SL)\right)\right)}{\ln\left(\left|\frac{2(SL_L) - (U+L)}{U-L}\right|\right)}$$

$U = (AC_U + SL_U)/2$ **(Eq. 4)**

$L = (AC_L + SL_L)/2$

With: $(SL_L - AC_L) = (AC_U - SL_U)$

And: $AC_U > SL_U > SL_L > AC_L$

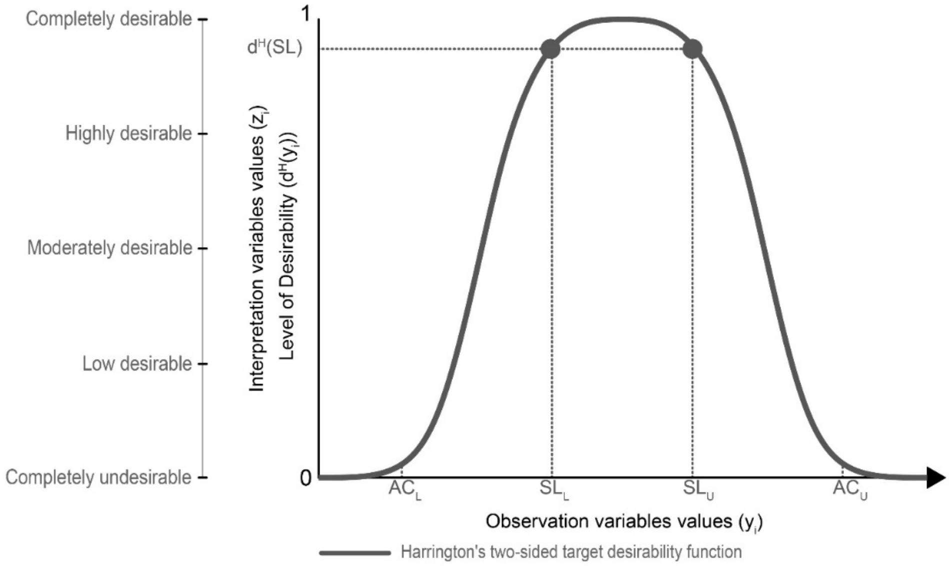


***Figure 19:** graphical representation of Harrington's two-sided desirability function*

Harrington's desirability functions are beneficial for interpreting y_i. The AC allows defining a point where the desirability level decreases dramatically (exponentially of an exponential). Minor violation of AC results in different low desirability levels; not all the solutions that violate the AC are equally desirable. On the other hand, the SL provides a point where the desirability values change softly toward complete desirability. However, this function can result in z_i values so close to zero that they are equal to zero in the floating-point number space of a computer, which can negatively affect the aggregation.

2.2.2 Derringer & Suich's desirability functions

In 1980, Derringer & Suich (Derringer & Suich, 1980) introduced a modified version of Harrington's desirability functions. Two different formulations of this function do exist, one-sided and two-sided. They are piecewise-defined and continuous.

The one-sided formulation offers two different versions. The first version is for minimization (Eq. 5); the lower the observed value y_i, the higher the level of the desirability z_i. The other is for maximization (Eq. 6); the higher the observed value y_i, the higher the level of the desirability z_i.

$$d^D(y_i) = \begin{cases} 1 & y_i \leq L \\ \left(\frac{U - y_i}{U - L}\right)^c & L < y_i < U \quad \text{with} \quad c \in \mathbb{R}_+^* \\ 0 & y_i \geq U \end{cases} \qquad \textbf{\textit{(Eq. 5)}}$$

$$d^D(y_i) = \begin{cases} 1 & y_i \geq U \\ \left(\frac{y_i - L}{U - L}\right)^c & L < y_i < U \quad \text{with} \quad c \in \mathbb{R}_+^* \\ 0 & y_i \leq L \end{cases} \qquad \textbf{\textit{(Eq. 6)}}$$

The two y_i that defines the bounds are the Upper bound (U) and the Lower bound (L); L < U. The level of desirability is described as the following for minimization $d^D(L) > d^D(U)$ and $d^D(y_i \leq L) = 1$, $d^D(y_i \geq U)=0$, while for maximization problem $d^D(L) < d^D(U)$ and $d^D(y_i \leq L) = 0$, $d^D(y_i \geq U)=1$. The parameter (c) allows adjusting the variations of desirability between the two bounds, changing the value of the parameter c changes the function's slope; consequently, it allows the function to match the preferences of the designer. (Quirante, 2012) (see Figure 20).

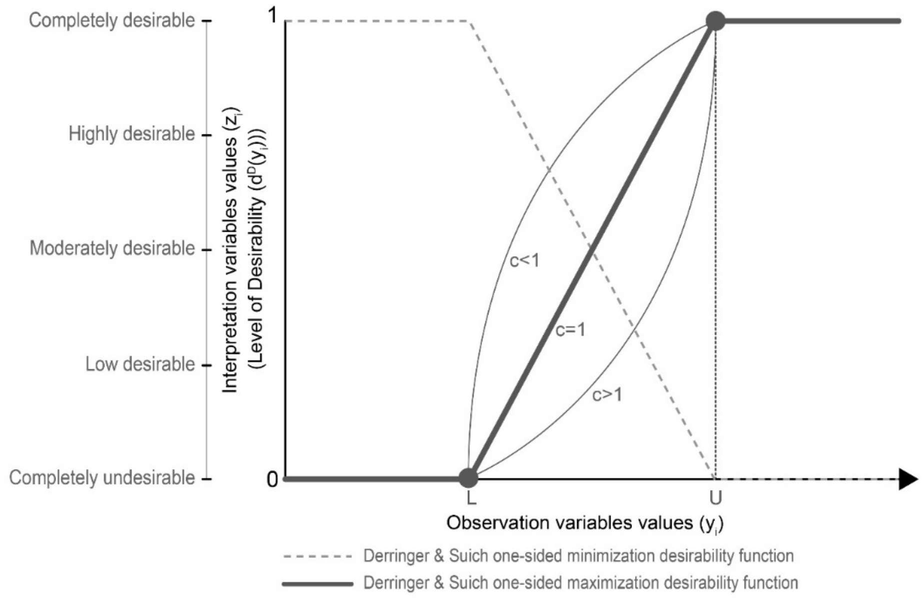


Figure 20: *Graphical representation of Derringer & Suich's one-sided desirability functions*

In the two-sided formulation (Eq. 7), two parameters that define the Lower bound (L) and the Upper bound (U) are required. Also, a third parameter that defines the Targeted value (T) is needed. Additionally, there are two other parameters (c, u) for adjusting the function's slope, one for each side of T (see Figure 21).

$$d^D(y_i) = \begin{cases} 0 & y_i \leq L \\ \left(\frac{y_i - L}{T - L}\right)^c & L < y_i \leq T \\ \left(\frac{y_i - U}{T - U}\right)^u & T < y_i < U \\ 0 & y_i \geq U \end{cases} \quad c \in \mathbb{R}_+^*, u \in \mathbb{R}_+^* \qquad \textbf{\textit{(Eq. 7)}}$$

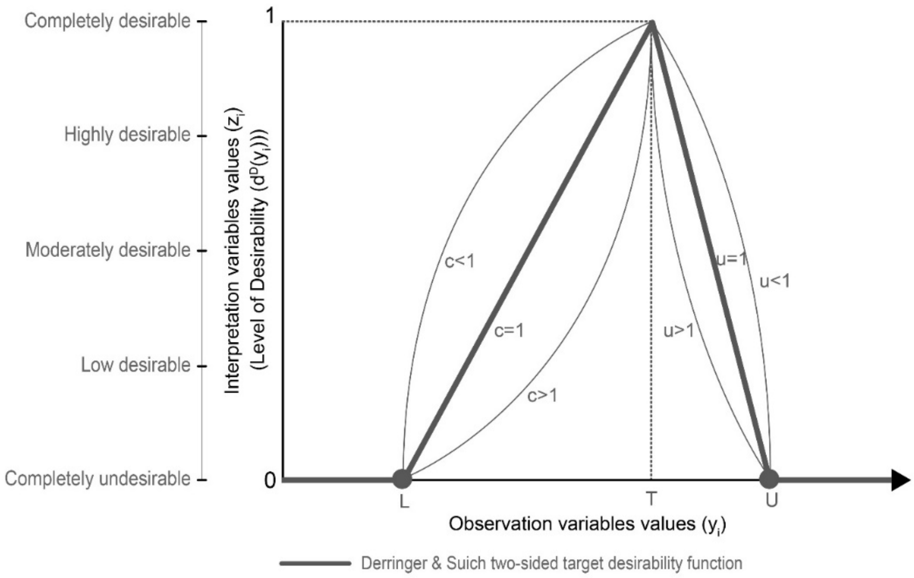


Figure 21: *Graphical representation of Derringer & Suich's two-sided desirability function*

Derringer & Suich's desirability functions interpret the observed values $y_{i,}$ which violate L or U as equally desirable; the function cannot differentiate the level of desirability when L or U is violated. For any solution where $y \geq U$ the $d^D(y_i) = 1$. For any solutions where $y_i \leq L$ the $d^D(y_i)=0$, in the aggregation. We may need an aggregation function that avoids transferring extreme values of y_i to a satisfaction value equal to zero.

2.2.3 The "arctan soft" desirability function

We proposed to use the "arctan soft" desirability function (d^{arctan}) (see Eq. 8). This function is both continuous and derivable everywhere in the real number space such as the Harrington's function. However, it is much softer than the Harrington's function which is computed from the exponential function of an exponential function. The function can be used for both maximization and minimization. The two acceptability thresholds that define the bounds in this function are y_i^+ and y_i^-. The designer has to assign the interpretation value z_i^+ ($d^{arctan}(y_i^+)$) for which represents the level of satisfaction of $y_i^{+.}$ Figure 22 demonstrates a graphical representation of this function.

$$z_i = d^{\arctan}(y_i) = \frac{1}{\pi} \times \arctan\left(\tan\left(\left(z_i{}^{+} - \frac{1}{2}\right) \times \pi\right) \times \left(\frac{y_i - ((y_i{}^{+} + y_i{}^{-})/2)}{y_i{}^{+} - ((y_i{}^{+} + y_i{}^{-})/2)}\right)\right) + \frac{1}{2}$$

Where:

For Maximization *(Eq. 8)*

$z_i{}^{+} = \text{The desirability of } y_i{}^{+} = d^{\arctan}(y_i{}^{+}) \qquad 0.5 < z_i{}^{+} < 1$

For Minimization

$z_i{}^{+} = \text{The desirability of } y_i{}^{+} = d^{\arctan}(y_i{}^{+}) \qquad 0 < z_i{}^{+} < 0.5$

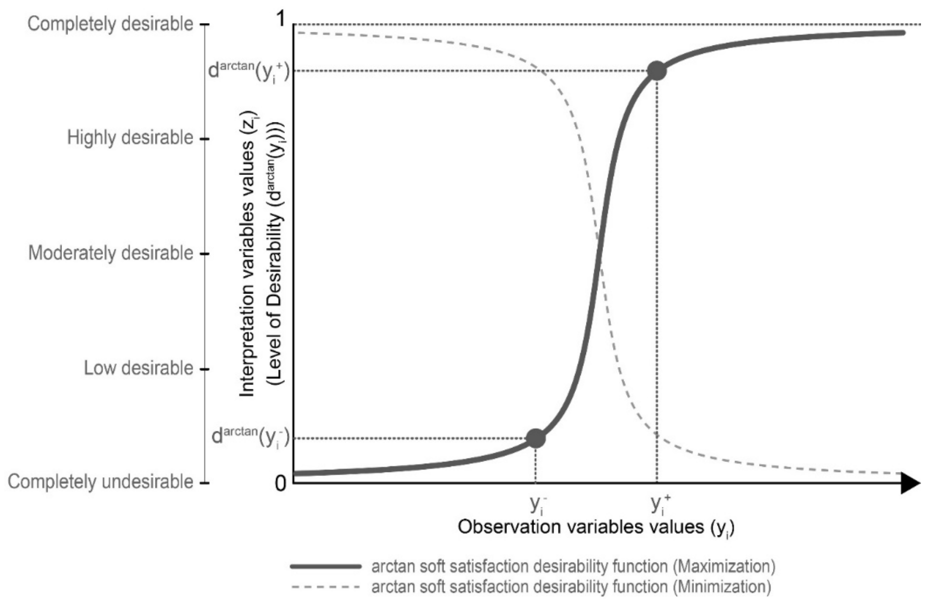


***Figure 22:** Graphical representation of arctan soft desirability function*

Even if the observed values y_i violate the requirements, the solution can still be acceptable, and the desirability values z_i never equals to zero ($0 < z_i \leq 1$). In addition, the function can differentiate the level of desirability from values that violate the limits. This function is suitable for interpreting the values of observational variables in design problems.

2.3 Aggregation

"Architecture is made up of choices. Beautiful buildings block the view. Elegant, thin façades waste of energy. Pleasantly enclosed spaces prevent people from taking the shortest way through. Monumental buildings diminish their surroundings. It's not easy to choose. A checklist can never weigh the options for you; only experience can do that" (Waern & Windgardh, 2015)

The Aggregation model (ζ) classifies the candidate solutions according to the design objectives. According to MOIA, the model aggregates the interpretation variables **z** (multi-criteria) into multiple Design Objective Indices **DOI (**multi-objective). Finally, from all the **DOI**, this model computes the Global Desirability Index GDI (single-objective). Later, the Morphogenesis model generates new values of **x** that maximize the GDI by using GOA.

Starting from a Pseudo-function, we can define two different types of aggregation functions ordinal and cardinal (see Figure 23). The ordinal aggregation functions do not combine criteria and the objectives into a single objective GDI as MOIA suggests; it is not possible to perform mathematical operations on ordinal information. Consequently, these ordinal functions are multi-objective and can result in many solutions that are equally optimized based on different objectives. When using an ordinal aggregation function, there is no need to interpret the observed variables **y**. On the other hand, the cardinal aggregation functions can combine the criteria and objectives into a single objective GDI, as MOIA suggests. These functions aggregate interpretation variables **z** resulting from an interpretation function; it is not possible to directly use the observed variables **y**. Ordinal aggregation functions are considered weakly negentropic, while cardinal aggregation functions are considered strongly negentropic.

The choice of an aggregation function is a decisive choice. This section intends to investigate three different aggregation functions. The first is Pareto's aggregation function, which is ordinal in nature. The second is the Maximin aggregation function, which is cardinal. The third is the Derringer & Suich's aggregation function, which is cardinal. While Pareto's function is low in negentropy, the two other functions are high in negentropy (see Figure 23). The figure graphically demonstrates each of the three aggregation functions intended to be discussed in this section. The figure also defines the basic requirements for precisely defining each of the three functions. At the end of this section, an additional function introduced by Scott & Antonson (Scott & Antonsson, 1998) that links the three aggregation functions is presented to show the continuity between the three functions.

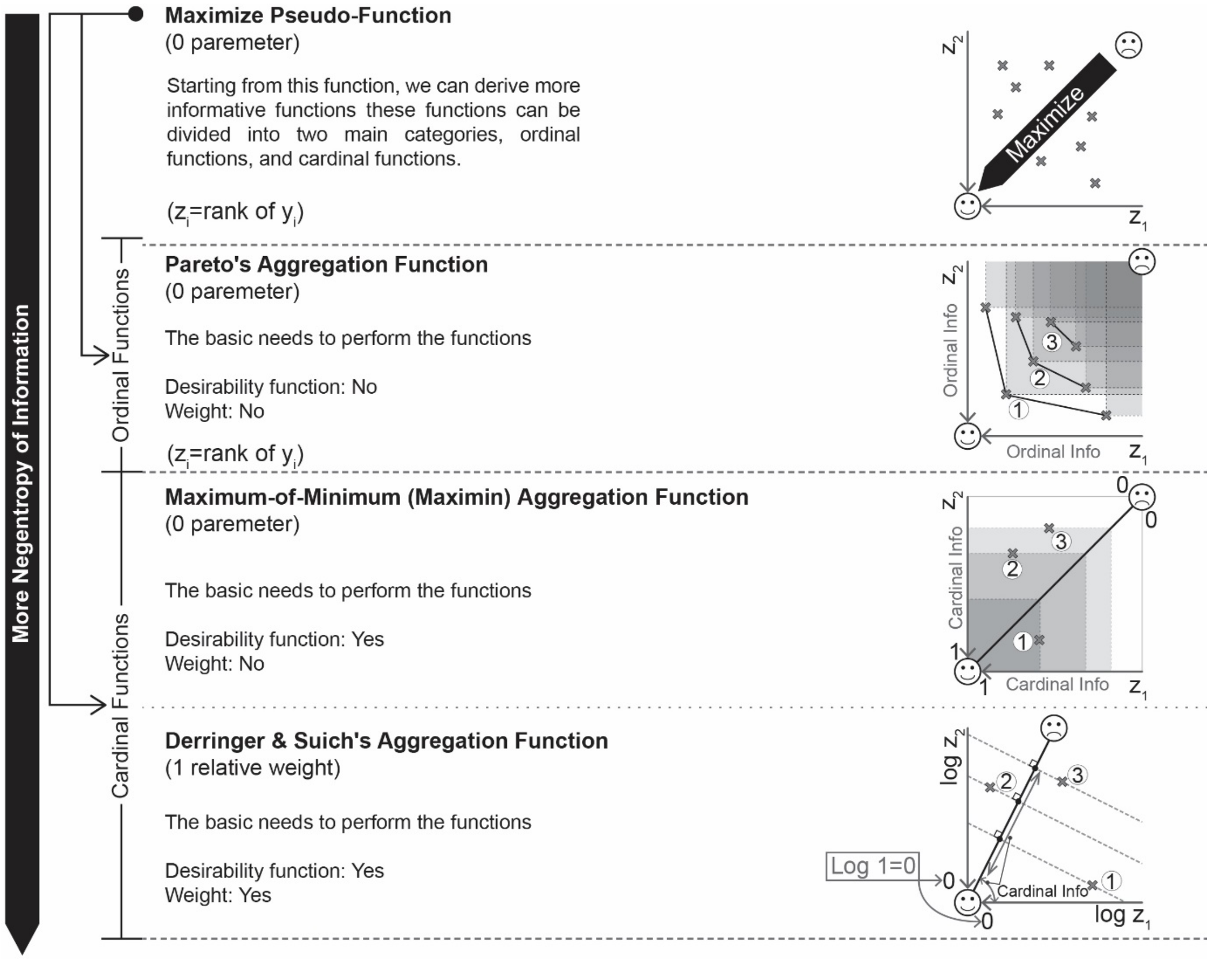


Figure 23: Graphical representation of mentioned aggregation functions

2.3.1 Pareto's function

In 1881 at King's College, Edgeworth described the optimization problem by saying, *"If we are optimizing a problem with two objectives. It is required to find a point such that, in whatever direction we take an infinitely small step, Objective (1) and Objective (2) do not increase together, but that, while one increases, the other decreases."*(Edgeworth, 1881) In 1893, this concept was expanded by Pareto. He describes, *"The optimum allocation of the resources of a society is not attained so long as it is possible to make at least one individual better off in his own estimation while keeping others as well off as before in their own estimation."*(Pareto, 1896).

Today, this definition of Pareto's method leads to improve trade-offs among several objectives. For the sake of homogeneity, simplicity, and coherence with other functions, Pareto's method is introduced as an aggregation function. This function can also be called Chebyshev scalarizing function (Giagkiozis & Fleming, 2015). Based on Pareto's function, the optimal solutions are called non-dominated or non-inferior solutions (Lobato & Steffen, 2017). Pareto's function is used as a decision-making logic that allows the computer to classify the different design variables **x** based on the observation variables **y**, which are pure objective performance; no interpretation of these values is required. Marsault highlights that *"most of the multi-objective evolutionary algorithms have a shared characteristic: they manipulate a*

(Pareto front).” (Marsault, 2018) Pareto’s function is considered as the classical function and is used by the available multi-objective design optimization tools in the market.

Figure 24, from A to D, displays a geometrical explanation of Pareto’s function classification process among a finite set of solutions. For the sake of simplicity, this geometrical interpretation is performed on a two objective problem. Both indicators of the objectives must be minimized in the proposed problem. Pareto’s function classification divides the solutions into two categories of solutions, which are “non-dominated/optimum” and “dominated/non-optimum” (see Figure 24 A). One solution dominates another, only if it achieves a better rank in all the objectives involved in the design. The set of non-dominated solutions can be linked together to form the Pareto front, namely a polyline fitting the external part of the solution set (see Figure 24 B). According to Pareto’s function, these non-dominated solutions are the optimum solutions; all the solutions which belong to the Pareto front are equally optimum.

A solution belonging to the non-dominated category may eliminate many other dominated solutions. From a geometrical point of view, this means that one single solution, which is S2, dominates any solution belonging to the grey rectangle presented in Figure 24-C. By intersecting the rectangles defined by every non-dominated solution, several non-dominated solutions may dominate the same solution (see Figure 24 D). While the non-dominated solutions are classified as the optimum, two different methods can be used to classify the dominated solutions. One method is to count the solutions that dominate each solution; the less is the solutions that dominate a solution, the better is the class of this solution. The other method is to ignore the non-dominated solutions and apply Pareto’s function to the rest of the solutions; this can be repeated to create multiple fronts; each represents one class.

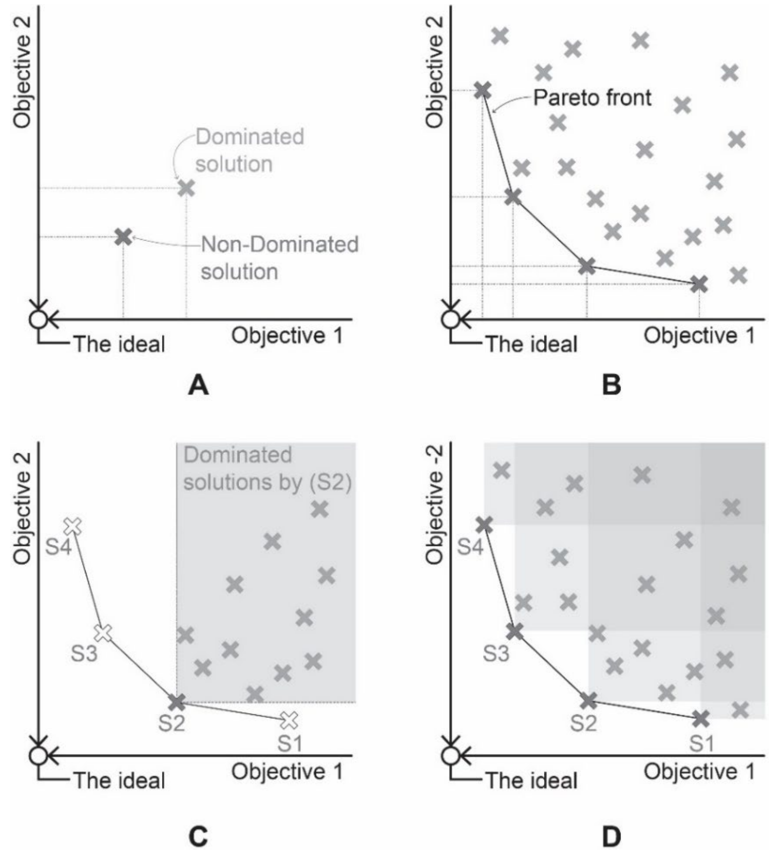


***Figure 24:** Geometrical explanation of the Pareto classification process*

Using Pareto's function can result in different problems. As a first point, the Pareto front usually includes numerous solutions with different **x** values and different **y** values that are considered equally optimum. Figure 25 shows the classification of a set of candidate solutions compared to a single solution S_1 based on Pareto's function. The solutions are divided into three zones. Any solution in "Zone A" dominates S_1, any solution in "Zone B" is equal to S_1, and any solution belongs to "Zone C" is dominated by S_1. If no solutions in Zone A exist, Pareto's function conder S_1 as non-dominated solutions. Pareto's function does not consider some concepts that can foster some solutions by regarding an objective as more important than another. Hence, solutions must be post-treated to select the best ones, which can make the process very inefficient and confusing. To avoid confusion, humans tend to compare design solutions by introducing concepts such as risk in their reasoning.

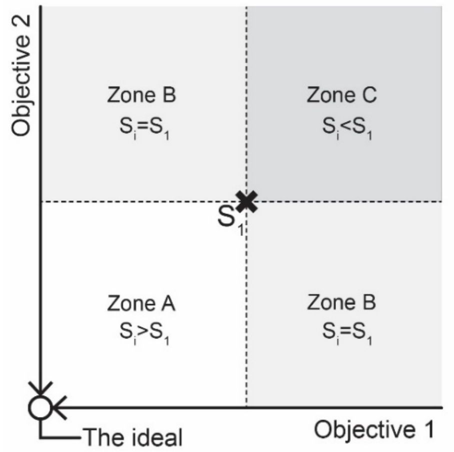


Figure 25: *The relation between the solutions according to Pareto's function*

As a second point, the Pareto front usually contains solutions that may seem irrational to humans and more especially to human experts of design. For example, Figure 26-A displays solution S5, which is dominated by the solution S2 and non-dominated by solution S4. Due to its position on the front, S4 may be regarded as an extreme solution since its performance score is very low, according to objective 2 and very high, according to objective 1. Concomitantly S5 seems to be a balanced solution as its performance scores are proportionate, as can be seen in Figure 26-B. Human reasoning often counterbalances the concepts of domination and equilibrium, and S5 may be regarded as a more acceptable solution than S4 (see Figure 26-C). If solution S2 never existed, such as presented in Figure 26-D, then S5 would be considered as non-dominated based on Pareto's function, this makes S5 equal to S4. However, Pareto's function is of major interest in the context of information scarcity.

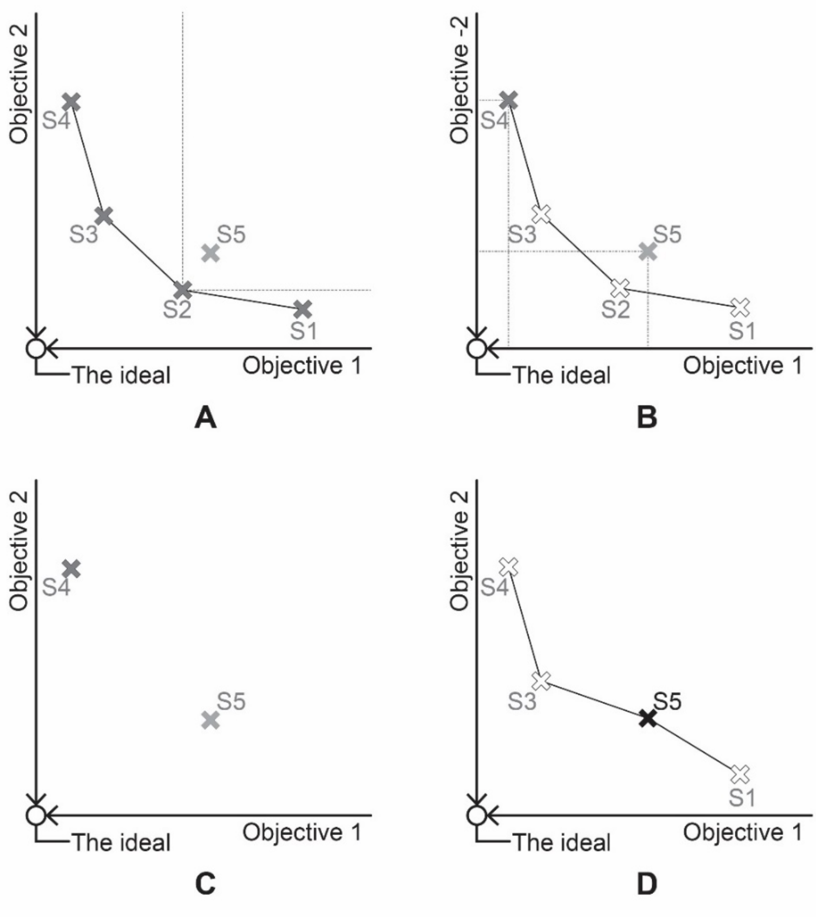


Figure 26: *Elimination issue of Pareto's function*

2.3.2 Maximin aggregation function

Maximum of Minimum (Maximin) is an aggregation function introduced by Kim and Lin in the design domain (Kim & Lin, 2006) (Eq. 9). Maximin is based on a compromisation logic. This function underestimates the solutions, which attain very low levels of satisfaction for at least one objective. Maximin can be regarded as a precautionary principle that avoids extreme and unsafe solutions.

$$\zeta(\mathbf{z}) = max(min(z_i)) \quad \textit{(Eq. 9)}$$

For a better understanding of the function, Figure 27 graphically explains the classification carried out by the function. Figure 27-A shows that the function works as if a square started to grow from the ideal point, and the first solution encountered by the square is classed $1^{st,}$ and the second is classed 2^{nd} and so on; the last solution "n" encounters the square which ranks "n^{th}." In Figure 27-B, the same Pareto front presented earlier (see Figure 26-A (2.3.1)) can be seen. In contrast to Pareto's function, Maximin function reduces the frontier to a single solution, namely S3. As Maximin uses cardinal information, it is possible to aggregate the different objectives into a single objective GDI. In Figure 27-C the solutions S4 and S5, which presented before (see Figure 26-C (2.3.1)) are classified by Maximin function, in contrast to Pareto's function classification, Maximin can differentiate them.

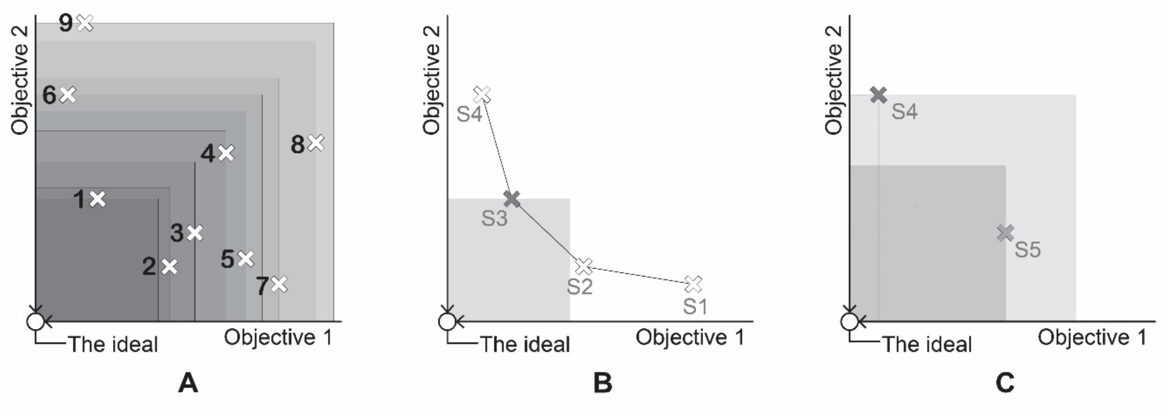


Figure 27: *Graphical explanation of Maximin aggregation function* (Kim & Lin, 2006)

Figure 28 explains how Maximin function aggregates the solutions. The function first determines the minimum z_i for each solution; z_i computed from y_i via a desirability function. In the figure, the solid gray lines link the solutions to these minimum values. For each solution, the minimum z_i is its GDI, and the higher is z_i value, the better the solution.

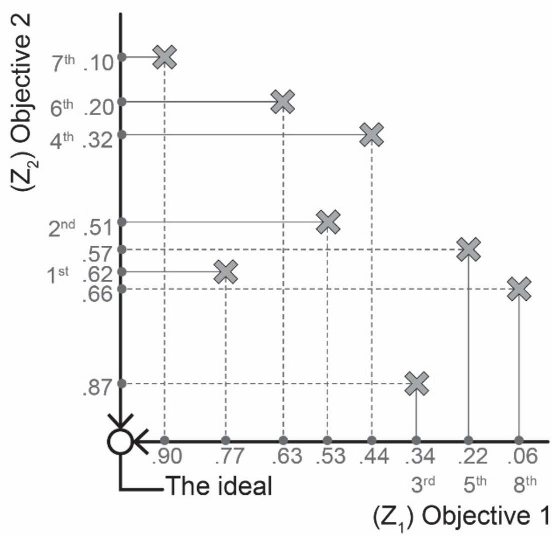


Figure 28: *Graphical representation of Maximin classification process*

However, in some situations, as shown in Figure 29, Maximin classification can be irrational for designers du to the position of the solutions. According to Maximin, S2 is better than S1. In this example, S1 obtains a high level of satisfaction in relation to objective 1, while S2 obtains a very low level of satisfaction in relation to the same objective. On the other hand, S1 obtains a very low level of satisfaction in relation to objective 2, while S2 obtains a very low but slightly better level of satisfaction than S1 in relation to the same objective. As a result, S1 may seem a more rational solution to designers if compared to S2, which is the opposite of Maximin classification. This happens because Maximin is non-compensatory.

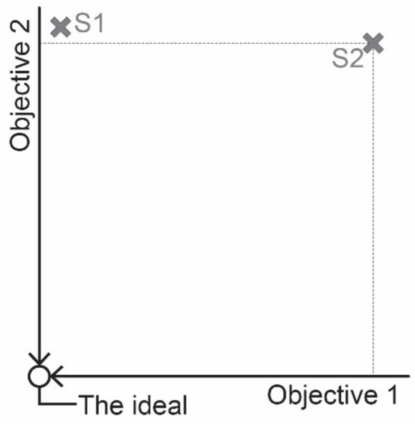


Figure 29: *Location Issue of Maximin*

Figure 30 represents the classification of an infinite set of candidate solutions compared to a single solution S_1 based on Maximin function. According to Maximin, any solution belongs to "Zone A" dominates S_1, any solution belongs to "Zone B" is dominated by S_1. However, any solution on the dotted line has shared the same class with S_1. However, the situation where more than one solution shares the same class is relatively rare because Maximin uses cardinal information.

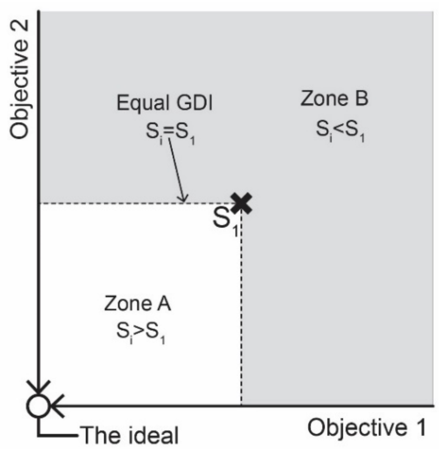


Figure 30: *The relation between the solutions in Maximin*

Maximin is low in computational cost; thus, it is suitable when a fast decision is required. Although Maximin does not require assigning weights, it requires using a desirability function. Because Maximin aggregation function does not require assigning weights, it is not capable of distinguishing the level of importance for the different objectives. Using this function allows the designers to express their preferences in the interpretation model but not in the aggregation model.

2.3.3 Derringer & Suich's aggregation function

Derringer & Suich's aggregation function (Derringer & Suich, 1980) (Eq. 10) entails assigning different weights to the different design objectives. This concept seems appealing for many designers because *"The various criteria of performance are not likely to be equally important, so some weighting system is needed"*(Lawson, 2005). While Pareto's function does not consider any human preference and Maximin involves human preference within the interpretation model, this function considers human preference within the interpretation and the aggregation models of MOIA.

$$\zeta(\mathbf{z}) = \prod_{i=1}^{n} (z_i)^{\omega_i}$$

$$\zeta(\mathbf{z}) = (z_1)^{\omega_1} \cdot (z_2)^{\omega_2} \cdot (z_3)^{\omega_3} \ldots (z_n)^{\omega_n}$$

Where: ω_i are the weight of the objectives or criteria

And: z_i are the performance values of the objectives or criteria

(Eq. 10)

And: $\sum \omega_i = 100\%$

And: $0\% \leq \omega_i \leq 100\%$

Table 2 demonstrates how Derringer & Suich's aggregation function works. The table compares seven different candidate solutions against two different objectives. In the used example, the assigned weight for Objective 1 is 60%, and the assigned weight for Objective 2 is 40%. The table also shows why using an interpretation function that can result in $z_i = 0$ is not suitable when we use this aggregation function; all the solutions that include at least one $z_i = 0$ are equal no matter what are the other z_i values (GDI=0) (see Table 2 (solutions A, and B)).

Solution	Objective-1 (w=60%)	Objective-2 (w=40%)	$\zeta(\overline{z})$=(GDI) (see Eq.11)	Ranking
Solution A	1.00	0.00	$1.00^{0.6} \cdot 0.00^{0.4}=0.00$	6th
Solution B	0.00	1.00	$0.00^{0.6} \cdot 1.00^{0.4}=0.00$	6th
Solution C	0.75	0.75	$0.75^{0.6} \cdot 0.75^{0.4}=0.75$	2nd
Solution D	0.50	0.50	$0.50^{0.6} \cdot 0.50^{0.4}=0.50$	3rd
Solution E	0.80	1.00	$0.80^{0.6} \cdot 1.00^{0.4}=0.87$	1st
Solution F	0.90	0.10	$0.90^{0.6} \cdot 0.10^{0.4}=0.37$	4th
Solution G	0.10	0.90	$0.10^{0.6} \cdot 0.90^{0.4}=0.24$	5th

***Table 2:** An example that demonstrates Derringer & Suich's aggregation function*

Figure 31 uses the solutions presented in Table 2 to illustrate how Derringer & Suich's aggregation function works; the solutions A and B are not used because they include at least one value where $z_i = 0$. Derringer & Suich's aggregation sets the desirability of the solutions on a logarithmic scale. To aggregate the solutions a preference line that starts from the center (the ideal) with a slope that represents the relative weight of the importance of the objectives. For the example presented in the figure (θ=90/(60+40)×60=54°, β=90/(60+40)×40=36°) and with an infinite length. Then, from each solution, a perpendicular line to the preference line is generated; this can be described as a projection of the solutions on the preference line. The intersection point between each of the perpendicular lines and the preference line represents the level of preference of the solution; the closer is the intersection point to the center (the ideal), the higher the satisfaction of the solution.

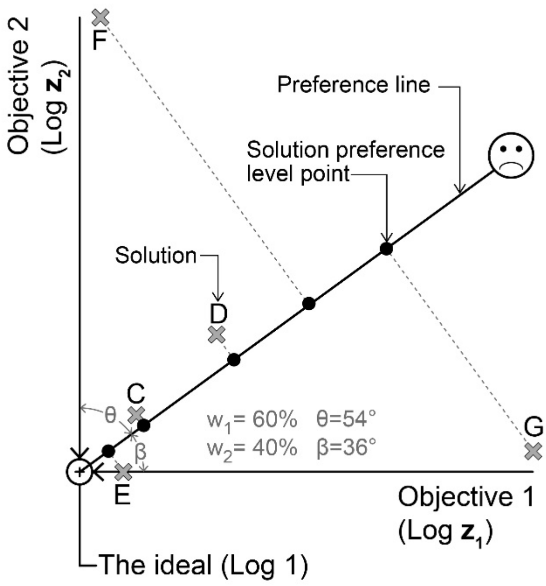


Figure 31: *Graphical representation of Derringer & Suich aggregation function classification*

To compare Derringer & Suich's function to the other aggregation functions presented earlier, Figure 32 graphically represents Derringer & Suich's aggregation function classification on a linear scale (non-logarithmic). In this example, equal weight for the objectives is used to simplify the concept. Each of the presented curves represents one GDI value; the solutions belonging to one curve share the same GDI. However, It is extremely rare to find different solutions with the same GDI; this function is excellent in differentiating the solutions' classes. In contrast to Maximin function, this function is compensatory.

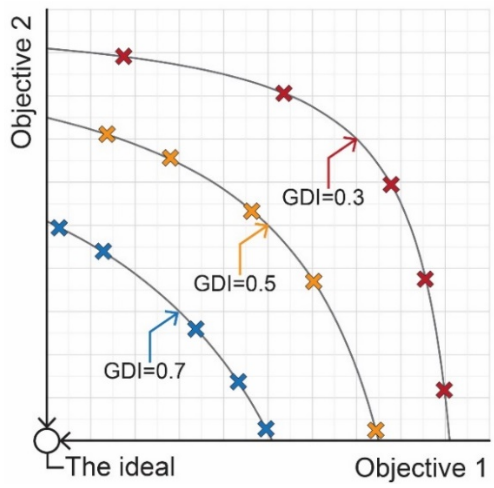


Figure 32: *Graphical representation of Derringer & Suich aggregation on a linear scale*

Figure 33 represents the relation between the solutions in this function by comparing a single solution S_1 to a set of infinite solutions; to simplify the idea, the figure uses equal objective weights. According to this function, any solution in "Zone A" dominates S_1, any solution in "Zone B" is dominated by S_1. However, any solution on the dotted line has an equal GDI to S_1. As mentioned earlier, having different solutions that are equally optimum is extremely rare because Derringer & Suich's aggregation uses weighted cardinal information to find a single objective (GDI).

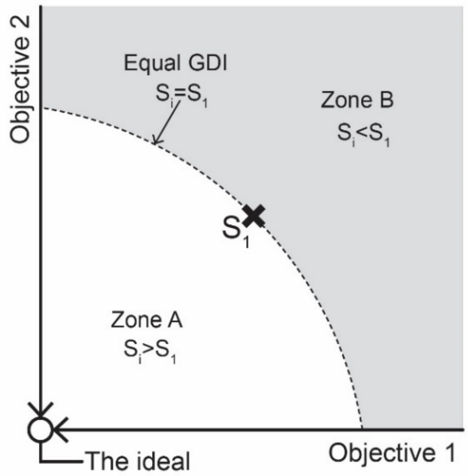


Figure 33: *The relation between the solutions in Derringer & Suich's aggregation function*

Derringer & Suich's aggregation function is the highest in negentropy compared to the other aggregation functions presented earlier. This function allows the designers to express their preferences in the aggregation model. Also, because it uses **z** values resulting from a desirability function, the designers can express their preference in the interpretation. The main difficulty facing Derringer & Suich's aggregation function is that the designer needs to assign weights for the different objectives, which can be tricky. The modeling cost is higher in this function compared to the other aggregation functions presented earlier.

2.3.4 Scott & Antonsson function

Scott & Antonsson (Collignan, 2011; Quirante, 2012; Scott & Antonsson, 1998, 1999) have presented a function that helps to link different aggregation functions. The function shows the continuity between Pareto's, Maximin, and Derringer & Suich's functions by only changing one value (s) (Eq. 11). The different values of (s) change this function to one of the three mentioned aggregation functions.

$$\zeta(\mathbf{z}) = \left(\sum \omega_i \, (y_i)^s\right)^{1/s}$$

With, ω_i = weight

Case (1): Pareto's function ($\omega_i = 1$); not sensitive to weight

Case (2): Maximin function ($\omega_i = 1$); not sensitive to weight

Case (3): Derringer & Suich's function ($\omega_i \in \mathbb{R}_+^*$)

(Eq. 11)

And y_i=Observation value

The value of (s) is related to the aggregation function:

Case (1): Pareto's function ($s = +\infty$)

Case (2): Maximin function ($s = -\infty$)

Case (3): Derringer & Suich's function ($s = 0$)

Figure 34 connects the different values of (s) to the different aggregation functions (Quirante, 2012; Scott & Antonsson, 1998, 1999). Using (s) value bigger than zero is not appropriate for the design. Thus, Pareto's aggregation function is not suitable for design. The figure also shows that increasing the value of (s) increases the compensation of the function. Consequently, Maximin faces some difficulties because the value of (s) is extremely low ($-\infty$). Hence, we can infer that Derringer & Suich's aggregation is more suitable for the design than the two other functions. However, Derringer & Suich's aggregation requires more information (weight), which increases the modeling cost.

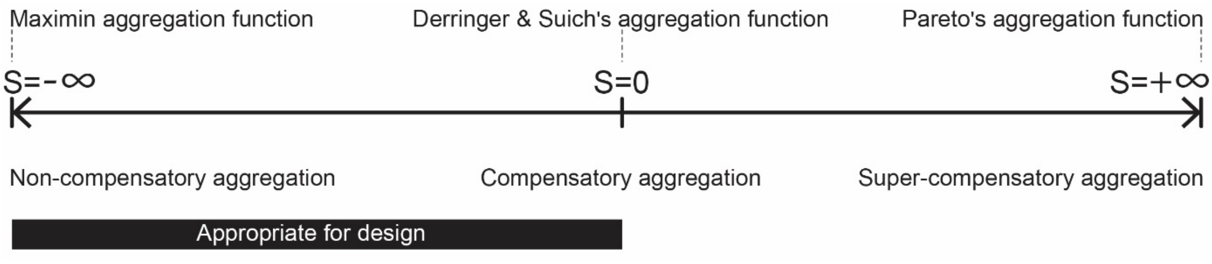


Figure 34: *Comparison between different values of (s) in Scott and Antonson function* (Collignan, 2011; Quirante, 2012; Scott & Antonsson, 1998, 1999)

2.4 Morphogenesis

In biology, the genome is encoded in the chromosomes as DNA, which nucleotides (A, T, G, C). In MOIA, we can link design variables **x** to the DNA and the values of **x** to the DNA nucleotides. The different values of **x** distinguish each individual from its population. In nature, the Morphogenesis transforms the genotype to phenotype. Österlund defines *"Natural morphogenesis is a process of evolutionary development and growth that causes an organism to develop its shape through the interaction of system-intrinsic capacities and external environmental forces"*(Österlund, 2010) In MOIA, the Morphogenesis is a process that computes (evolves) sets of **x** values that characterize candidate solutions which maximize the GDI. In MOIA, to evolve new **x** values, the Morphogenesis model links **x** to the GDI through a Morphogenesis algorithm. To initiate the iteration, MOIA uses random **x** values.

The Morphogenesis algorithms are either gradient or non-gradient based optimization algorithm. The gradient-based algorithms require the calculation of derivatives. It's not the case for non-gradient algorithms, which are more flexible. Gradient algorithms are efficient for finding local optima, whereas non-gradient algorithms are generally used to compute the global optimum of a problem (Papalambros & Douglass, 2017). *"When the problem has continuously differentiable functions, a gradient-based method is the right solution choice. If derivatives are not available, non-gradient methods are the only recourse."*(Papalambros & Douglass, 2017) However, the computing cost of the non-gradient algorithm is generally high compared to gradient ones (Papalambros & Douglass, 2017). In MOIA, the Morphogenesis algorithm is usually a non-gradient GOA.

For design problems, metaheuristics approaches can be very effective since design problems are uncertain. *"In fact, due to the high complexity and difficulty of optimization problems under uncertainty, often classical approaches (that guarantee to find the optimal solution) are feasible only for small size instance of the problems, and they could require a lot of computational effort. In contrast, approaches based on metaheuristics are capable of finding good and sometimes optimal solutions to problem instances of realistic size, in a generally smaller computation time."*(Bianchi, Dorigo, Gambardella, & Gutjahr, 2009).

According to Sorensen and Glover (Sörensen & Glover, 2013). *"A metaheuristic is a high-level problem-independent algorithmic framework that provides a set of guidelines or strategies to develop heuristic optimization algorithms. The term is also used to refer to a problem-specific implementation of a heuristic optimization algorithm according to the guidelines expressed in such a framework."*(Sörensen, Sevaux, & Glover, 2018). Moreover, according to Osman and Laporte, *"A metaheuristic is formally defined as an iterative generation process which guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space, learning strategies are used to structure information in order to find efficiently near-optimal solutions."*(Osman & Laporte, 1996).

Many metaheuristic algorithms have been developed to tackle optimization complex problems under uncertainty *"Nearly all metaheuristic algorithms share the same following characteristics: they are nature-inspired."*(Cheng & Prayogo, 2014). Agarwal and Mehta state that the Nature-Inspired Algorithm (NIA) *"are mainly categorized into evolutionary algorithms and swarm intelligence "based algorithms."*(Agarwal & Mehta, 2014).

The Evolutionary algorithms (EAs) category includes any algorithm inspired by the evolution in nature, (e.g., Genetic algorithms (GA), Immune algorithm, Differential evolution, etc.). One of the most popular algorithms used in design optimization is GA; these algorithms mimic species evolution by natural selection. GA is robust and commonly used for design optimization, which means that it is reliable for solving a wide range of problems. GA is the most representative algorithm of the evolutionary type. Many versions of GA are available; one example of these versions is the EpiGenetic Algorithm (EGA); it adapts the concept of epigenetics to the GA(Birogul, 2016). Hassan and Cohanim say, *"The GA and its many versions have been popular in academia and the industry mainly because of its intuitiveness, ease of implementation, and the ability to effectively solve highly nonlinear, mixed integer optimization problems that are typical of complex engineering systems."* (Hassan et al., 2004).

In GA to evolve the set of genes, three operators called selection, crossover, and mutation are applied to the population through an iterative process. The selection operator selects solutions based on their satisfaction with the criteria and the objectives; in MOIA, we use GDI. The crossover operator evolves the characteristics of the individuals by mixing the DNA nucleotides of the genes; in MOIA, we use the values of **x**. The evolved individuals nevertheless preserve the genes of the original generation. The mutation adds random characteristics to some of the genes, making it possible to explore entirely new solutions. Knowing that these specific genes are not necessarily present in the initial population, this exploration procedure makes it possible to find the global optimum of the design problem.

The designing activity must achieve an appropriate balance between exploitation (selection and crossing) and exploration (mutation). *"Intense exploration does not give optimal solution while deep exploitation traps an algorithm in local optima."*(Agarwal & Mehta, 2014). These steps evolve a set of globally optimized genes.

Papalambros and Douglass describe that *"Genetic algorithms offer several advantages because they are versatile, require no mathematical knowledge, and are easy to program. There is a variety of different crossover and mutation operators that can be chosen, as well as a variety of different methods for parent selection."*(Papalambros & Douglass, 2017). However, *"The drawback of the GA is its expensive computational cost."*(Hassan et al., 2004).

The Swarm intelligence (SI) is another category of GOA (e.g., particle swarm, ant colony, artificial bee colony, and bacterial foraging algorithms, etc.) The algorithms belonging to this category are designed to *"optimize the certain problem by mimicking the collective behavior of natural swarms."*(Agarwal & Mehta, 2014).

In 1995, Eberhart and Kennedy introduced the Particle Swarm Optimization (PSO) algorithm (Kennedy & Eberhart, 1995), which has proven to be very efficient for the global optimization of particular types of problems. PSO can converge very quickly and may require relatively little computational time compared to GA. *"PSO uses the analogy of social interactions among particles in nature, such as insects or birds, to search for optimal solutions."*(Papalambros & Douglass, 2017).

PSO can be described as a network of particles acting together. Each solution corresponds to a particle in which direction and velocity are computed from the PSO algorithm. The interaction between particles is based on mechanisms that are not operators but determine the velocity of the particles from inertia and attraction towards the most performant positions inside the design search space. The particles move inside the search space until the solutions are fixed to one particular point, which is the global optimum. *"Particle swarm optimization is* *an extremely wimple algorithm that seems to be effective for optimizing a wide range of functions."*(Kennedy & Eberhart, 1995).

Both GA and PSO are heuristic population-based search methods. Papalambros and Douglass distinguish the difference between PSO and GA *"The algorithm is similar to GA in that a population of samples together determines the search directions for the next iteration. However, samples in PSO follow trajectories, whereas those in GA jump in the design space."*(Papalambros & Douglass, 2017). GA and PSO are two of the most popular heuristic non-gradient GOA. However, many other heuristics non-gradient GOA can be used to compute the global optimum of a design problem.

2.5 Optimality and acceptability

Simon (Herbert A. Simon, 1973) defines design problems as ill-structured problems; the structure of design problems lacks some definitions. Solving these problems involves subjective judgments and objective knowledge of the problem characteristics. Subjective judgment is difficult to process and is non-reducible to pure mathematical logic. In many cases, optimum design solutions can be less acceptable than some other candidate solutions based on the designer's subjective judgment. Optimality and acceptability are two different notions.

Optimality reflects the performance measurements and concerns the computation of numerical objectives based on mathematical logic. Hence, optimization alone is not enough to determine the preferred design solutions from the designers' points of view. The acceptability, on the other hand, concerns human perceptions. To assess design acceptability, the designer's preference should be the center of reasoning and judgment. The word acceptability consists of two parts accept-ability, which means the ability to accept or "*capable or worthy of being accepted"*(Merriam-Webster, 2019a). Through the acceptability, it is possible to process subjective judgments.

It is crucial to state that the successful design process has to consider both the optimality and acceptability. In design, decision-making is the bridge linking optimality and acceptability. Numerical optimization supports the reasoning process of the designer's preferences. In design, we compute optimality and ensure acceptability. The integration between optimality and acceptability in the design process builds a computational path that leads to optimal and acceptable solutions. However, integrating optimality and acceptability requires a deep understanding of the design process.

Throughout history, humans developed numerous decision support systems. The first ones developed and used were graphical; geographical maps are an example of these tools. However, these systems are non-autonomous. Eventually, humans developed different mechanical decision support systems; these were the first autonomous decision support systems. Since then, humans searched for faster, more responsive, less bulky, flatter, lighter, more stealthy, cheaper, more upgradable, more accessible, more understandable, and more intelligent autonomous decision support systems. After discovering electricity, many electrical decision support systems were invented. Later, many electronic decision support systems were introduced and are still widely used. Nowadays, Artificial Intelligence AI allowed us to develop advanced decision support systems that can compete with the human brain in many fields. AI can help us solve a wide variety of complex problems.

Design problems are ill-structured. Thus, humans must be inside the definition of the problem's structure (Herbert A. Simon, 1973). Using a closed and completely autonomous decision support system to solve ill-structured problems based on Artificial Intelligence AI is ineffective as it is independent of humans; no human is inside the definition of the problem's structure. The concept of Intelligence Augmentation IA is a more suitable approach for solving ill-structured problems as it puts humans inside the definition. In design, we need decision support systems based on IA.

MOIA framework is both a design approach and a modeling method that allows us to put the human inside the definition of design problem structure. MOIA intrinsically carries out an IA approach. It is suitable for developing decision support systems related to design problems. MOIA is capable of integrating both optimality and acceptability by allowing designers to express their preferences inside the design optimization process. Such systems can increase the probability of generating solutions that are optimized mathematically and are accepted by humans.

In MOIA, the integration between optimality and acceptability takes place in the Interpretation model (δ) and the Aggregation model (ζ). Using interpretation and aggregation functions that are high in negentropy in these models can enhance the integration of the acceptability within the optimization process. Moreover, it can preserve the valuable information resulting from the observation model and thus can derive precise results. However, these functions, which are high in negentropy, require more information, which increases the cost of modeling.

I recommend developing decision support systems for design based on *acceptimality* (acceptability and optimality). A deep understanding of MOIA and the possible approaches for its models are incredibly vital for such development as it provides a design approach based on *acceptimality*. MOIA can provide a new paradigm for developing decision support systems adapted to design problems. Developing such systems can improve design outcomes and can attract more designers' to optimize.

CHAPTER 3 Software typologies

"Give ordinary people the right tools, and they will design and build the most extraordinary things." (Gershenfeld, n.d.)

To proceed with a design process, the designers usually use different workflows. According to the Merriam-Webster dictionary, a workflow is *"the sequence of steps involved in moving from the beginning to the end of a working process"*(Merriam-Webster, 2019c). To be more specific, a workflow here is *"how digital tools have been adopted by architects and engineers and merged with building delivery methods"*(Garber, 2017) The designer can use different tools to complete one workflow.

In the mid of the 20th century, the computer was introduced to the world of design. Two decades later, the commercial use of computers in design started. The term Computer-Aided Design (CAD) is used to describe this new paradigm. Since then, CAD has tremendously improved; many software-based tools are developed for designers.

There is no doubt that CAD helped the designers to work much faster and more accurately than before. It made the design industry more efficient. Many repetitive and time-consuming jobs in the design industry were improved or replaced by the computer. Consequently, the designers gained more time to focus on design decisions. Most of the available CAD tools are made for drawing production and not for design decisions. Consequently, some designers prefer to call CAD Computer-Aided Drawings or Computer Aided Drafting. Only a few tools in the market are developed to serve design decisions.

Nowadays, most of the tools used by architects are computer-based. This chapter explores the popular typologies of these typologies, and it also links them to MOIA. This exploration is vital as it helps determine the tools that can support the design decision. Later, this will allow us to define, study, and evaluate different decision support workflows for design.

3.1 Drafting software

Drafting software is the firstborn CAD typology in the market. These tools helped the designer to draft much more accurately and faster. The tools belong to this typology, such as the early versions of AutoCAD® uses lines and points as drawing on papers. However, these lines and points do not contain any information more than its geometrical and graphical characteristics. The software that belongs to this typology is not capable of collecting the necessary information needed to run analyses that insight design decisions. Hence, they do not support design decision-making. In other words, the drafting tools do not serve MOIA models. Nowadays, it is rare to find software that only serves two-dimensional drafting, as most of these tools were upgraded to include three-dimensional capabilities.

Kensek and Nobel explain, *"Early CAD was essentially divorced from building analysis, and CAD was frequently employed by a separate technical team within the firm. Sharing even basic CAD data was initially difficult, and the pace of CAD adoption in professional practice was slow. Architects could integrate CAD into the workflow without* *significantly improving the way consultant, clients, and contractors worked. The decision to adopt CAD usually involved discussions of drafting speed, ease of making updates, and limited benefits that might accrue with enhanced accuracy. In later incarnations of CAD, three-dimensional computing added capabilities, including visualization and clash-detection."* (Kensek & Noble, 2014)

3.2 Massing software

Massing software typology focuses on 3D modeling. In the market, there are several massing software developed to aid designers, engineers, and artists. Some of these tools, such as AutoCAD®, are upgraded from 2D drafting tools to include 3D modeling capabilities. Some of the massing tools include additional capabilities such as rendering, animation, and virtual reality, which makes these tools excellent for presentation.

There are three main categories of massing modeling. Each category is based on different techniques. These categories are polygon mesh modeling, non-uniform rational B-spline (NURBS) modeling, and subdivision modeling (see Figure 35).

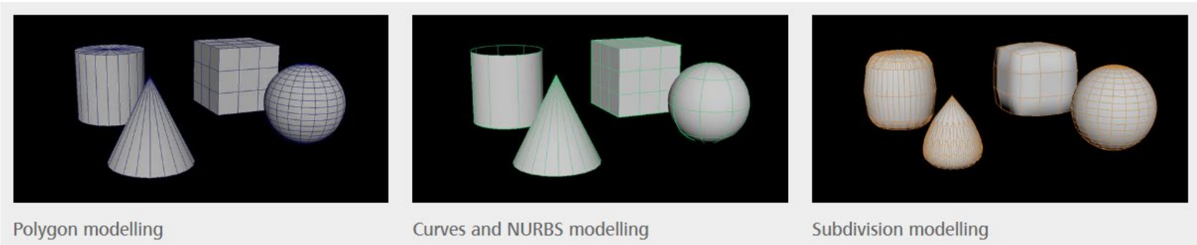


Figure 35: *The types of massing modeling techniques* (Autodesk Inc, 2018)

3.2.1 Polygon mesh modeling

Polygon mesh modeling uses a group of flat polygons to represent the mass surface; the polygons are usually three-sided (triangles) or four-sided (quadrilaterals). The more polygons we use, the more accurate the model is, and the more massive the file size is (see Figure 36). Polygon mesh modeling is unsuitable for smooth surfaces. It cannot create a real smooth surface. However, this modeling technique is relatively easy to create, and control, which makes it perfect when high accuracy is not required, and quick calculations are needed, such as in video games.

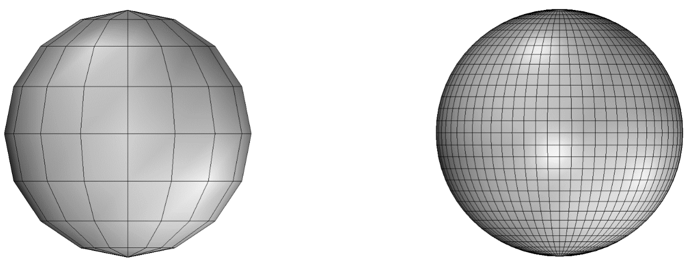


Figure 36: *The relation of the surface divisions counts and the smoothness of the mass*

3.2.2 NURBS modeling

In contrast to polygon mesh modeling, NURBS models are based on curves defined mathematically. Thus, it constructs smooth and accurate models. It is suitable for engineering, product design, and any other purpose where precision is needed. Rhinoceros® (see Figure 37) is one of the most popular NURBS modeling software among architects nowadays.

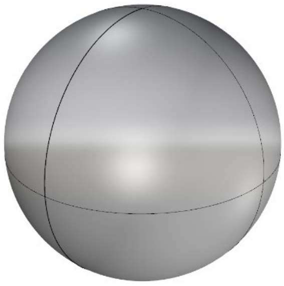


Figure 37: *sphere based on NURBS modeling*

3.2.3 Subdivision modeling

Subdivision modeling has the characteristics of both polygon mesh and NURBS modeling. *"Like NURBS surfaces, subdivision surfaces are capable of producing smooth organic forms and can be shaped using relatively few control vertices. Like polygon surfaces, subdivision surfaces allow you to extrude specific areas and create detail in your surfaces when it is required."* (Autodesk Inc, 2010a). This technique allows the user to control the levels of details and the smoothness of the surface by controlling the number of surface divisions (see Figure 38). It is relatively easy to create and control a subdivision model. Subdivision modeling is widely used in the film making industry because of its characteristics.

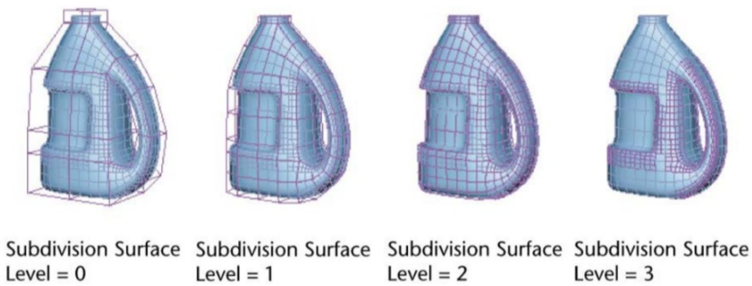


Figure 38: *level of details in Subdivision* (Autodesk Inc, 2010b)

In conclusion, the massing software typology mainly models three-dimensional masses. These masses contain only the geometrical and graphical information of the model parts. Thus, the tools belonging to this category are not prepared for performing physical and functional analysis to evaluate the design. Furthermore, in these models, there are no constraints that link the parts of the models. Using massing software does not serve any of MOIA models.

3.3 Rendering software

Rendering software is mainly developed for creating realistic images or videos of the final design. Some of these tools are stand-alone and can exchange information with the modeling tools through exporting and importing. However, some rendering software is plugins for modeling tools. These tools are usually used after the definition of the final design. The tools of this typology do not serve any of MOIA models.

3.4 Graphics software

Graphics are vital to express ideas in design. The designers use graphics tools in almost every single project. Graphics software is separated into two different categories, raster graphics editors and vector graphics editors.

3.4.1 Raster graphics editor

This category is mainly made for editing images. Raster editors divide the image into small squares that are equal in size; each square is called a pixel. Every single pixel is made of one-color fill and no outline. The resolution of the raster image relays on the number of pixels in the image area; the more the pixels in the area, the smaller the size of the pixel, and the higher the resolution. Thus, increasing the size of an image can lead to undesired pixelization. Many raster file formats are available in the market, for example (PNG, JPEG, TIFF, and GIF).

3.4.2 Vector graphics editor

This category is mainly used for diagraming, illustration, typography, and layout design. The main advantage of vector graphics is that the drawings contain mathematical information. Thus, it allows the user to control the drawing efficiently and precisely in the coordinate system. In contrast to the raster graphics editors, which based on pixels, the user can enlarge the drawings without compromising the resolution (see Figure 39). There are many vector file formats, for example (PDF, EPS, WMF, VML, and SVG). Lately, some vector software started adopting some of the raster software features, and some raster software started adopting vector features.

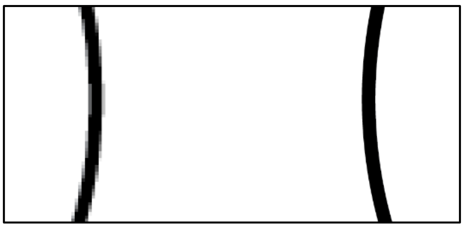


Figure 39: *Pixelization; the enlarged curve on the left represents Raster the other represents Vector*

Both raster and vector graphics are mainly made for producing and editing graphics, including photo editing, typography, and drawings. It is clear that graphics software is not made for design decision making. These tools are not capable of serving any of MOIA models, and they mainly serve presentation purposes.

3.5 Observation software

During the past decades, many observation software was developed for Architecture, Engineering, and Construction (AEC) Industry. The software that belongs to this typology helps to assess the performance of the candidate design solutions. It provides the decision-maker with valuable feedback. Observation is essential for design decision support systems; using these tools is essential for the Observation model of MOIA.

For a long time, integrating observation in designer's workflows, which mainly focuses on modeling, formed a challenge for both the developers and the users. During the last few years, software developers started to focus on solving this problem by integrating observation tools within the modeling tools. For example, in 2015, Autodesk Ecotect®, which is a software for observation, was discontinued. At the same time, Insight®, a new observation tool integrated into Autodesk Revit®, was introduced. Ecotect® case represents a new direction adopted by the developers. They seek to embed the observation tools in parametric modeling such as Building Information Modeling (BIM) platforms; parametric modeling is discussed next. Kensek and Noble emphasize that *"BIM is engaging design analytics in ways that allow architects to make far better performance-based decisions."*(Kensek & Noble, 2014).

Building Information Modeling (BIM) is defined by the National Building Information Modeling Standard (NBIMS) as *"a digital representation of physical and functional characteristics of a facility. A BIM is a shared knowledge resource for information about a facility forming a reliable basis for decisions during its life-cycle; defined as existing from earliest conception to demolition. A basic premise of BIM is collaboration by different stakeholders at different phases of the life cycle of a facility to insert, extract, update or modify information in the BIM to support and reflect the roles of that stakeholder."*(National Institute of Building Sciences, n.d.). Figure 40 presents the BIM Process.

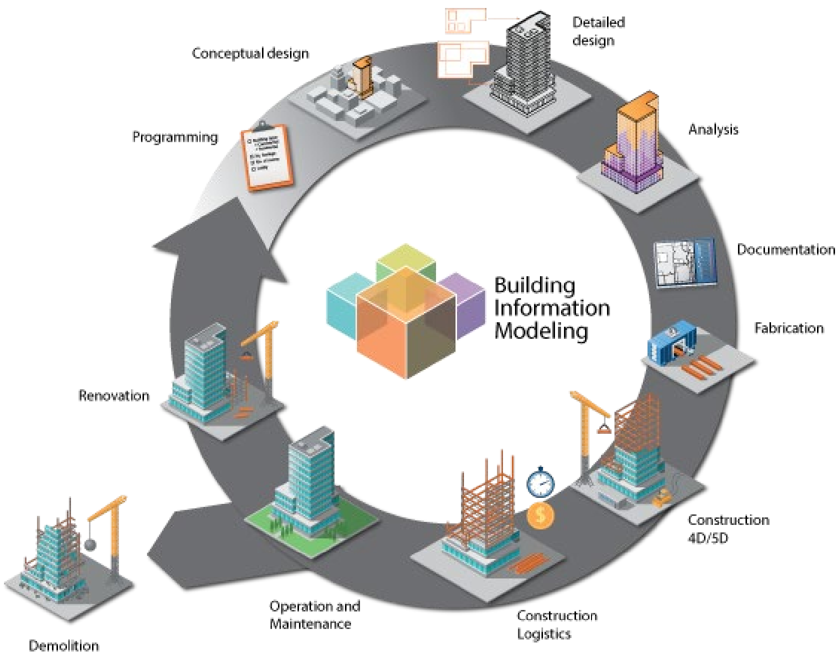


Figure 40: *BIM process* (Narke, n.d.)

3.6 Parametric Modeling

"Design is change. Parametric modeling represents change. It is an old idea, indeed one of the very first ideas in computer-aided design." (Woodbury, Yüce Gün, Peters, & Sheikholeslami, 2010). In parametric modeling, the model's parts are related to each other; changing one part affects the other part. *"Parametric modeling is a general methodology for defining models with constraints and variable parameters"* (Chuck Eastman, Teicholz Paul, Sacks Rafael, 2011). Compared to traditional modeling, change is easy in Parametric modeling because of the relations between the model's parts. It allows the designer to explore many design options easily.

Whitehead states: *"Parametrics is more about an attitude of mind than any particular software application. It has its roots in mechanical design, as such, for architects it is borrowed thought and technology"* (Woodbury et al., 2010). The conventional mediums of design are paper, pencil, and eraser. We use these tools to add or erase marks. The designers shifted to CAD, but the main idea of adding and erasing marks remained the same in non-parametric tools (Woodbury et al., 2010). However, *"Parametric Modeling (also known as constraint modeling) introduces a fundamental change: "marks", that is, parts of a design, relate and change together in a coordinated way. No longer must designers simply add and erase. They now add, erase, relate and repair."* (Woodbury et al., 2010).

The concept of parametric modeling has been adopted in many design disciplines *"In some design disciplines, like mechanical engineering, they are now the normal medium for work. In others such as architecture, their substantial effects started only about the year 2000"* (Woodbury et al., 2010). Nowadays, BIM platforms are one of the most popular parametric modeling tools among building designers. *"Technologies, that allow users to produce building models that consist of parametric objects are considered BIM authoring tools"*(Chuck Eastman, Teicholz Paul, Sacks Rafael, 2011).

BIM platforms such as Revit® and ArchiCAD® are growing very fast. Indeed, they are the building industry standard. Kensek and Noble highlights, *"much more so than CAD, BIM is revolutionizing the way the building partners practice and document their work, even changing the nature of the design process."* (Kensek & Noble, 2014). By adopting BIM, the building industry became more efficient, *"when adopted well, BIM facilitates a more integrated design and construction process that results in better quality buildings at lower cost and reduced project duration."*(Chuck Eastman, Teicholz Paul, Sacks Rafael, 2011)

Figure 41 presents the interface of Revit® to illustrates the concept of parametric modeling. The figure shows a list of variable parameters (type properties) of a door. The variable parameters of the door are connected to the variable parameters of the wall. In this example, by changing the width of the door, the wall opening will adapt to fit the new width. If we increased the thickness of the wall, the thickness of the door frame would adapt to the new wall thickness. This happens because of the constraints that define the relationship between the parameters of the door and the wall.

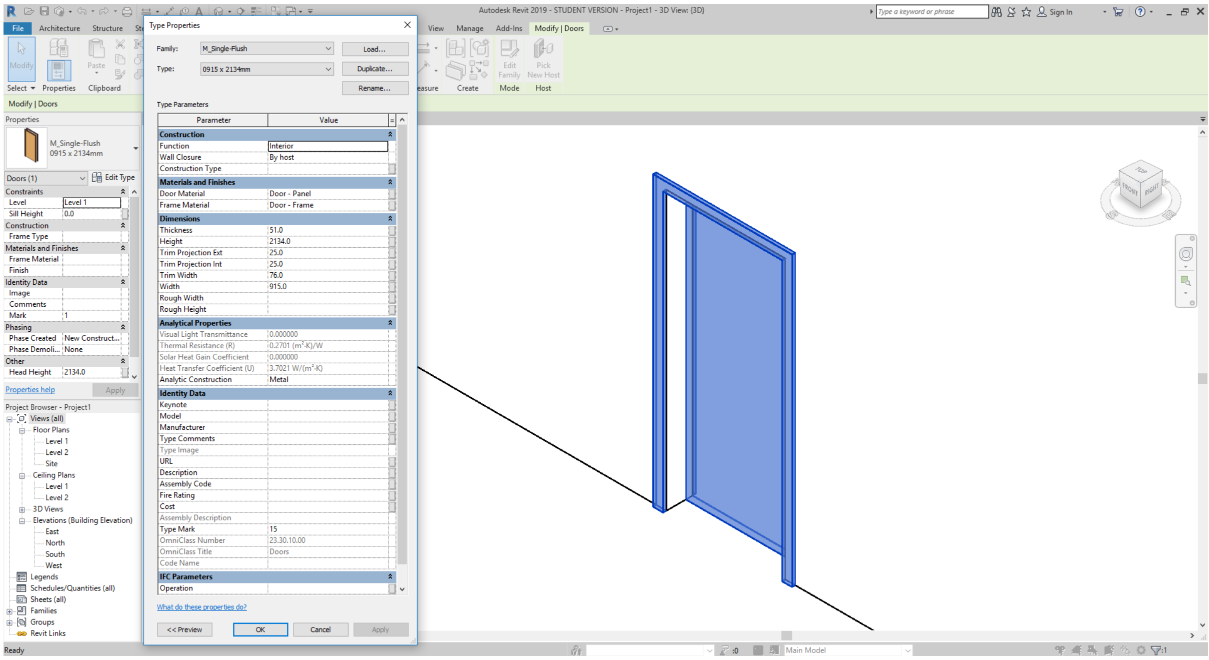


Figure 41: *controlling the model's properties through its parameters in Revit®*

Parametric modeling is highly beneficial for MOIA as it allows us to define the design variables **x** in the Morphogenesis model. Parametric modeling facilitates exploring more options by manipulating **x**. In conventional modeling, the model's parts are not connected, and exploration is challenging. Some parametric tools can observe design solutions. These tools usually allow the designer to define the design scenario, which is essential for the Observation model of MOIA.

However, defining a model with a complex set of constraints and variable parameters is complicated and sometimes limited. Another method that allows us to construct and control such complex relations between the model parts are required. To solve this problem, we use algorithmic modeling, which is an advanced parametric modeling technique. It allows us to define parametric models through algorithms. In what follows, the main features of algorithmic modeling are discussed.

3.7 Algorithmic Modeling

Algorithmic Modeling is significant for building a sophisticated structure of connections between model parts. However, in computers, we write algorithms through a programming language. The designers are not programmers *"It is well-known that conventional programming languages are difficult to learn and use, requiring skills that many people do not have"*(Lewis & Olson, 1987). However, to make programming accessible for designers, we can use visual programming. *"It seems clear that a more visual style of programming could be easier to understand and generate for humans, especially for non-programmers or novice programmers"*(Myers, 1990). Visual Programming (VP) tools can help the designers to define complex parametric models.

Software developers have introduced many VP tools for designers; during the last few years, those tools became quite popular among architects. *"In recent years many software* *houses have developed visual tools in order to make scripting more accessible to users with little to no programming skills."*(Tedeschi et al., 2014). VP allows the designers to build algorithms by graphically connecting pre-made basic formulas. These tools contain most of the mathematical operations needed. Many formulas can be easily installed as a plugin. Mixing VP with textual programming is possible in these tools. The result of the algorithm implemented in these tools can appear directly as a 3D digital model.

Two of the most popular VP tools among architects nowadays are Grasshopper® and Dynamo®. Grasshopper® (Figure 42) is a VP environment originally made for Rhinoceros®; thus, they can easily exchange information. A very diverse library of plugins supporting Grasshopper® is available. On the other side, Dynamo® (Figure 43) is a VP plugin for Revit®. However, a standalone version of the tool is available. Dynamo® can use the information contained in the BIM models of Revit®. Dynamo® enhances the parametric capabilities of Revit®. These tools usually integrate limitless observation tools and methods, which can give insight to the decision-maker. VP is excellent for defining the design variables **x** and the design scenario in both the Morphogenesis and the Observation models of MOIA.

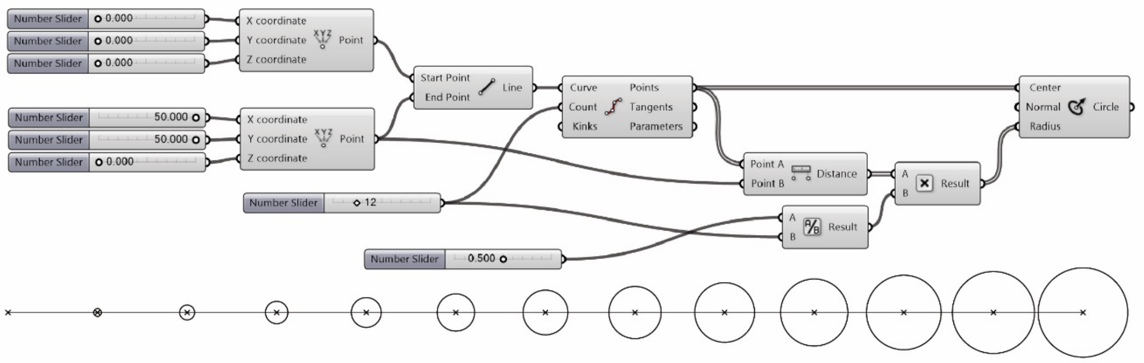


Figure 42: *Grasshopper® algorithm developed to control the radius of many circles based on their distance from a point.*

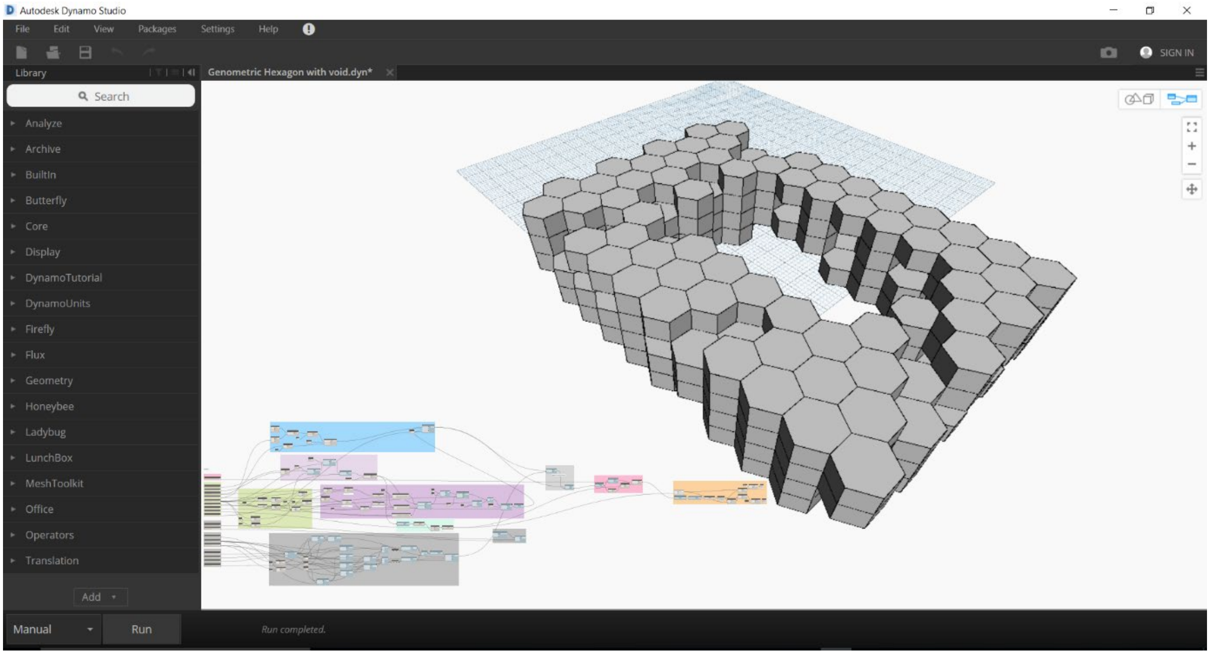


Figure 43: *Dynamo® algorithm that controls a vast number of hexagonal units*

3.8 Generative Design

In parametric and algorithmic modeling, the designer defines the design model via a set of design variables **x** and constraints. Based on parametric modeling, the generative tools manipulate the variables with respect to the constraints to generate many design options from one model. However, randomly manipulating **x** values is inefficient. Hence, combining observation tools and generative parametric modeling is essential. The observation tools compute the observation variables **y** of each version of the design (candidate solution characterized by **x**). Both **x** and **y** are then used to guide the evolution towards better design solutions. This process is the core concept of generative design (see 1.2.3).

This combination that can generate optimized design solutions is essential for the design form-finding process. Indeed, *"They shift the emphasis from (form-making) to (form-finding)"*(Agkathidis, 2015). These tools are designed to help the design decision, they are recognized as decision support systems. *"Computational tools have introduced innovative form-finding techniques, revolutionizing architectural design and production"*(Agkathidis, 2015). However, only recently, a few numbers of workflows adopting this approach were introduced to the market. These workflows are still limited and not widely used among architects. It seems that the workflows based on generative design are the closest existing workflows to serve MOIA models.

CHAPTER 4 Decision support workflows

"Design is nothing if not decision making." (Petroski, n.d.)

Using design frameworks such as MOIA provides us with better design outcomes. To execute such frameworks for approaching specific problems, we use workflows. The design frameworks are broad and can be used to approach various problems in different design disciplines. In contrast, a workflow is a detailed executive plan of a design framework to approach specific tasks in a particular design discipline. In one workflow, many tools can be involved. *"Since, in architecture, the design process is not confined to a single stage of a project, an architect will probably use many different tools to achieve the necessary work for each phase. There are some architects who, based on their approach to design, spend a great deal of the design process using computer software, but their actual process (the steps or stages) may seem very similar to that of an architect who sketches and draws using traditional tools. What we must recognize is that the design process is not specific to the tools that are used, but to the architect approach."*(Makstutis, 2018)

While sometimes a single tool seems useless, within a workflow, it can be significant. Reviewing each tool independently can be misleading. However, reviewing the tools within the context of the workflow is more useful. Based on MOIA and after extensive exploration of many design processes and tools typologies, this chapter intends to investigate different decision support workflows for architects. The selected workflows are capable of generating design options, evaluating them, and evolving better options based on this observation autonomously. Additionally, the chosen workflows are capable of helping architects during the early stages of the design process.

4.1 Workflow 1

Workflow 1 consists of using Grasshopper® and its generative evolutionary solver Galapagos©. It allows the designers to define design variables **x** and design scenarios, which represent the observation variables **y**. Many observation tools are available for Grasshopper®. Some of these tools can interpret the observed values. However, this interpretation is usually strict; (0 (not satisfied), 1(satisfied)) or (-1(not satisfied), 0(satisfied), 1(not satisfied)). This interpretation is not suitable for high in negentropy aggregation functions.

Galapagos© evolves the design variables **x** by linking it to the satisfaction level of a single objective. It seeks to evolve the fitness landscape toward optimizing a single objective by using EA (see Figure 44). The fitness landscape consists of all the observed candidate solutions. Rutten explains, *"The term "Evolutionary Computing" may very well be widely known at this point in time, but they are still very much a programmer's tool. 'By programmers for programmers' if you will. The applications out there that apply evolutionary logic are either aimed at solving specific problems, or they are generic libraries that allow other programmers to piggyback along. It is my hope that Galapagos will provide a generic platform for the application of Evolutionary Algorithms to be used on a wide variety of problems by non-programmers."*(Rutten, 2011). However, this workflow does not involve any methods to compute this single objective from different fitness (observation variables). In architectural design, several fitnesses are usually involved.

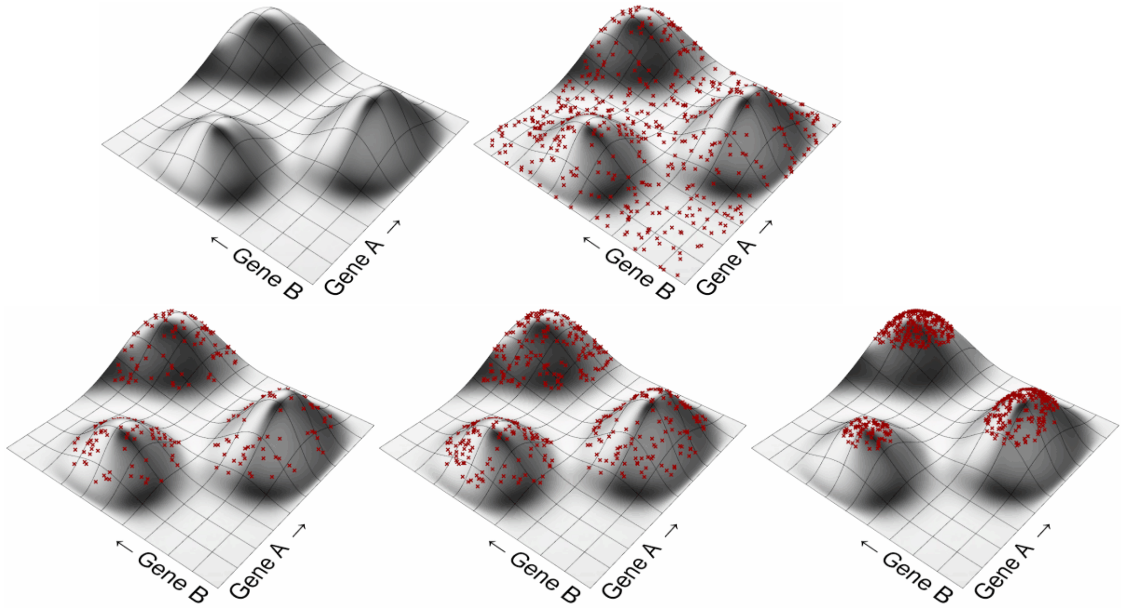


Figure 44: *Different generations of fitness landscape in evolutionary solvers* (Rutten, 2011)

Figure 45 shows the interface of Galapagos©. Figure 46 demonstrates the results of two tests performed to optimize the location of the building (A) regarding solar gain based on this workflow. In the figure, the example on the left minimizes solar gain, and the example on the right maximizes it. Because this workflow uses Grasshopper®, thus, the designer can watch the form of each solution in Rhinoceros®.

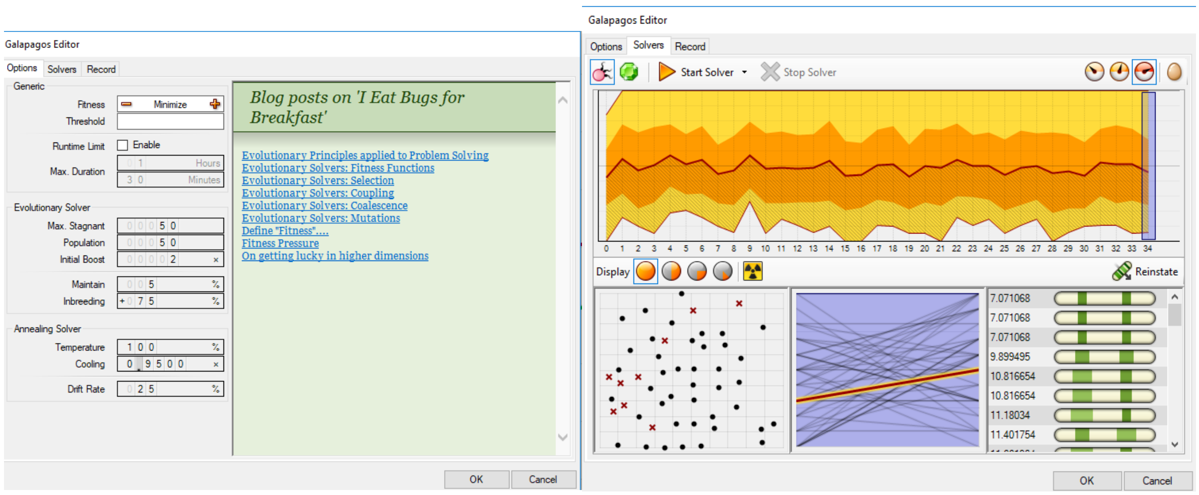


Figure 45: *Galapagos© interface; left (option panel), right (solver panel)*

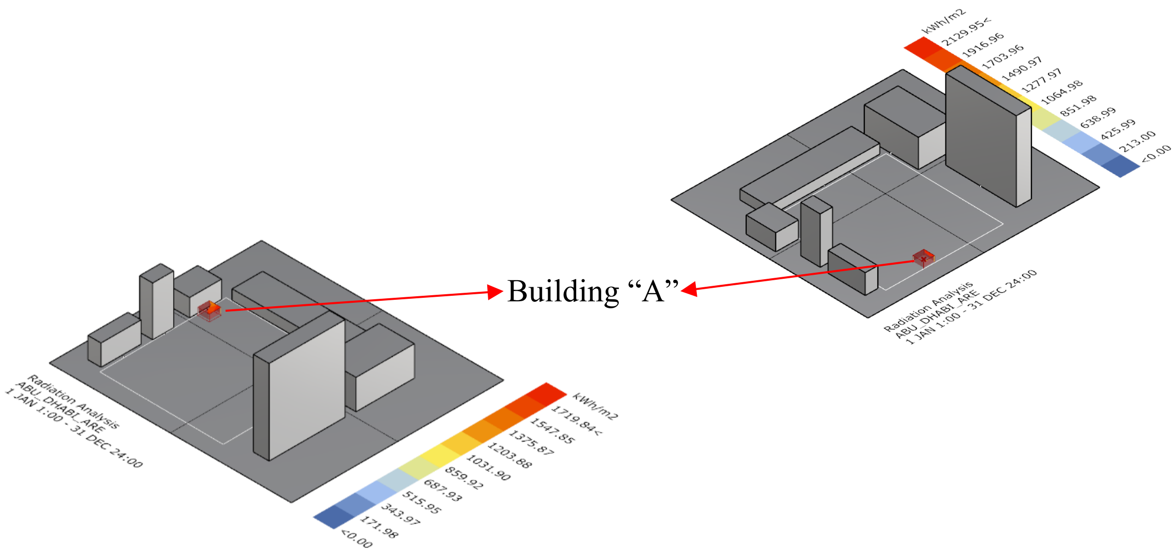


Figure 46: *Optimizing the position of a building “A” based on observing the solar gain; left (minimization, right (Maximization).*

4.2 Workflow 2

Workflow 2 consists of using Grasshopper® VP and its multi-objective optimization tool Octopus©. Vierlinger defines, *“Octopus is a plug-in for applying evolutionary principles to parametric design and problem-solving. It allows the search for many goals at once, producing a range of optimized trade-off solutions between the extremes of each goal.”*(Vier, 2019). He explains, *“Octopus introduces multiple fitness values to the optimization. The best trade-offs between those objectives are searched, producing a set of possible optimum solutions that ideally reach from one extreme trade-off to the other.”*(Vier, 2019). Octopus© generates solutions by respecting the constraints and manipulating the design variables **x**, which is predefined by the designer in Grasshopper®. Then, each candidate solution is observed by using one of the many tools and methods available in Grasshopper®. Finally, Octopus© uses Pareto’s function (see 2.3.1) to aggregate the solutions based on the observed variables **y** (Vierlinger, 2013). This process is iterative and autonomous.

Octopus© uses a 3D coordinate system to visually represent the candidate solutions’ classification (Vierlinger, 2013) (see Figure 47). Each of the three axes represents one observation variable. Every candidate solution is represented in the coordinate system by a point. It is possible to replace the points with the forms of solutions. By default, the points use color code ranging between light green and dark red to indicate the solution class; dark red represents non-dominated “Pareto front” solutions. In addition to the solutions’ position within the scatterplot, the user can use sizes and colors to represents the objectives. Octopus© allows the user to isolate the elite solutions and non-dominated solutions. It also can represent the solutions in a multi-axis parallel coordinates view.

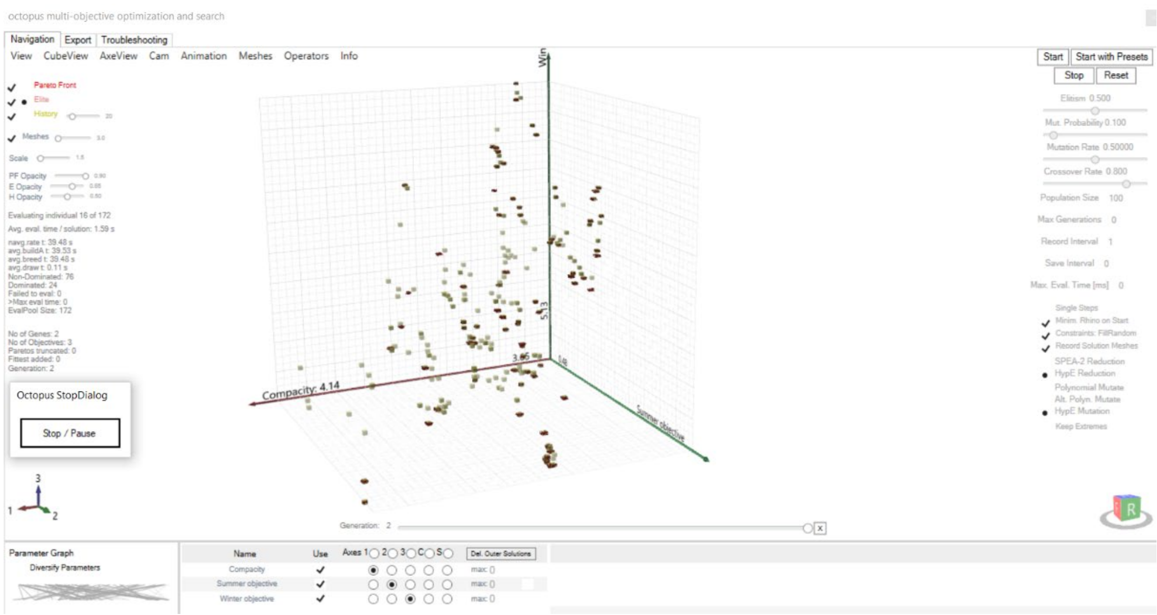


***Figure 47:** Octopus© user interface*

Using a parallel coordinate for Grasshopper®, such as Design-Explorer© (see Figure 48), is useful to filter the candidate solutions in this workflow. *"Parallel coordinates is a general-purpose visual multidimensional coordinate system."*(Inselberg, 2009). The parallel coordinates are a robust method for exploring, sorting, and filtering a large number of solutions based on the defined design variables **x** and observation variables **y**. While the scatterplot can represent up to three variables, parallel coordinates allow to represent an unlimited number of variables visually. It can filter the candidate solutions very efficiently. However, if only three or fewer objectives are involved, the scatterplot can be more informative, *"Despite the popularity, the parallel coordinates plot is not as straightforward as the scatterplot in presenting the information contained in a solution set. Due to mapping multi-dimensional data onto a lower 2D space, the loss of information is inevitable."*(Li, Zhen, & Yao, 2017)

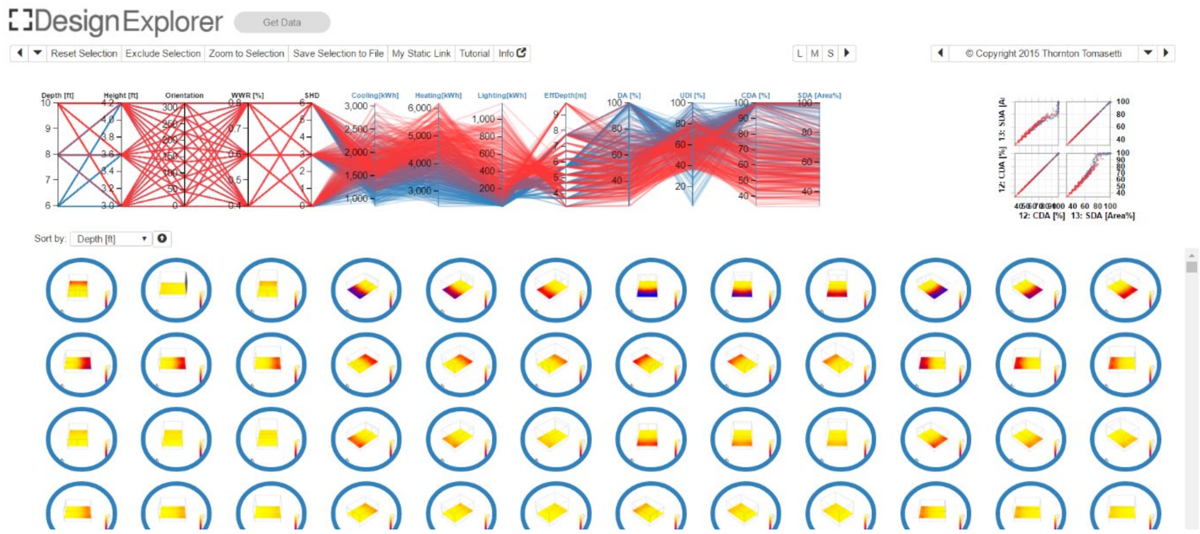


***Figure 48:** Design-Explorer© page* (Tomasetti, 2017)

4.3 Workflow 3

Workflow 3 consists of using Grasshopper® VP to define the design variables **x** and observation variables **y**. Then we use Biomorpher© to evolve new **x** values. Harding describes Biomorpher© as; *"Interactive Evolutionary Algorithms for Rhino Grasshopper."*(Harding, n.d.-a). He explains, *"As opposed to setting objective functions (As with Galapagos for example), Interactive Evolutionary Algorithms (IEAs) allow designers to engage with the process of evolutionary development itself. This creates an involved experience, helping to explore the wide combinatorial space of parametric models without always knowing where you are headed."*(Harding, n.d.-a).

In Biomorpher©, four different types of inputs from Grasshopper® are used (genome, geometry, performance, initial population). The genome is the design variables **x**. The geometry is the candidate solutions' 3D forms. The performance is the observation variables **y**; it is computed from assessing the solutions based on the fitnesses by using one or more of the observations approaches available in Grasshopper®. Finally, the initial population is an optional input that can be used to start the generative process with non-random values of **x**.

In this workflow, the designer first defines the variable parameters, constraints, and scenarios in Grasshopper®. Then, in Biomorpher©, the user defines the population size and the mutation probability (see Figure 49) to compute the first generation (see Figure 50). Once the first generation is computed, the designer can define a target (no optimization, maximize, minimize) (Figure 50) to evolve the solutions. On top of each evolved solution, circles that represent the satisfaction level of each objective are presented (see Figure 50). The user can select the preferred candidate solutions to direct the next generations; Biomorpher© use IEAs. This process can be repeated for evolving many generations. Eventually, the user can select parents from different generations to evolve new solutions (see Figure 51).

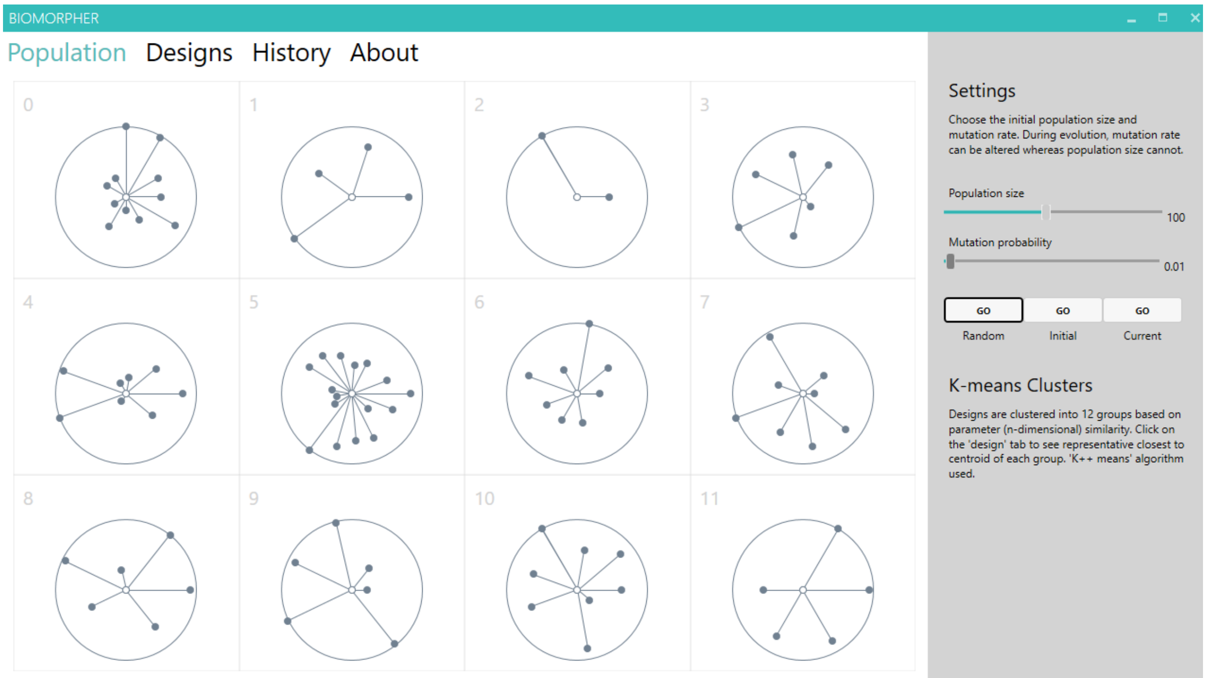


Figure 49: *Population in Biomorpher©, each circle represents a cluster of similar solutions*

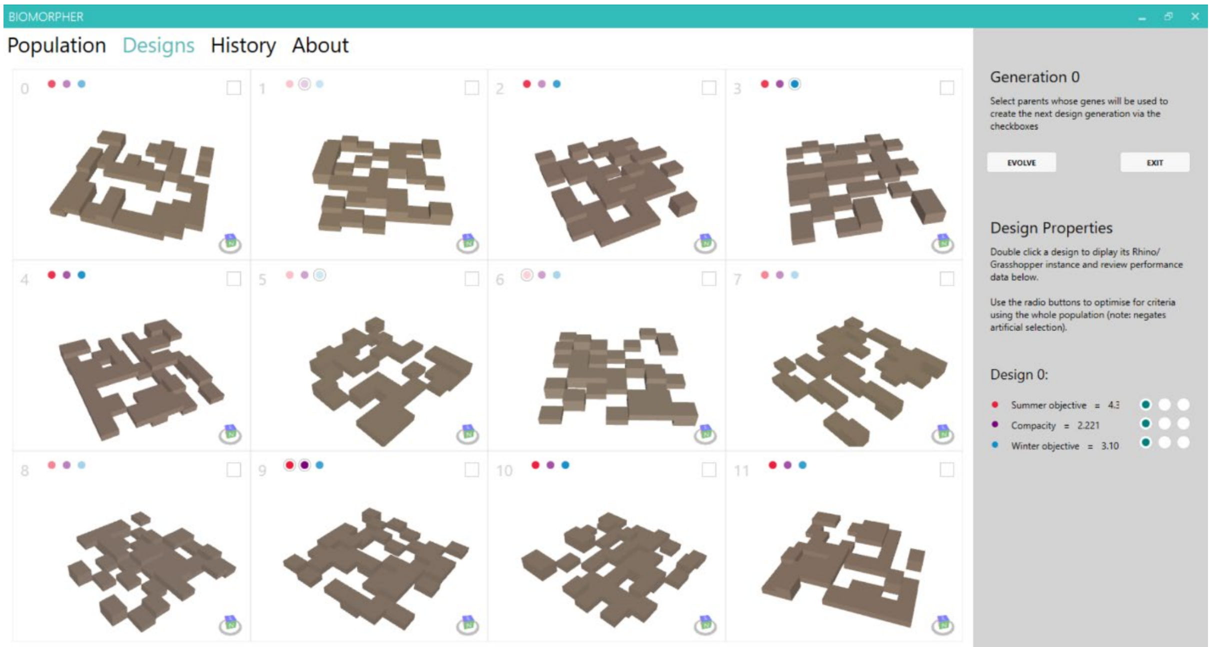


Figure 50: *Candidate solutions in Biomorpher©*

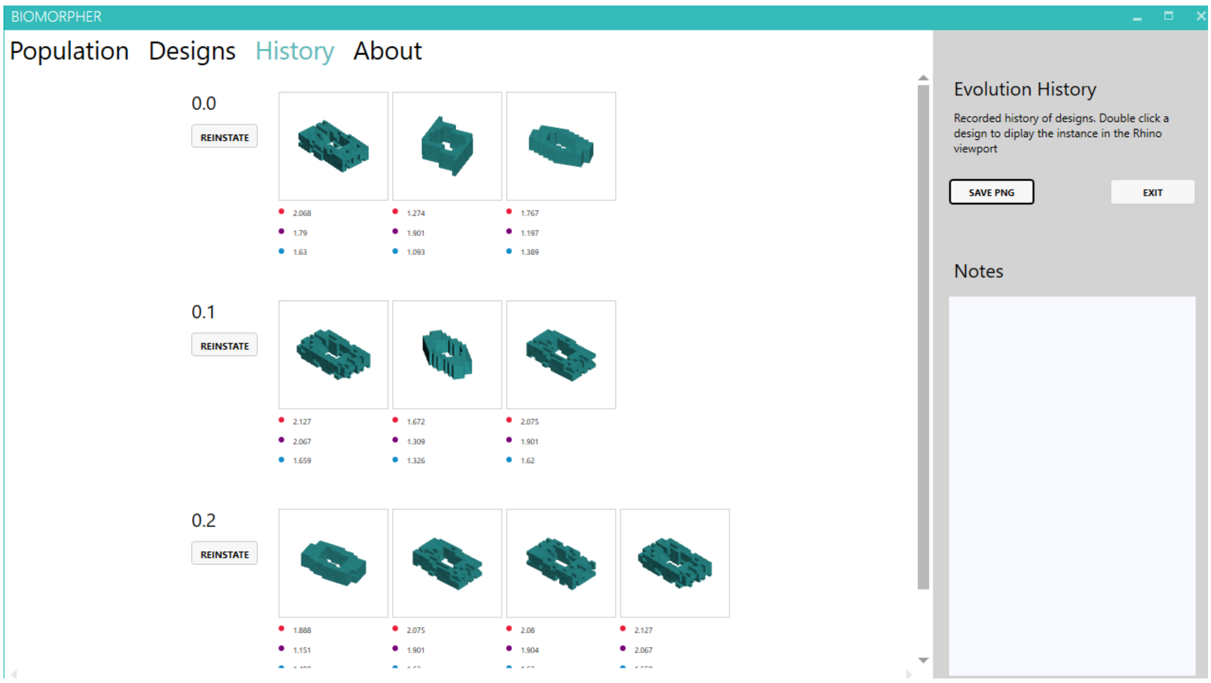


Figure 51: *Different generations in Biomorpher©*

To classify the candidate solutions, fitness ranging from (0.0 to 1.0) is assigned for each phenotype (Harding, n.d.-b). The fitness is then used in a roulette wheel selection (fitness proportionate selection) (Goldberg, 1989; Harding, n.d.-b) to evolve new solutions. Elitism is not used in this process. According to Harding, to compute these values, Biomorpher© uses a process that consists of several steps (Harding, n.d.-b):

1. *All fitnesses are reset to zero.*
2. *If a manual selection is made (i.e., the tickbox checked), then all designs in that cluster will have fitness set to 1.0 regardless of performance-based fitness. They can go no higher.*
3. *If no performance-based criterion is specified, go to selection.*
4. *If one performance-based criteria is specified, then normalise performance values for the whole population and assign this as the fitness. If fitness is already 1.0 (due to manual selection) then do nothing. Note that if you are minimizing a performance value, we normalize and take 1-x of course.*
5. *If two or more performance-based criteria are used, then simply normalise performance values, sum these and divide by the number of criteria.*

This classification method can result in solutions that are inferior in performance because the selection in IEA allows the user to select solutions based on subjective opinions and not only based on the performance. Harding clarifies, *"this is different to typical multi-objective optimization methods in that you could in theory evolve something that is poorly performing for all performance-based objectives (i.e., nowhere near a Pareto front), simply because you like the look of it."* (Harding, n.d.-b).

4.4 Workflow 4

Workflow 4 consists of using Dynamo® VP and its generative tool, Refinery®. Dynamo® allows the designers to construct parametric models defined by design variables **x** and design constraints. It also allows the designer to use many observation tools and methods. Once the designer defines **x** and the scenario, Refinery® is used to evolve a new solution by linking design variables **x** values to the design variables **y** values. Refinery® can optimize multi-objective based on many fitnesses simultaneously. It classifies the solutions based on Pareto's function according to the observation variables **y**.

Refinery® allows the designers to choose from four different generation methods (optimize, cross product, randomize, like this). The tools offer three different options to regard the observation variables (ignore, maximize, minimize). Before the generative process starts, the user must specify the population size, the number of generations, and seed value to control the randomization. Once all these choices are assigned, Refinery® can start evolving design options.

Figure 52 presents the interface of Refinery®. On the left side of the interface, a list of the different studies is located. On the right side, a group of thumbnails that represent the generated solution is located. On top of the thumbnails, a scatterplot represents the solution's performance. The user can choose one objective for each of the plot's two axes. The user can easily change the objectives represented in each axis to compare the solutions based on different objectives. The position of the solutions within the scatterplot is not the only option to compare the solution visually. The user can use color and size to illustrate the performance of the objectives. Eventually, the user can use a parallel coordinate in Refinery® to filter the results (see Figure 53).

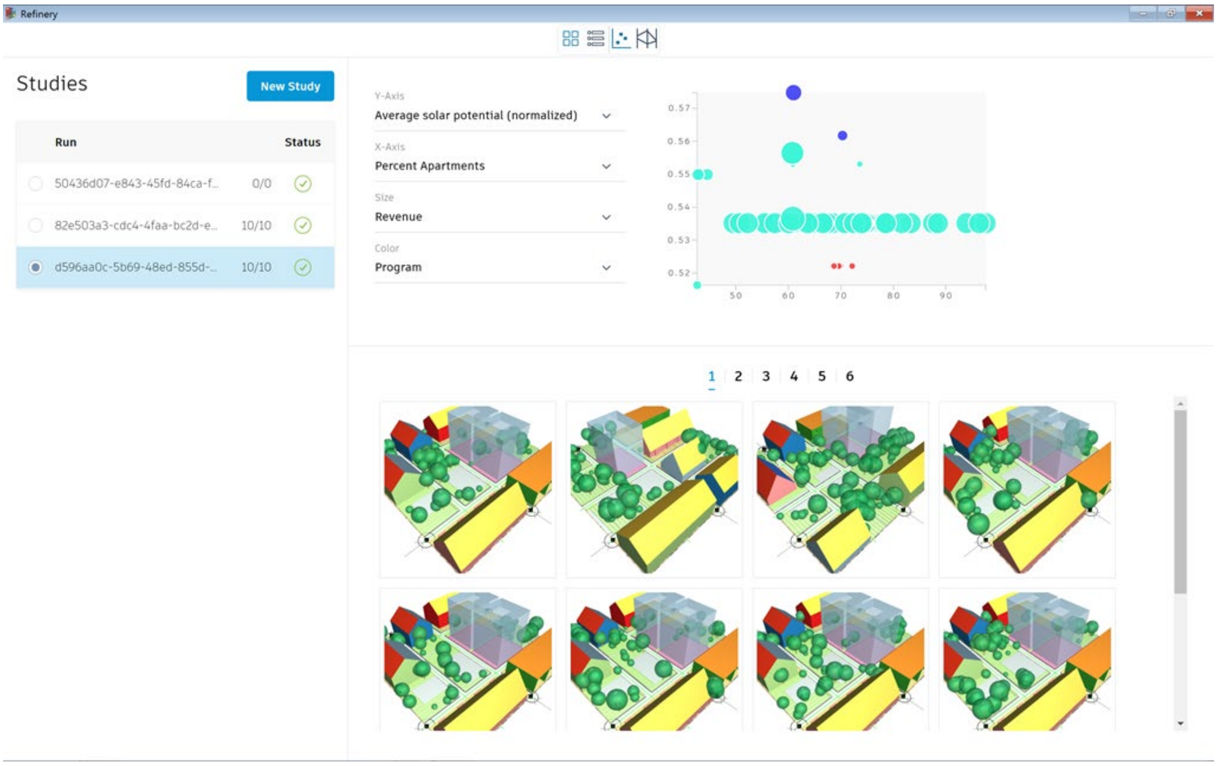


Figure 52: *Refinery® interface (A) scatterplot* (Walmsley, n.d.)

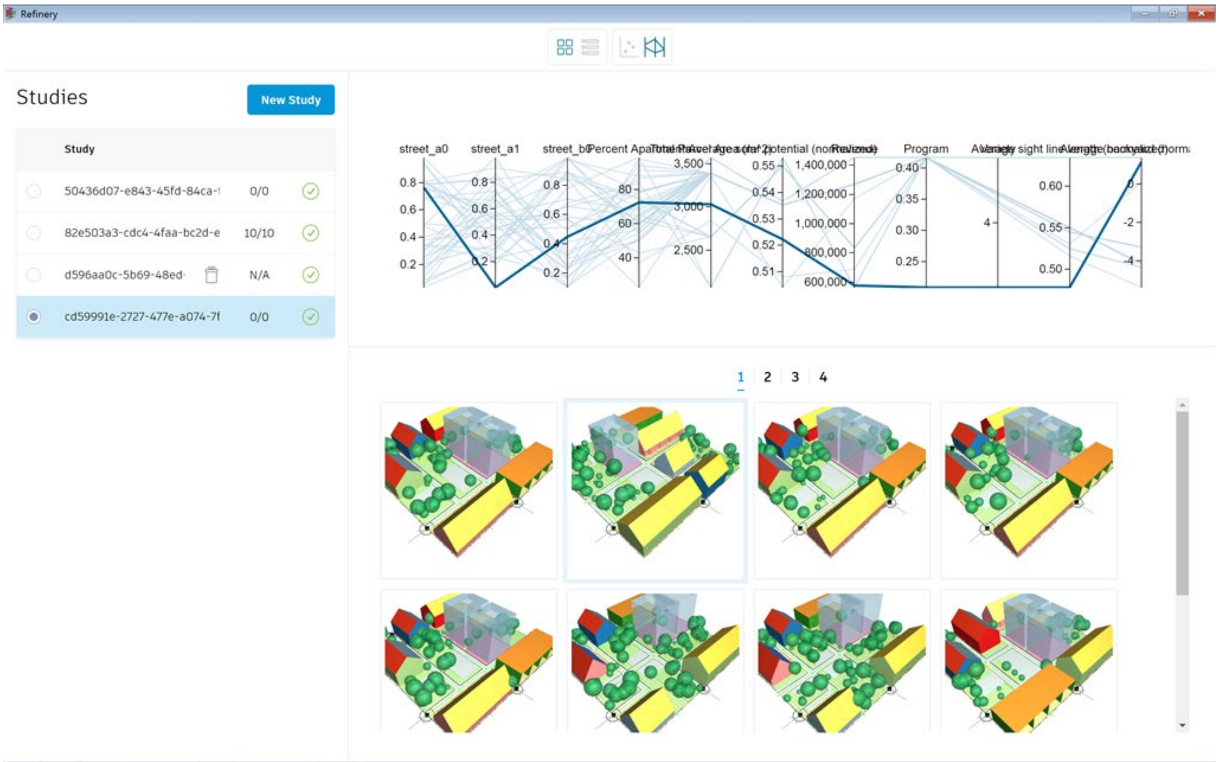


Figure 53: *Refinery® interface (B) parallel coordinate* (Walmsley, n.d.)

4.5 Workflow 5

Workflow 5 consists of using Dynamo® VP and its generative optimization tool Optimo®. Dynamo® allows the designer to define design variables **x** and constraints. It also allows the designer to use many observation tools and methods. Once the designer defines the parametric model and the scenario, Optimo© is used to generate new solutions by evolving the values of **x** by linking the original **x** to **y**. Optimo© can optimize multi-objective based on many finesses simultaneously. Optimo© classifies the solutions via Pareto's function according to the observation variables **y**. *"Optimo - a BIM-based multi-objective optimization tool - was developed to enable rapid building performance optimization in the process of design. Optimo is an open-source application for parametrically interacting with BIM models for design optimization. Optimo provides the option to optimize multiple objective functions with respect to multiple parameters and works based on the Nondominated Sorting Genetic Algorithm-II (NSGA-II) (Deb et al., 2002)"* (Deb, Pratap, Agarwal, & Meyarivan, 2002; Rahmani ASL, 2015). It is recommended by Rahmani, the developer of the tool, to use an interactive parallel coordinate to narrow the options *"Visualizing the results in an interactive parallel coordinates plot allows the various iterations to be evaluated by the designer. "* (Rahmani ASL, 2015).

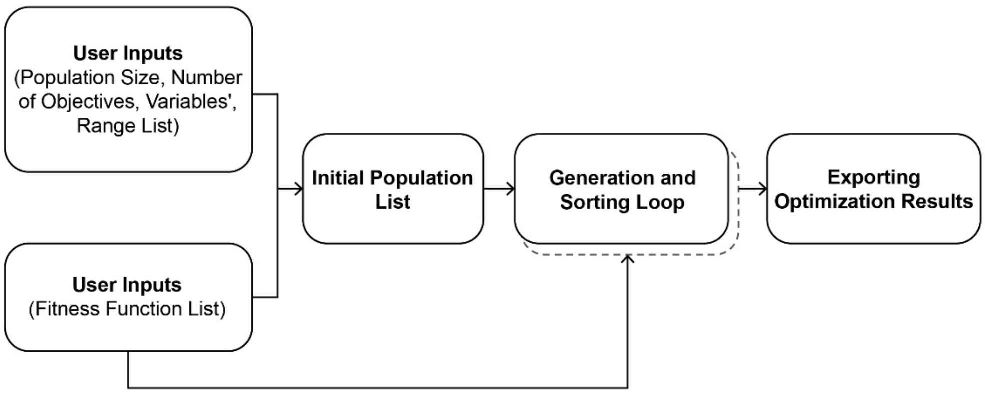


Figure 54: *Optimo© structure* (Rahmani ASL, 2015)

4.6 Workflow 6

In this workflow, we use EcoGen© to generate optimized modular buildings. In his book, Marsault, the researcher at MAP-Aria based in Lyon and the leading developer of EcoGen©, defines *"EcoGen is a software wizard for architectural eco-design, a source of proposals and analytical data, assisting the designer in the creation phase. Its components are designed to reduce the disconnect between the post-design creation and optimization phases by means of a continuous and gradual process."*(Marsault, 2018).

EcoGen© helps the architects and urban designers during the early stage of the design process to find ecologically optimized modular building's forms. It uses a genetic algorithm to generate modular solutions that respond to predefined objectives. *"EcoGen belongs to the family of generative software tools based on population evolution. Its principle is to iteratively generate a number of solutions, using two engines: one morphological, the other genetic. Some solutions, deemed effective, are crossed with each other and/or mutated to generate new ones, which will then be assessed based on certain criteria chosen at the outset by the user and, of course, modified depending on the results obtained."* (Marsault, 2018).

The software uses Pareto's function to classify the candidate solutions based on their performance. Therefore, EcoGen© is capable of optimization multi-objective simultaneously. *"Each time EcoGen is launched for the same initial site and program data, the random generator is initialized with a different value. This makes it possible to obtain an approximation of the Pareto front in just a few runs via various convergence trajectories and therefore to temporarily."*(Marsault, 2018).

To initiate the generative process of EcoGen© (v2.1), the user must specify some settings. First, the user must identify the objectives fitnesses intended to be optimized. Currently, four different objectives are available (compacity, heating, solar gain, daylight factor). However, two other objectives are under development (life cycle cost, economic cost). Additionally, the user must specify the percentage of the different program types (bright, blind, luminous). This helps evaluate heating, solar gain, and daylight factor accurately. The user must also specify the voxel size (width, length, height), number of floors, targeted floor area, and tolerance limit for the targeted area. The user can also specify the number of steps (generations). Once all this information is specified, the user can run the calculation.

According to the developer, *"EcoGen attempts to achieve two objectives permanently: searching through a vast number of diversified solutions and, at the same time, increasing the efficacy of the families of solutions that seem best adapted to the situation"*(Marsault, 2018). Once the candidate solutions are computed, the user can specify the ones that reflect his personal preference. The preferred candidate solutions help to orient the evolution; EcoGen© uses an Interactive Genetic Algorithm (IGA). *"EcoGen can work in autonomous mode (without human intervention, except for pause or stop), and, ultimately, propose a list of optimized solutions for the program and criteria selected by the user. However, it can also work in assisted mode (interactive). In this case, each time the partial results are consulted, the user can tell it which solution/s of those displayed he is interested in, depending on subjective criteria (morphological, esthetic) or objective criteria (efficiency-based, constructive, functional), and thus guide evolution in one or more preferred directions."* (Marsault, 2018)

Figure 55 demonstrates the main interface of EcoGen©. Figure 56 shows the perspective mode of EcoGen©. In this mode, the designer can explore each candidate solution separately within its context from different angles. Moreover, in this mode, the values of the observation variables **y** are presented numerically.

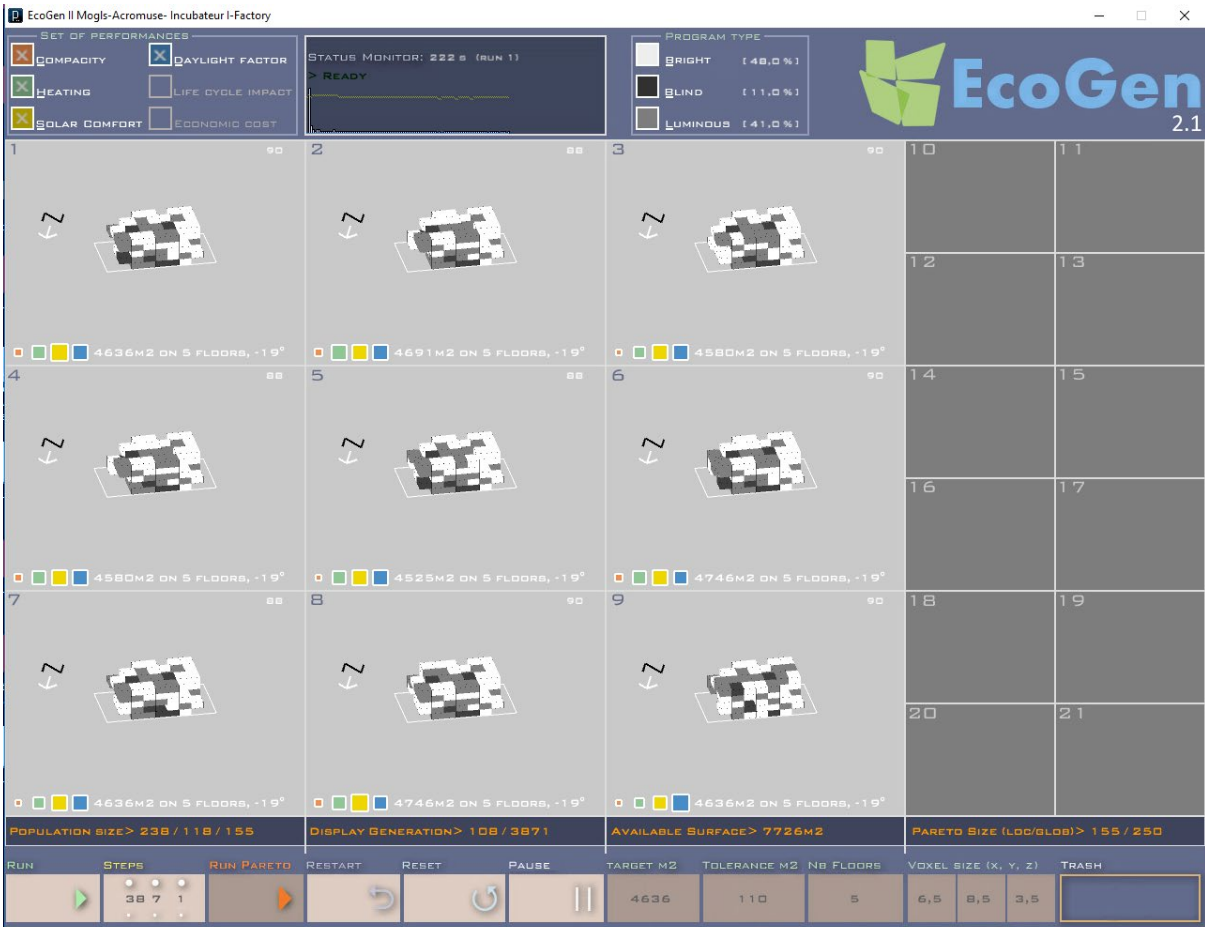


Figure 55: *EcoGen© (version 2.1) user interface*

Figure 56: *EcoGen© (version 2.1) perspective mode*

Compared to other workflows presented earlier and based on VP, EcoGen© is fast and easy to use. However, In EcoGen©, the significant part of **x** and **y** is predefined. It always generates modular form based on random identical voxels. The observation is limited to only four predefined performance fitness (six in the future). Compared to the workflows that are based on using VP, this can be a significant disadvantage as it limits the role of the designers during structuring the problem, which can restrict their creativity.

4.7 *Acceptimality* and decision support systems

In this chapter, different decision support workflows for designers have been reviewed. Linking these workflows to MOIA helps to identify the difficulties they face, which prevents them from achieving *acceptimality*.

Workflow 1 is only capable of approaching design problems with a single objective. In design, to use a single objective, this objective must be global. This workflow does not propose any method to compute a global objective from many observation variables **y**. For example, in MOIA, the GDI is a global objective computed from multiple **DOI** and criteria.

The main difficulty of Workflow 3 is that it does not adopt a reliable method to approach optimality. Therefore, it can result in solutions that are inferior in performance. Furthermore, there is no reliable interpretation method the designer can use to express his personal preference of the criteria and the objectives. However, this workflow adopts a remarkable approach for selection that increases the interaction between the tools and the designer. In MOIA, the selection occurs within the Morphogenesis model.

Workflows 2, 4, 5, and 6 use a similar approach. Thus, they share similar difficulties. These workflows use EA to generate solutions; workflow 6 uses interactive IEA. Then, they observe the generated candidate solutions. Next, they use Pareto's function to aggregate the observed solutions. Based on the aggregation, EA is used to evolve new solutions. These workflows are the closest to MOIA among the discussed workflows. However, they focus on approaching optimality and not *acceptimality*.

Figure 57 represents the framework adopted by Workflows 2, 4, 5, and 6. Comparing this framework to MOIA (see Figure 14 (Chapter 2)) is essential to understand the difficulties that confront these workflows. From this comparison, we can recognize that this framework does not allow the designer to express his preference in the optimization process. Consequently, these workflows cannot approach *acceptimality*. This problem occurs because these workflows use Pareto's function. This function is ordinal and low in negentropy, as mentioned before. Consequently, it does not allow the designer to express his personal preference for the objectives. Furthermore, it classifies the solutions based on the observation variables **y** without interpretation; the designer is not allowed to express his preference of the criteria.

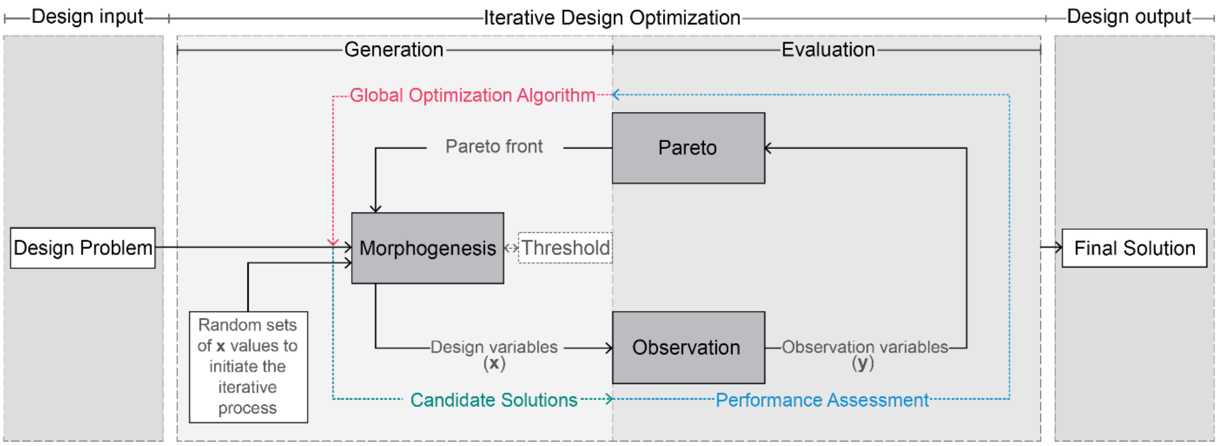


Figure 57: *The design framework adopted by Workflows 2, 4, 5, 6*

Replacing Pareto's function with other aggregation functions that allow the designer to express his preferences can result in decision support systems for approaching *acceptimality*. The high in negentropy cardinal functions such as Maximin or Derringer & Suich's aggregation functions can be a suitable alternative to Pareto's function. Investigating the different aggregation functions is highly recommended. Such investigations can help in developing decision support systems based on *acceptimality*.

As mentioned before, Workflows 2, 4, 5, and 6 share the same framework. However, based on the tools' typologies they use, they are divided into two categories. The first category includes Workflows 2, 4, and 5 and is based on VP, and it uses GA. The second category is Workflow 6, this workflow is not based on VP, and it uses an IGA. Comparing these categories is also recommended.

CHAPTER 5 Tools and workflows

"If we knew what it was we were doing, it would not be called research, would it?" (Einstein, n.d.)

The support systems available in the market for multi-objective design optimization are mainly adopting a similar approach based on generative design. They usually start from a set of candidate solutions based on random values. Then, they observe these candidate solutions. Next, they use Pareto's function to classify the solutions based on design objectives. Finally, an optimization algorithm is used to evolve new solutions. In chapter 4, four different workflows adopting this approach are presented (Workflows 2, 4, 5, and 6). However, these workflows are not identical as they are based on different tools. As mentioned earlier, we can separate these workflows into two different categories. One category is based on VP and uses GA (EA) (Workflows 2, 4, and 5); the other category is based on non-VP and uses IGA (IEA) (Workflow 6). In this chapter, we adopt an experimental approach to compare designers' acceptability of these workflows based on the tools types and the interface they use.

5.1 Experiment 1: comparing workflow 2 and 6

This experiment intends to study and compare users' acceptability of the two different categories, presented before. The aim is to study the acceptability of the workflows based on user experience by comparing Workflow 2 and Workflow 6.

5.1.1 Methodology

Samples of architects and engineers, from the domain of building design, are invited to participate in a work session that explores both categories represented by Workflow 2 and Workflow 6. The work session is divided into four consecutive parts, presentation, time of guided manipulation, time of autonomous manipulation, and an individual questionnaire. The work session is performed four times at different dates with different groups.

5.1.1.1 Presentation

The work session starts with a presentation consisting of two parts. The first part the presentation discussed the idea of "building lifecycle," which includes planning, construction, and operation concepts; designing is part of the planning stage. Next, an example of an architectural design process stage was presented, followed by introducing design optimization in the early stage of the design process (see Figure 58). The presentation highlighted the importance of design optimization during the early stage of the design process, which is the concern of this work session. Decisions at this stage of the design process profoundly impact the characteristics of the design solutions.

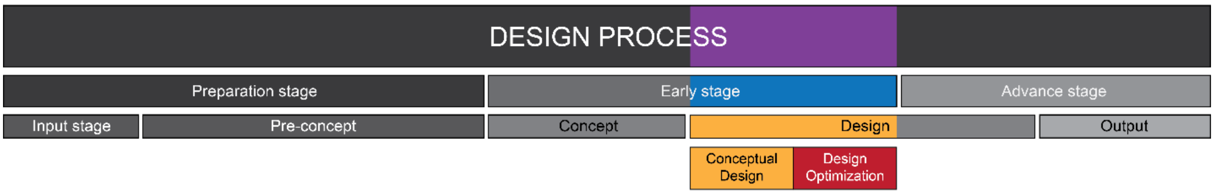


Figure 58: *Design optimization in the early stage of the design process*

Then, a video of conceptual examples of optimization case studies (Upgreengrade Team, 2016) was presented to present the idea of design optimization. The examples included a set of simple architectural design optimization problems. Each problem was based on one of the following objectives:

- minimize the distance of a building to a group of buildings,
- optimize the location of a building to maximize the shadow received from the surroundings,
- optimize the location of a building based on radiation analysis,
- optimize the form of a building based on radiation analysis.

All these examples have been solved with Grasshopper® and Galapagos© and Ladybug© tools. Afterward, London city hall (Foster and Partners, 2002) was presented as a case study of an optimized building to present the importance of multi-response optimization in architectural design (see Figure 59). According to the designer, there were two main objectives involved in the optimization of the form *"minimizing the surface area exposed to direct sunlight"* and *"provide shading for the naturally ventilated offices"*(Foster and Partners, 2002). Then Pareto's function (see 2.3.1) was explained to the participants by using different methods. First, a graphical explanation that demonstrates Pareto's function in a simple situation with two design objectives was presented. Second, a numerical explanation of Pareto's function by comparing four different solutions regarding three different objectives was presented.

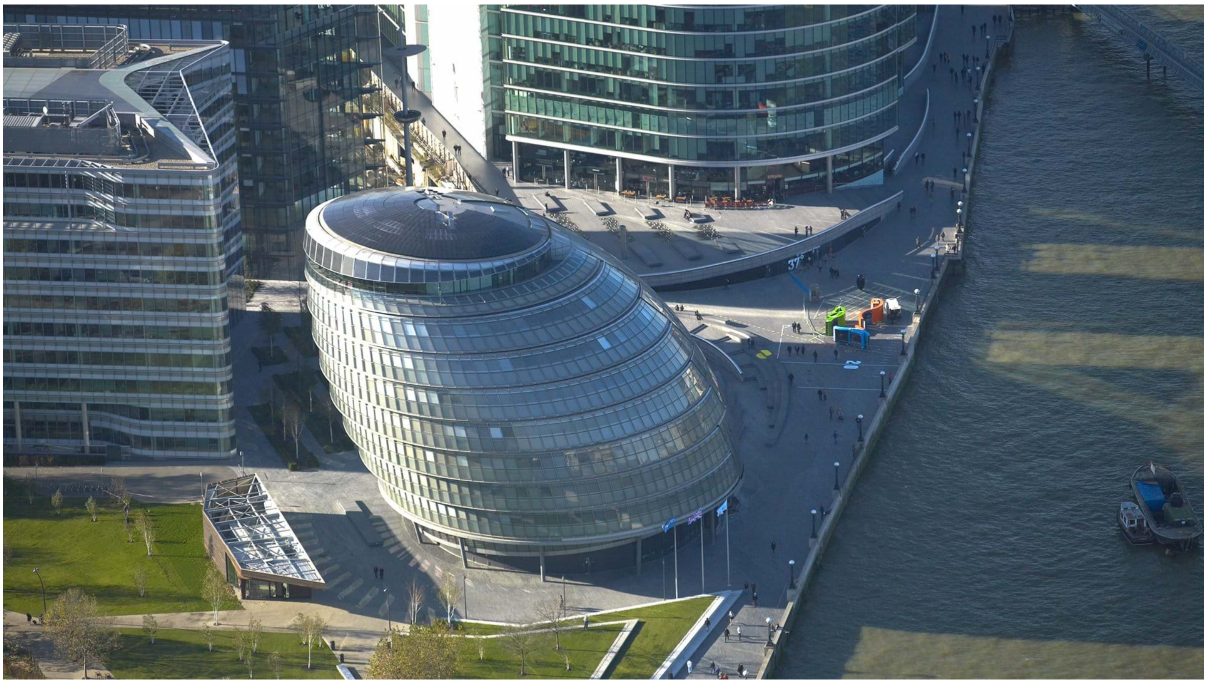


Figure 59: *London city hall* (Foster and Partners, 2002)

The second part of the presentation focuses on presenting Workflow 2 (see 4.2) and Workflow 6 (see 4.6) to the participants. First some generative multi-objective design optimization tools are mentioned (EcoGen©, Optimo©, Octopus©, Biomorpher©) and classified to VP based and non-VP based tools. For Workflow 2, the presentation introduces the concept of VP, which is then followed by a tour of the Grasshopper® user interface to show how it works and its capabilities. Next, a list of Grasshopper® plug-ins is presented to show the extended capabilities of the software. This includes Octopus© (see 4.2), Biomorpher© (see 4.3), Design Explorer© (see 4.2), Ladybug©, and Honeybee©; the last two are tools for environmental observation. For Workflow 6, the presentation defines EcoGen©, how it works, its capabilities through a tour of its user interface.

5.1.1.2 Guided manipulation

After the presentation, a guided manipulation of both EcoGen© and Grasshopper® was performed to familiarize the participants with the tools. First, the participants followed a guide for using EcoGen©. The guided manipulation taught the participants how to start EcoGen©, how to manipulate its parameters, and how to evolve solutions. They were also guided to use the interactive selection of the phenotypes to orient the evolution. Finally, the participants were guided to explore and navigate the solutions.

Next, the participants followed a guided manipulation of Grasshopper®. The manipulation began by explaining how to start Grasshopper®. A series of steps were used to demonstrate the complexity that can be reached by using Grasshopper®. The participants were guided to construct and manipulate a single point, then, to construct another point and use both points to create a parametric line. Next, The participants were guided to build a grid of points and an “A” point isolated from this grid. Each point that belongs to the grid was used as a center for constructing a circle. Based on the distance between the point “A” and the center of each circle, the radius dimensions of the circle were assigned. As a result, the closer the circle is to point “A”, the smaller it is. After, the participants were guided to replace the circles with spheres (see Figure 60). Subsequently, they were guided to optimize the position of point “A” in order to maximize the total volume of the spheres and to minimize the total distances between the centers and the point “A” by using Octopus©; the interface of Octopus© was explained during this process.

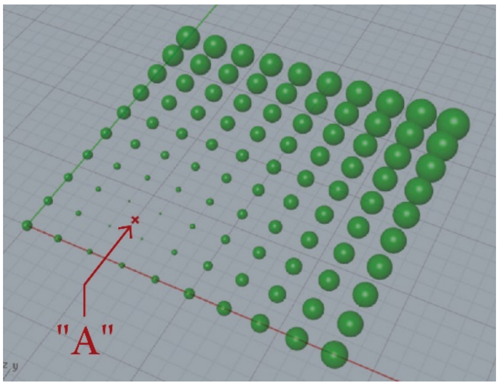


Figure 60: *A parametric grid of spheres, sized based on distance*

5.1.1.3 Autonomous manipulation

During the autonomous manipulation, the participants were asked to generate an optimized modular building by using both workflows (2 and 6). First, they were asked to use EcoGen© v 2.0 (the latest version at this moment) to generate an optimized modular building form (see Figure 61). The participants had the possibility to manipulate the parameters of EcoGen© and to select the preferred solutions. The participants were encouraged to use all the available objectives at the experiment time, compacity, solar gain, and heating; these were the objectives available in EcoGen©.

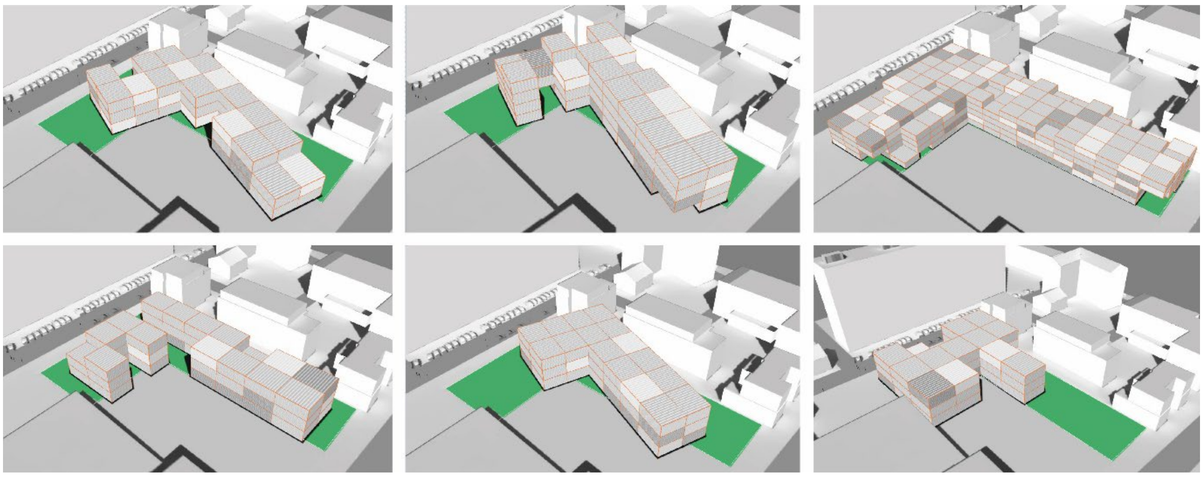


Figure 61: *Examples of solutions created by EcoGen©*

Second, the participants were asked to use Grasshopper® and Octopus© to generate an optimized modular building form. For this part of the experiment, a Grasshopper® algorithm was predeveloped for the participants. The algorithm generates modular three-dimensional grids consisting of voxels (see Figure 62 and Figure 63) by using a set of variables (see Table 3).

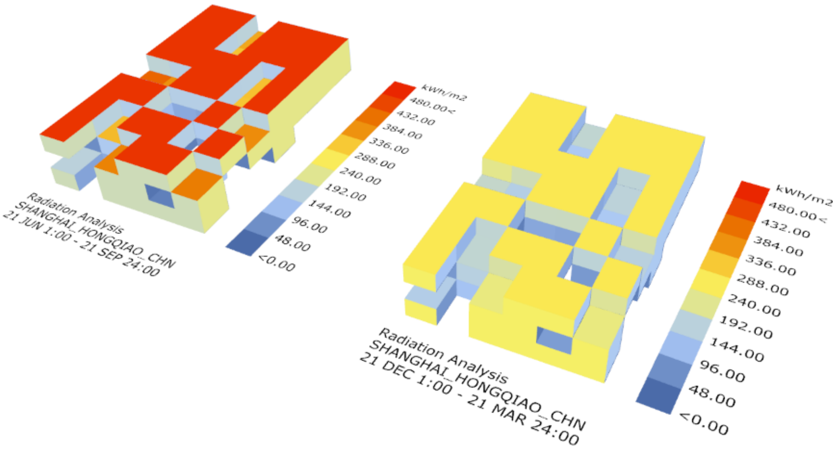


Figure 62: *Example of a solution created by the predeveloped algorithm*

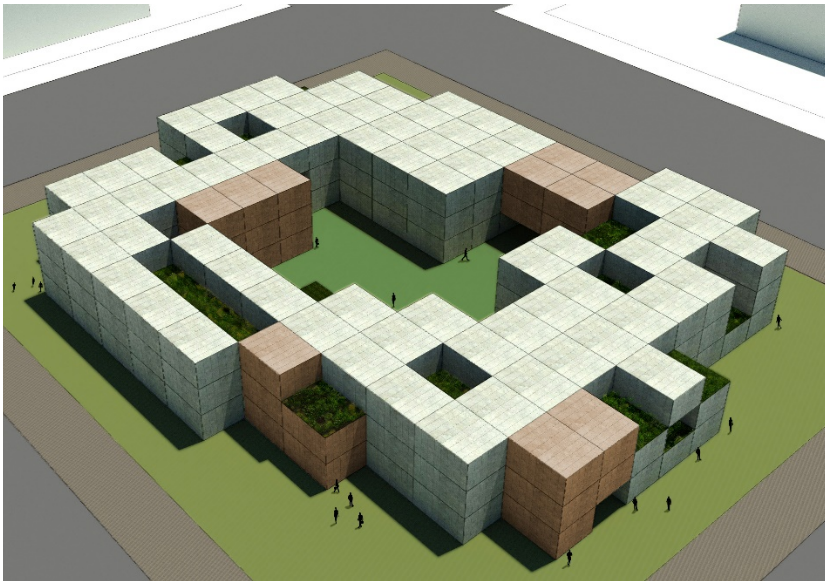


Figure 63: *Massing model of a design solution generated by the predeveloped algorithm*

	Variable name	Variable function
1	Shuffle function	changes the positions of the voxels (voxel's presence or absence in each cell of the three-dimensional grid)
2	Building area width	defines the land lot width
3	Building area length	defines the land lot length
4	Building area origin x	defines the x value of the land lot origin point
5	Building area origin y	defines the y value of the land lot origin point
6	Building area orientation	controls the orientation of the land lot
7	Unit width	controls the width of the modules (the basic voxel)
8	Unit length	controls the length of the modules (the basic voxel)
9	Unit height	controls the height of the modules (the basic voxel)
10	Number of levels	specifies the maximum number of floors allowed
11	Program area	defines the total area of the required interior program
12	Form orientation	controls the orientation of the forms
13	Courtyard width	defines the width of a rectangular courtyard
14	Courtyard length	defines the length of a rectangular courtyard
15	Courtyard origin x	defines the x value of a rectangular courtyard origin point
16	Courtyard area origin y	defines the y value of a rectangular courtyard origin point

***Table 3**: Experiment 1, the list of variables used in the Grasshopper® predeveloped algorithm*

The algorithm is prepared to observe three different fitness: (1) the compacity of the form (maximize), which corresponds to the ratio between the form's surface of the envelope and the total surface of the floors, (2) the solar gain during the summer period (minimize), and (3) the solar gain during the winter period (maximize). The algorithm uses Octopus© to optimize the solutions. For observing solar gain, the algorithm uses Ladybug©; *"Ladybug performs detailed analysis of climate data to produce customized, interactive visualizations for environmentally-informed design."*(Ladybug Tools Team, 2020). For imitating EcoGen©, the algorithm prepared to generate a variety of modular forms by autonomously manipulating the "form orientation," and the "shuffle function" (see Table 3).

Many tests were performed before the experiment to confirm the reliability and flexibility of the algorithm. A real case study was selected to test and demonstrate the abilities of the algorithm. A small U-shape house designed by the Baytikool team (BaityKool team, 2018) (see Figure 64) for Solar Decathlon Middle East 2018 competition in Dubai ("Solar Decathlon Middle East," n.d.) was used for that purpose. The test used the house as a unit (voxel). The goal was to find the positions and the orientation of a group of units to create an optimized modular collective housing based on the three objectives defined in the algorithm. The Baitykool team was asked to select between two design approaches, low-rise with a two-stories limit or mid-rise with twelve-stories limit. The design team decided to focus on the low-rise approach.

Figure 64: *Baytikool, Solar Decathlon Middle East 2018 competition in Dubai* (BaityKool team, 2018)

The chosen approach was then used to test the algorithm. A suitable neighborhood was selected, modeled, and imported into the algorithm (see Figure 65). The way to define modules free areas was explained to the participants. These areas can be used for creating spaces for transportation, public spaces, services, etc. They can also be used for avoiding existing structures or landscapes on the site.

Figure 65: *Optimized collective housing based on Baitykool unit*

The tests showed that the algorithm is flexible and reliable, and it can be used for the experiments. The algorithm appears to improve the control of the variables of the different solutions. However, the main disadvantage is relative to the computation convergence times, which are high in comparison with EcoGen©. By autonomously testing both workflows based on EcoGen© and the predeveloped algorithm for Grasshopper®. The participants were able to give their opinions by responding to a questionnaire.

5.1.1.4 Questionnaire

A digital questionnaire was given to each participant to collect their feedback (see Appendix I). The questionnaire was divided into four different sections. The first section compares both workflows. The second section collects the participants' opinions of the used tools in general. The third section collects basic information about the participants' background and skills. The final section collects general personal information about the participants.

5.1.2 Participants

The total number of participants in the experiment was 11 persons (see Appendix I). 36.4 % are specialized in architecture, while 63.6 % are engineers specialized in buildings design. Students represent 63.6% of the participants, professionals represent 27.3 %, and only 9.1 % are university professors. In terms of age, 54.5 % are between 20-29 years old, and 27.3% are between 30-39 years old, while 9.1% are between 40-49, and 9.1 % are 50 years old or above. Having the majority of participants younger designers is positive for this experiment since the younger designers are more familiar with the digital tools in general. However, the majority of participants indicated that they had no or low experience in programming or visual programming (see Table 4), which is excellent for more neutral judgment. Most of the participants were not familiar with using multi-objective design optimization methods (Figure 66). This is also positive for the experiment because it aims to encourage those designers who are not familiar with optimization to optimize.

Level of skill	Programming skills	Visual programming skills
0//1	18.2 %	27.3 %
1/10	9.1 %	36.4 %
2/10	18.2 %	9.1 %
3/10	9.1 %	0.00 %
4/10	9.1 %	9.1 %
5/10	18.2 %	18.2 %
6/10	0.00 %	0.00 %
7/10	9.1 %	0.00 %
8/10	0.00 %	0.00 %
9/10	9.10 %	0.00 %
10/10	0.00 %	0.00 %

Table 4: *Experiment 1, participants experience in programming*

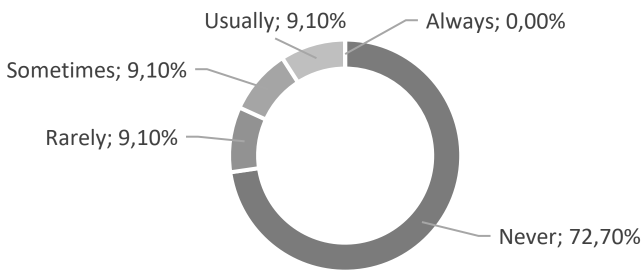


Figure 66: *Experiment 1, participants experience in multi-objective design optimization*

5.1.3 Results

To compare both workflows, the participants answered a list of questions based on their experience during the work session (see Appendix I). Each question allows the user to select between two answers; each answer represents one workflow. Table 5 presents all the questions used for the comparison, and it also demonstrates the statistics of the participants' answers. The questions were targeting five different points. The first two questions concern the design process. Questions 3 and 4 focus on the tools' ability to involve designers' creativity. Questions 5 and 6 represent performance optimization. Questions 7 and 8 focus on the tool interface. Finally, the last question compares the participants' general preferences. By observing these results, it was clear that the participants prefer Workflow 2 (Grasshopper® & Octopus©) over Workflow 6 (EcoGen©). However, for the interface, they appreciated a little more Workflow 6 over Workflow 2.

	Question	Workflow 2	Workflow 6
1	Which workflow do you think can help you to find design solutions?	72.7 %	27.3 %
2	Which workflow is more suitable for your design process?	72.7 %	27.3 %
3	In which workflow do you feel more involved in the design?	81.8 %	18.2 %
4	Which workflow stimulates your creativity more?	81.8 %	18.2 %
5	Which workflow do you prefer to filter the results?	63.6 %	36.4 %
6	Which workflow do you believe helps produce better results?	90.9 %	9.1 %
7	Which workflow is easier to understand?	27.3 %	72.7 %
8	Which user interface do you prefer?	45.5 %	54.5 %
9	Which workflow do you prefer?	100 %	0.00 %

Table 5: *Experiment 1, comparison between Grasshopper® and EcoGen©*

The second group of questions was focused on more general comments regarding the presented tools (see Appendix I). Table 6 presents all the questions used for these general questions, and it also demonstrates the statistics of the participants' answers. These questions are separated into four groups. Questions 1 and 2 focus on the designers' opinions of the generative multi-objective design optimization tools in general. It is clear that the majority of the participants are willing to use these tools for their academic work. However, slightly more than half of them are not sure about using it in their professional work; the rest of them are willing to use these tools. Questions 3 to 6 focuses on the designers' opinions of VP. The answers showed highly favorable opinions toward VP. Questions 7 through 10 focus on the designers' opinions of EcoGen©. For these questions, the answers varied. About half of the designers find this tool suitable and want to know more about it. However, overall, not many participants are considering using it in the future. Finally, the majority of the participants prefer EcoGen© as a plug-in for Grasshopper®.

	Question	Yes	No	Maybe
1	If you are a student, do you consider using these tools in your schoolwork?	57.1 %	28.6 %	14.3 %
2	Do you consider using these tools in your professional work?	45.5 %	0.00 %	54.5 %
3	Do you think that visual programming is suitable for architects?	100 %	0.00 %	0.00 %
4	If you are a student, do you consider using visual programming in your future schoolwork?	57.1 %	28.6 %	14.3 %
5	Do you consider using visual programming in your future professional work?	45.5 %	0.00 %	54.5 %
6	Do you wish to learn more about visual programming?	100 %	0.00 %	0.00 %
7	Do you think that EcoGen© is suitable for architects?	54.5 %	18.2 %	27.3 %
8	If you are a student, do you consider using EcoGen© in your future schoolwork?	14.3 %	57.1 %	28.6 %
9	Do you consider using EcoGen© in your future professional work?	27.3 %	36.4 %	36.4 %
10	Do you wish to learn more about EcoGen©?	54.5 %	27.3 %	18.2 %
11	Do you prefer if EcoGen© became a plugin for Grasshopper®?	63.6 %	9.1 %	27.3 %

Table 6: *Experiment 1, general feedback about the tools*

In general, the results show a certain homogeneity of the answers regardless of the characteristics of the participants and their skills. The feedback demonstrates that Workflow 2, which, based on VP, is more acceptable for the participants in comparison to Workflow 6, which based on EcoGen©. However, this acceptability is ambiguous because the designers preferred the interface of EcoGen©. It will be necessary to complement this experiment with another one in order to understand the preferences of the designers better. The additional experiment must compare the workflow 2, which is preferred by the designer to another workflow based on VP based on an interface similar to EcoGen©. This will improve our investigations.

5.2 Experiment 2: Design optimization in visual programming

Both Biomorpher© and Octopus© are two different multi-fitness evolutionary solvers for Grasshopper®. On one side, Octopus© interface is based on graphically representing the Pareto front (see 4.2). On the other side, similarly to EcoGen©, Biomorpher© interface is based on presenting the 3D massing of the candidate solutions (see Figure 67). Both Biomorpher© and EcoGen© use IEA; they allow to select the solution based on their personal preferences.

Figure 67: *The interfaces of EcoGen© (left), and Biomorpher© (right)*

Comparing Workflow 2 (see 4.2), which is based on Octopus© and Workflow 3 (see 4.3), which is based on Biomorpher©, can help us to focus the investigation and understand the results of Experiment 1. The fact that Octopus© and Biomorpher© are both based on VP helps to neutralize many of the dispersive factors. However, in contrast to Octopus© and EcoGen©, Biomorpher© does not use Pareto's function. This experiment intends to focus the participants' attention on the interface and the selection method. Therefore, in this experiment, Octopus© and Biomorpher© are introduced as two evolutionary tools for Grasshopper ® based on Pareto's function with different interfaces and mechanisms (interactive, non-interactive).

5.2.1 Methodology

Samples of architects and engineers from the domain of building design were invited to participate in a work session that explores both workflows. Each work session was divided into four consecutive parts. These parts correspond to a presentation, time of guided manipulation, time of autonomous manipulation, and an individual questionnaire. The work session was performed four times at different dates with different groups.

5.2.1.1 Presentation

The presentation is divided into three parts. The first part is identical to the first part of the presentation presented in Experiment 1 (see 5.1.1.1). The second part of the presentation focuses on the VP. A tour of the user interface of Grasshopper® was performed to explain the concept of VP. Next, a list of Grasshopper® plug-ins was presented to show the extended capabilities of the software; this includes Octopus© (see 4.2), Biomorpher© (see 4.3), Design-Explorer© (see 4.2), Ladybug©, and Honeybee©.

The third part of the presentation focuses on explaining the user interface of both Octopus© and Biomorpher© in detail. It is important to emphasize that the experiment aims to compare the acceptability based on the experience of the users of these tools and not the actual performance of the solutions they evolve. Both plug-ins are introduced as two multi-objective optimization tools that use a similar aggregation method and different interfaces or algorithms (GA Octopus© and IEA for Biomorpher©); in reality, these tools use different aggregation methods (see 4.2, 4.3). This strategy increases the focus on comparing the interfaces by neutralizing the aggregation method.

5.2.1.2 Guided manipulation

After the presentation, the participants followed a guided manipulation of Grasshopper®. The same procedure used to guide the participants to manipulate Grasshopper® in Experiment 1 is used (see 5.1.1.2). However, in this experiment, the participants were guided to optimize the position of point “A” to maximize the total volume of the spheres and to minimize the total distances between the centers of the spheres and the point “A” by using both Octopus© and Biomorpher© (see Figure 60 (5.1.1.2)).

5.2.1.3 Autonomous manipulation

For the autonomous manipulation time, the predeveloped algorithm presented and tested in Experiment 1 (see 5.1.1.3) is used to test both Octopus© and Biomorpher©; this will keep the link between both experiments. The algorithm was explained to the participants, and they were asked to manipulate the variables for a better understanding of the algorithm. Finally, the participants were asked to generate optimized modular forms by using the algorithm with Octopus© and then with Biomorpher©. Three objectives were defined for the experiment, to minimize the form compacity, solar gain in summer, and to maximize the solar gain in winter.

5.2.1.4 Questionnaire

A questionnaire was given to the participants to collect their feedback. The questionnaire is divided into four different sections. The first section compares both workflows. The second section collects the opinions of the participants about the VP in general. The third section collects basic information about the participants’ background and skills in multi-objective design optimization and programming. The final section collects basic personal information about the participants.

5.2.2 Participants

The total number of participants is 13 persons, 46.2% are engineers specialized in buildings design, 46.2% are architects, and 7.7% are architects/engineers. Students represent 69.2%, professors are 7.7%, professionals are 15.4%, and finally, 7.7% is professor/professional. In terms of age, 20-29 years old represent 84.6% of the participants, and only 15.4% are 30 years old or older. Most of the participants indicated that they had no or low experience in programming or VP (see Table 7). The majority of the participants never used multi-objective design optimization (see Figure 68).

Level of skill	Programming skills	Visual programming skills
0/10	38.5 %	61.54%
1/10	0.00 %	0.00%
2/10	7.7 %	15.4 %
3/10	7.7 %	15.4 %
4/10	23.1 %	0.00 %
5/10	7.7 %	7.7 %
6/10	0.00 %	0.00%
7/10	7.7 %	0.00%
8/10	0.00%	0.00%
9/10	7.7 %	0.00%
10/10	0.00 %	0.00%

Table 7: *Experiment 2, participants' experience in programming*

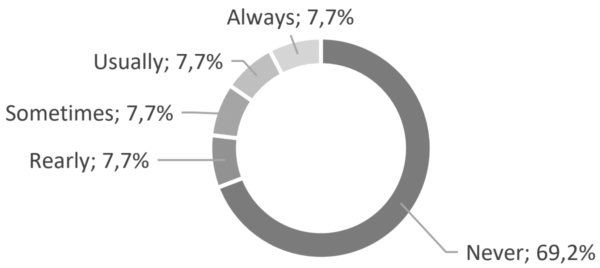


Figure 68: *Experiment 2, participants' experience in multi-objective design optimization*

5.2.3 Results

To compare both tools, the participants responded to a list of questions based on their experience during the work session. Each question allows the user to select between two answers, and each answer represents one of the tested tools. Table 8 presents all the questions used for the comparison, and it also demonstrates the statistics of the participants' answers. The questions are targeting five different points. The first two questions concern the design process. Questions 3 and 4 focus on the tool's ability to involve the designers' creativity. Questions 5 and 6 concern performance optimization. Questions 7 to 9 focus on the user interface. Finally, the last question compares the participants' general preferences of both tools.

	Question	Octopus©	Biomorpher©
1	Which tool do you think can help you to find design solutions in different situations?	46.2 %	53.8 %
2	Which tool is more suitable for your design process?	30.8 %	69.2 %
3	In which tool do you feel more involved in the design?	15.4 %	84.6 %
4	Which tool stimulates your creativity more?	15.4 %	84.6%
5	Which tool do you prefer to filter the results?	76.9 %	23.1 %
6	Which tool do you believe helps produce better results?	69.2 %	30.8 %
7	Which tool is easier to understand?	15.4 %	84.6 %
8	Which tool is more interesting to you?	53.8 %	46.2 %
9	Which interface do you prefer?	30.8 %	69.2 %
10	Which tool do you prefer?	38.5 %	61.5 %

Table 8: *Experiment 2, comparison between Biomorpher© and Octopus©*

Observing the results from Table 8 shows that each tool has its strength. Regarding the first two parts design process and involving designers' creativity, the participants preferred Biomorpher©. According to the participants, Biomorpher© is more visual and interactive. They consider that Biomorpher© involves them and stimulates their creativity more than Octopus©. However, those who prefer Octopus© find the scatterplot based on Pareto's function very useful as it allows them to compare the solutions easily. Interestingly two-thirds of those who preferred Octopus© have a background in architecture. In Experiment 1, most of the participants preferred Workflow 2, which uses Octopus© over Workflow 6, which uses EcoGen© and an interface similar to Biomorpher©. However, in Experiment 2, the participants prefer Biomorpher© over Octopus©. From that, we can infer that in Experiment 1, the participants chose Workflow 2 because it uses VP and not because of Octopus© interface. Once an IEA approach with a visual interface similar to EcoGen© interface was introduced to VP, the participants preferred it.

Regarding the performance optimization, the participants prefer Octopus© over Biomorpher©. No matter if we use VP or non-VP approach, the participants trust Octopus© over Biomorpher© and EcoGen©. The participants find Octopus© more informative as it graphically represents the performance of the solutions based on Pareto's function. This interface style decreases the confusion and allows for comparing the candidate solutions easily. Furthermore, the participants believe that because Octopus© involves less human subjective judgment, the chances of finding optimized solutions are higher.

Regarding the interface and based on the participants' comments, the majority of the participants preferred Biomorpher© because the interface is more visual and interactive. However, slightly more than half of the participants find Octopus© more interesting. By comparing the results of the two experiments concerning the interface, we can see that it doesn't matter if we use VP or not. The participants prefer the interface style of Biomorpher© and EcoGen© over the Octopus© interface style.

Finally, the participants prefer using Biomorpher© over Octopus©. Comparing this result to the results of Experiment 1, where all the participants preferred Workflow 2 over Workflow 6 can help us to infer two points. First, the designers prefer to define the structure of the problem in VP. Second, the designers prefer the interface style of Biomorpher© and EcoGen© approach the solutions; these two interfaces are similar.

The second group of questions focuses on more general feedback about VP (see Table 9). Questions 1 and 2 concentrates on the tools used during the work session in general. The rest of the questions concern VP in general. The results show that the majority of the designers are willing to use these tools for their professional work. In general, VP is highly acceptable among designers.

	Question	Yes	No	Maybe
1	If you are a student, do you consider using these tools in your schoolwork?	55.6 %	0.00 %	44.4 %
2	Do you consider using these tools in your professional work?	53.8 %	7.7 %	38.5 %
3	Do you think that visual programming is suitable for architects?	84.6 %	7.7 %	7.7 %
4	If you are a student, do you consider using visual programming in your future schoolwork?	66.7 %	7.7 %	33.3 %
5	Do you consider using visual programming in your future professional work?	61.5 %	7.7 %	30.8 %
6	Do you wish to learn more about visual programming?	84.6 %	7.7 %	7.7 %

Table 9: *Experiment 2, general feedback about the tools*

In general, the results of experiment 2 show a certain homogeneity of the answers regardless of the characteristics of the participants and their skills. The results demonstrate that the participants prefer to use the interface of Biomorpher© interface rather than Octopus©. However, the designers find Octopus© is more reliable for optimization. Therefore, developing tools based on VP that are a middle way between Biomorpher© and Octopus© has a high potential to be accepted by designers.

5.3 Interactive Visual Programming

From the results of experiments 1 and 2, we can conclude that the decision support workflows based on VP have high potential to be accepted among the designers. To generate optimized solutions, parametric modeling is of high interest. Programming is a high-end tool for Parametric modeling. However, textual programming is non-accessible for most of the designers. Using VP allows designers to describe sophisticated parametric models.

The interaction between the tool and the designer is essential to achieve acceptability and, thus, acceptimality. This interaction can increase the acceptability of the tools. From the results, we can infer tow points that can enhance this interaction. Firstly, using an interactive optimization algorithm such as IEA allows the designers to guide the optimization process based on their subjective judgment. Secondly, the graphical aspects of the tool are crucial as they insight the selection of designers. From the experiments, two graphical qualities are crucial for the tools. First, the graphical representation of the performances of the candidate solutions is crucial. This can increase the confidence of the designers. It provides them with vital information about the candidate solutions. Using a scatterplot such as the used in Octopus© is a good example of how to represent the performance of the solutions. It allows the designer to compare the solutions. Second, it is imperative to present the massing model of the candidate solutions. This allows the designers to explore the solutions, which is essential for forming a personal judgment of the solution.

Refinery® (see 4.4) meets most of the points discussed above. However, Refinery® does not use an interactive algorithm. Unfortunately, when these experiments were performed, Refinery® did not exist. Experiment 2 was published in IBPSA 2018 conference (Afandi, Barlet, Sebastian, Bruneau, & Marsault, 2018).

CHAPTER 6 Aggregation for *acceptimality*

"The problem with digital architecture is that an algorithm can produce endless variations, so an architect has many choices." (Eisenman, n.d.)

This chapter presents different experiments that test different aggregation functions, which can potentially replace Pareto's function (see 2.3.1) in the context of design optimization. These aggregation functions are Maximin (see 2.3.2) and Derringer & Suich's function (see 2.3.3). The experiments aim to link the functions to designers' acceptability. In contrast to Pareto's function, Maximin and Derringer & Suich's are cardinal and high in negentropy. These functions take into account designers' acceptability in the optimization process by allowing designers to express their preferences in the interpretation model of MOIA (see chapter 2). Furthermore, Derringer & Suich's function allows the designers to assign different weights to the design objectives in the aggregation model of MOIA. Thus, each function classifies the solutions differently. Figure 69 represents the three functions graphically.

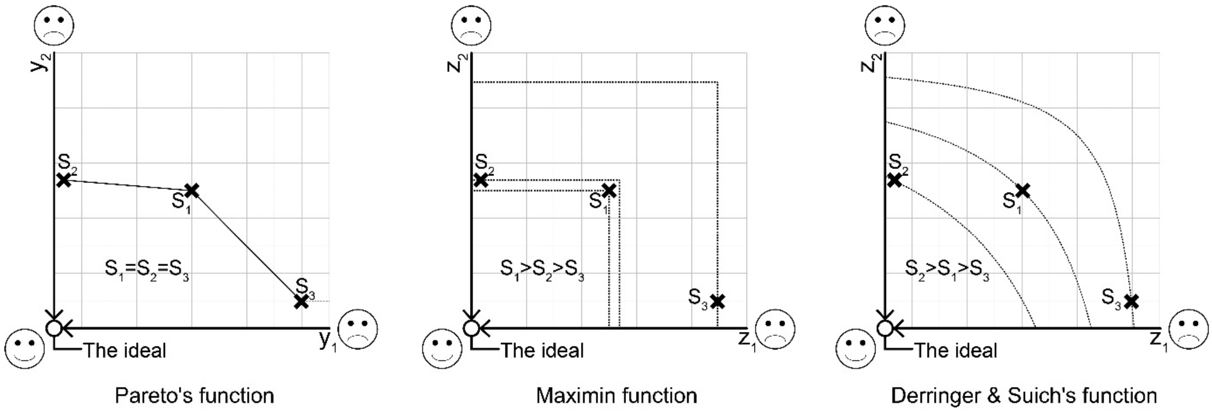


Figure 69: *Graphical representation of Pareto's, Maximin, Derringer & Suich's aggregation functions*

One of the challenging problems of design is language development for form definition. This problem raises the question of creating grammars of form that are consistent with the needs of designers (Garcia, 2017). Mainly for economic factors, architects often use forms made of simple primary volumes. These forms are usually simpler and cheaper to construct in comparison to rounded or streamlined ones. The simple primary forms are also consistent and relatively inexpensive when we apply the physical analysis of performances during the early stages of the design process. The human brain can envisage these forms quickly. The experiments of this chapter use form grammar that results in design solutions consisting of identical voxels clustered in a three-dimensional grid. The use of modular forms to study the relationship between numerical optimality and human acceptability is more effective than the use of complex forms because the complexity of the forms could complicate our study of acceptability. It also links the experiments of this chapter to the ones of the previous one.

6.1 Experiment 3

Experiment 3 investigates both Maximin and Derringer & Suich's aggregation functions on a panel of experts. The aim is to compare these functions to designers' judgments.

6.1.1 Procedure

A group of participants was invited to a work session. The work session started with a presentation that is identical to the first part of the presentation presented in Experiment 1 (see 5.1.1.1). The aim is to prepare the participants with the basic knowledge required to complete the experiment. It is essential to avoid presenting Maximin or Derringer & Suich's aggregation functions to keep the participants spontaneous in their judgment. Finally, two tests that seek to investigate the functions based on designers' judgments are performed.

6.1.1.1 Testing of Derringer & Suich's aggregation function

The test is performed during the work session, and it aims to measure two different aspects related to Derringer & Suich's aggregation function (see Appendix III). It first sets out to observe and measure whether the architects need to make use of objective weighting for conveying design objectives. The second aspect concerns the measurement of the consistency of the objective weighting values of the designers.

The participants were asked to place themselves in the situation of working in the early stage of the design process of four office buildings. Each building was positioned in a different climate zone; each zone corresponds to one particular scenario. The first zone is an extremely hot climate (Dubai, United Arab Emirates), the second is a moderate climate (San Diego, United States of America), the third is an extremely cold climate, (Yakutsk, Russia) and the fourth is a contrasted climate which is hot in summer and cold in winter (Shanghai, China). For each location, the participants received the information of the monthly average high temperature, the monthly average low temperature, and the monthly sunshine hours. Five different objectives to optimize were considered (see Appendix III):

- Objective 1: Maximize the compacity of the building form.
- Objective 2: Maximize the direct sunlight.
- Objective 3: Maximize natural lighting, which includes indirect sunlight.
- Objective 4: Minimize heating energy consumption.
- Objective 5: Minimize cooling energy consumption.

To test both aspects, the participants were asked to set weights for the different objectives presented above without further details on the concept of importance or precise meaning of the term optimization. The participants' appreciation must remain spontaneous.

For the first aspect of the test, The participants must evaluate the weight of the five objectives for each scenario. Based on Derringer & Suich's aggregation, each scenario needs five relative weights ranging between 0% and 100% for each objective. The sum of the weights of all the objectives together for each scenario should equal 100%. In this part of the test, the weight evaluation is non-redundant. The non-redundant evaluation of the weights means that for each scenario, the participants assign a weight for each objective by directly deciding its value (see Appendix III). We can link the high variation of weights to the high need for using the concept of assigning weights for design objectives, which is the core principle of Derringer & Suichs' aggregation function.

For the second aspect of the test, the focus was on only one scenario, which was the case of the extremely hot climate (Dubai, United Arab Emirates). The evaluation of consistency can be performed by using the Analytic Hierarchy Process (AHP) introduced by Saaty (Saaty, 2013; Saaty & Vargas, 2000). In this method, the decision-maker has to evaluate the different objectives in a pairwise comparison. In contrast to the previous evaluation, the pairwise comparison is redundant because we evaluate each objective more than one time by pairing it with the other objectives. The number of the pairwise combination is "$(n^2-n)/2$" where n is the number of objectives; in this case $((5^2-5)/2)=10$ combinations. For each comparison, the weighting is based on a scale that consists of 9 possible choices (see Figure 70). The middle of the scale means that both objectives are equally important. Each participant should fill all the possible matches of pairwise objectives.

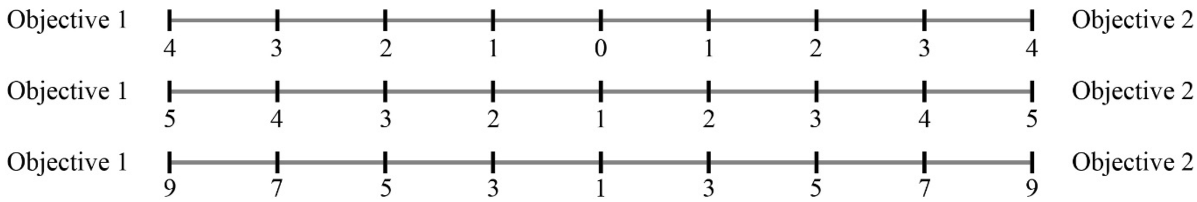


Figure 70: *Examples of different pairwise comparison scales*

The scale-based evaluation (see Appendix III) then has to be filled into a judgment matrix (see Table 10). The dark grey cells in the matrix are always neutral and equal to the middle of the scale because it compares each objective to itself. The values in the light grey cells are always the opposite values of the values in white cells.

	Objective 1	Objective 2	Objective 3
Objective 1	1	1/9	1/7
Objective 2	9	1	5
Objective 3	7	1/5	1

Table 10: *An example of the Judgment matrix*

The matrix is then used to calculate the consistency ratio. If "n" is the number of objectives and "λ_{max}" is the maximal eigenvalue, the consistency ratio is computed from (Eq. 12) (Saaty, 2013; Saaty & Vargas, 2000). According to Saaty, for responses to be considered highly consistent, the consistency ratio must be less than 10% (Saaty & Vargas, 2000).

$$\text{Consistency Ratio} = \frac{\lambda_{\max} - \mathrm{n}}{\mathrm{RI} \times (\mathrm{n} - 1)} \qquad \textit{(Eq. 12)}$$

With,

RI = random index (see Table 11)

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
RI	0	0	0.52	0.89	1.11	1.25	1.35	1.40	1.45	1.49	1.51	1.54	1.56	1.57	1.58

Table 11: *The random index values according to AHP (Saaty, 2013; Saaty & Vargas, 2000)*

6.1.1.2 Test of Maximin aggregation function

The test aims to investigate two different ideas. Firstly, we want to observe how far the scatterplot, which represents the Pareto front graphically, affects the architects' decision. Secondly, we want to compare participants' judgment to both Pareto's function and Maximin functions.

The participants were asked to imagine themself in the early stage of designing a mixed-use building (offices, residential, commercial) in a mixed climate, hot summer, cold winter (Shanghai, China). The participants were given the monthly average high temperature, monthly average low temperature, and monthly sunshine hours. The building is modular, and the size of one module is (Width = 6m, Length = 8m, Hight = 4m). The total area of the program is 12000 m^2. The lot area is 8000 m^2. The building has three stories at maximum. A courtyard 25 m × 30 m in the middle of the building was required as a part of the design. The objective was to maximize solar gain in winter and minimize solar gain in summer (see Appendix III).

For the first aspect of the test, a list of 10 different generated forms was shown to the participants (see Figure 71); each form represents one solution (see Appendix III). The forms were generated by using the predeveloped Grasshopper® algorithm for Experiment 1 (see 5.1.1.3). All of the generated solutions belong to the Pareto front. However, the participants have no access to the Pareto front or to the scatterplot that represents the performance of the solution graphically. Each form represents the solar gain in (KWh/m2) by using a color scheme in both summer and winter. The participants must select only one form that they prefer. After, the scatterplot that presents the ten solutions, including Pareto's front (see Figure 72), was given to the participants. Again, the participants must select only one form. The aim is to observe if the scatterplot, which represents the solutions and compares them based on their performance visually, will affect the designers' decisions. For the second aspect, the final selection of the participants was compared to Maximin classification of the solutions.

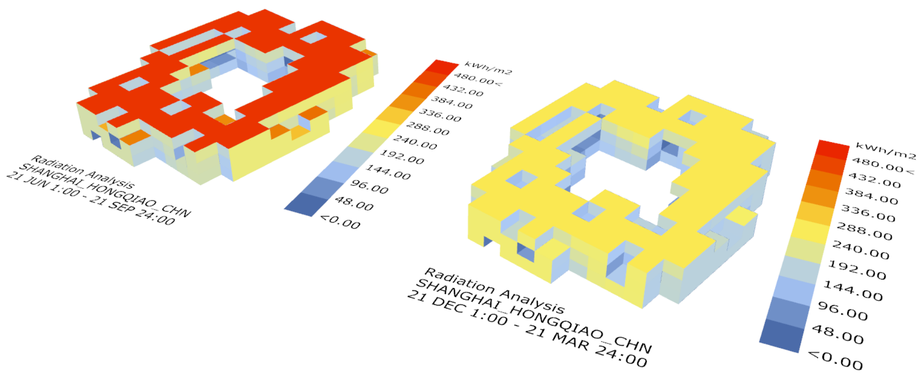


Figure 71: *Experiment 3, an example of the used Solutions*

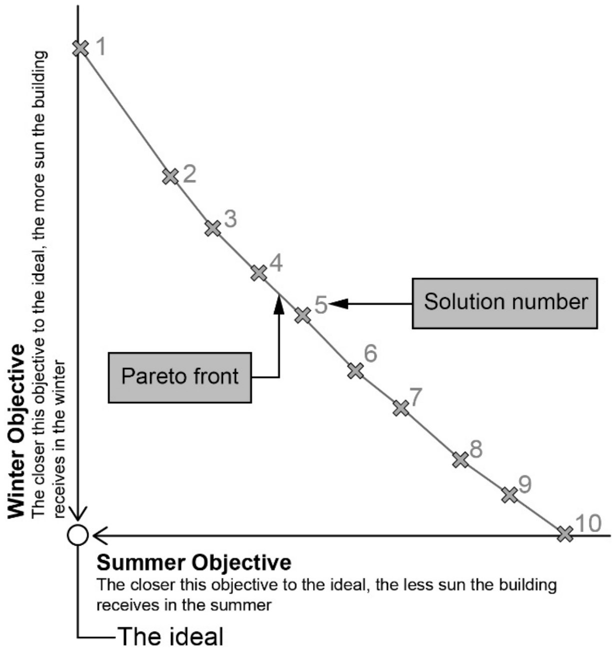


Figure 72: *Experiment 3, scatterplot, including Pareto front for the solutions 1 to 10.*

6.1.2 Participants

The total number of participants of Experiment 3 was 32 persons; about 87.5% of the participants were between 20-29 years old (see Table 12). Table 13 demonstrates the specialty and professional status of the participants. Finally, Table 14 shows the frequency of using multi-objective optimization by the participants in their work.

Age group	Count	Percentage
20-24	18	56%
25-29	8	25%
30-34	2	6%
35-39	1	3%
40-44	2	6%
45-49	1	3%

Table 12: *Experiment 3, Participants' age groups*

Categories	Count	Percentage
Student/Architecture	24	75 %
Student/Engineering	2	6 %
Student/(Architecture/Engineering)	2	6 %
Professional/Architecture	2	6 %
(Professor/Professional)/(Architecture/Engineering)	1	3 %
Professor/Architecture	1	3 %

Table 13: *Experiment 3, Participants' specialty and professional status*

Frequency	Count	Percentage
Never	16	52 %
Rarely	10	32 %
Sometimes	5	16 %
Usually	0	0 %
Always	0	0 %

Table 14: *Experiment 3, Frequency of using multi-objective design optimization by the participants.*

6.1.3 Results

6.1.3.1 Results of Derringer & Suich's aggregation function test

Comparing the five different objectives weights for each of the four scenarios shows a variation in the objectives' weights for each scenario. Tracking each objective in each scenario also shows a wide range of weights, which means that the different scenarios and the different objectives alter the weights (see Figure 73). Thus, we can infer that building designers tend to use different weights for different objectives. The weights in this analysis were computed from the average of participants' non-redundant evaluation (see 6.1.1.1).

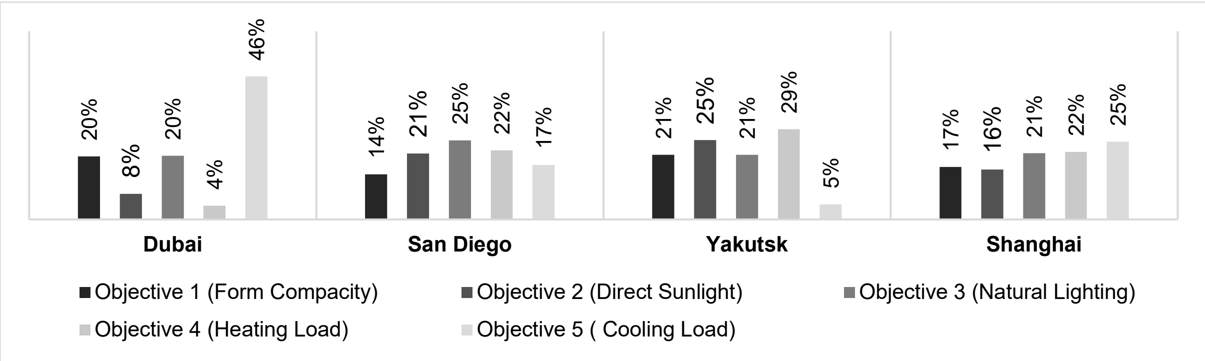


Figure 73: *Experiment 3, average weights resulting from the non-redundant evaluation*

In this experiment, it is clear that the location significantly affected the judgment of the participants. For example, Dubai climate is extremely hot in summer (41°C on average in July and August), and the temperature is relatively high in winter (above 23°C). The sunshine hours are long (250 to 340 hours per month). As a result, on average, the participants evaluated the heating load and the direct sunlight to low importance (4% and 8%), and the cooling load to high importance (46%). In contrast, these objectives are highly important for the scenario of Yakutsk, Russia, as it is an extremely cold climate.

However, it is likely that participants do not differentiate between the satisfaction of objectives and criticality. Satisfaction refers to the level, as a physical quantity, it leads to satisfaction (is this level relatively high or low?). Criticality concerns the difficulties or the negative consequences to achieve satisfaction (is the satisfaction achievement problematic?). For the scenario of Dubai, the cooling load objective, the level of satisfaction, and the criticality are both high. Because of the high average temperatures, the energy consumption for cooling is high; the electricity is becoming more and more expensive in the city. Other solutions that reduce energy demand can be used. These solutions can lower the cooling load, such as improving the insulation of the buildings.

Based on the average responses of the participants, Figure 74 compares the weight values derived from both the non-redundant evaluation (see Figure 73) and the redundant pairwise comparison for the case of Dubai. Unexpectedly, despite the fact that the participants have different points of view, both evaluations resulted in very close values for the objective weights.

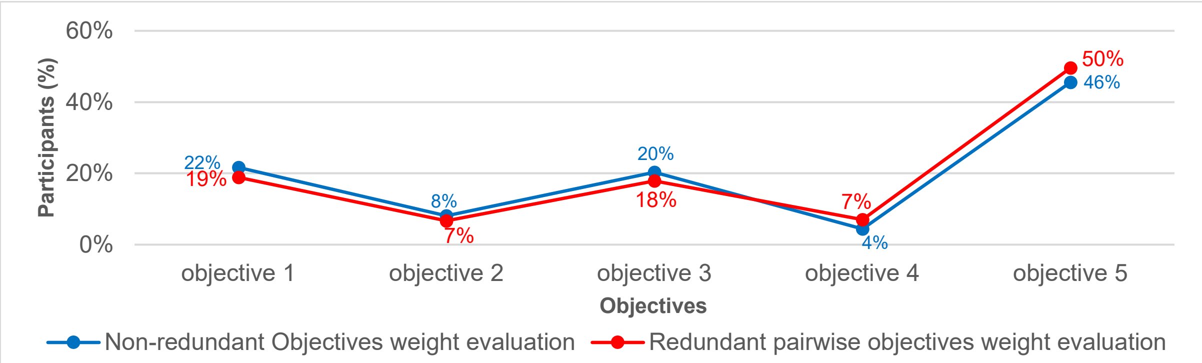


Figure 74: *Experiment 3, average weights values derived from the non-redundant, and the redundant (pairwise) comparison for the five objectives involved in the case of Dubai.*

However, the previous results of the non-redundant evaluation are not enough to assess consistency. The redundant pairwise comparison (see 6.1.1.1) of each participant can be consistent or inconsistent. Figure 75 demonstrates the consistency ratios calculated from the pairwise comparison by using the AHP (Saaty, 2013; Saaty & Vargas, 2000) consistency analysis for each participant. Based on the figure, 22% of the participants attained a consistency ratio lower than 10%, which is regarded by Saaty (Saaty, 2013; Saaty & Vargas, 2000) as acceptable; in the figure, the green columns represent those participants. From a more accurate observation of the participants' responses, we regard deviation values lower than 30% as low values. 19% of the participants attained a consistency ratio higher than 30%; in the figure, the red columns correspond to those participants. These are regarded as random evaluations. 59% of the participants are in between very low and acceptable consistency. The average responses of the pairwise redundant evaluation are similar to the direct non-redundant evaluation, which is regarded as entirely consistent. However, the consistency ratio of the separate responses is low, according to AHP (Saaty, 2013; Saaty & Vargas, 2000).

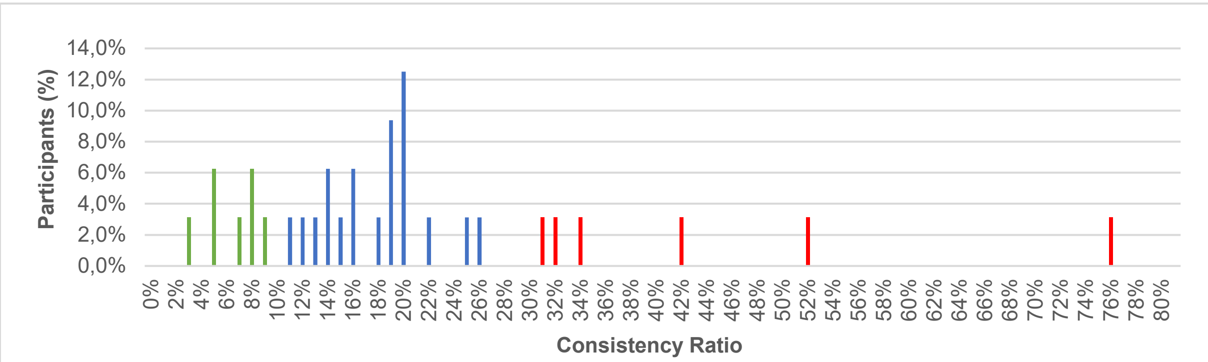


Figure 75: *Experiment 3, consistency ratios of participants' evaluation of the objectives based on the pairwise comparison*

These results highlight the consequences of using weighting aggregation functions for generative design applications; the consistency of the designers is low. One solution to resolve this issue is to infer the set of every possible consistent series of weights from participant's responses; thus, the outcome of the weighting process is a set of series of weights rather than a single series of weights. The set of design solutions computed from the set of series would then be a set of clusters of solutions. However, this mathematical approach is expected to increase the obstacle of understanding the results of the design. It will also affect the designers' judgment. Investigating this approach is part of our perspectives research.

In conclusion, weight-based aggregation is required among the building's designers. Derringer & Suich's aggregation function can increase designers' satisfaction by increasing their role in decision-making in the optimization process. In a situation where fast design decision is required, the decision-maker has a high chance to be inconsistent with evaluating objectives weights. It is highly suggested to adopt methods that help the designer to be more consistent in evaluating the weight of the design objectives. The pairwise comparison based on the AHP method can be used for that purpose; the consistency ratio can be a warning indicator.

6.1.3.2 Results of Maximin aggregation function test

After each participant selected one solution before and after seeing Pareto front, 75% of the participants changed their decision (Figure 76). 67% of the people who did change after the Pareto front was presented moved to Solution (5), which is ranked first by Maximin. In the final selection, 16 participants who represent 50% of the participants selected Solution (5). Most of them explained that the reason for changing to Solution (5) was that this solution could compromise between summer and winter. The second most selected solution was Solution (4) with 25%, and the third selected solution is Solution (6), with almost 19%, which again matches Maximin function classification logic. In their response, the participants who selected the Solutions (4) or (6) explained that they wanted to compromise, but they believed that one of the objectives was more important than the other because Solution (4) is better in summer than Solution (6) and vice versa. Solutions (4, 5, 6), which represents 30% of the options were selected by 89% of the participants.

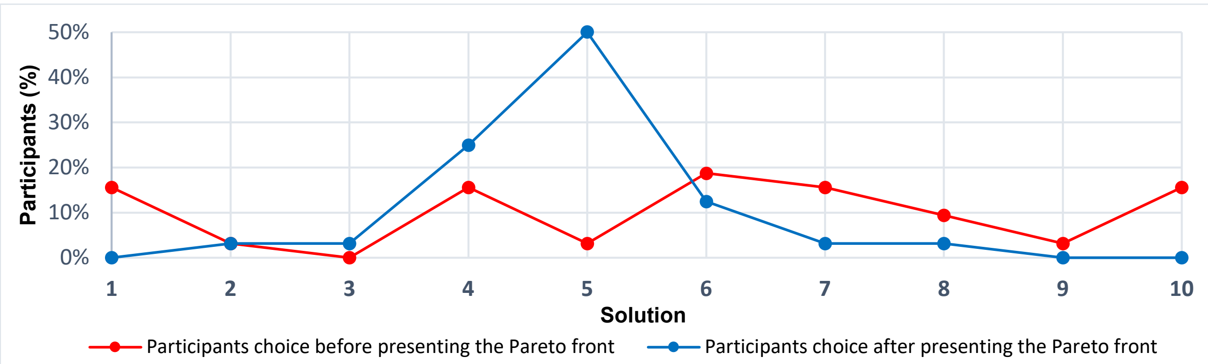


Figure 76: *Experiment 3, comparison of participants' selections before and after presenting the Pareto front*

In conclusion, Maximin resulted in a classification that is similar to the classification of the participants. Figure 77 shows the classifications based on the participants' responses after they saw the Pareto's front. Also, it presents the classification of the solutions based on Maximin aggregation function. The figure helps to observe the difference between the three classifications (Maximin, Pareto's function, the participants). With this figure, it is clear that the participants' classification matches Maximin classification. However, Maximin classification method was not presented to the participants; they adopted the approach of Maximin spontaneously.

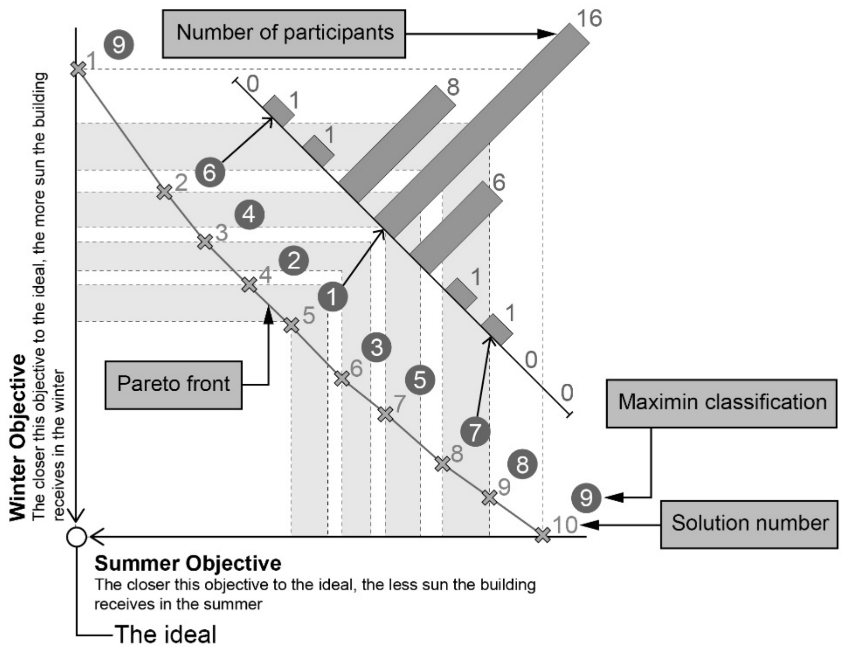


Figure 77: Experiment 3, comparison of Pareto front, Maximin, and participants' selections

These are remarkable results because they emphasize the limits of Pareto's aggregation function. As presented earlier, Maximin function uses a precautionary principle, and the participants seem to accept this principle spontaneously. However, it is evident that the designers need to see the scatterplot, including the Pareto front, to adopt the precautionary principle. The most reasonable justification is that, before presenting the scatterplot, the participants' were confused by the type of available information. By seeing the scatterplot, including the Pareto front, the information was organized, which allowed the participants to exploit it. This can be linked to particular cognitive biases.

As Pareto's aggregation function is widely used in many scientific domains related to optimization, these results lead to significant consequences. Pareto's function may result in unacceptable design solutions. However, Pareto's function classification can help in organizing the solutions. Pareto's function classification is calculable from a simple ordinal classification of the design solutions according to each design criterion. Though, Pareto's function is often ineffective as it may classify many solutions at the same level (many solutions can belong to the Pareto front). Hence, it can be regarded as a low-efficiency filter. When the number of solutions that belong to the Pareto front is high, Pareto's classification may become not very helpful for designers.

6.2 Experiment 4

From analyzing different aggregation functions (see 2.3), it seems that information availability is critical for classifying a set of solutions. Each function uses different information related to its negentropy. A function cannot benefit from the information that exceeds its capacity, and cannot operate with less than the information it requires. In contrast, humans are more flexible; they can adapt to the available information. Experiment 4 intends to study the influence of information availability on designers' judgment. It also intends to compare the classification resulting from Pareto's function to the classification of Maximin function by using designers' judgment as a benchmark. The experiment performs the investigation on two sets of solutions, each presented on a scatterplot. Each plot is used to test three different levels of information availability; low (scatterplot), Medium (scatterplot and numerical data), High (scatterplot, numerical data, 3D analyzed forms).

6.2.1 Procedure

Six online-based tests were prepared and sent to groups of building designers. Half of the tests use the set of solutions presented in Scatterplot 1 (see Figure 78). The other half of the tests uses the set of solutions presented in Scatterplot 2 (see Figure 79). Each scatterplot represents two objectives. The first objective is concerned with maximizing the solar gain during the wintertime. The second objective is concerned with minimizing solar gain during the summertime. For each scatterplot, each of the three tests presents different levels of information (see Table 15) to the participants. In each test, the participants were asked to rank the seven-candidate solutions (A, B, C, D, E, F, G) aiming to optimize the objectives (see Appendix IV). Each candidate represents a different proposal for mixed-use buildings (office, residential, commercial) in a mixed climate, which is hot in summer and cold in winter (Shanghai, China).

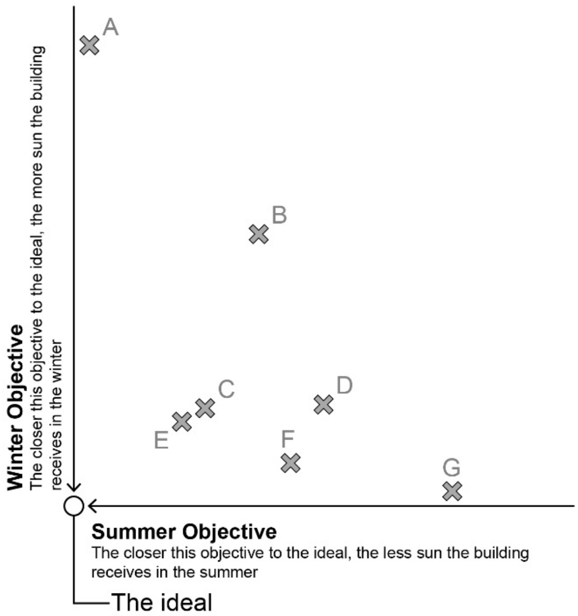


Figure 78: *Experiment 4, Scatterplot 1*

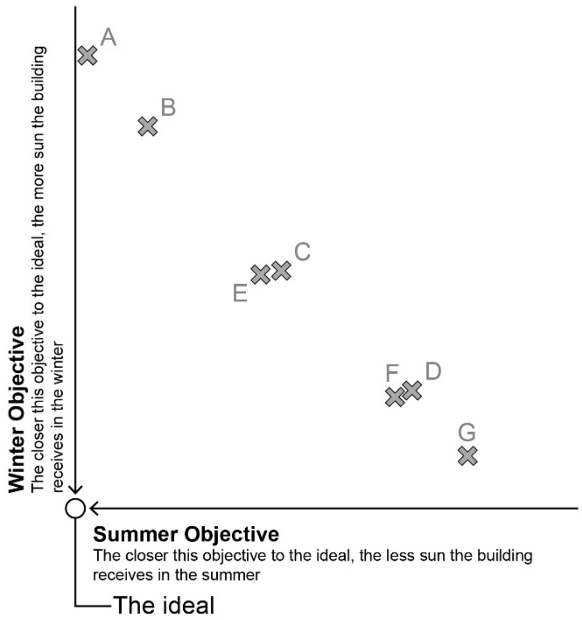


Figure 79: *Experiment 4, Scatterplot 2*

Level of information	Scatterplot	Building Description*	Analyzed 3D Forms**
Low	✔	✖	✖
Medium	✔	✔	✖
High	✔	✔	✔

Table 15: *Experiment 4, the different level of information*

* Building description includes numerical information about the building and its location. The following list demonstrates this information:

- The building is modular, the module size is: (Width=6m, Length=8m, Hight=4m).
- The required interior area is 12000 m^2.
- The land lot size is 8000 m^2 (W=80m, L=100m).
- The maximum number of levels allowed is three stories.
- A (25m X 30m) courtyard is required in the center of the building.
- The monthly average low temperature of Shanghai, China (see Appendix IV)
- The monthly average high temperature of Shanghai, China (see Appendix IV)
- The monthly sunshine hours of Shanghai, China were (presented in the test)

** Figure 80 shows an example of the analyzed 3D forms. The analysis represents the performance of one solution in regard to the objectives. The color code is used to represent the amount of solar radiation the building receives.

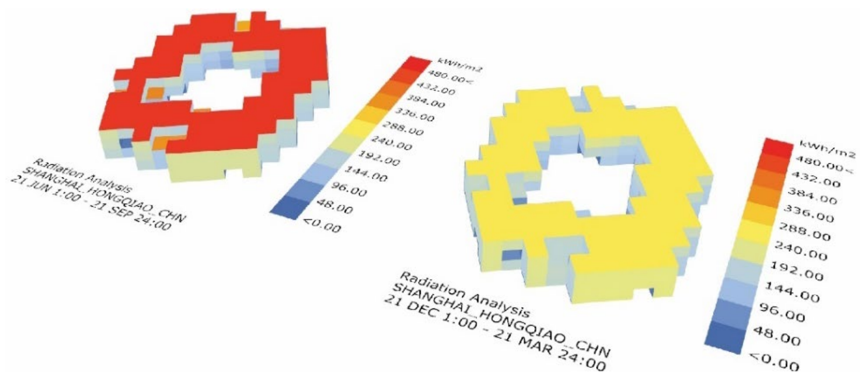


Figure 80: *Experiment 4, an example of the used solutions, the form on the left represents summer objective, and the other represents winter objective.*

6.2.2 Participants

In this part, the information about the participants of each of the six tests is presented.

6.2.2.1 Scatterplot 1 with a low level of information

The total number of participants in this test was 124 persons. Table 16 demonstrate the participants' Category/Speciality. Table 17 shows how frequent the participants use multi-objective design optimization in their work.

Category/Speciality	Count	Percentage
Student/Architecture	50	40.3%
Student/Engneering	6	4.8%
Professional/Architecture	39	31.5%
Professional/Engineering	7	5.6%
Professor/Architecture	14	11.3%
Professor/Engineering	4	3.2%
Professor, Professional/Architecture	2	1.6%
Professor, Professional/Architecture, Engineering	1	0.8%
Professional/Real-estate developer	1	0.8%

Table 16: *Experiment 4, Scatterplot 1, low level of information participants categories/specialty*

Frequency	Count	Percentage
Never	35	28.2%
Rarely	19	15.3%
Sometimes	33	26.6%
Usually	29	23.4%
Always	8	6.5%

Table 17: *Experiment 4, Scatterplot 1, low level of information, frequency of using multi-objective design Optimization*

6.2.2.2 Scatterplot 1 with a medium level of information

The total number of participants in this test was 23 persons. Table 18 demonstrates the participants' Category/Speciality. Table 19 shows how frequent the participants use multi-objective design optimization in their work.

Category /Speciality	Count	Percentage
Student/Architecture	14	60.9%
Student/Engneering	3	13%
Professor/Architecture	1	4.3%
Professor/Engineering	3	13%
Professor, Professional/Architecture	1	4.3%
Professor, Professional/Architecture, Engineering	1	4.3%

Table 18: *Experiment 4, Scatterplot 1, medium level of information participants categories/specialty*

Frequency	Count	Percentage
Never	2	8.7%
Rarely	4	17.4%
Sometimes	9	39.1%
Usually	7	30.4%
Always	1	4.3%

Table 19: *Experiment 4, Scatterplot 1, medium level of information, frequency of using multi-objective design Optimization*

6.2.2.3 Scatterplot 1 with a high level of information

The total number of participants in this test was 21 persons. Table 20 demonstrate the participants' Category/Speciality. Table 21 shows how frequently the participants use multi-objective design optimization in their work.

Category/Speciality	Count	Percentage
Student/Architecture	13	61.9%
Student/Engneering	2	9.5%
Professional/Architecture	1	4.8%
Professor/Architecture	2	9.5%
Professor/Engineering	2	9.5%
Professor, Professional/Architecture	1	4.8%

Table 20: *Experiment 4, Scatterplot 1, high level of information participants categories/specialty*

Frequency	Count	Percentage
Never	4	19%
Rarely	3	14.3%
Sometimes	7	33.3%
Usually	5	23.8%
Always	2	9.5%

Table 21: *Experiment 4, Scatterplot 1, high level of information, frequency of using multi-objective design Optimization*

6.2.2.4 Scatterplot 2 with a low level of information

The participants of this test are the same participants of scatterplot 1 with a medium level of information (see 6.2.2.2).

6.2.2.5 Scatterplot 2 with a medium level of information

The participants of this test are the same participants of scatterplot 1 with a high level of information (see 6.2.2.3).

6.2.2.6 Scatterplot 2 with a high level of information

The total number of participants in this test was 39 persons. Table 22 demonstrate the participants' Category/Speciality. Table 23 shows how frequently the participants use multi-objective design optimization in their work.

Category/Speciality	Count	Percentage
Student/Architecture	27	2.6%
Student/Engneering	4	10.3%
Professional/Architecture	2	5.1%
Professor/Architecture	1	2.6%
Professor/Engineering	3	7.7%
Professor, Professional/Architecture	1	2.6%
Professor, Professional/Architecture, Engineering	1	2.6%

Table 22: *Experiment 4, Scatterplot 2, high level of information participants categories/specialty*

Frequency	Count	Percentage
Never	9	23.1%
Rarely	5	12.8%
Sometimes	11	28.2%
Usually	11	28.2%
Always	3	7.7%

Table 23: *Experiment 4, Scatterplot 2, high level of information, frequency of using multi-objective design Optimization*

6.2.3 Results

The data collected from each test are analyzed to investigate two different issues. Firstly, to observe if the different levels of information significantly affect the decisions of the designers. Secondly, to compare the classification of Maximin functions from one side to Pareto's function from the other side based by linking them to designers' classification.

6.2.3.1 Scatterplot 1

Before presenting the designers' point of view, it is vital to present the classification of the solutions of Scatterplot 1 derived from Pareto's function and Maximin (see Table 24); both classification methods of Pareto's function which were presented earlier (see 2.3.1) result in identical classification for the solutions of Scatterplot 1.

Solution	Pareto's function	Maximin
Solution A	1 (Pareto front)	7
Solution B	6	5
Solution C	5	2
Solution D	6	4
Solution E	1 (Pareto front)	1 (The best among the population)
Solution F	1 (Pareto front)	3
Solution G	1 (Pareto front)	6

Table 24: *Experiment 4, Scatterplot 1 solutions' classification based on Pareto's function and Maximin function*

Figure 81 to Figure 83 shows the responses of the participants for the three tests of Scatterplot 1; each figure represents a different level of information (see Table 15 (6.2.1)). From the figures, it is clear that the dominant solution for every class is identical to Maximin classification, which is E > C > F > D > B > G > A (E ranked 1st, A Ranked 7th) for the three tests. The average classification of the three tests based on Scatterplot 1 is also E > C > F > D > B > G > A (E ranked 1st, A Ranked 7th). This average classification is calculated based on a point system consisting of four steps; the system averages the solutions by taking into account the ranks of the solutions. First, for each solution, the participants' selection for every class is counted; this must be repeated for the three tests. Second, the average of the participants count for each solution is computed for each class. Third, for each solution, the average of each class is multiplied by ((solutions' count + 1) − the class rank). Fourth, the multiplication results of all the classes for each solution are averaged. The result represents the points of the solutions, which then are used to classify them. In general, the participants' classification is similar to Maximin function classification and distinct from Pareto's function classification.

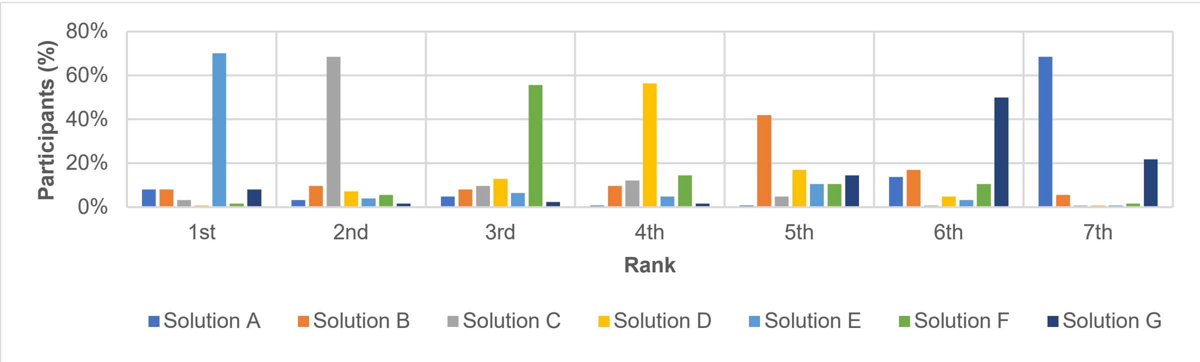


Figure 81: *Experiment 4, participants' responses of Scatterplot 1 with a low level of information test*

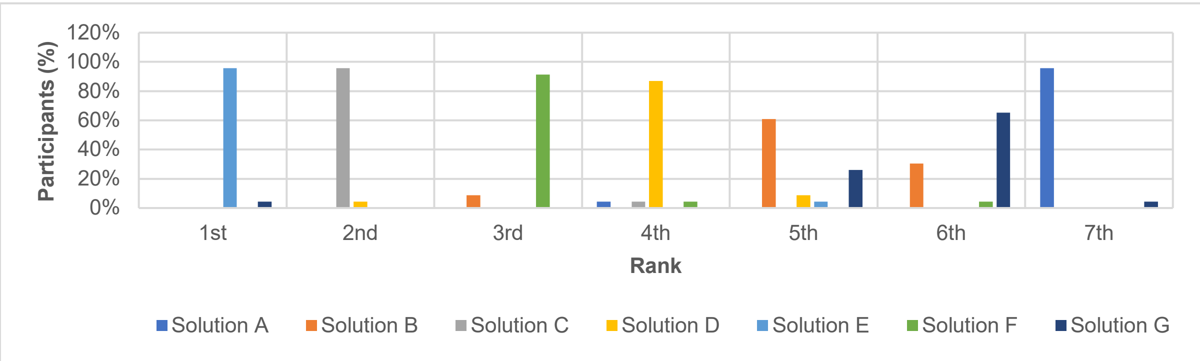


Figure 82: *Experiment 4, participants' responses of Scatterplot 1 with a medium level of information test*

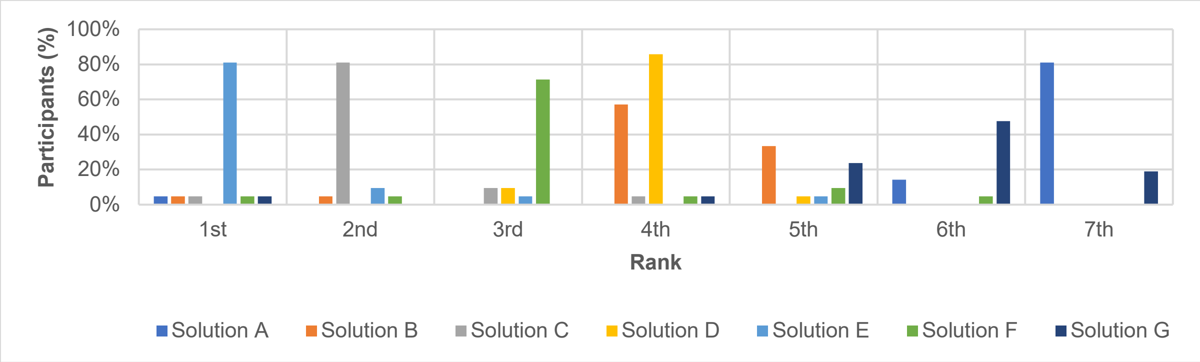


Figure 83: *Experiment 4, participants' responses of Scatterplot 1 with a high level of information test*

Figure 84 to Figure 90 compare designers' classification for each solution resulting from the three tests of Scatterplot 1. Each figure observes one solution when a different level of information was given. From the figures, it is clear that the different levels of information have no significant impact on designers' decisions when faced with Scatterplot 1, except for solution B, which slightly changed more than the other, but still, these changes are not drastic.

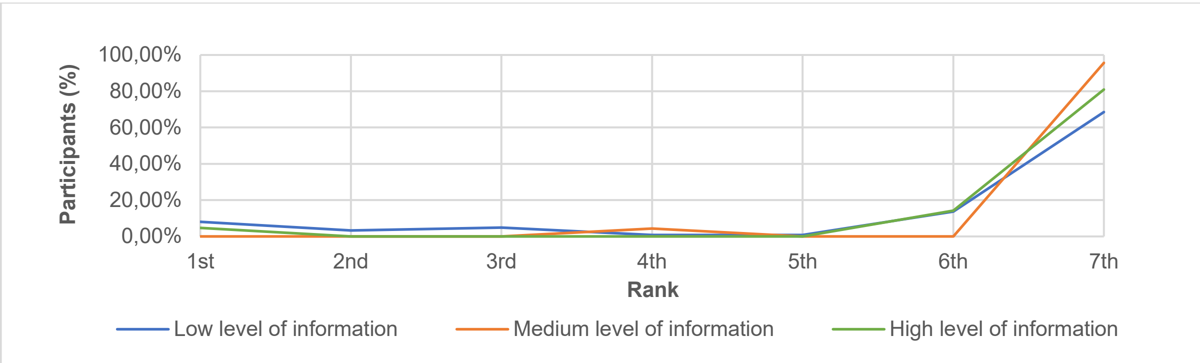


Figure 84: *Experiment 4, Scatterplot 1, Solution A, participants' classification when faced with different information*

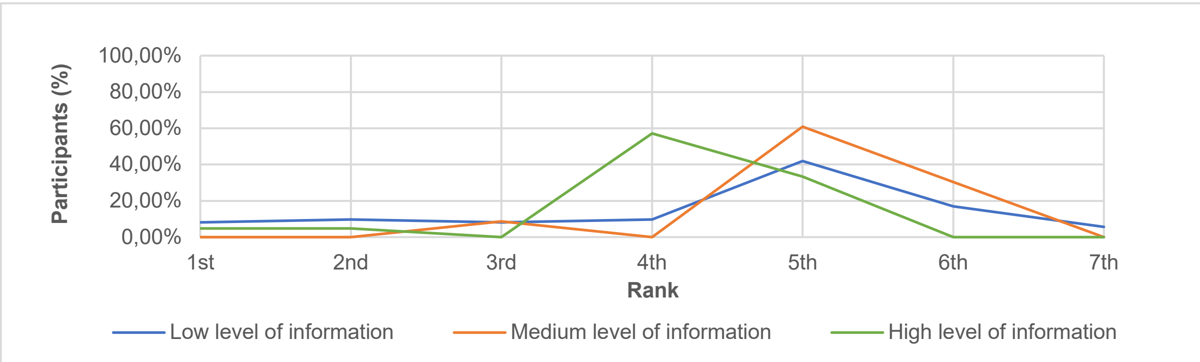


Figure 85: *Experiment 4, Scatterplot 1, Solution B, participants' classification when faced with different information*

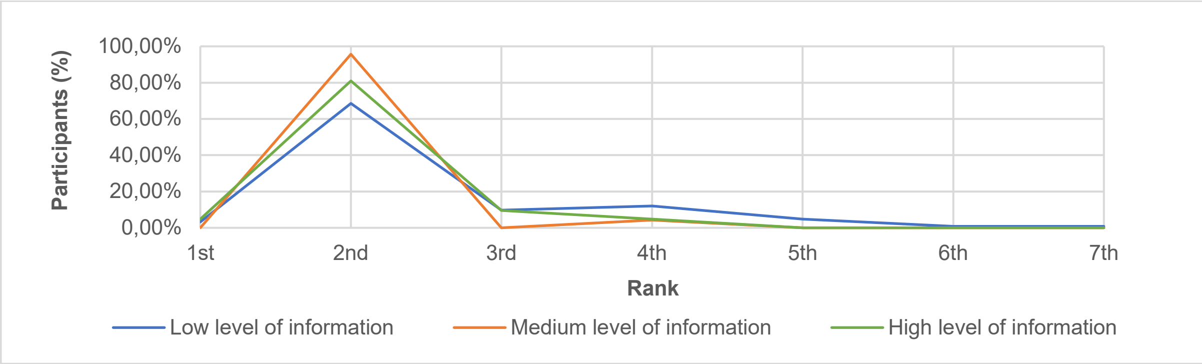


Figure 86: *Experiment 4, Scatterplot 1, Solution C, participants' classification when faced with different information*

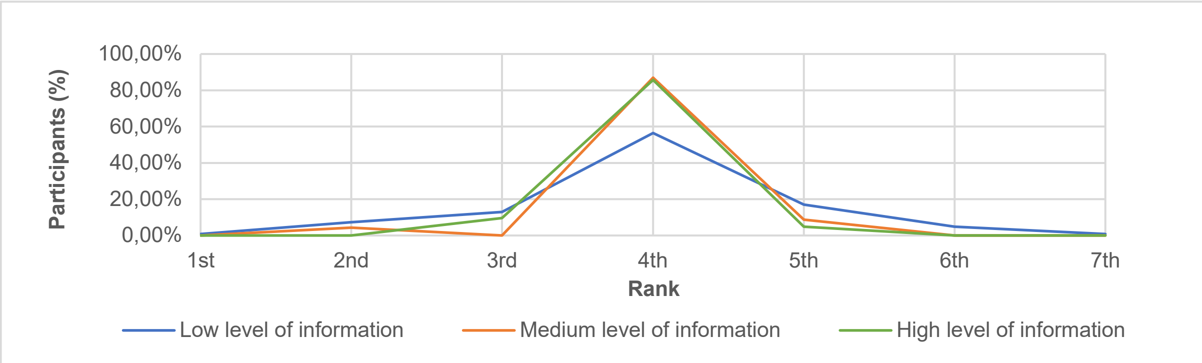


Figure 87: *Experiment 4, Scatterplot 1, Solution D, participants' classification when faced with different information*

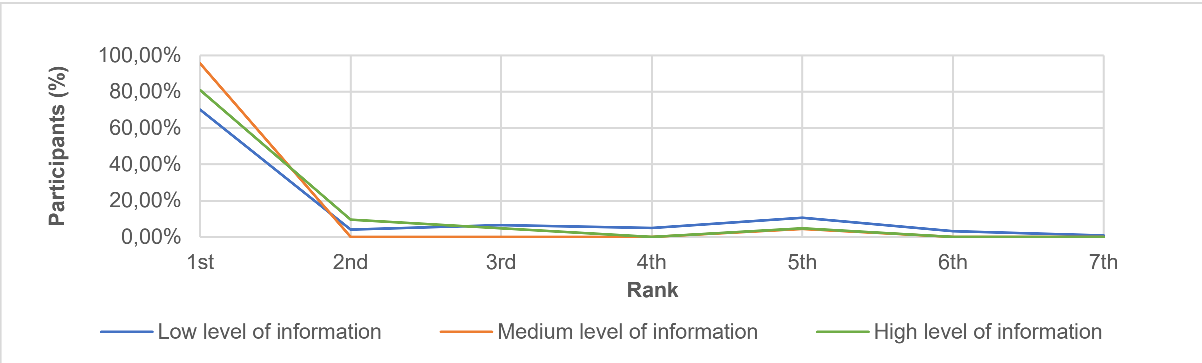


Figure 88: *Experiment 4, Scatterplot 1, Solution E, participants classification when faced with different information*

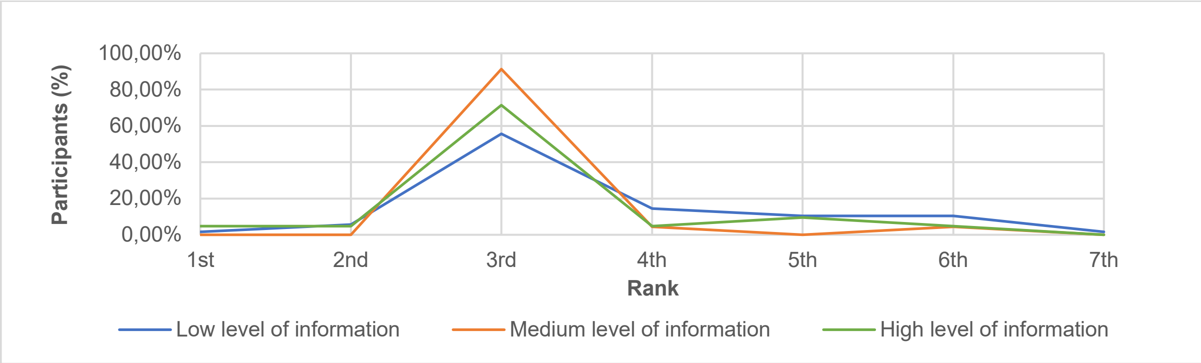


Figure 89: *Experiment 4, Scatterplot 1, Solution F, participants' classification when faced with different information*

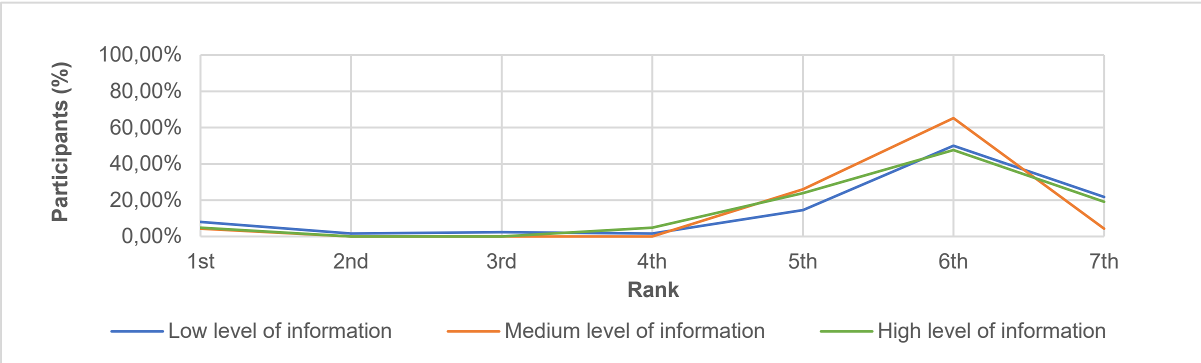


Figure 90: *Experiment 4, Scatterplot 1, Solution G, participants' classification when faced with different information*

The previous analysis, which is based on using a scoring system to calculate the average classification of the participants' is not enough. By using the average, the answers may compensate each other, which can be misleading. Another analysis of the deviation between the classification derived from each function and the classification of each participant separately is required. Figure 91 to Figure 93 demonstrates these deviations for Scatterplot 1; each figure represents one test based on a different level of information (see Table 15 (6.2.1)). Every deviation is computed from four steps. First, we must assign a number to represent each solution (A=1, B=2, C=3, D=4, E=5, E=6, F=7). Then, we compute the absolute differences between Pareto's function or Maximin function classification and the classification of one participant for each solution. Next, we compute the absolute differences between the classification resulting from each function and the reverse of that classifications. The inversed classification is far from the original classification; it serves as a benchmark for comparison. Finally, to calculate the deviation, the sum of the first absolute difference is divided by the second absolute difference for each function. In the figures, we can see that the difference between Pareto's function classification and participants' classification is far; in some responses further than the benchmark, some relative deviations are over 100%. Maximin classification is closer to the classification of the participants.

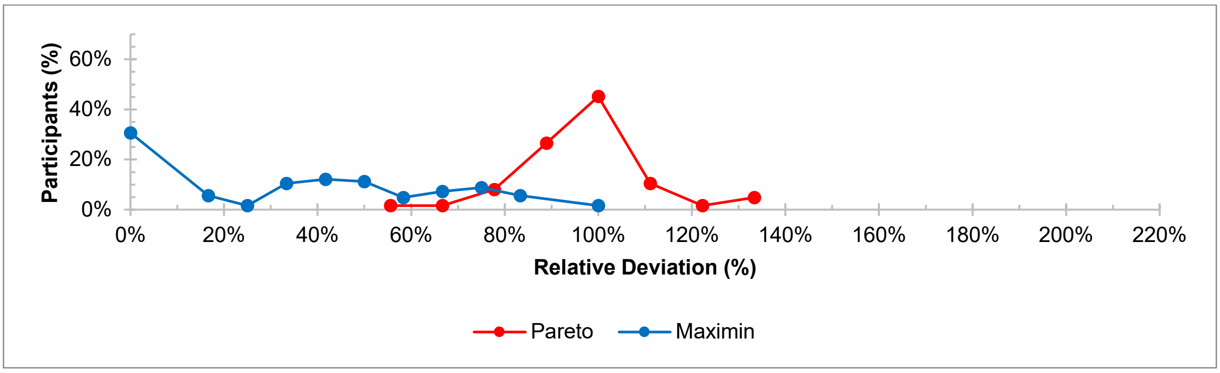


Figure 91: *Experiment 4, Scatterplot 1, low level of information, the relative deviation between different aggregation functions and designers' classification*

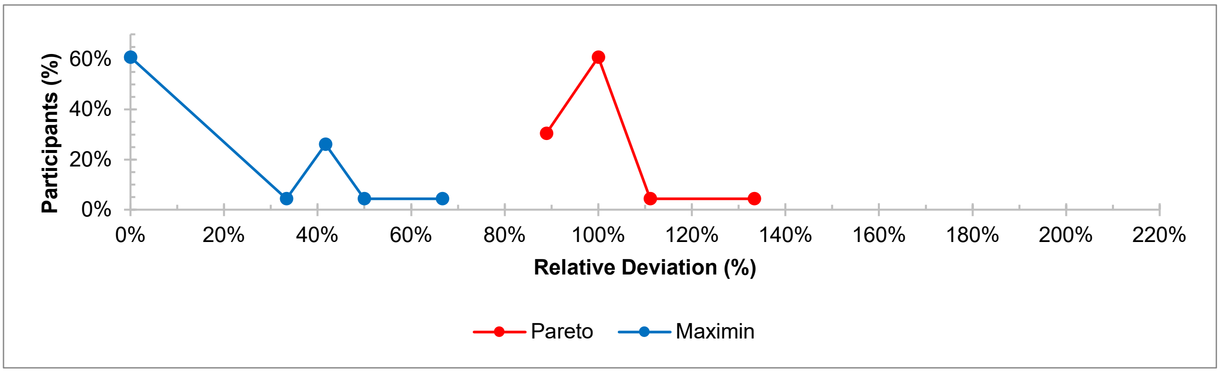


Figure 92: *Experiment 4, Scatterplot 1, medium level of information, the relative deviation between different aggregation functions and designers' classification*

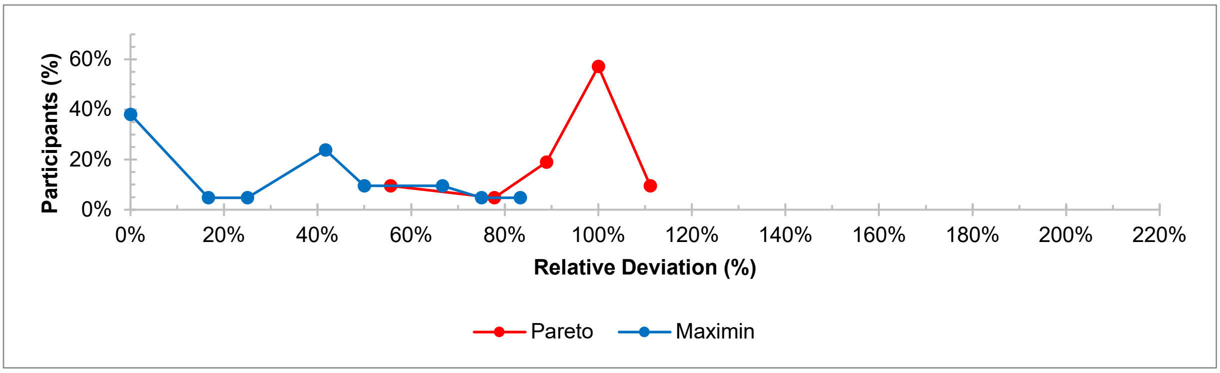


Figure 93: *Experiment 4, Scatterplot 1, high level of information, the relative deviation between different aggregation functions and designers' classification*

From a more accurate observation of the participants' responses, we regard deviation values lower than 20% as low values. Consequently, for Maximin, 40% of the answers are regarded as low in deviation, for Pareto's function classification, 0% of the participants have a deviation lower than 20%. In fact, for the classification derived from Pareto's function, the majority of deviation values are around 100%. Thus, we can infer that a non-negligible amount of the participants adopt the use of Maximin aggregation function spontaneously.

However, we can derive the same classification of Maximin for the set of solutions presented in Scatterplot 1 by using Derringer & Suich's aggregation function and by assuming that the weights of the objectives are equal. Hence, another experiment that compares these two aggregation functions by using designers' classification as a benchmark is required. Though the figures show that the majority of the participants recommend classifications that we are unable to interpret, since the information and the proposed process are straightforward, we can assume that the responses are not misled. Most likely, those participants used an approach based on their own experience in a further creative manner compared to the other participants.

6.2.3.2 Scatterplot 2

As in Scatterplot 1, before presenting designers' classification, it is essential to present the classification of the solutions of Scatterplot 2 based on the different aggregation functions (see Table 25); both classification methods of Pareto's function which was presented earlier (see 2.3.1) results in identical classification for the solutions of Scatterplot 2.

Solution	Pareto	Maximin, Derringer & Suich (equal weights)
Solution A	1 (Pareto front)	7
Solution B	1 (Pareto front)	5
Solution C	6	2
Solution D	6	4
Solution E	1 (Pareto front)	1 (the best among the population)
Solution F	1 (Pareto front)	3
Solution G	1 (Pareto front)	6

Table 25: *Experiment 4, Scatterplot 2 solutions' classification based on Pareto's function and Maximin function*

Figure 94 to Figure 96 shows the responses of the participants of scatterplot 2; each figure represents a different level of information (see Table 15 (6.2.1)). We can observe that the dominant solution for every rank is identical to Maximin classification, which is E > C > F > D > B > G > A (E ranked 1st, A Ranked 7th) (see Table 25). The average classification of the participants is calculated by following the same point system used for Scatterplot 1 (see 6.2.3.1). The average classification of the three tests of Scatterplot 2 is also E > C > F > D > B > G > A (E ranked 1st, A Ranked 7th). The participants' classification is similar to Maximin function classification and distinct from Pareto's function classification.

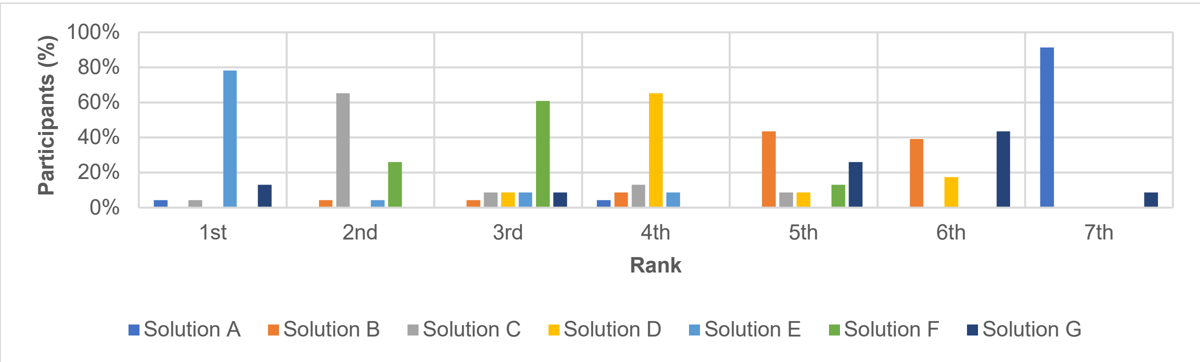


Figure 94: *Experiment 4, participants' responses of Scatterplot 2 with a low level of information test*

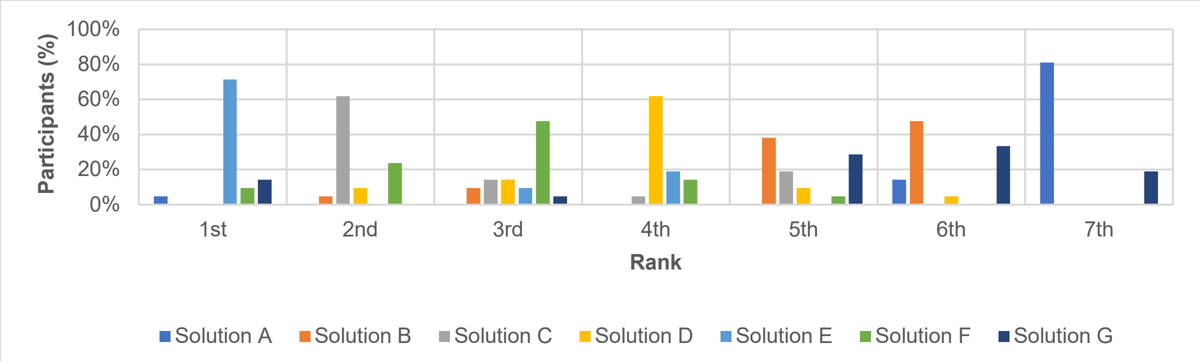


***Figure 95:** Experiment 4, participants' responses of Scatterplot 2 with a medium level of information test*

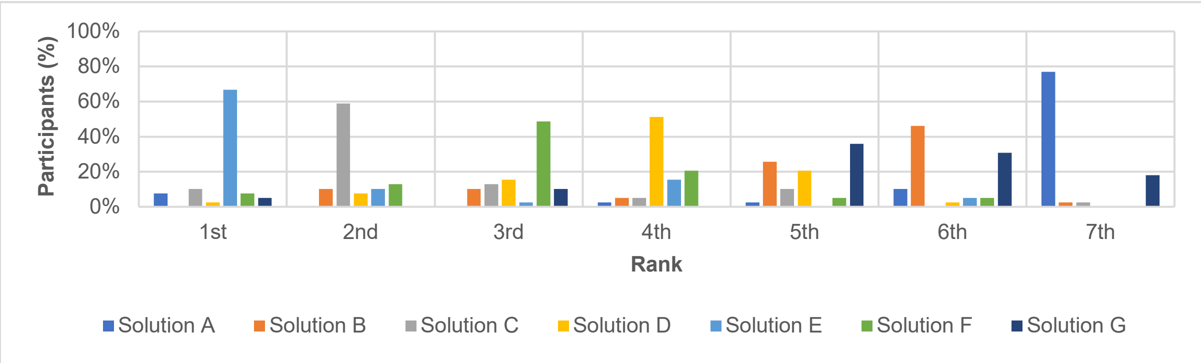


***Figure 96:** Experiment 4, participants' responses of Scatterplot 2 with a high level of information test*

Figure 97 to Figure 103 compare designers' classification for each solution resulting from the three tests of Scatterplot 2. Each figure observes one solution when a different level of information was given. From the figures, it is clear that the different level of information has no significant impact on designers decisions when faced with Scatterplot 2, including for solution B.

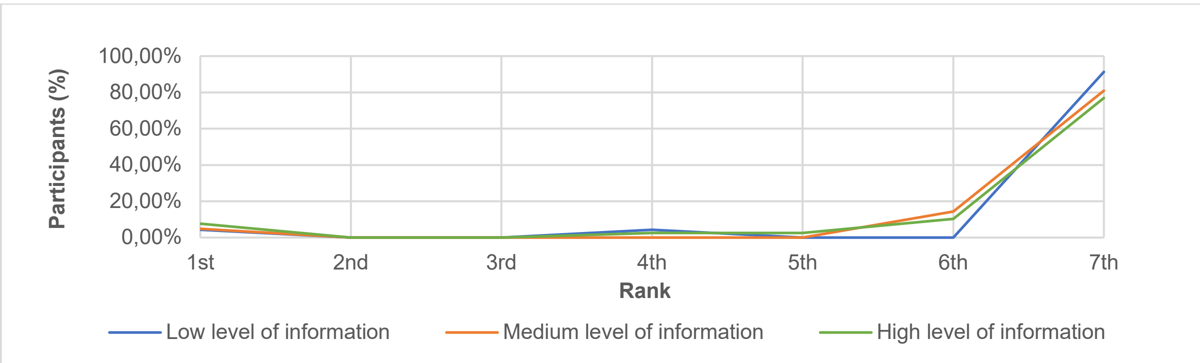


***Figure 97:** Experiment 4, Scatterplot 2, Solution A, participants' classification when faced with different information*

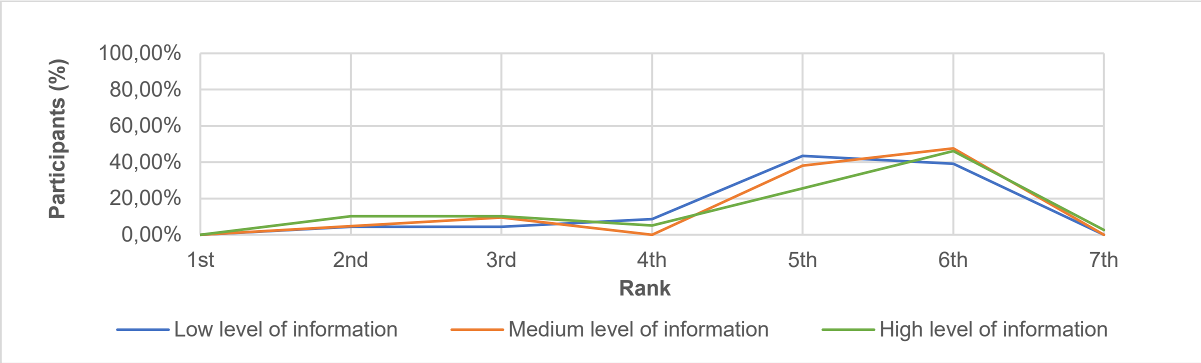


Figure 98: *Experiment 4, Scatterplot 2, Solution B, participants' classification when faced with different information*

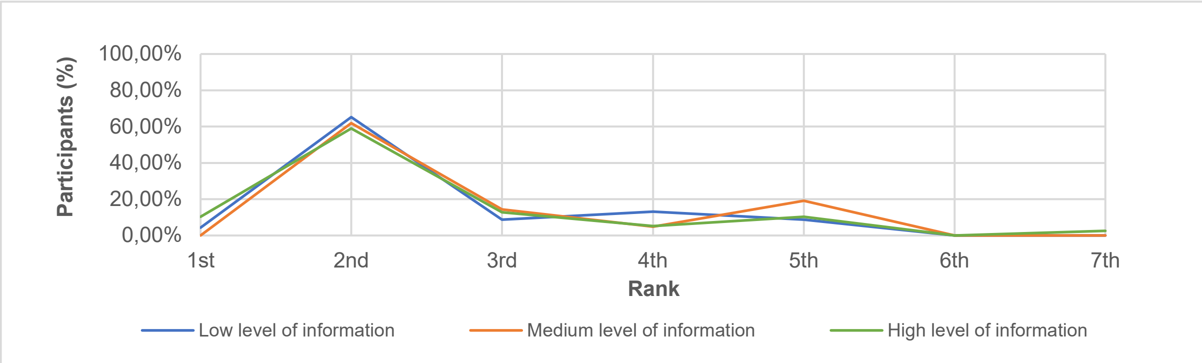


Figure 99: *Experiment 4, Scatterplot 2, Solution C, participants' classification when faced with different information*

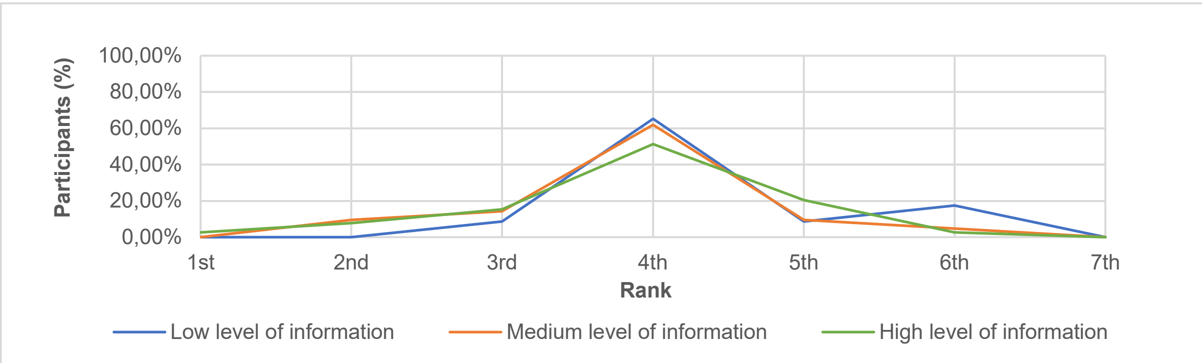


Figure 100: *Experiment 4, Scatterplot 2, Solution D, participants' classification when faced with different information*

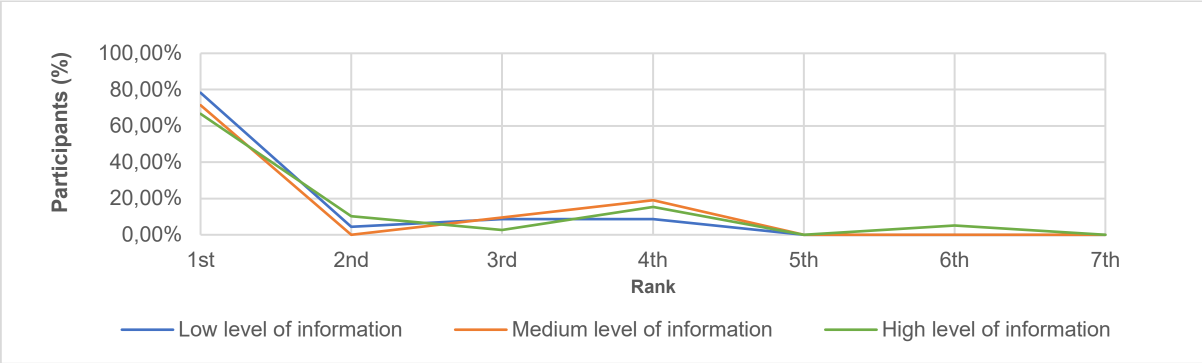


Figure 101: *Experiment 4, Scatterplot 2, Solution E, participants' classification when faced with different information*

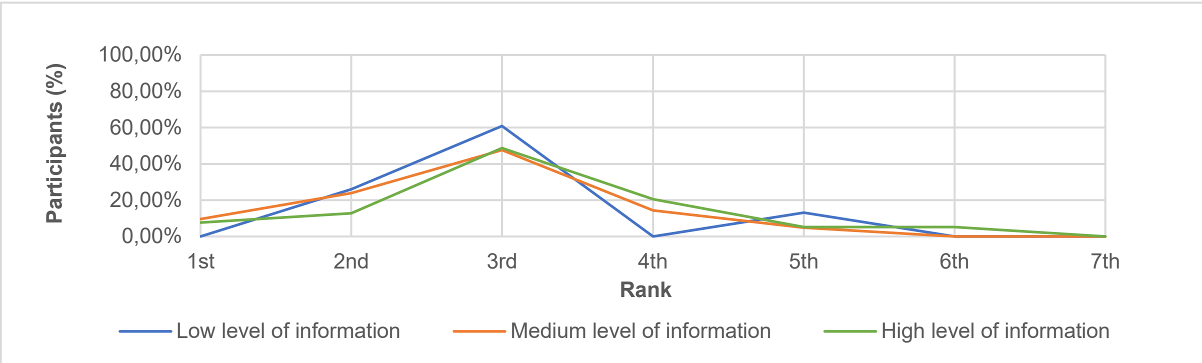


Figure 102: *Experiment 4, Scatterplot 2, Solution F, participants' classification when faced with different information*

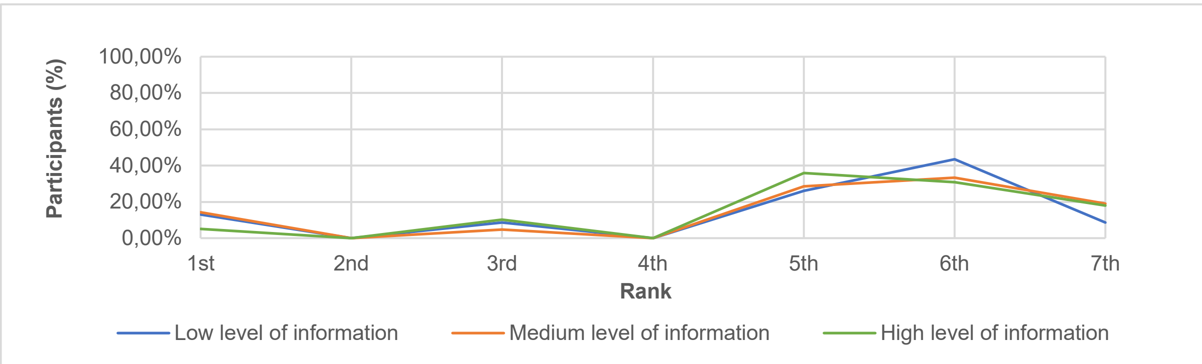


Figure 103: *Experiment 4, Scatterplot 2, Solution G, participants' classification when faced with different information*

As in Scatterplot 1, it is crucial to compute the deviation between the classification of the different functions and the classification of each participant in Scatterplot 2. Figure 104 to Figure 106 demonstrates the deviations of the different classification approaches for Scatterplot 2; each figure represents one test based on different levels of information (see Table 15 (6.2.1)). The deviation is calculated by following the same steps used to calculate the deviation for Scatterplot 1 (see 6.2.3.1). In the figures, we can see that the difference between Pareto's function classification and participants' classification is far; in some responses further than the benchmark, some relative deviations are over 100%.

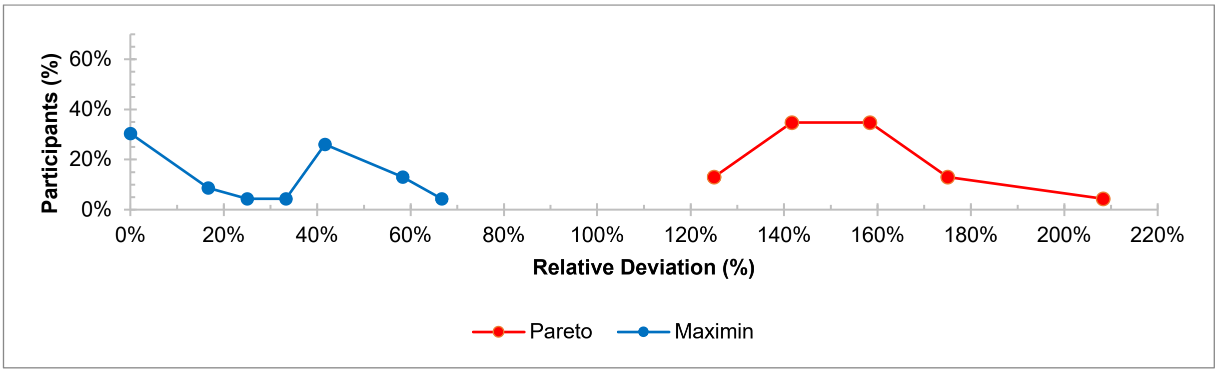


Figure 104: *Experiment 4, Scatterplot 2, low level of information, the relative deviation between different aggregation functions and designers' classification*

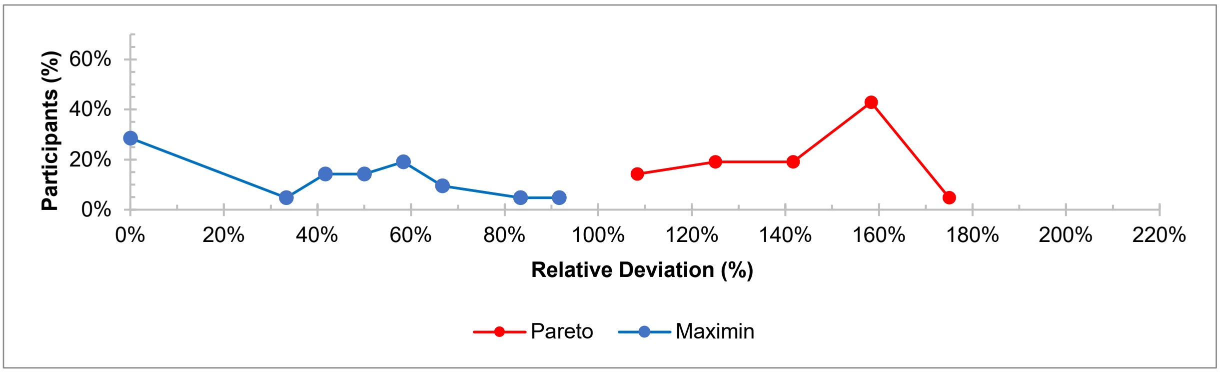


Figure 105: *Experiment 4, Scatterplot 2, medium level of information, the relative deviation between different aggregation functions and designers' classification*

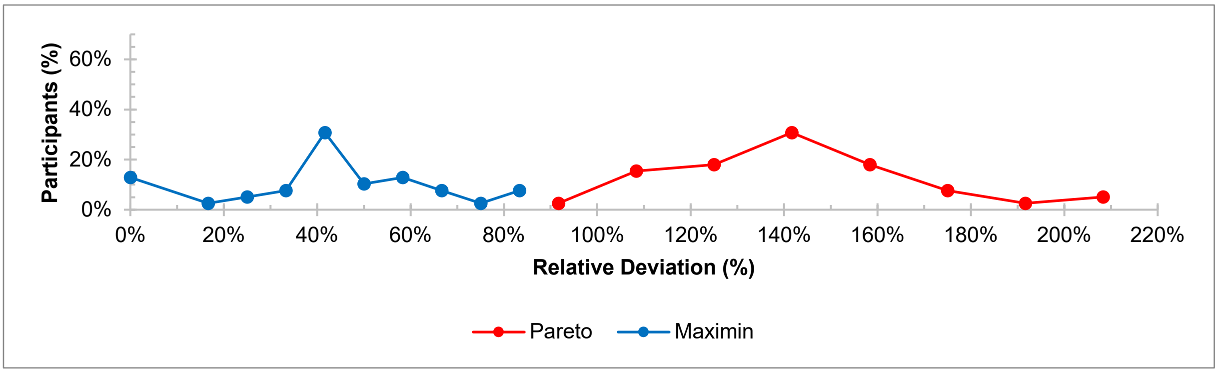


Figure 106: *Experiment 4, Scatterplot 2, high level of information, the relative deviation between different aggregation functions and designers' classification*

Similar to Scatterplot 1, to observe the deviation, the deviation of less than 20% are regarded as low values. Accordingly, for Maximin, 25% of the answers are regarded as low in deviation. However, for Pareto's function, 0% have a deviation lower than 20%; this is similar to what has been observed in Scatterplot 1. Furthermore, as in Scatterplot 1 for the classification derived from Pareto's function, the majority of deviation values are over 100%. Therefore, it can be concluded that a significant group of the participants is spontaneously adopting Maximin aggregation function.

However, similar to Scatterplot 1, for the set of solutions presented in Scatterplot 2, we can derive the same classification resulted from Maximin by using Derringer & Suich's aggregation function if we assume that the objectives share identical weights. Therefore, an additional experiment that compares both functions by using designers' classification as benchmarks is needed. Though, like Scatterplot 1, most of the participants adopt classifications that we are unable to interpret. Given That the proposed process and the involved information are simple, it is reasonable to assume the responses are not misleading. Compared to the other participants, those participants probably adopted an approach based on their own experience in a further creative way.

From the results, we can conclude the classification resulting from the high in negentropy aggregation functions "Maximin function or Derringer & Suich's function (equal objectives weights)" are relatively close to designers' classification in comparison to the classification resulting from the low in negentropy function "Pareto's function." It is highly recommended to use these high in negentropy functions when the available information is sufficient. Another experiment that compares the classification of Maximin function and Derringer & Suich's function (equal objectives weights) to designers' classification is required. However, if scarcity of information is the case, then Pareto's function can be used. The results also show that giving information more than scatterplot, building type, and location did not significantly influence designers' decisions. From Experiments 1 and 2, we can say that the graphical representation of the solution's performance profoundly influences the designers' judgment. This type of representation organizes the information and facilitate comparing the candidate solutions.

6.3 Experiment 5

The previous experiment finds that designers' reasoning is relatively close to Maximin. However, the set of solutions used for Experiment 4 results in identical classification when Maximin or Derringer & Suich's aggregation functions (equal weights) are used. Therefore, another experiment that compares these two functions is critical. This experiment might improve our understanding of the designers' reasoning. Experiment 5 intends to perform a comparison of these two functions. Figure 107 represents the difference between the classification of both functions. The figure shows two solutions "S1" and "S2". According to Maximin function, "S2" is better than "S1", while based on Derringer & Suich's function (equal weights), "S1" is better than "S2". According to these differences, the experiment must use sets of solutions that result in different classification when aggregated by each of the two functions. The solutions of these sets must be classified by buildings designers to provide a benchmark for the comparison.

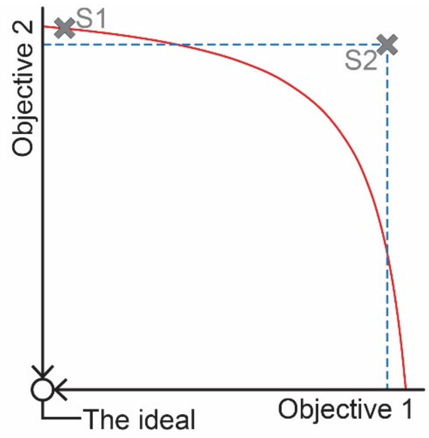


Figure 107: *Comparing the classification of Maximin and Derringer & Suich (equal weights)*

It is worth noting that if reliable objectives weights are available, then the Derringer & Suich's aggregation function must be the designers' choice. However, if reliable objectives weights are not available, then Maximin or Derringer & Suich's function (equal weights) or Maximin can be used. By using these two functions, the designer can express his preference only within the Interpretation model of MOIA. In that case, the main difference between both functions is that Maximin is non-compensatory, and it avoids extreme interpretation variables **z**, while Derringer & Suich's function (equal weights) is compensatory.

6.3.1 Procedure

The participants were invited to complete two online-based tests (see Appendix V). In each test, they were asked to classify a set of five solutions (A, B, C, D, E). In the first test (see Appendix V), the participants must classify the solutions to satisfy two objectives. The objectives are related to solar gain in summer and winter. Each solution represents a design alternative of a building in a mixed climate that is hot in summer and cold in winter (Shanghai, China). The solutions' satisfaction of the objectives were presented to the participants (see Table 26); 1 means complete satisfaction, and 0 means no satisfaction. Also, a scatterplot (Figure 108) and a parallel coordinate (Figure 109) were provided to represent the solutions' satisfaction of the objectives graphically. The classification of the participants will serve as a benchmark to compare the classification of the same set of solutions resulting from the two aggregation functions.

Solution	Summer objective level of satisfaction	Winter objective level of satisfaction
Solution A	0,9	0,45
Solution B	0,5	0,5
Solution C	0,8	0,25
Solution D	0,4	0,6
Solution E	0,35	0,95

Table 26: *Experiment 5, test 1, solutions' satisfaction of the objectives*

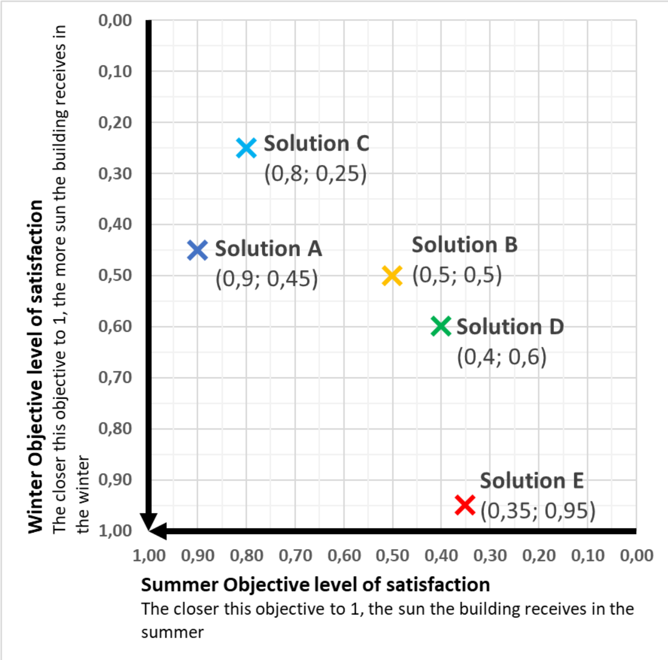


Figure 108: *Experiment 5, test 1, a scatterplot represents solutions' satisfaction of the objectives*

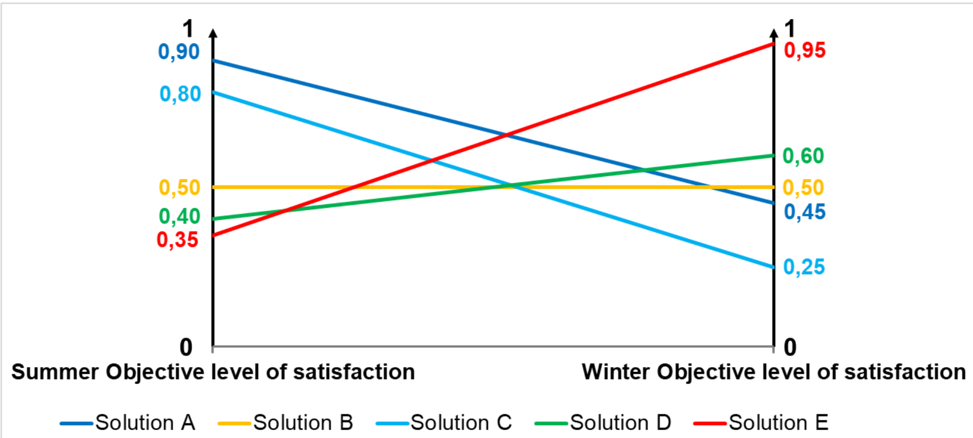


Figure 109: *Experiment 5, Test 1, a parallel coordinate represents solutions' satisfaction of the objectives*

For the second test (see Appendix V), the participants must classify the solutions to satisfy four objectives. In contrast to the previous test, in this test, the objectives, climate, and location are not specified. The absence of these essential information will lead the participants to assume that these objectives are equal, which helps to neutralize the weight factor; it is evident that when weights are not equal, and the information is accessible, Derringer & Suich's function is suitable. The solutions' satisfaction of the objectives are presented to the participants (see Table 27); 1 is complete satisfaction, and 0 is no satisfaction. Since the scatterplot cannot represent more than three dimensions, a parallel coordinate (Figure 110) was used to represents the solutions' satisfaction of the objectives graphically to the participants. The classification of the participants will serve as a benchmark to compare the classification of the same set of solutions when aggregated from the two functions.

Solution	Objective 1	Objective 2	Objective 3	Objective 4
Solution A	1	1	1	0,3
Solution B	0,5	0,5	0,5	0,5
Solution C	0,4	0,4	0,4	0,4
Solution D	0,9	0,9	0,2	0,7
Solution E	0,2	0,2	0,5	0,4

Table 27: *Experiment 5, test 2, solutions' satisfaction of the objectives*

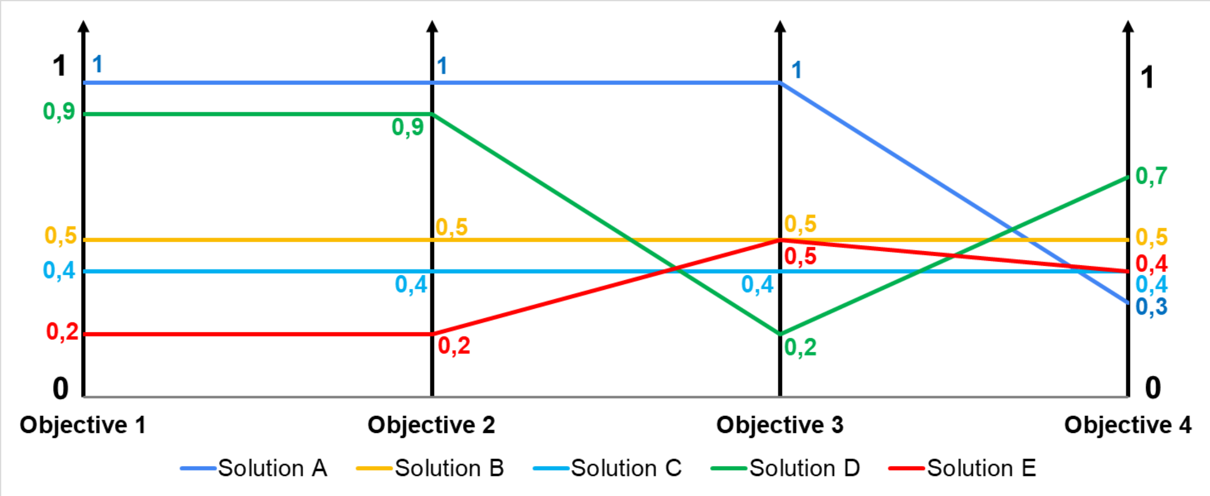


Figure 110: *Experiment 5, test 2, a parallel coordinate represents solutions' satisfaction of the objectives*

6.3.2 Participants

The total number of participants was 51 persons. Table 28 demonstrates the participants' Speciality/Category. Most of the participants have a background in architecture. All the participants of these experiments work in building design. Table 29 shows how frequently the participants use multi-objective design optimization in their work.

Specialty/Category	Count	Percentage
Architect/Student	14	27%
Architect/Professional	20	39%
Architect/Professor	8	16%
Engineer/Student	3	6%
Engineer/Professional	4	8%
Engineer/Professor	2	4%

Table 28: *Experiment 5, participants' specialty/categories*

Frequency	Count	Percentage
Never	11	22%
Rarely	9	18%
Sometimes	11	22%
Usually	12	24%
Always	8	16%

Table 29: *Experiment 5, participants' frequency of using multi-objective design optimization*

6.3.3 Results

Table 30 presents the classification of the solutions of test 1 of experiment 5 (see Table 26, Figure 108, Figure 109 (6.3.1)) of both aggregation functions, namely Maximin and Deringer & Suich's function (equal weights). The classification of Pareto's function is also presented to link the experiment to the previous ones.

Solution	Maximin	Derringer & Suich (equal weights)	Pareto's function
Solution A	2	1	1
Solution B	1	3	1
Solution C	5	5	5
Solution D	3	4	1
Solution E	4	2	1

Table 30: *Experiment 5, test 1, solutions classification based on three aggregation functions*

Figure 111 shows the responses of the participants for the first test of experiment 5. As can be seen in the figure, the dominant solutions of each class is A > B > B, C > D > E (A is the dominant of the 1st class, E is the dominant of the 5th class). The average classification of is A > B > D > C > E (A ranked 1st, E ranked 5th). This average classification is calculated based on the point system used in experiment 4 (see 6.2.3.1).

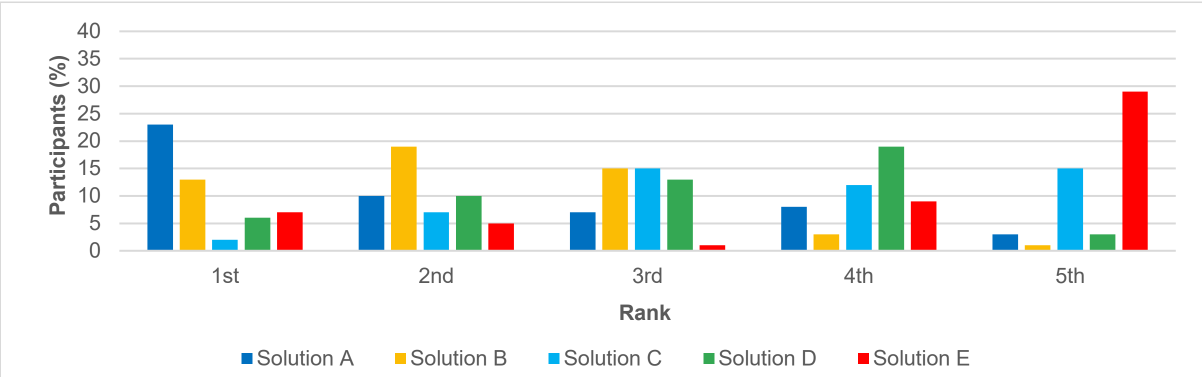


Figure 111: *Experiment 5, test 1, participants' responses*

By comparing the average classification resulting from the participants to the classification resulting from the two functions, we can observe the following. In both the average classification of the participants and of Derringer & Suich's (equal weights), Solution A is at the highest class. For the rest of the solutions, Derringer & Suich's present a similar classification to the participants, the only difference is that Solution E is shifted to be in the second class (see Figure 112).

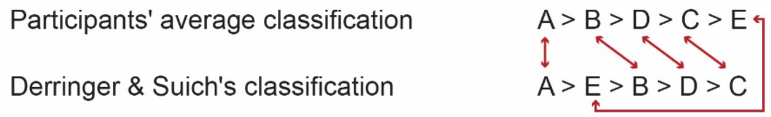


***Figure 112:** A comparison between the classification based on participants' average and Derringer & Suich's function (Experiment 5, test 1)*

Maximin also shares some similarities with the classification of the participants. The main difference is that compared to participants' average classification, Maximin switches the first two solutions and the last two solutions (see Figure 113). Hence, the participants' classification is mixed, and it does not entirely follow these functions. However, both functions are closer to the participants' classification average if compared to Pareto's function classification (see Table 30)

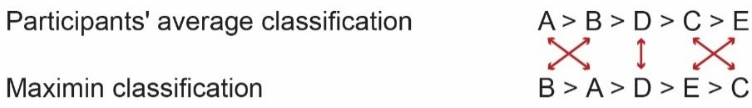


***Figure 113:** A comparison between the classification based on participants' average and Maximin function (Experiment 5, test 1)*

The previous analysis, which is based on scoring to calculate the average classification, is not enough. Another analysis of the deviation between the classification of both functions and the classification of each participant is required. Figure 114 demonstrates the deviations between the classifications of participants and the functions. To compute the deviation, we must assign a number to represent each solution (A=1, B=2, C=3, D=4, E=5). Then we used the same method presented in Experiment 4 (see 6.2.3.1). In the figure, we can see that the individuals' responses are closer to Maximin in comparison to Derringer & Suich's function.

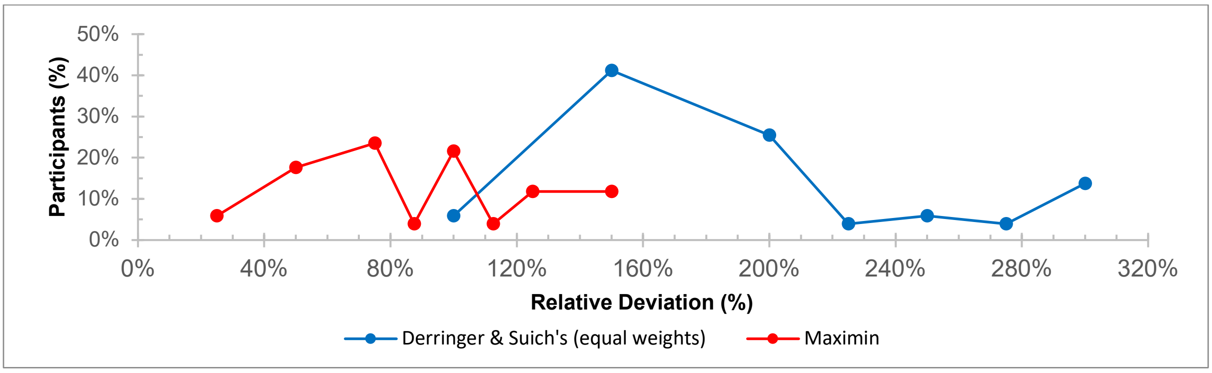


***Figure 114:** Experiment 5, test 1 the relative deviation between different aggregation functions and designers' classification*

Table 31 presents the classification of the solutions of the second test of experiment 5 (see Table 27, Figure 110 (6.3.1)) of both aggregation functions, namely Maximin and Deringer & Suich's function (equal weights). The classification of Pareto's function is also presented to link the experiment to the previous ones.

Solution	Maximin	Derringer & Suich (equal weights)	Pareto's function
Solution A	3	1	1
Solution B	1	3	1
Solution C	2	4	5
Solution D	4	2	1
Solution E	4	5	1

Table 31: *Experiment 5, test 2, solutions classification based on three aggregation functions*

Figure 115 shows the responses of the participants for the second test of experiment 5. As can be seen in the figure, the dominant solutions of each class is A > B > C > D > E (A is the dominant of the 1st class, E is the dominant of the 5th class). The average classification is B > A > C > D > E (B ranked 1st, E Ranked 5th). This average classification is calculated based on the point system used in experiment 4 (see 6.2.3.1).

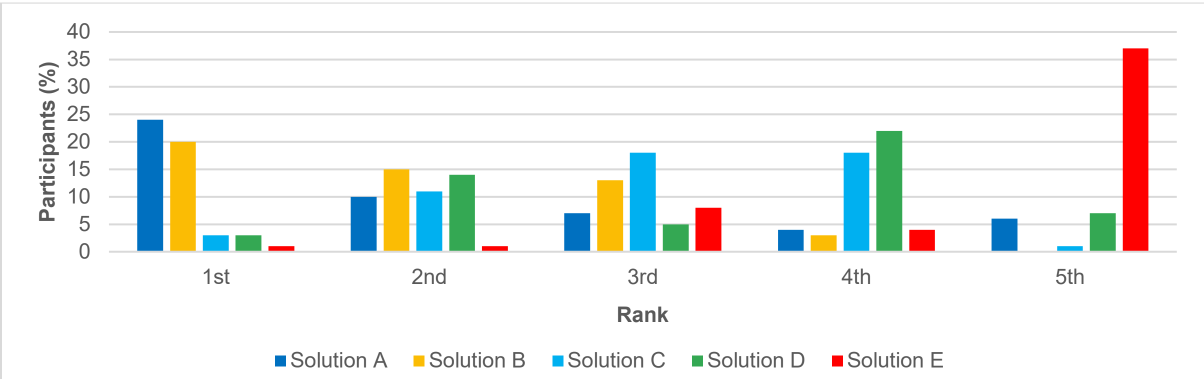


Figure 115: *Experiment 5, test 2, participants' responses*

By comparing the average classification resulting from the participants to the classification resulting from the two functions, we can realize the following. Both the participants' average and Maximin classified Solution B at the highest class (see Figure 116) However, Derringer & Suich's function (equal weights) classify Solution A as the highest, which matches the dominant solution of first-class based on participants' responses (see Figure 117). However, Maximin considers solutions D and E as equal solutions. Finally, we can realize that for the participants and both functions, Solution E is in the lowest class (see Figure 116, Figure 117) according to Pareto's function, Solution E belongs to Pareto front. Similar to the first test of this experiment, both functions are resulting in a classification that is closer to participants' judgment in comparison to Pareto's function (see Table 31).

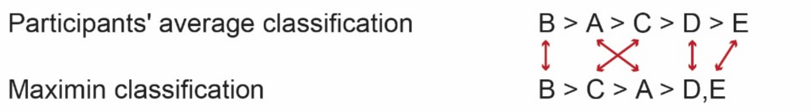


Figure 116: *A comparison between the classification based on participants' average and Maximin function (Experiment 5, test 2)*

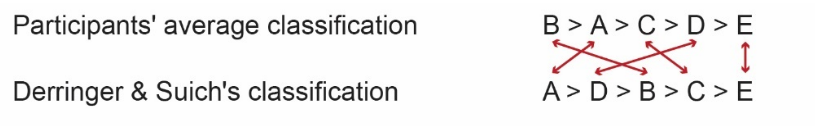


***Figure 117:** A comparison between the classification based on participants' average and Derringer & Suich's function (Experiment 5, test 2)*

As in the previous test, to compute the deviation, we must assign a number to represent each solution (A=1, B=2, C=3, D=4, E=5). Then we used the same method presented in Experiment 4 (see 6.2.3.1). Figure 118 demonstrates the deviations of both functions. In the figure, we can see that the individuals' responses are slightly closer to Derringer & Suich's function (equal weights) than Maximin. This is expected and justified because many objectives are involved, and Derringer & Suich's is compensatory while Maximin is not. The absence of the scatterplot in this test may also affect the designer judgment; the location of the solution is not presented visually, and the participants relied more on the numbers, which might lead them to compensate. According to Experiment 3 (see 6.1.3.2), the presence of the scatterplot that represents can lead the designers to adopt a classification logic that is similar to Maximin; this was tested based on the situations of two objectives.

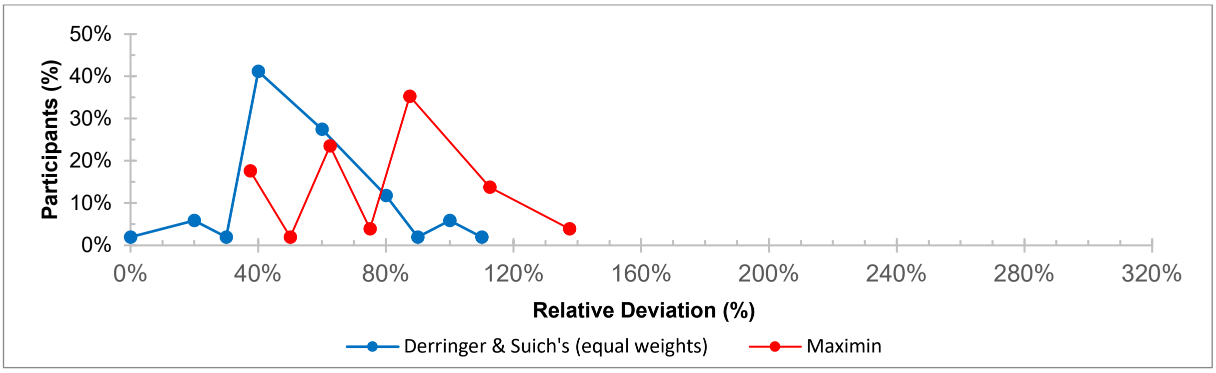


***Figure 118:** Experiment 5, test 2 the relative deviation between different aggregation functions and designers' classification*

6.4 Information availability and *acceptimality*

To achieve *acceptimality* in MOIA the selecting the aggregation function is a crucial decision. Generally, the higher the negentropy of the function, the more likely it helps us approach *acceptimality*. Here we encounter two problems. From one side, the low in negentropy aggregation function may result in unacceptable solutions or low informative data. From the other side, the high in negentropy aggregation functions need additional information. From the experiments presented in this chapter and based on MOIA, three different information scenarios can decide which aggregation function can be used.

Scenario one, if the essential information for the interpretation based on desirability functions is available (see 2.2), and the weights of the objectives are decidable and consistent, then Derringer & Suich's aggregation function is appropriate. Unfortunately, the chance of inconsistency in evaluating the weights is high among the designers. However, a methodology such as the pairwise comparison based on AHP can be used to provide the designer with feedback. It can detect if the weights are consistent or inconsistent.

Scenario two, if the essential information for the interpretation based on desirability functions is available (see 2.2), but the weights of the objectives are undecidable or inconsistent, then Maximin or Derringer & Suich's (equal weights) aggregation functions can be used. If avoiding extreme interpretation variables is a priority, then the non-compensatory approach of Maximin is appropriate. If the designer prefers to compensate between the objectives, then Derringer & Suich's (equal weights) is appropriate. Increasing objectives can increase the designers' desire to compensate.

Scenario three, if the essential information for the interpretation based on desirability functions is not available (see 2.2). Then Pareto's function is the left option. Scenario one and two allow MOIA to achieve acceptability by adopting high in negentropy functions. If scenario three is the situation, then classical multi-objective optimization is the only choice, and MOIA cannot achieve *acceptimality*. In this case, the designers have to work on the solutions manually after the aggregation is done.

CHAPTER 7 Conclusion

"It is comparatively easy to make computers exhibit adult level performance on intelligence tests or playing checkers, and difficult or impossible to give them the skills of a one-year-old when it comes to perception and mobility." (Moravec, 1988)

7.1 Recommendations

The dissertation adopted five different experiments to define a set of recommendations. The recommendations aim to help the designers and the developers to develop a new generation of decision support systems for designers in general and architects in particular. These systems should increase the acceptability of the process and the solutions, and thus of the decision support systems. Consequently, it can attract more designers to adopt the profitable behavior of optimization. Furthermore, the new systems allow the designer to approach *acceptimality.* These experiments are divided into two categories. The first two experiments investigated different design optimization workflows based on different digital tools. The other three experiments investigated different aggregation functions. Based on MOIA and from the results of these experiments, a set of four recommendations are proposed.

7.1.1 Generative visual programming

To generate design solutions, we usually use generative design tools based on parametric modeling. The parts of these models are connected by a set of constraints. Algorithmic modeling can be described as advanced parametric modeling. It offers high control of the model's parts and the constraints. To define an algorithm, we use programming. However, the classical textual programming is not accessible for designers in general and architects in specific as it requires extensive training. Visual Programming (VP) is a relatively new programming approach, and it is accessible to designers as it is relatively simple. The first two experiments of the dissertation show that VP is highly accepted among designers. It is highly recommended for future developments to adopt a VP approach.

7.1.2 Interactive algorithm

From the first two experiments, it was clear that using an interactive algorithm such as IEA can increase the users' acceptability of the tools. By using these algorithms, the designer can express his preference in the selection stage during the Morphogenesis model of MOIA. These algorithms allow the designers to judge the solutions based on their subjective opinions. As a result, the systems that use these algorithms can increase the acceptability of the solutions. However, this might negatively affect the optimality of the solution. The designer can decide when to use this option and when not to use it. Usually, if no selection are made by the designer, the interactive algorithm performs as a non-interactive algorithm. It is recommended that future developments of design tools and workflows use an interactive algorithm.

7.1.3 Visual interface

The interface of the tools can profoundly affect designers' acceptability of the tools. From the results of Experiments 1, 2, and 3, we can infer that the graphical representation of the solutions and their performance can strongly influence designers' judgment. First, it is highly recommended to use graphical representations that represent the performance of all the solutions simultaneously. The number of objectives can define possible types of graphical representation. For two objective problems, a two-dimensional scatterplot or a parallel coordinate can be used (see Figure 119). For three objective problems, a three-dimensional scatterplot or a parallel coordinate can be used (see Figure 119). If four or more objectives are involved, a parallel coordinate can be used (see Figure 119). However, the two-dimensional and the three-dimensional scatterplots can represent more than two or three objectives by using scales of colors, opacity, sizes to the solutions to represent the additional objectives. Second, it is highly recommended to show the 3D masses of the candidate solutions. This allows the designers to explore the solutions, which is essential for the selection stage within the Morphogenesis model of MOIA, especially if an interactive algorithm is used.

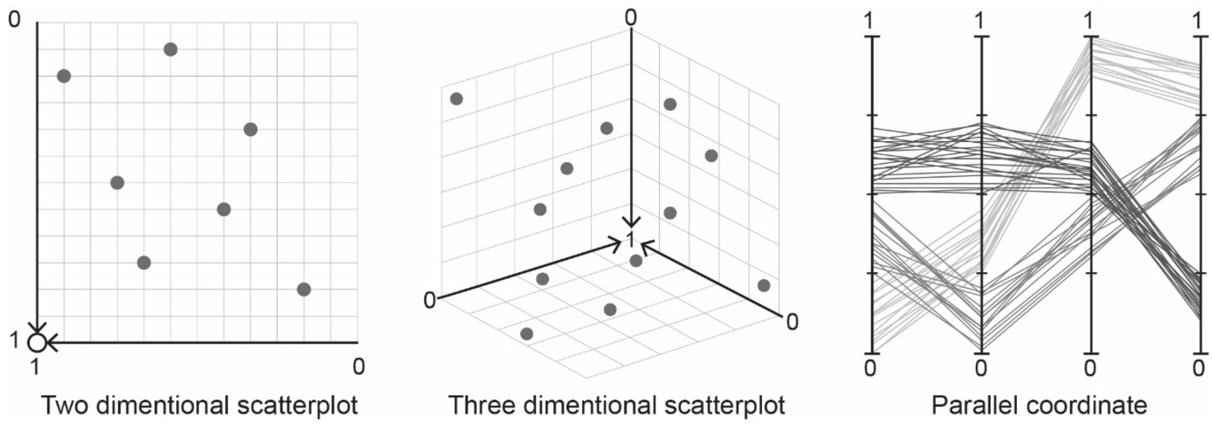


Figure 119: *Different graphical representation of the information*

7.1.4 Adapting to different information

Most of the contemporary design optimization digital systems use Pareto's function to trade off the objectives. Pareto's function is low in negentropy, and it uses the rank of the observed variables to classify the solutions. The ordinal information that Pareto's function uses do not allow us to apply mathematical operations to compute a global objective. When additional information such as cardinal values is available, Pareto's function cannot benefit from the surplus of information. Based on the results of the last three experiments, other functions that are regarded as high in negentropy cardinal can be used to benefit from the additional information. It is highly recommended to allow the designers to select between different functions to adapt to the availability of the information. The algorithm presented in Figure 120 demonstrates how the systems must adapt to information available based on MOIA.

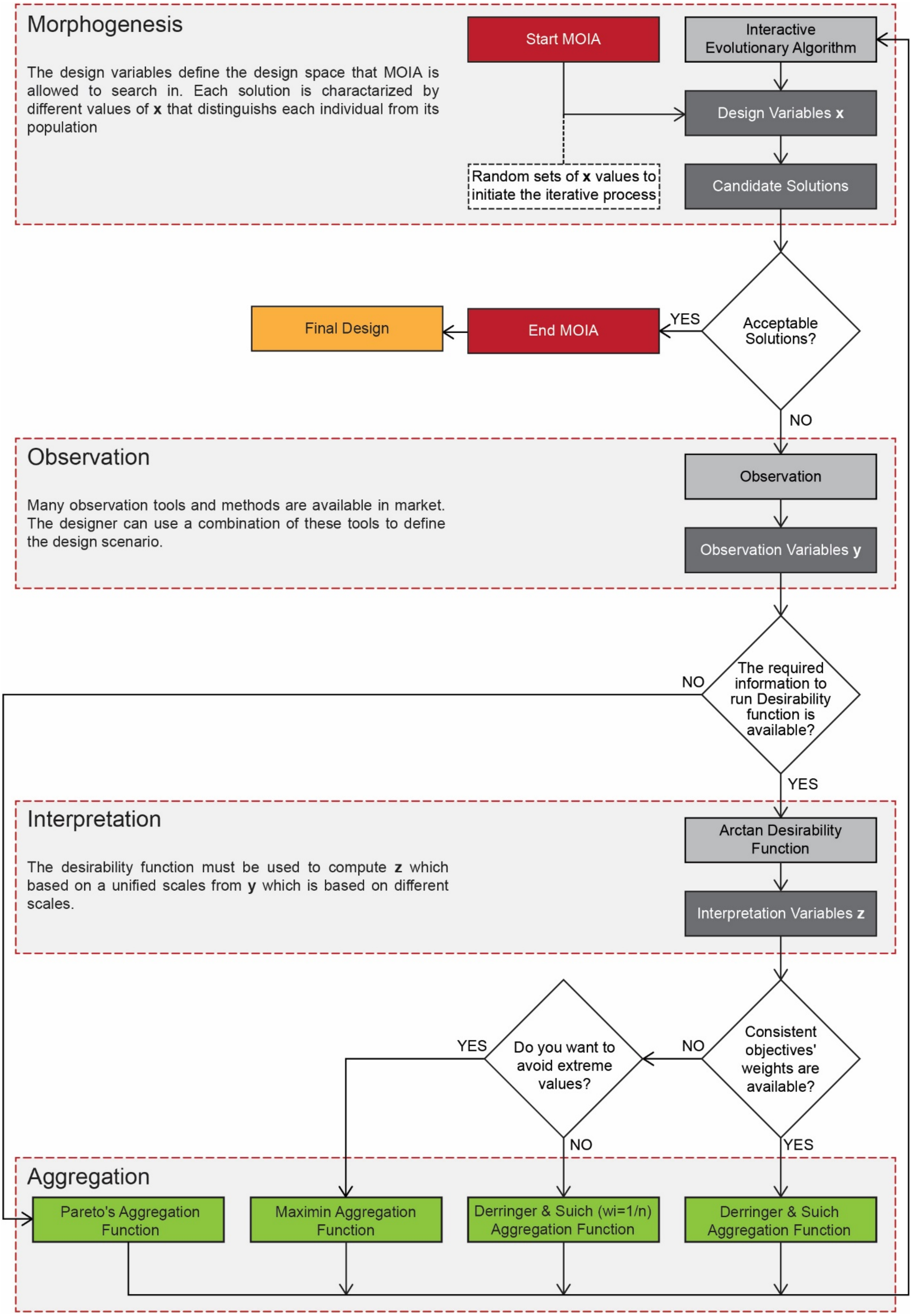


Figure 120: *A recommended algorithm based on MOIA that respond to information availability*

7.2 Conclusion

We spend most of our lives in our buildings. The building sector consumes more than half of the electricity that we produce and about one-third of the total energy. The performance of our buildings is significant. Design optimization allows the designer to find solutions with the best possible performance. However, a limited number of architects use the existing design optimization tools.

For a long time, design problems were recognized as ill-defined problems. They lack the precise and objective knowledge in their definitions. Also, they are considered vague and partly formalized. Furthermore, they involve subjectivity and objectivity. The interpretation and creativity of designers are required to solve such problems. To specify the required elements for the definition of the design process, experienced humans must use concepts such as goals, objective, criterion, importance, etc. these concepts are features of intentionality.

In design, the solutions should be optimal from a numerical point of view and acceptable by the designers. Both acceptability and optimality are essential, and they lead to *acceptimality*. However, the tools in the market are not well prepared to successfully integrate designers' acceptability and performance optimality. They do not consider designers' preferences of the criteria or the objectives inside the optimization process. Consequently, the solutions resulting from these tools are not necessarily acceptable by the designers.

The unbalanced collaboration between the designers and the tools can be the central cause of architects' reluctance to use these tools. The dissertation proposed a set of recommendations that can help the developers to create a new generation of design optimization tools that potentially can attract more designers. The recommendations aim to enhance the collaboration between the tools and the designers.

The research began by exploring many design processes. From this exploration, a design framework based on four models Morphogenesis, Observation, Interpretation, and Aggregation (MOIA), is proposed. MOIA allows integrating designers' preferences within the process of optimization. In MOIA, the designers' can express their preferences of the criteria and the objectives within the Interpretation model and the Aggregation model. By using an interactive algorithm, MOIA can also allow the designers to express his preference within the selection stage of the Morphogenesis model; some of the existing tools already offer this option.

The main common characteristic among the generative multi-objective design optimization tools is that they use Pareto's function to classify the solution. This function does not consider human preferences. Other functions that consider human preferences must be used to approach *acceptimality*. Two aggregation functions that can consider human preferences and can alternate Pareto's function are proposed. These aggregation functions are Maximin and Derringer & Suich's function. These functions are cardinal and high in negentropy. These functions mainly combine the design objectives into a GDI, which regarded as a global objective.

Maximin aggregate the solutions based on compromisation logic. It adopts a precautionary principle that evades extreme and unsafe solutions. Derringer & Suich's aggregation entails assigning weights to designs' objectives. In contrast to Pareto's function, these functions use information resulting from a desirability interpretation function, which considers human preferences within the Interpretation model of MOIA. Additionally, Derringer & Suich's aggregation function considers human preferences within the Aggregation models of MOIA by allowing the designers to assign weights.

It is crucial to investigate the classification resulting from the different aggregation functions by using designers' classification as a benchmark. However, it was vital to investigate the generative design optimization workflows, which are used by the architects first. The workflows and the tools involved in them can influence designers' acceptability. Both the acceptability of the workflows and the solutions are vital and relative.

The dissertation adopted an experimental approach to evaluate the different workflows and different aggregation functions. For that, five different experiments are performed throughout this research. The first two experiments compare different generative design optimization workflows by using designers' judgment as a reference. The other three experiments compared different aggregation functions by using designers' judgment as a benchmark.

By comparing the different workflows, it was clear that the VP is suitable for future tools. VP is highly accepted among the designers. Furthermore, it was evident that using an interactive optimization algorithm can increase the acceptability of the tools as it allows the designers' to express their subjective judgment in the selection stage of the Morphogenesis model of MOIA. Allowing the designers to explore the massing of the solutions visually can increase the overall acceptability of the tools. It is essential for the selection stage when an interactive optimization algorithm is used.

By comparing the different aggregation functions, it was clear that the high in negentropy functions that allow the designers to express their preferences can increase the acceptability of the solutions. If the required information is available, using the high in negentropy aggregation functions such as Maximin or Derringer & Suich's are more likely to be used as it results in a classification that is closer to designers' classification.

Based on the experiments, four different recommendations are proposed. The recommendations aim to enhance the collaboration between the designers and the tools. These recommendations suggest using VP as a base for the new tools. Additionally, it recommends using an interactive algorithm that allows the designers to express their subjective judgment of the forms during the selection stage of the Morphogenesis model. Also, it recommends using graphical representations that represent the performance of the candidate solution simultaneously. This can be extremely important as it insight designers' decisions during the selection. Finally, the used aggregation function must respond to information availability. Following these recommendations can increase the chance of attracting more designers to adopt the profitable behavior of optimization. It allows the designers to approach acceptimality.

7.3 Future work

At present, a set of generative tools and workflows based on the presented recommendations are under development. A series of experiments that use designers' judgment as a benchmark must be applied to observe the acceptability of the solutions resulting from these tools. Moreover, the performance and the acceptability of the solutions resulting from these tools must be compared to other solutions resulting from the classical generative optimization tools and the solutions resulting when no generative tools are used. The experiments should also observe designers' acceptability of the developed workflow process and tools interface. The proposed experiments are crucial to develop reliable systems.

The proposed systems are based on approaching *acceptimality* by using MOIA. They are recognized as Intelligence Augmentation IA systems. Adopting these systems increases the interaction between humans and the machine in the field of design. In the future, we can link these systems to machine learning. *"Adoption of machine learning-powered systems can more rapidly increase if they are designed in a user-oriented way where the goal is to empower the human users rather than to replace them."* (Kane, 2019). Integrating machine learning in the proposed systems can improve the interaction between the designers and the computer. Consequently, the computer augments the capabilities of the designers to approach optimality, and the designers help the computer to improve its capabilities in approaching acceptability. *"By using an intelligence augmentation (IA) approach, where the technology seeks to empower and complement people's innate skills, a huge amount of value can be brought both to people's personal lives and to the enterprise, with the machine learning capabilities that exist today."* (Kane, 2019). In the future, these systems can propose different solutions for a problem based on a period of interaction with a specific designer or a particular school of thought. It is highly recommended to start another research that focuses on integrating machine learning into the proposed system. The research can consist of a team that includes architects, engineers, and programmers.

References

Afandi, A., Barlet, A., Sebastian, P., Bruneau, D., & Marsault, X. (2018). Outils d'optimisation de la conception multi-objectifs utilisant la programmation visuelle : tests utilisateurs d'acceptabilité. In *Conférence Francophone de l'International Building Performance Simulation Association* (pp. 231–237). Bordeaux: IBPSA France.

Agarwal, P., & Mehta, S. (2014). Nature-Inspired Algorithms: State-of-Art, Problems and Prospects. *International Journal of Computer Applications*, *100*(14), 14–21. https://doi.org/10.5120/17593-8331

Agkathidis, A. (2015). Generative Design Methods Implementing Computational Techniques in Undergraduate Architectural Education. In *ECAADE 2015: REAL TIME - EXTENDING THE REACH OF COMPUTATION, VOL 2*. https://doi.org/10.13140/RG.2.1.3767.1120

Alexander, C. (1964). *Notes on the Synthesis of Form* (7th ed.). Cambridge: Harvard University Press.

Autodesk Inc. (2010a). *Subdivision Surface Modeling Copyright Notice Autodesk ® Maya ® 2011 Software*. Retrieved from http://download.autodesk.com/us/support/files/subds.pdf

Autodesk Inc. (2010b). *Subdivision Surface Modeling Copyright Notice Autodesk ® Maya ® 2011 Software*. Retrieved from http://download.autodesk.com/us/support/files/subds.pdf

Autodesk Inc. (2018). Common Types of Geometry in 3D Modelling. Retrieved August 5, 2018, from https://www.autodesk.in/solutions/3d-modeling-software

BaityKool team. (2018). BaityKool Project. Retrieved May 25, 2019, from https://baitykool.com/project.html

Bellman, R. (1961). *Adaptive Control Processes*. Princeton: Princeton University Press.

Bianchi, L., Dorigo, M., Gambardella, L. M., & Gutjahr, W. J. (2009). A survey on metaheuristics for stochastic combinatorial optimization. *Natural Computing*, *8*(2), 239–287. https://doi.org/10.1007/s11047-008-9098-4

Birogul, S. (2016). EpiGenetic Algorithm for Optimization: Application to Mobile Network Frequency Planning. *Arabian Journal for Science and Engineering*, *41*(3), 883–896. https://doi.org/10.1007/s13369-015-1869-5

Cheng, M. Y., & Prayogo, D. (2014). Symbiotic Organisms Search: A new metaheuristic optimization algorithm. *Computers and Structures*, *139*, 98–112. https://doi.org/10.1016/j.compstruc.2014.03.007

Ching, F. D. K. (1995). *A Visual Dictionary of Architecture* (First Edit). New York: John Wiley & Sons, Inc.

Chuck Eastman, Teicholz Paul, Sacks Rafael, L. K. (2011). *BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and*

Contractors (Second Edi). Hoboken: John Wiley & Sons, Inc.

Churchill, W. (1943). We shape our buildings. *Famous Quotations and Stories*.

Collignan, A. (2011). *Méthode d'optimisation et d'aide à la décision en conception mécanique : Application à une structure aéronautique.* Université Sciences et Technologies - Bordeaux I. Retrieved from http://www.theses.fr/2011BOR14395

Dantzig, G. (n.d.). TOP 8 QUOTES BY GEORGE DANTZIG | A-Z Quotes. Retrieved March 19, 2019, from https://www.azquotes.com/author/29405-George_Dantzig

Davis, F. D., & Venkatesh, V. (2004). Toward preprototype user acceptance testing of new information systems: Implications for software project management. *IEEE Transactions on Engineering Management*. https://doi.org/10.1109/TEM.2003.822468

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*. https://doi.org/10.1109/4235.996017

Derringer, G., & Suich, R. (1980). Simultaneous Optimization of Several Response Variables. *Journal of Quality Technology*. https://doi.org/10.1080/00224065.1980.11980968

Dubberly, H. (2004). *How do you design?* San Francisco: Dubberly Design Office. https://doi.org/10.1016/j.bbamcr.2006.09.022

Edgeworth, F. Y. (1881). Mathematical psychics - An essay on the application of mathematices to the moral sciences. *C. Kegan Paul & Co., London*, *6*, 581–583. https://doi.org/10.2307/2223848

Einstein, A. (n.d.). Albert Einstein Qoutes - BrainyQuote. Retrieved March 27, 2019, from https://www.brainyquote.com/authors/albert_einstein

Eisenman, P. (n.d.). Peter Eisenman Quotes - BrainyQuote. Retrieved March 27, 2019, from https://www.brainyquote.com/authors/peter_eisenman

Engelbart, D. C. (1962). *Augmenting Human Intellect: A Conceptual Framework. Summary Report AFOSR-3223 under Contract AF 49 (638)-1024, SRI Project 3578 for Air Force Office of Scientific Research* (Vol. 1). Washington. Retrieved from http://csis.pace.edu/~marchese/CS835/Lec3/DougEnglebart.pdf

Foster and Partners. (2002). City Hall | Architecture Projects | Foster + Partners. Retrieved February 15, 2019, from https://www.fosterandpartners.com/projects/city-hall/

Garber, R. (2017). *Workflows : Expanding Architecture's Territory in the Design and Delivery of Buildings.* John Wiley & Sons, Incorporated. Retrieved from https://www.wiley.com/en-fr/Workflows:+Expanding+Architecture's+Territory+in+the+Design+and+Delivery+of+Buildings-p-9781119317845

Garcia, S. (2017). Classifications of Shape Grammars. In *Design Computing and Cognition '16* (pp. 229–248). Cham: Springer International Publishing. https://doi.org/10.1007/978-

3-319-44989-0_13

Gero, J. (1990). Design Prototypes: A Knowledge Representation Schema for Design. *AI Magazine*. https://doi.org/10.1609/aimag.v11i4.854

Gero, J. S., & Kannengiesser, U. (2004). The situated function-behaviour-structure framework. *Design Studies*. https://doi.org/10.1016/j.destud.2003.10.010

Gero, J. S., & Kannengiesser, U. (2014). The Function-Behaviour-Structure Ontology of Design. In A. Chakrabarti & L. T. M. Blessing (Eds.), *An Anthology of Theories and Models of Design* (pp. 263–283). London: Springer. https://doi.org/10.1007/978-1-4471-6338-1_13

Gershenfeld, N. (n.d.). Quotefancy. Retrieved March 18, 2019, from https://quotefancy.com/quote/1714488/Neil-Gershenfeld-Give-ordinary-people-the-right-tools-and-they-will-design-and-build-the

Giagkiozis, I., & Fleming, P. J. (2015). Methods for multi-objective optimization: An analysis. *Information Sciences*. https://doi.org/10.1016/j.ins.2014.08.071

Goldberg, D. E. (David E. (1989). *Genetic algorithms in search, optimization, and machine learning* (First edition). Boston: Addison-Wesley Pub. Co.

Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*. https://doi.org/10.1006/knac.1993.1008

Harding, J. (n.d.-a). Biomorpher - Grasshopper. Retrieved May 2, 2019, from https://www.grasshopper3d.com/group/biomorpher

Harding, J. (n.d.-b). Optimization using Biomorpher - Grasshopper. Retrieved June 1, 2019, from https://www.grasshopper3d.com/group/biomorpher/forum/topics/optimization-using-biomorpher

Harrington, E. C. (1965). The Desirability Function. *Industrial Quality Control*, 21(10):494-498.

Hassan, R., Cohanim, B., Weck, O. De, Venter, G., Associate, P., Systems, E., … Systems, E. (2004). A Comparison of Particle Swarm. *American Institute of Aeronautics and Astronautics*.

Hawking, S. (2001). Human Intelligence Quotes. Retrieved March 19, 2019, from https://www.brainyquote.com/quotes/stephen_hawking_447572?src=t_human_intelligence

Howarth, D. (2017). Generative design software will give designers "superpowers" Retrieved February 25, 2019, from https://www.dezeen.com/2017/02/06/generative-design-software-will-give-designers-superpowers-autodesk-university/

IEA. (2017). Energy Technology Perspectives 2017 - Executive Summary. *International Energy Agency (IEA) Publications*. https://doi.org/10.1787/energy_tech-2014-en

Inselberg, A. (2009). *Parallel coordinates: Visual multidimensional geometry and its applications*. *Parallel Coordinates: Visual Multidimensional Geometry and Its Applications*. New York: Springer Science+Business Media, LLC. https://doi.org/10.1007/978-0-387-68628-8

Jabi, W. (2013). *Parametric Design for Architecture*. London: Laurence King Publishing Ltd.

Kane, J. (2019). Intelligence Augmentation and Machine Learning | Cogito. Retrieved June 7, 2020, from https://www.cogitocorp.com/intelligence-augmentation-machine-learning-for-the-real-world/

Kasparov, G., & Greengard, M. (2017). *Deep Thinking: Where Machine Intelligence Ends and Human Creativity Begins*. New York: Public Affairs.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks* (Vol. 4, pp. 1942–1948). Perth, WA, Australia: IEEE. https://doi.org/10.1109/ICNN.1995.488968

Kensek, K. M., & Noble, D. E. (2014). *Building information modeling: BIM in Current and Future Practice*. John Wiley & Sons, Inc.

Kim, K. J., & Lin, D. K. J. (2006). Optimization of multiple responses considering both location and dispersion effects. *European Journal of Operational Research*. https://doi.org/10.1016/j.ejor.2004.06.020

Koberg, Don, Bagnall, J. (1972). *The Universal Traveler: a soft systems guide to creativity,problem-solving and the process of reaching goals*. W. Kaufmann.

Ladybug Tools Team. (2020). Ladybug Tools | Ladybug. Retrieved March 3, 2020, from https://www.ladybug.tools/ladybug.html

Lawson, B. (2005). *How designers think: The design process demystified* (Fourth Edi). New York: Routledge. https://doi.org/10.1016/0142-694X(81)90033-8

Lee, D. H., Jeong, I. J., & Kim, K. J. (2018). A desirability function method for optimizing mean and variability of multiple responses using a posterior preference articulation approach. *Quality and Reliability Engineering International*. https://doi.org/10.1002/qre.2258

Lewis, C., & Olson, G. (1987). Can principles of cognition lower the barriers to programming? In G. Olson, S. Sheppard, & E. Soloway (Eds.), *Empirical Studies of Programmers: Second Workshop* (pp. 248–263). Norwood: Ablex Publishing Corp.

Lexico. (2020a). Framework | Definition of Framework by Lexico. Retrieved January 28, 2020, from https://www.lexico.com/en/definition/framework

Lexico. (2020b). Ontology | Definition of Ontology by Lexico. Retrieved January 28, 2020, from https://www.lexico.com/en/definition/ontology

Lexico Dictionary. (2020). Value Judgment | Definition of Value Judgment by Lexico. Retrieved September 26, 2019, from https://www.lexico.com/en/definition/value_judgment

Li, M., Zhen, L., & Yao, X. (2017). How to Read Many-Objective Solution Sets in Parallel Coordinates [Educational Forum]. *IEEE Computational Intelligence Magazine*, *12*(4), 88–100. https://doi.org/10.1109/MCI.2017.2742869

Lobato, F. S., & Steffen, V. (2017). *Multi-Objective Optimization Problems Concepts and Self-Adaptive Parameters with Mathematical and Engineering Applications*. Springer. Retrieved from http://www.worldcat.org/isbn/9783319585659

Makstutis, G. (2018). *Design Process in Architecture: From Concept to Completion*. London: Laurence King Publishing.

Markoff, J. (2015). *Machines of Loving Grace: the quest for common ground between humans and robots*. 2015: HarperCollins Publishers.

Marsault, X. (2018). *Eco-generative design for early stages of architecture*. *Eco-generative Design for Early Stages of Architecture*. ISTE Lyd and John Wiley & Sons, Inc. https://doi.org/10.1002/9781119422044

McCarthy, J., Minsky, M., Rochester, N., Shannon, C. E. (1955). A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence. Dartmouth.

Merriam-Webster. (2019a). Acceptability | Definition of Acceptability. Retrieved July 10, 2019, from https://www.merriam-webster.com/dictionary/acceptability

Merriam-Webster. (2019b). Desirability | Definition of Desirability. Retrieved July 4, 2019, from https://www.merriam-webster.com/dictionary/desirability

Merriam-Webster. (2019c). Workflow | Definition of Workflow by Merriam-Webster. Retrieved April 30, 2019, from https://www.merriam-webster.com/dictionary/workflow?utm_campaign=sd&utm_medium=serp&utm_source=jsonld

Merriam-Webster. (2020). Optimization | Definition of Optimization by Merriam-Webster. Retrieved June 1, 2020, from https://www.merriam-webster.com/dictionary/optimization

Moravec, H. (1988). *Mind Children: The Future of Robot and Human Intelligence*. Cambridge: Harvard University Press.

Myers, B. A. (1990). Taxonomies of visual programming and program visualization. *Journal of Visual Languages and Computing*. https://doi.org/10.1016/S1045-926X(05)80036-9

Nagy, D. (2017). Learning from nature. Retrieved July 19, 2019, from https://medium.com/generative-design/learning-from-nature-fe5b7290e3de

Narke, A. (n.d.). Building Information Modeling (BIM) Market - Global Industry Forecast & Growth Analysis Report till 2022 | Market Research | The BIM Hub. Retrieved February 25, 2019, from https://thebimhub.com/2016/08/25/building-information-modeling-bim-market-global/#.XHQCbehKiUk

National Institute of Building Sciences. (n.d.). Frequently Asked Questions About the National BIM Standard-United States^{TM}. Retrieved May 2, 2019, from https://www.nationalbimstandard.org/faqs

Niemeyer, O. (n.d.). Oscar Niemeyer Quotes - BrainyQuote. Retrieved March 2, 2019, from https://www.brainyquote.com/authors/oscar_niemeyer

Nilsson, N. J. (2010). *The Quest for Artificial Intelligence: A History of Ideas and Achievements*. Cambridge University Press. https://doi.org/10.1017/CBO9780511819346

Osman, I. H., & Laporte, G. (1996). Metaheuristics: A bibliography. *Annals of Operations Research*, *63*(5), 511–623. https://doi.org/10.1007/BF02125421

Österlund, T. (2010). *Methods for morphogenesis and ecology in architecture*. University of Oulu. Retrieved from http://jultika.oulu.fi/Record/isbn978-951-42-6257-9

Papalambros, P., & Douglass, W. (2017). *Principles of Optimal Design: Modeling and Computation* (Third edit). Cambridge, UK: Cambridge University Press. Retrieved from http://principlesofoptimaldesign.org/

Pareto, V. (1896). *Cours d'Économie Politique: vols. I and II. The ANNALS of the American Academy of Political and Social Science*. Lausanne: F. Rouge. https://doi.org/10.1107/S0567740882007778

Parker, M. (n.d.). Mark Parker Quotes - BrainyQuote. Retrieved March 2, 2019, from https://www.brainyquote.com/quotes/mark_parker_891788

Petroski, H. (n.d.). Henry Petroski Quotes. Retrieved May 15, 2019, from https://www.brainyquote.com/quotes/henry_petroski_694526

Piotrowski, C. (2011). *Problem Solving and Critical Thinking for Designers* (1st ed.). Hoboken: John Wiley & Sons.

Pressman, A. (2012). *Designing Architecture: The elements of process*. *Designing Architecture: The Elements of Process*. https://doi.org/10.4324/9780203122174

Quirante, T. (2012). *Modelling and Numerical Optimization Methods for Decision Support in Robust Embodiment Design*. University of Bordeaux.

Quirante, T., Sebastian, P., & Ledoux, Y. (2013). A trade-off function to tackle robust design problems in engineering. *Journal of Engineering Design*, *24*(1), 64–81. https://doi.org/10.1080/09544828.2012.691160

Rahmani ASL, M. (2015). *A Bbuilding Information Model (BIM) Based Framework for Pperformance Optimization*. Texas A&M University. https://doi.org/10.1360/zd-2013-43-6-1064

Rutten, D. (2011). Evolutionary Principles applied to Problem Solving | I Eat Bugs For Breakfast. Retrieved February 25, 2019, from https://ieatbugsforbreakfast.wordpress.com/2011/03/04/epatps01/

S.Dushkes, L. (2012). *The Architect Says: qoutes, quips, and words of wisdom*. New York: Princeton Architectural Press.

Saaty, T. L. (2013). *Fundamentals of Decision Making and Priority Theory with the Analytic Hierarchy Process Saaty, Thomas L.. Fundamentals of Decision Making and Priority Theory With the Analytic Hierarchy Process* (ebook Seco). Pittsburgh, PA Saaty, Thomas L.. Fundamentals of Decision Making and Priority Theory With the Analytic Hierarchy Process . RWS Publications. Édition du Kindle.

Saaty, T. L., & Vargas, L. (2000). Fundamentals of Decision Making and Priority Theory With the Analytic Hierarchy Process. *RWS Publications*.

Scott, M. J., & Antonsson, E. K. (1998). Aggregation functions for engineering design trade-offs. *Fuzzy Sets and Systems*, *99*(3), 253–264. https://doi.org/10.1016/S0165-0114(97)00032-8

Scott, M. J., & Antonsson, E. K. (1999). Arrow's Theorem and engineering design decision making. *Research in Engineering Design - Theory, Applications, and Concurrent Engineering*. https://doi.org/10.1007/s001630050016

Shannon, C. E. (1948). A Mathematical Theory of Communication. *Bell System Technical Journal*. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x

Simon, H. A. (1956). Rational choice and the structure of the environment. *Psychological Review*. https://doi.org/10.1037/h0042769

Simon, Herbert A. (1973). The structure of ill structured problems. *Artificial Intelligence*, *4*(3–4), 181–201. https://doi.org/10.1016/0004-3702(73)90011-8

Smith, Albert C, Smith, K. S. (2015). *Developing your Design Process: six key concepts for studio* (First). New York: Routledge.

Solar Decathlon Middle East. (n.d.). Retrieved May 30, 2019, from https://www.solardecathlonme.com/

Sörensen, K., & Glover, F. W. (2013). Metaheuristics. In S. I. Gass & M. C. Fu (Eds.), *Encyclopedia of Operations Research and Management Science* (pp. 960–970). Boston, MA: Springer US. https://doi.org/10.1007/978-1-4419-1153-7_1167

Sörensen, K., Sevaux, M., & Glover, F. (2018). A history of metaheuristics. *Handbook of Heuristics*, *2–2*, 791–808. https://doi.org/10.1007/978-3-319-07124-4_4

Tedeschi, A., Andreani, S., Buono, A., Degni, M., Friesen, L., Galli, A., … Wirz, F. (2014). *AAD; Algorithms-Aided Design* (First Edit). Brienza: Le Penseur Publisher.

Tomasetti, T. (2017). Design Explorer 2. Retrieved February 25, 2019, from https://tt-acm.github.io/DesignExplorer/

Upgreengrade Team. (2016). Galapagos Parametric Optimizations _ Part01 - YouTube. Retrieved January 12, 2018, from https://www.youtube.com/watch?v=68rmW3vUh5k

Vier, R. (2019). Octopus - Grasshopper. Retrieved February 15, 2019, from https://www.grasshopper3d.com/group/octopus

Vierlinger, R. (2013). Multi Objective Design Interface, *2013*(April), 1–83. https://doi.org/10.13140/RG.2.1.3401.0324

Villaggi, L., & Walmsley, K. (2018). Generative Urban Design: A Collaboration Between Autodesk Research and Van Wijnen | Autodesk University. Retrieved February 15, 2019, from https://www.autodesk.com/autodesk-university/class/Generative-Urban-Design-Collaboration-Between-Autodesk-Research-and-Van-Wijnen-2018

Waern, R., & Windgardh, G. (2015). *What is architecture? and 100 other questions*. London: Laurence King Publishing Ltd. Retrieved from https://www.laurenceking.com/product/what-is-architecture/

Walmsley, K. (n.d.). Project Refinery: implement generative design workflows for AEC - Through the Interface. Retrieved May 8, 2019, from https://www.keanw.com/2018/11/project-refinery-implement-generative-design-workflows-for-aec.html

Woodbury, R., Yüce Gün, O., Peters, B., & Sheikholeslami, M. (Roham). (2010). *Elements of Parametric Design*. Oxon: Routledge.

Zablit, P., & Zimmer, L. (2001). Global aircraft predesign based on constraint propagation and interval analysis. In *Proceedings of CEAS Conference on multidisciplinary Aircraft design and Optimization* (pp. 77–86). Köln, Germany: CEAS.

Appendicies

Appendix I: Experiment 1

A. Workflows comparison *(*required)*:

Question 1*: Which workflow do you think can help you to find design solutions?
- Grasshopper®
- EcoGen©

Question 2: Why? ---

Question 3*: Which workflow is more suitable for your design process?
- Grasshopper®
- EcoGen©

Question 4: Why? ---

Question 5*: In which workflow do you feel more involved in the design?
- Grasshopper®
- EcoGen©

Question 6: Why? ---

Question 7*: Which workflow stimulates your creativity more?
- Grasshopper®
- EcoGen©

Question 8: Why? ---

Question 9*: Which workflow do you prefer to filter the results?
- Grasshopper®
- EcoGen©

Question 10: Why? ---

Question 11*: Which workflow do you believe helps produce better results?
- Grasshopper®
- EcoGen©

Question 12: Why? ---

Question 13*: Which workflow is easier to understand?
- Grasshopper®
- EcoGen©

Question 14: Why? ---

Question 15*: Which user interface do you prefer?
- Grasshopper®
- EcoGen©

Question 16: Why? ---

Question 17*: Which workflow do you prefer?
- Grasshopper®
- EcoGen©

Question 18: Why? ---

Question 19*: What are the cons and pros of each tool? ----------------------------------

B. General feedback about the tools *(*required)*:

Question 20*: If you are a student, do you consider using these tools in your schoolwork?
- Yes
- No
- Maybe
- I am not a student

Question 21*: Do you consider using these tools in your professional work?
- Yes
- No
- Maybe

Question 22*: Do you think that visual programming is suitable for architects?
- Yes
- No
- Maybe

Question 23*: If you are a student, do you consider using visual programming in your future schoolwork?
- Yes
- No
- Maybe
- I am not a student

Question 24*: Do you consider using visual programming in your future professional work?
- Yes
- No
- Maybe

Question 25*: Do you wish to learn more about visual programming?
- Yes
- No
- Maybe

Question 26*: Do you think that EcoGen© is suitable for architects?
- Yes
- No
- Maybe

Question 27*: If you are a student, do you consider using EcoGen© in your future schoolwork?
- Yes
- No
- Maybe
- I am not a student

Question 28*: Do you consider using EcoGen© in your future professional work?
- Yes
- No
- Maybe

Question 29*: Do you wish to learn more about EcoGen©?
- Yes
- No
- Maybe

Question 30*: Do you prefer if EcoGen© became a plugin for Grasshopper®?

- Yes
- No
- Maybe

Question 31: Do you have any comments? --

C. Background and experience *(*required)*:

Question 32*: Do you have any experience in programming? *(e.g., C#, Python™, VB, etc.), (0 = I have no experience, 10= I am an expert)*

0 2 3 4 5 6 7 8 9 10

Question 33*: Do you have any experience in visual programming? *(e.g., Grasshopper, Dynamo, Generative Components, etc.), (0 = I have no experience, 10= I am an expert)*

0 2 3 4 5 6 7 8 9 10

Question 34*: Do you use any of the tools on the list? *(check all that apply)*

- ☐ Grasshopper®
- ☐ Design Explorer©
- ☐ Dynamo®
- ☐ Project Fractal®
- ☐ Generative Components®
- ☐ EcoGen©
- ☐ I do not use any of these tools

Question 35*: Do you use multi-objective design optimization in your work?

- ○ Never
- ○ Rarely
- ○ Sometimes
- ○ Usually
- ○ Always

Question 36*: What are the tools that you use for multi-objective design optimization? *(check all that apply)*

- ☐ I don't do any multi-objective design optimization
- ☐ Octoput©
- ☐ Biomorpher©
- ☐ Optimo©
- ☐ EcoGen©
- ☐ Other ----------------------------

D. Personal information *(*required)*:

Question 36: Your email address? --

Question 37: Your name? --

Question 38: Your family name? --

Question 39*: Your Age? (--)

Question 40*: You are?

- ☐ Student
- ☐ Professional
- ☐ Other ---------------------------

Question 41*: What is your major?

- ☐ Architect
- ☐ Engineer
- ☐ Other ------------------------

Appendix II: Experiment 2

A. Workflows comparison *(*required)*:

Question 1*: Which tool do you think can help you to find design solutions in different situations?
- Octopus©
- Biomorpher©

Question 2: Why? ---

Question 3*: Which tool is more suitable for your design process?
- Octopus©
- Biomorpher©

Question 4: Why? ---

Question 5*: In which tool do you feel more involved in the design?
- Octopus©
- Biomorpher©

Question 6: Why? ---

Question 7*: Which tool stimulates your creativity more?
- Octopus©
- Biomorpher©

Question 8: Why? ---

Question 9*: Which tool do you prefer to filter the results?
- Octopus©
- Biomorpher©

Question 10: Why? ---

Question 11*: Which tool do you believe helps produce better results?
- Octopus©
- Biomorpher©

Question 12: Why? ---

Question 13*: Which tool is easier to understand?
- Octopus©
- Biomorpher©

Question 14: Why? ---

Question 15*: Which tool is more interesting to you?
- Octopus©
- Biomorpher©

Question 16: Why? ---

Question 17*: Which interface do you prefer?
- Octopus©
- Biomorpher©

Question 18: Why? ---

Question 19*: Which tool do you prefer?

Question 20: Why? ---

Question 21*: What are the cons and pros of each tool? ---

B. General feedback about the tools *(*required)*:

Question 20*: If you are a student, do you consider using these tools in your schoolwork?
- Yes
- No
- Maybe
- I am not a student

Question 21*: Do you consider using these tools in your professional work?
- Yes
- No
- Maybe

Question 22*: Do you think that visual programming is suitable for architects?
- Yes
- No
- Maybe

Question 23*: If you are a student, do you consider using visual programming in your future schoolwork?
- Yes
- No
- Maybe
- I am not a student

Question 24*: Do you consider using visual programming in your future professional work?
- Yes
- No
- Maybe

Question 25*: Do you wish to learn more about visual programming?
- Yes
- No
- Maybe

Question 26: Do you have any comments? ---

C. Background and experience *(*required)*:

Question 27*: Do you have any experience in programming? *(e.g., C#, Python™, VB, etc.), (0 = I have no experience, 10= I am an expert)*

0 2 3 4 5 6 7 8 9 10

Question 28*: Do you have any experience in visual programming? *(e.g., Grasshopper®, Dynamo®, Generative Components®, etc.), (0 = I have no experience, 10= I am an expert)*

0 2 3 4 5 6 7 8 9 10

Question 29*: Do you use any of the tools on the list? *(check all that apply)*

- ☐ Grasshopper®
- ☐ Design Explorer©
- ☐ Dynamo®
- ☐ Project Fractal®
- ☐ Generative Components®
- ☐ I do not use any of these tools

Question 30*: Do you use multi-objective design optimization in your work?

- ○ Never
- ○ Rarely
- ○ Sometimes
- ○ Usually
- ○ Always

Question 31*: What are the tools that you use for multi-objective design optimization? *(check all that apply)*

- ☐ I don't do any multi-objective design optimization
- ☐ Octoput©
- ☐ Biomorpher©
- ☐ Optimo©
- ☐ Other ---------------------------

D. Personal information *(*required)*:

Question 32: Your email address? --

Question 33: Your name? --

Question 34: Your family name? --

Question 35*: Your Age? (--)

Question 36*: You are?

- ☐ Student
- ☐ Professional
- ☐ Other ----------------------------

Question 37*: What is your major?

- ☐ Architect
- ☐ Engineer
- ☐ Other -------------------------

Appendix III: Experiment 3

A. Test of Derringer & Suich's aggregation function *(*required)*

You are in the early stage of designing four office buildings, each in different place.

Location 1: Extremely hot climate (Dubai, United Arab Emirates).

Month	Average High	Average Low	Sunshine hours
January	23.6°C	14.2°C	254.2 h
February	25.4°C	15.3°C	229.6 h
March	28.2°C	17.5°C	254.2 h
April	33°C	21°C	294.0 h
May	37.6°C	24.9°C	344.1 h
June	38.7°C	27.2°C	342.0 h
July	40.8°C	30.2°C	322.4 h
August	41°C	30.2°C	316.2 h
September	39°C	27.7°C	309.0 h
October	35.6°C	24.3°C	303.8 h
November	30.3°C	20°C	285.0 h
December	26.2°C	16.2°C	254.2 h

Climate data (Dubai, United Arab Emirates)

Location 2: Moderate climate (San Diego, United States).

Month	Average High	Average Low	Sunshine hours
January	19°C	10°C	217.0 h
February	19°C	11°C	224.0 h
March	19°C	12°C	248.0 h
April	20°C	13°C	240.8 h
May	21°C	15°C	248.0 h
June	22°C	17°C	240.8 h
July	24°C	19°C	310.0 h
August	25°C	20°C	279.0 h
September	25°C	19°C	270.0 h
October	23°C	16°C	248.0 h
November	21°C	12°C	240.0 h
December	19°C	10°C	248.0 h

Climate data (San Diego, United States)

Location 3: Extremely cold climate (Yakutsk, Russia).

Month	Average High	Average Low	Sunshine hours
January	-35.1°C	-41.5°C	18.6 h
February	-28.6°C	-38.2°C	98.0 h
March	-12.3°C	-27.4°C	232.5 h
April	1.7°C	-11.8°C	273.0 h
May	13.2°C	1°C	303.8 h
June	22.4°C	9.3°C	333.0 h
July	25.5°C	12.7°C	347.2 h
August	21.5°C	8.9°C	272.8 h
September	11.5°C	1.2°C	174.0 h
October	-3.6°C	-12.2°C	105.7 h
November	-23.1°C	-31°C	60.0 h
December	-34.3°C	-40.4°C	9.3 h

Climate data (Yakutsk, Russia)

Location 4: Mixed climate "hot summer, cold winter" (Shanghai, China).

Month	Average High	Average Low	Sunshine hours
January	8.1°C	2.1°C	114.3 h
February	10.1°C	3.7°C	119.9 h
March	13.8°C	6.9°C	128.5 h
April	19.5°C	11.9°C	148.5 h
May	24.8°C	17.3°C	169.8 h
June	27.8°C	21.7°C	130.9 h
July	32.2°C	25.8°C	190.8 h
August	31.5°C	25.8°C	185.7 h
September	27.9°C	22.4°C	167.5 h
October	22.9°C	16.8°C	161.4 h
November	17.3°C	10.6°C	131.1 h
December	11.1°C	4.7°C	127.4 h

Climate data (Shanghai, China)

You need to optimize five different objectives:

- Objective 1: Form compacity "Maximize": Optimize this objective should increase form compacity.
- Objective 2: Direct sunlight "Maximize": Optimize this objective should increase direct sunlight that reaches the building, "If this objective weight = 0 it means direct sunlight has no importance, for example; a building with no potential of receiving direct sunlight is acceptable"
- Objective 3: Natural lighting "Maximize": Optimize this objective should increase natural lighting (indirect sunlight) that reach the building, "If this objective weight = 0 it means natural lighting has no importance, for example; a building with no natural lighting at all is acceptable the result can be a building with no windows"
- Objective 4: Heating load "Minimize": Optimize this objective should decrease the building heating energy consumption

- Objective 5: Cooling load "Minimize": Optimize this objective should decrease the building heating energy consumption

Question 1*: Read the notes below then complete the table:

- For each location indicate the weight of each objective.
- The more weight an objective has the more important it is (0%=Not important, 100%=The only important).
- For each location, the sum of objectives weights should equal 100%.

Objective	Location 1	Location 2	Location 3	Location 4
Objective 1 (maximize)				
Objective 2 (maximize)				
Objective 3 (maximize)				
Objective 4 (minimize)				
Objective 5 (minimize)				
Total	100%	100%	100%	100%

Question 2*: For the case of the office building in Dubai, compare the objectives two by two in terms of importance based on the following scales (0 means that the two objectives are of equal importance).

Objective 1	4	3	2	1	0	1	2	3	4	Objective 2
Objective 1	4	3	2	1	0	1	2	3	4	Objective 3
Objective 1	4	3	2	1	0	1	2	3	4	Objective 4
Objective 1	4	3	2	1	0	1	2	3	4	Objective 5
Objective 2	4	3	2	1	0	1	2	3	4	Objective 3
Objective 2	4	3	2	1	0	1	2	3	4	Objective 4
Objective 2	4	3	2	1	0	1	2	3	4	Objective 5
Objective 3	4	3	2	1	0	1	2	3	4	Objective 4
Objective 3	4	3	2	1	0	1	2	3	4	Objective 5
Objective 4	4	3	2	1	0	1	2	3	4	Objective 5

B. Test of Maximin aggregation function *(*required)*

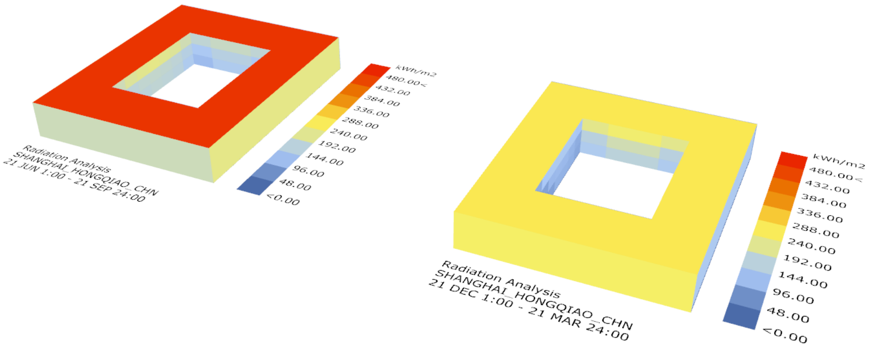
You are in the early stage of designing a mixed-use building (office, residential, commercial) in a Mixed climate “hot summer, cold winter” (Shanghai, China).

Month	Average High	Average Low	Sunshine hours
January	8.1°C	2.1°C	114.3 h
February	10.1°C	3.7°C	119.9 h
March	13.8°C	6.9°C	128.5 h
April	19.5°C	11.9°C	148.5 h
May	24.8°C	17.3°C	169.8 h
June	27.8°C	21.7°C	130.9 h
July	32.2°C	25.8°C	190.8 h
August	31.5°C	25.8°C	185.7 h
September	27.9°C	22.4°C	167.5 h
October	22.9°C	16.8°C	161.4 h
November	17.3°C	10.6°C	131.1 h
December	11.1°C	4.7°C	127.4 h

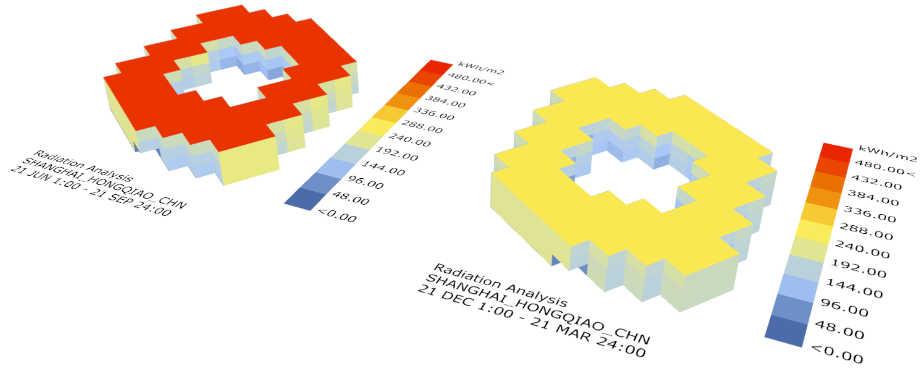
Climate data (Shanghai, China)

- The building is modular and the size of one module is (width=6m, Length=8m, Hight=4m).
- The total area of the program is 12000m^2.
- The lot area is (width 80m X length 100m) 8000 m^2.
- The building has only 3 stories.
- A (25m X30m) courtyard is required in the middle of the building.

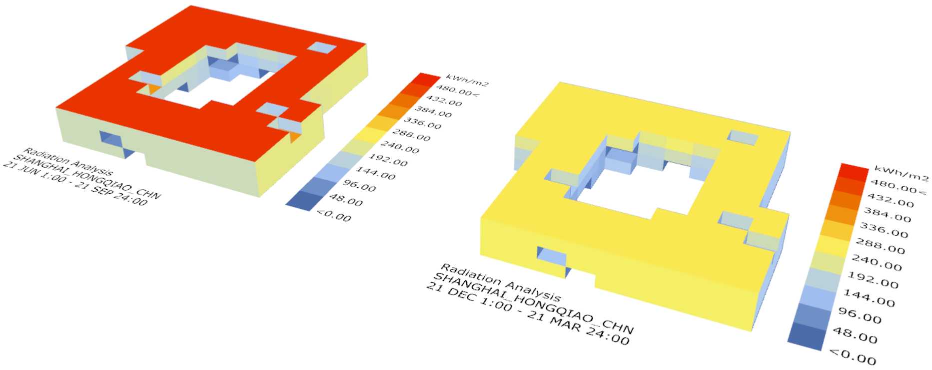
The following solutions represents ten candidates of the discribed building:



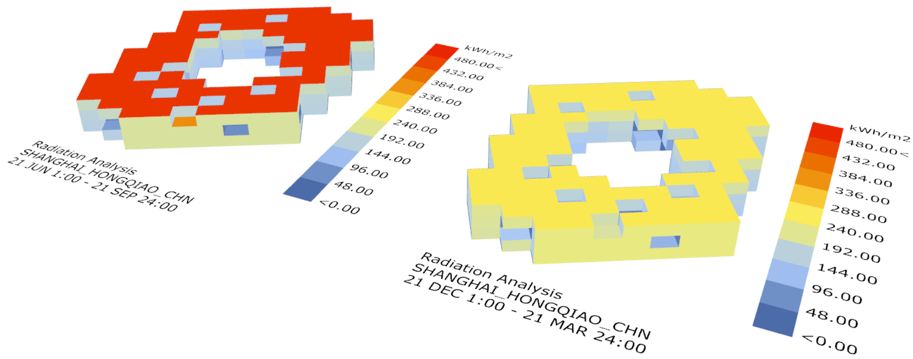
Solution 1



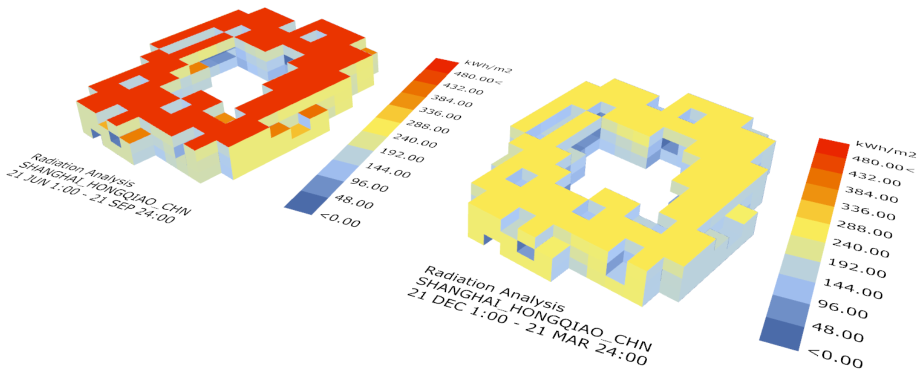
Solution 2



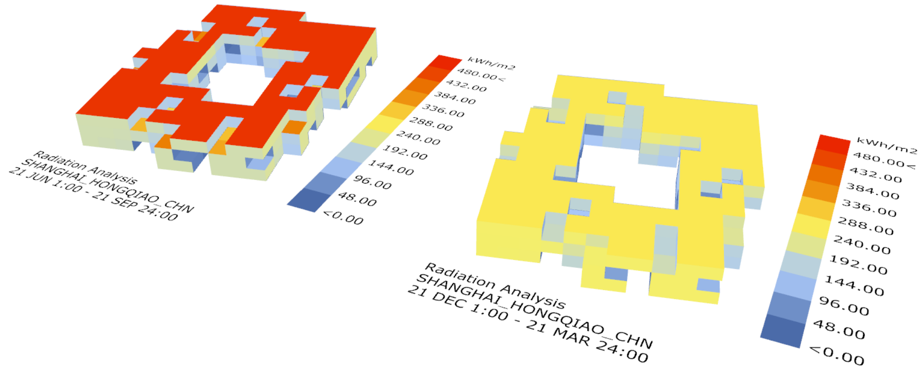
Solution 3



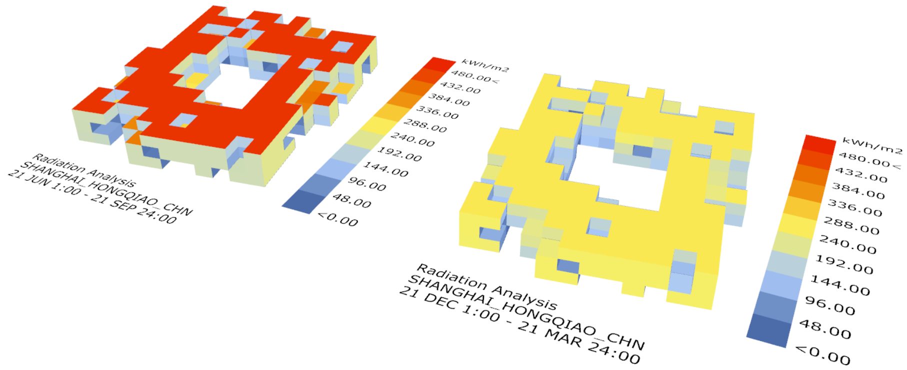
Solution 4



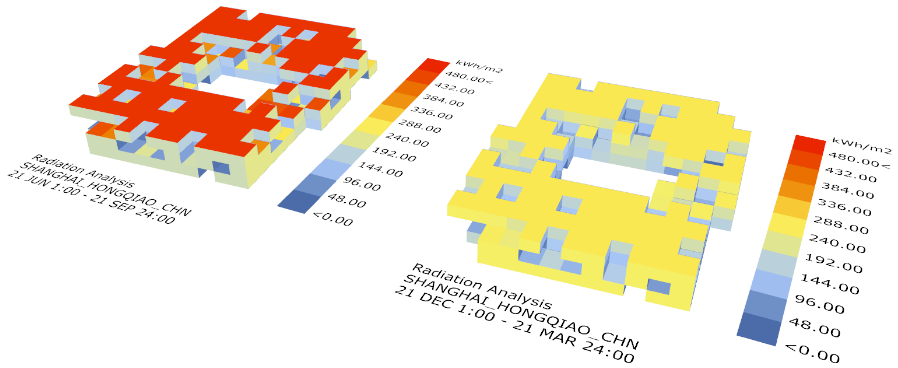
Solution 5



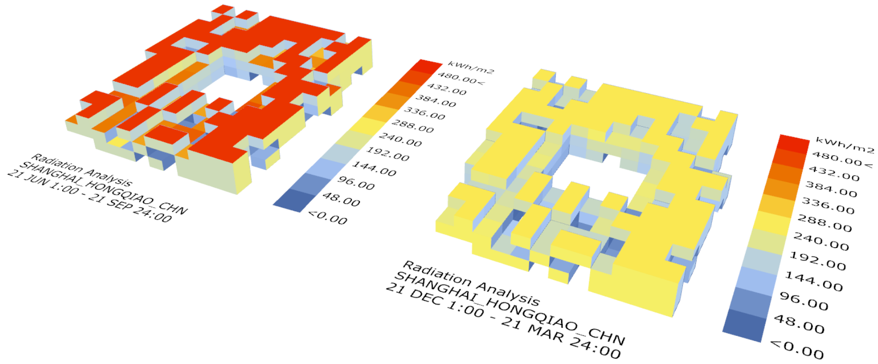
Solution 6



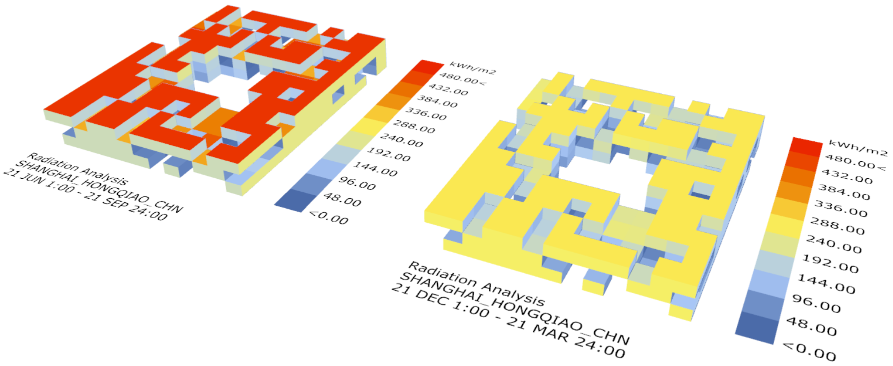
Solution 7



Solution 8



Solution 9



Solution 10

Question 3*: From the list of the solutions please select only one solution that you prefer more than the others, the objective is to maximize solar gain in winter and minimize solar gain in summer:

Please select the number of the solution: 1-2-3-4-5-6-7-8-9-10
Please explain why you did choose this solution?

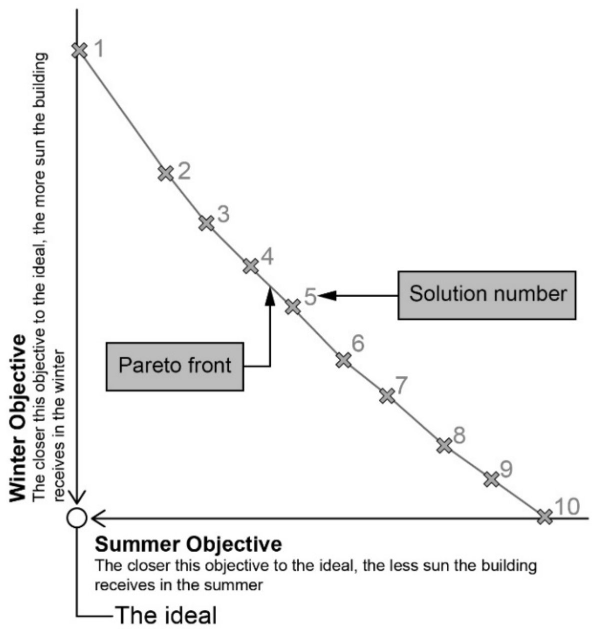
……………………………………………………………………………………………………

Question 4*: Please answer the previous question again, this time an extra piece of information will be given to you:

Please select a solution number: 1-2-3-4-5-6-7-8-9-10
Please explain why you did choose this form?

……………………………………………………………………………………………………

The extra piece of information is the follwing scatterplot, it present Pareto front of the solutions. All of the solutions are non-dominated (see the figure below).



C. Personal information *(*required)*:

Question 5*: Do you use multi-objective design optimization in your work?

- Never
- Rarely
- Sometimes
- Usually
- Always

Question 6*: Your Age? (--)

Question 7*: You are?

- Student (Architecture)
- Student (Engineering)
- Professor (Architecture)
- Professor (Engineering)
- Professional (Architecture)
- Professional (Engineering)
- Other -------------------------

Appendix IV: Experiment 4

This Experiment consist of different tests each test is performed with different group of participants

1. Scatterplot 1 with a low level of information:

A. Personal information *(*required)*:

Question 1*: Is your work related to buildings design?
- Yes
- No

Question 2*: You are?
- Student (Architecture)
- Student (Engineering)
- Professor (Architecture)
- Professor (Engineering)
- Professional (Architecture)
- Professional (Engineering)
- Other ------------------------

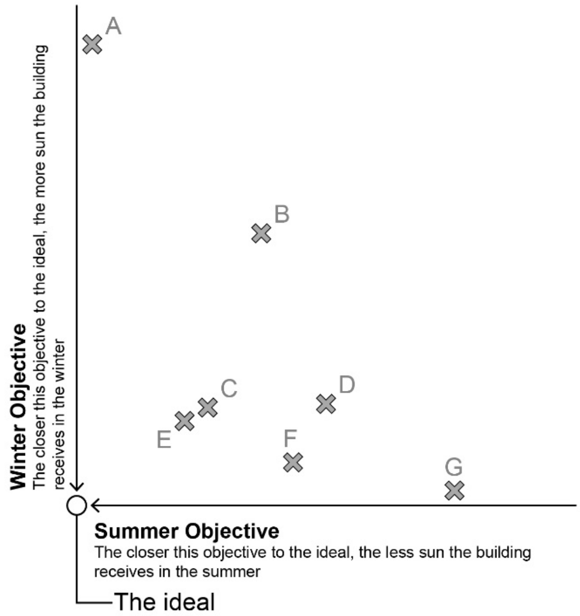
Question 3*: Do you use multi-objective design optimization in your work?
- Never
- Rarely
- Sometimes
- Usually
- Always

B. Solutions ranking *(*required)*:

Please read the information then answer the question:

You are at the early stage of designing a mixed-use building (office, residential, commercial) in a mixed climate (hot summer, cold winter-Shanghai, China).

Here is a solar gain graph that represents the performance of the candidate solutions in summer and in winter



A scatterplot that represents the solutions

Question 4*: We want to maximize the solar gain in winter and to minimize the solar gain in summer. Please rank the solutions according to your preference, "1st Place" represents the best while "7th Place" represents the worst.

Please, select only one circle for each solution

	Solution A	Solution B	Solution C	Solution D	Solution E	Solution F	Solution G
1st place	O	O	O	O	O	O	O
2nd place	O	O	O	O	O	O	O
3rd place	O	O	O	O	O	O	O
4th place	O	O	O	O	O	O	O
5th place	O	O	O	O	O	O	O
6th place	O	O	O	O	O	O	O
7th place	O	O	O	O	O	O	O

2. Scatterplot 1 with a medium level of information:

A. Personal information *(*required)*:

Question 1*: Is your work related to buildings design?

- Yes
- No

Question 2*: You are?

- Student (Architecture)
- Student (Engineering)
- Professor (Architecture)
- Professor (Engineering)
- Professional (Architecture)
- Professional (Engineering)
- Other ------------------------

Question 3*: Do you use multi-objective design optimization in your work?

- Never
- Rarely
- Sometimes
- Usually
- Always

B. Solutions ranking *(*required)*:

Please read the information then answer the question:

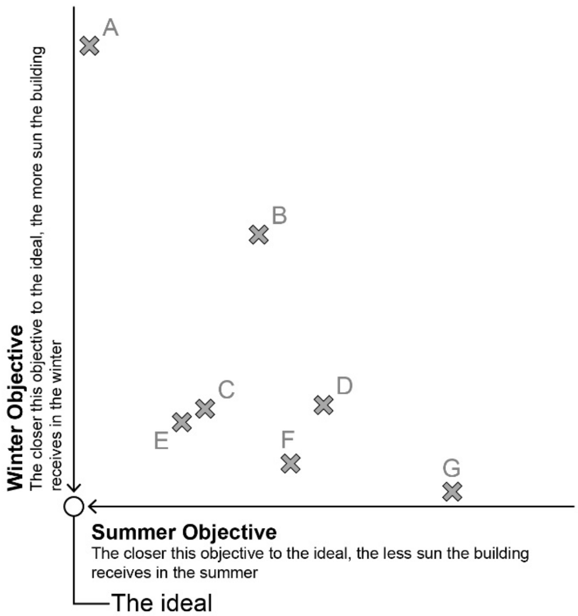
You are at the early stage of designing a mixed-use building (office, residential, commercial) in a mixed climate (hot summer, cold winter-Shanghai, China).

- The building is modular, the module size is: (Width=6m, Length=8m, Hight=4m).
- The required program is 12000 m^2.
- The land lot size is 8000 m^2 (W=80m, L=100m).
- The maximum number of levels allowed is 3 stories.
- A (25m X 30m) courtyard is required in the center of the building.

You have the climate data of shanghai and a solar gain performance graph (scatterplot) that represents seven candidate design solutions.

Month	Average High	Average Low	Sunshine hours
January	8.1°C	2.1°C	114.3 h
February	10.1°C	3.7°C	119.9 h
March	13.8°C	6.9°C	128.5 h
April	19.5°C	11.9°C	148.5 h
May	24.8°C	17.3°C	169.8 h
June	27.8°C	21.7°C	130.9 h
July	32.2°C	25.8°C	190.8 h
August	31.5°C	25.8°C	185.7 h
September	27.9°C	22.4°C	167.5 h
October	22.9°C	16.8°C	161.4 h
November	17.3°C	10.6°C	131.1 h
December	11.1°C	4.7°C	127.4 h

Here is a solar gain graph that represents the performance of the solutions in summer and in winter



A scatterplot that represents the solutions

Question 4*: We want to maximize the solar gain in winter and to minimize the solar gain in summer. Please rank the solutions according to your preference, "1st Place" represents the best while "7th Place" represents the worst.

Please, select only one circle for each solution

	Solution A	Solution B	Solution C	Solution D	Solution E	Solution F	Solution G
1st place	○	○	○	○	○	○	○
2nd place	○	○	○	○	○	○	○
3rd place	○	○	○	○	○	○	○
4th place	○	○	○	○	○	○	○
5th place	○	○	○	○	○	○	○
6th place	○	○	○	○	○	○	○
7th place	○	○	○	○	○	○	○

3. Scatterplot 1 with a high level of information:

A. Personal information *(*required)*:

Question 1*: Is your work related to buildings design?

- Yes
- No

Question 2*: You are?

- Student (Architecture)
- Student (Engineering)
- Professor (Architecture)
- Professor (Engineering)
- Professional (Architecture)
- Professional (Engineering)
- Other ------------------------

Question 3*: Do you use multi-objective design optimization in your work?

C. Never
D. Rarely
E. Sometimes
F. Usually
G. Always

B. Solutions ranking*(*required)*:

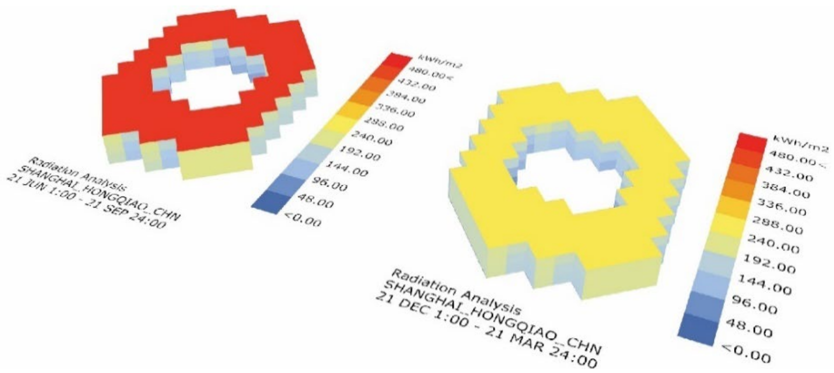
Please read the information then answer the question:

You are at the early stage of designing a mixed-use building (office, residential, commercial) in a mixed climate (hot summer, cold winter-Shanghai, China).

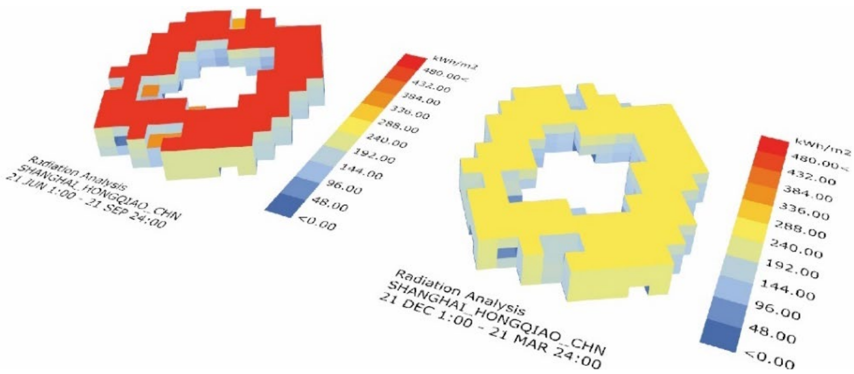
- The building is modular, the module size is: (Width=6m, Length=8m, Hight=4m).
- The required program is 12000 m^2.
- The land lot size is 8000 m^2 (W=80m, L=100m).
- The maximum number of levels allowed is 3 stories.
- A (25m X 30m) courtyard is required in the center of the building.

You have the climate data of shanghai and a solar gain performance graph (scatterplot) that represents seven candidate design solutions.

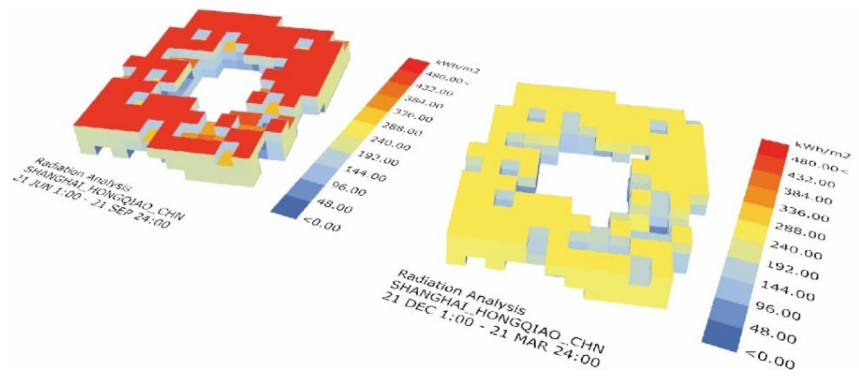
Month	Average High	Average Low	Sunshine hours
January	8.1°C	2.1°C	114.3 h
February	10.1°C	3.7°C	119.9 h
March	13.8°C	6.9°C	128.5 h
April	19.5°C	11.9°C	148.5 h
May	24.8°C	17.3°C	169.8 h
June	27.8°C	21.7°C	130.9 h
July	32.2°C	25.8°C	190.8 h
August	31.5°C	25.8°C	185.7 h
September	27.9°C	22.4°C	167.5 h
October	22.9°C	16.8°C	161.4 h
November	17.3°C	10.6°C	131.1 h
December	11.1°C	4.7°C	127.4 h



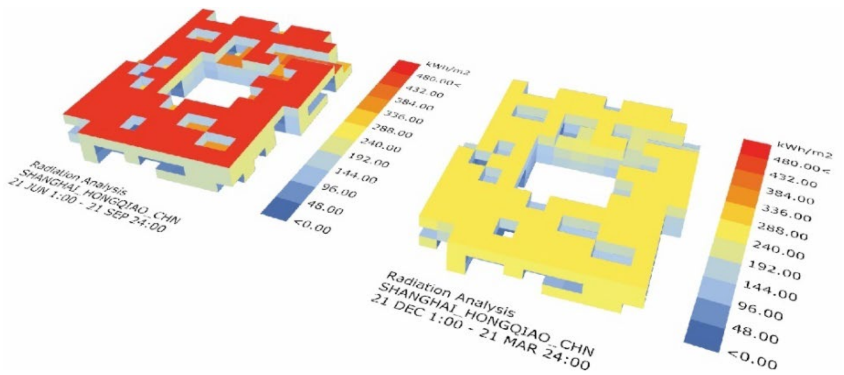
Solution A



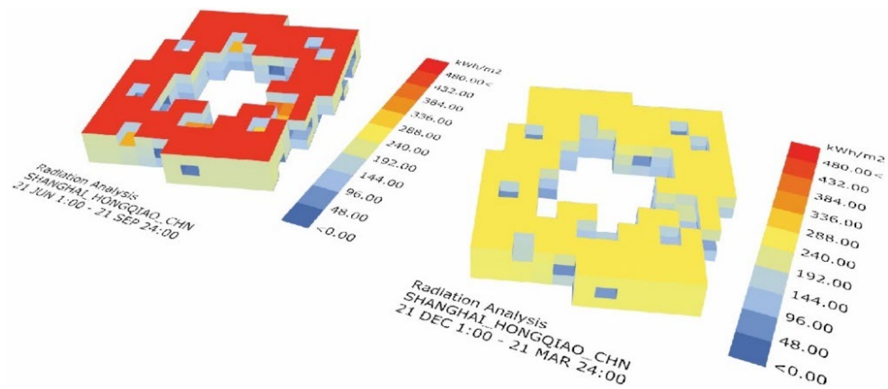
Solution B



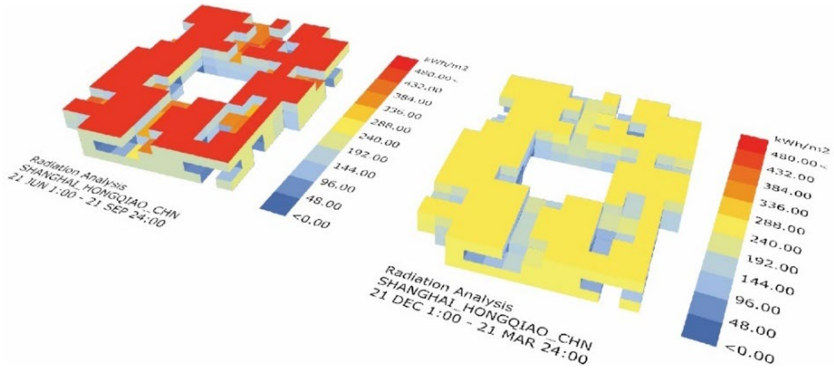
Solution C



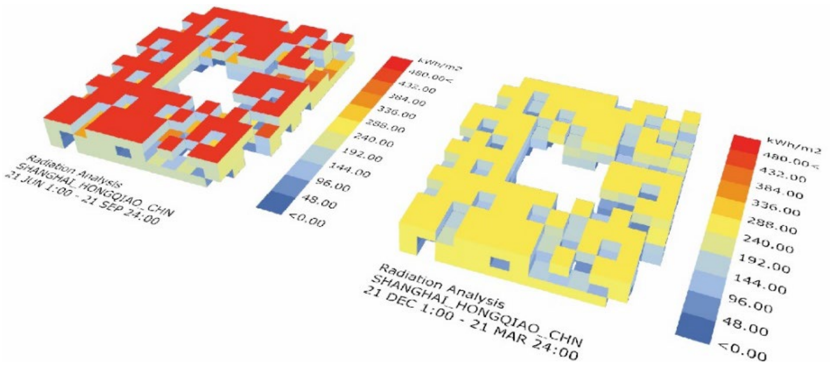
Solution D



Solution E

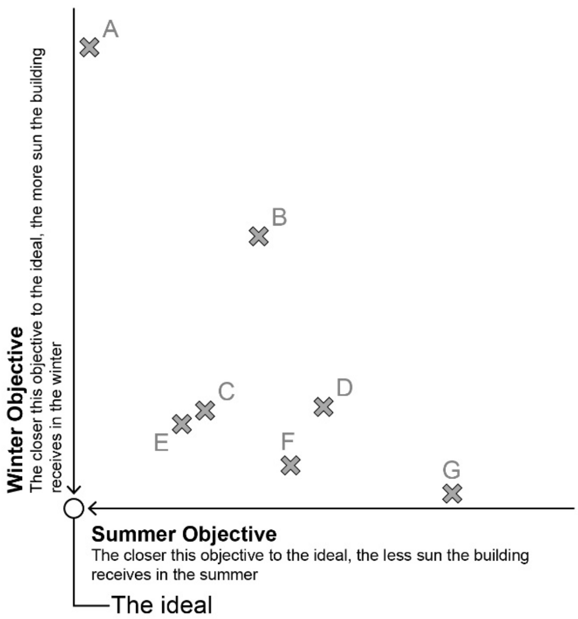


Solution F



Solution G

Here is a solar gain graph that represents the performance of the solutions in summer and in winter



A scatterplot that represents the solutions

Question 4*: We want to maximize the solar gain in winter and to minimize the solar gain in summer. Please rank the solutions according to your preference, "1st Place" represents the best while "7th Place" represents the worst.

Please, select only one circle for each solution

	Solution A	Solution B	Solution C	Solution D	Solution E	Solution F	Solution G
1st place	O	O	O	O	O	O	O
2nd place	O	O	O	O	O	O	O
3rd place	O	O	O	O	O	O	O
4th place	O	O	O	O	O	O	O
5th place	O	O	O	O	O	O	O
6th place	O	O	O	O	O	O	O
7th place	O	O	O	O	O	O	O

4. Scatterplot 2 with a low medium of information:

A. Personal information *(*required)*:

Question 1*: Is your work related to buildings design?

- Yes
- No

Question 2*: You are?

- Student (Architecture)
- Student (Engineering)
- Professor (Architecture)
- Professor (Engineering)
- Professional (Architecture)
- Professional (Engineering)
- Other -------------------------

Question 3*: Do you use multi-objective design optimization in your work?

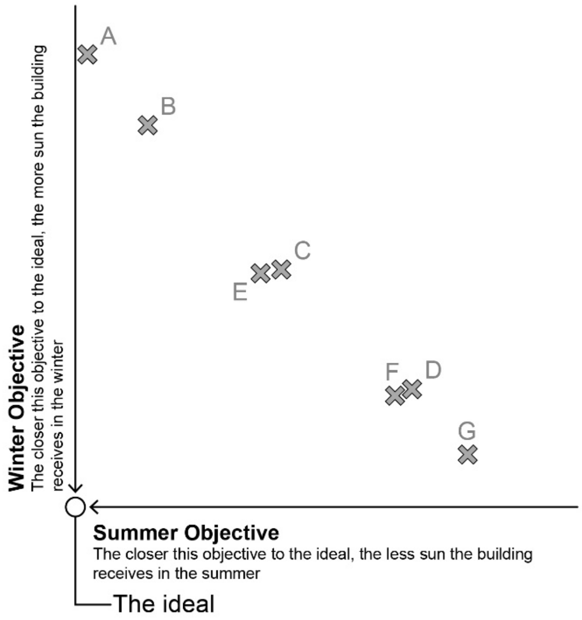
C. Never
D. Rarely
E. Sometimes
F. Usually
G. Always

B. Solutions ranking*(*required)*:

Please read the information then answer the question:

You are at the early stage of designing a mixed-use building (office, residential, commercial) in a mixed climate (hot summer, cold winter-Shanghai, China).

Here is a solar gain graph that represents the performance of the candidate solutions in summer and in winter



A scatterplot that represents the solutions

Question 4*: We want to maximize the solar gain in winter and to minimize the solar gain in summer. Please rank the solutions according to your preference, "1st Place" represents the best while "7th Place" represents the worst.

Please, select only one circle for each solution

	Solution A	Solution B	Solution C	Solution D	Solution E	Solution F	Solution G
1st place	○	○	○	○	○	○	○
2nd place	○	○	○	○	○	○	○
3rd place	○	○	○	○	○	○	○
4th place	○	○	○	○	○	○	○
5th place	○	○	○	○	○	○	○
6th place	○	○	○	○	○	○	○
7th place	○	○	○	○	○	○	○

5. Scatterplot 2 a with a high level of information:

A. Personal information *(*required)*:

Question 1*: Is your work related to buildings design?

- Yes
- No

Question 2*: You are?

- Student (Architecture)
- Student (Engineering)
- Professor (Architecture)
- Professor (Engineering)
- Professional (Architecture)
- Professional (Engineering)
- Other ------------------------

Question 3*: Do you use multi-objective design optimization in your work?

C. Never
D. Rarely
E. Sometimes
F. Usually
G. Always

B. Solutions ranking*(*required)*:

Please read the information then answer the question:

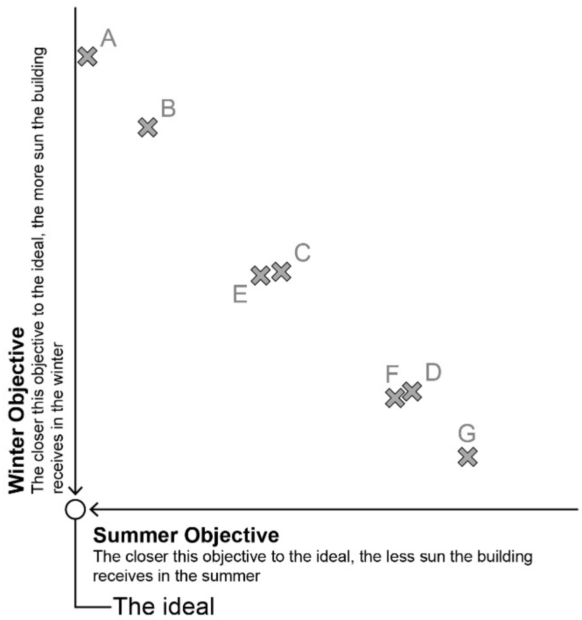
You are at the early stage of designing a mixed-use building (office, residential, commercial) in a mixed climate (hot summer, cold winter-Shanghai, China).

- The building is modular, the module size is: (Width=6m, Length=8m, Hight=4m).
- The required program is 12000 m^2.
- The land lot size is 8000 m^2 (W=80m, L=100m).
- The maximum number of levels allowed is 3 stories.
- A (25m X 30m) courtyard is required in the center of the building.

You have the climate data of shanghai and a solar gain performance graph (scatterplot) that represents seven candidate design solutions.

Month	Average High	Average Low	Sunshine hours
January	8.1°C	2.1°C	114.3 h
February	10.1°C	3.7°C	119.9 h
March	13.8°C	6.9°C	128.5 h
April	19.5°C	11.9°C	148.5 h
May	24.8°C	17.3°C	169.8 h
June	27.8°C	21.7°C	130.9 h
July	32.2°C	25.8°C	190.8 h
August	31.5°C	25.8°C	185.7 h
September	27.9°C	22.4°C	167.5 h
October	22.9°C	16.8°C	161.4 h
November	17.3°C	10.6°C	131.1 h
December	11.1°C	4.7°C	127.4 h

Here is a solar gain graph that represents the performance of the solutions in summer and in winter



A scatterplot that represents the solutions

Question 4*: We want to maximize the solar gain in winter and to minimize the solar gain in summer. Please rank the solutions according to your preference, "1st Place" represents the best while "7th Place" represents the worst.

Please, select only one circle for each solution

	Solution A	Solution B	Solution C	Solution D	Solution E	Solution F	Solution G
1st place	○	○	○	○	○	○	○
2nd place	○	○	○	○	○	○	○
3rd place	○	○	○	○	○	○	○
4th place	○	○	○	○	○	○	○
5th place	○	○	○	○	○	○	○
6th place	○	○	○	○	○	○	○
7th place	○	○	○	○	○	○	○

6. Scatterplot 2 with low level of information:

A. Personal information *(*required)*:

Question 1*: Is your work related to buildings design?
- Yes
- No

Question 2*: You are?
- Student (Architecture)
- Student (Engineering)
- Professor (Architecture)
- Professor (Engineering)
- Professional (Architecture)
- Professional (Engineering)
- Other ------------------------

Question 3*: Do you use multi-objective design optimization in your work?

C. Never
D. Rarely
E. Sometimes
F. Usually
G. Always

B. Solutions ranking*(*required)*:

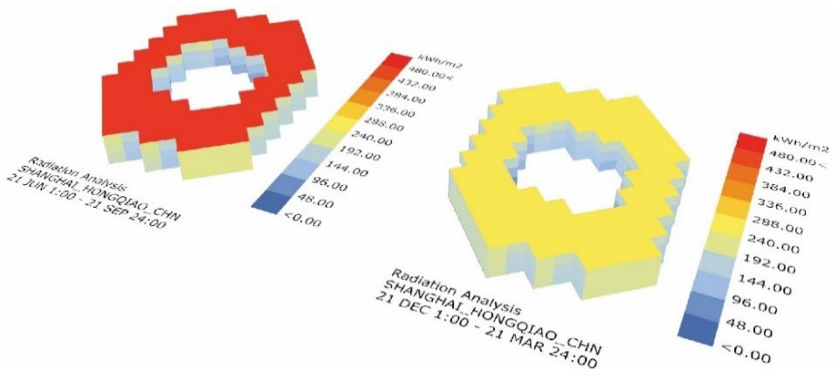
Please read the information then answer the question:

You are at the early stage of designing a mixed-use building (office, residential, commercial) in a mixed climate (hot summer, cold winter-Shanghai, China).

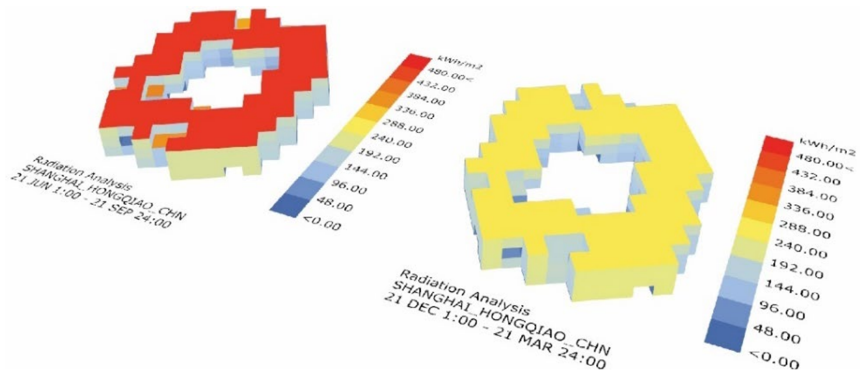
- The building is modular, the module size is: (Width=6m, Length=8m, Hight=4m).
- The required program is 12000 m^2.
- The land lot size is 8000 m^2 (W=80m, L=100m).
- The maximum number of levels allowed is 3 stories.
- A (25m X 30m) courtyard is required in the center of the building.

You have the climate data of shanghai, solar gain analysis for 7 candidate design solutions, and a solar gain performance graph (scatterplot).

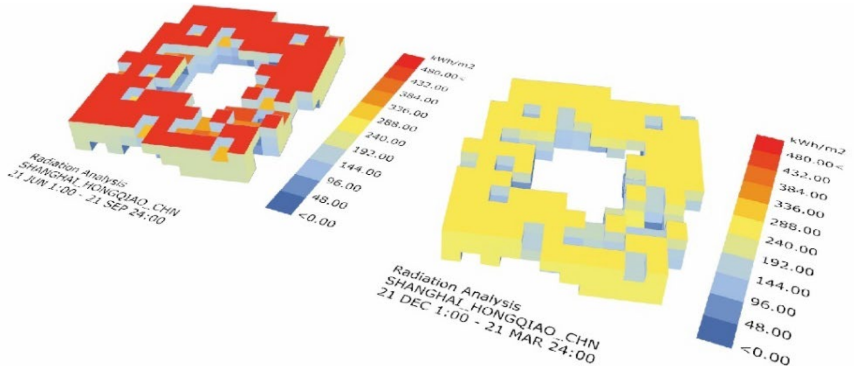
Month	Average High	Average Low	Sunshine hours
January	8.1°C	2.1°C	114.3 h
February	10.1°C	3.7°C	119.9 h
March	13.8°C	6.9°C	128.5 h
April	19.5°C	11.9°C	148.5 h
May	24.8°C	17.3°C	169.8 h
June	27.8°C	21.7°C	130.9 h
July	32.2°C	25.8°C	190.8 h
August	31.5°C	25.8°C	185.7 h
September	27.9°C	22.4°C	167.5 h
October	22.9°C	16.8°C	161.4 h
November	17.3°C	10.6°C	131.1 h
December	11.1°C	4.7°C	127.4 h



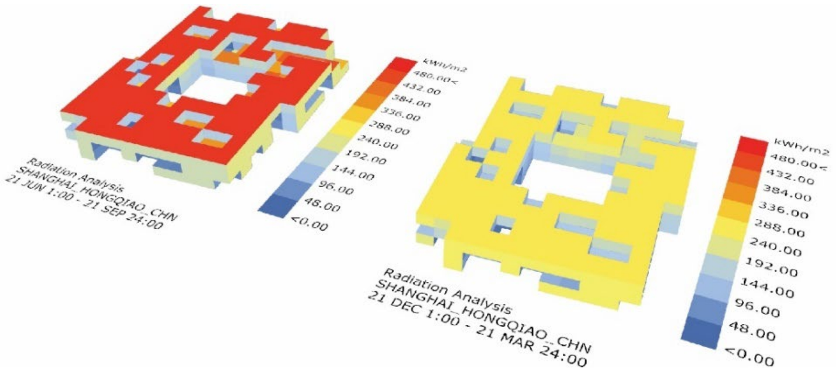
Solution A



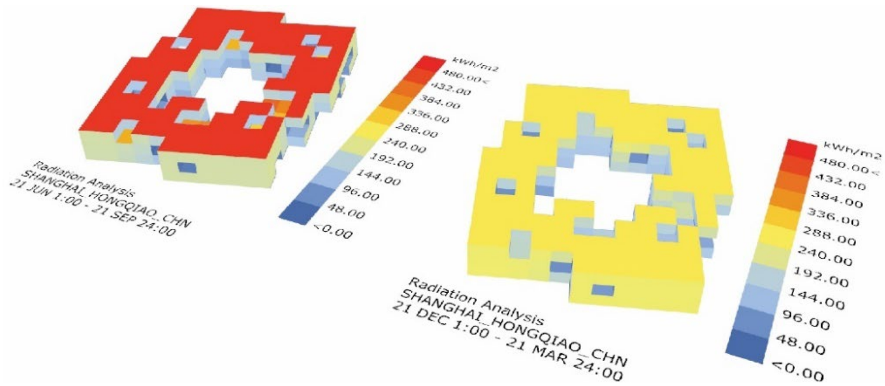
Solution B



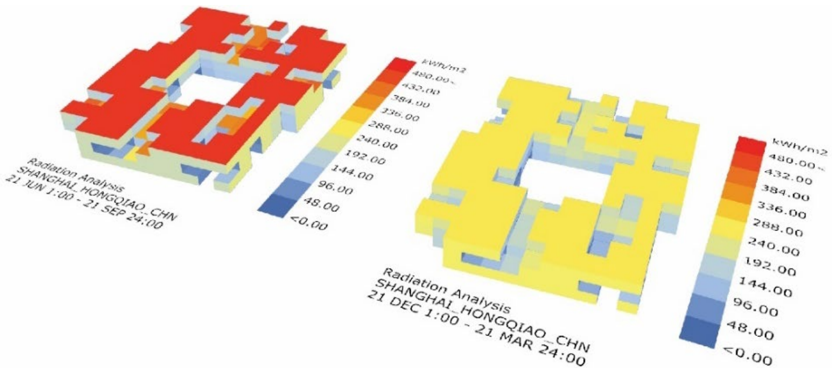
Solution C



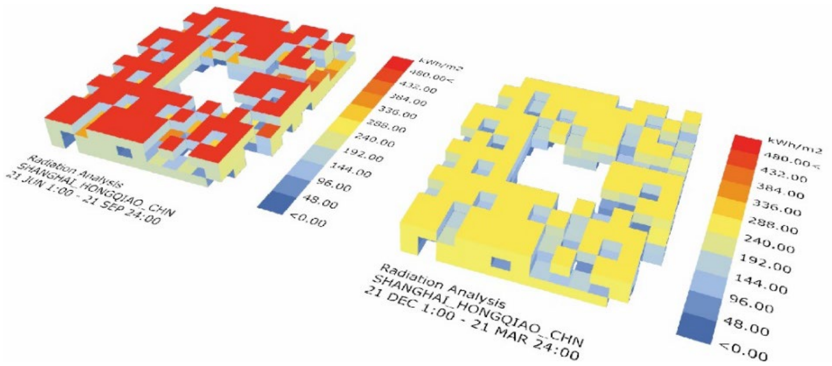
Solution D



Solution E

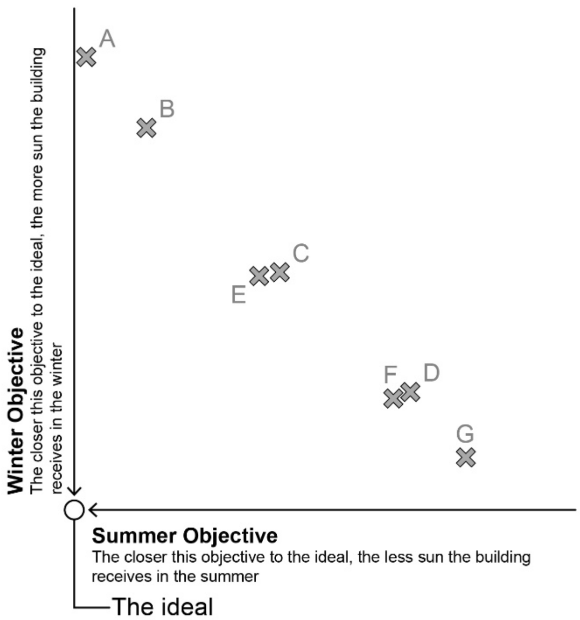


Solution F



Solution G

Here is a solar gain graph that represents the performance of the solutions in summer and in winter



A scatterplot that represents the solutions

Question 4*: We want to maximize the solar gain in winter and to minimize the solar gain in summer. Please rank the solutions according to your preference, "1st Place" represents the best while "7th Place" represents the worst.

Please, select only one circle for each solution

	Solution A	Solution B	Solution C	Solution D	Solution E	Solution F	Solution G
1st place	○	○	○	○	○	○	○
2nd place	○	○	○	○	○	○	○
3rd place	○	○	○	○	○	○	○
4th place	○	○	○	○	○	○	○
5th place	○	○	○	○	○	○	○
6th place	○	○	○	○	○	○	○
7th place	○	○	○	○	○	○	○

Appendix V: Experiment 5

A. Personal information *(*required)*:

Question 1*: Is your work related to buildings design?

- Yes
- No

Question 2*: You are?

- Student (Architecture)
- Student (Engineering)
- Professor (Architecture)
- Professor (Engineering)
- Professional (Architecture)
- Professional (Engineering)
- Other ------------------------

Question 3*: Do you use multi-objective design optimization in your work?

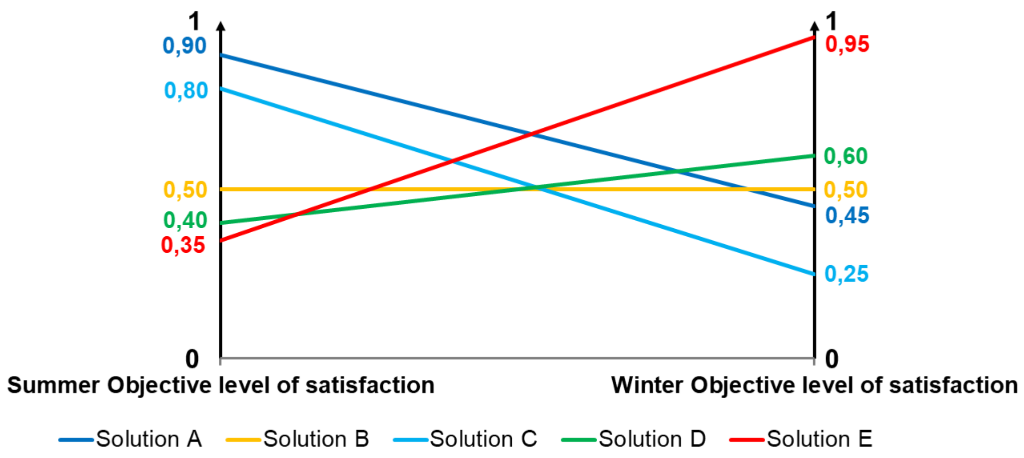
C. Never
D. Rarely
E. Sometimes
F. Usually
G. Always
H.

B. Two objectives test *(*required)*:

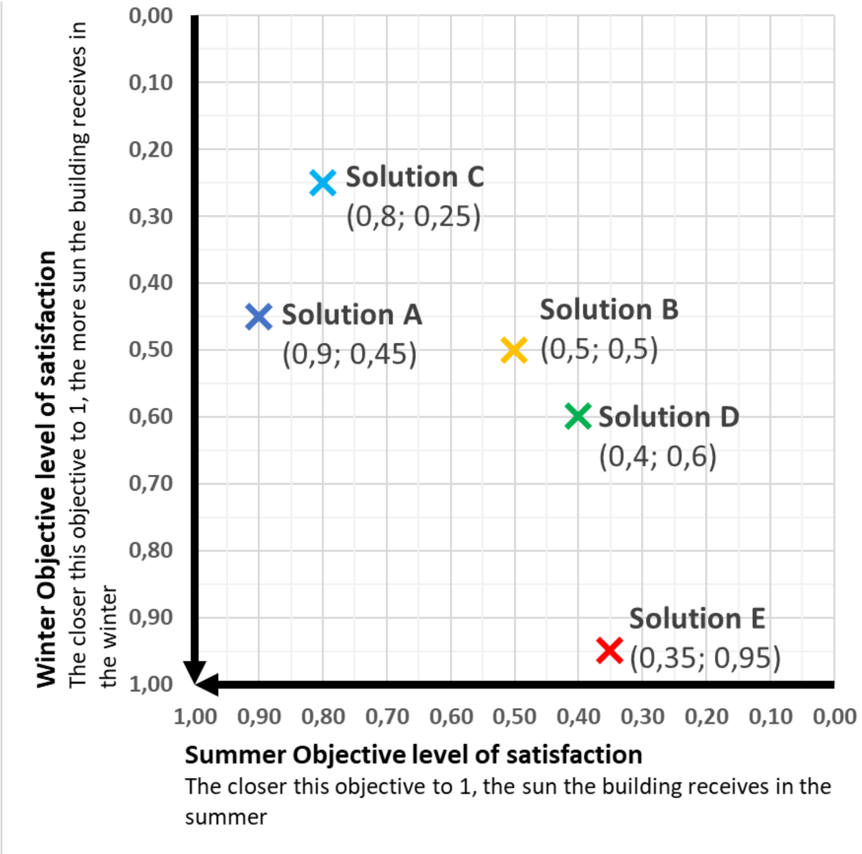
Question 1*: Based on the table and the figures below, Rank the solutions (A, B, C, D, E). 1st place represents the best while 5th represent the worst. The goal is to optimize the solar gain in summer and winter for a building in a mixed climate that is hot in summer & cold in winter (Shanghai, China). 1=High Satisfaction 0=Low Satisfaction.

	Summer Objective level of satisfaction	Winter Objective level of satisfaction
Solution A	0,9	0,45
Solution B	0,5	0,5
Solution C	0,8	0,25
Solution D	0,4	0,6
Solution E	0,35	0,95

The level of satisfaction of the solutions



A parallel coordinate that represents the solutions satisfaction of the objectives



A scatterplot that represents the solutions satisfaction of the objectives

Please, select only one circle for each solution

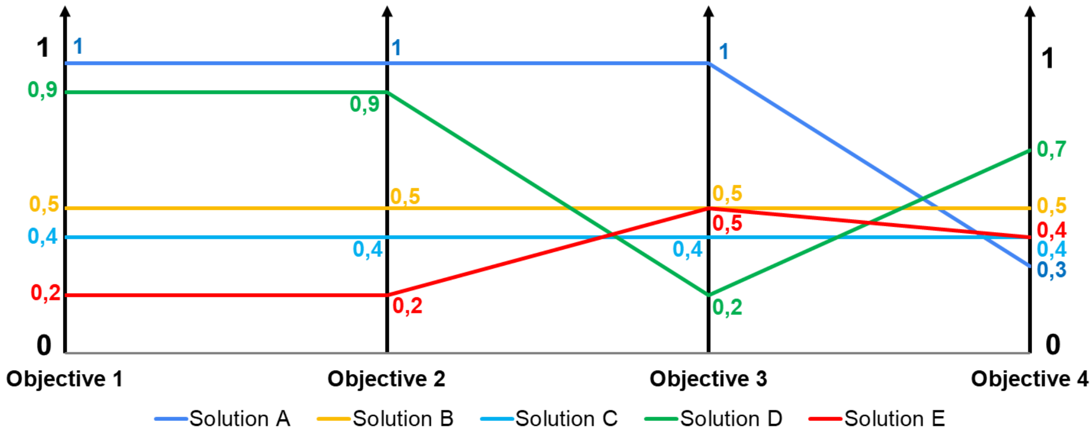
	Solution A	Solution B	Solution C	Solution D	Solution E
1st place	O	O	O	O	O
2nd place	O	O	O	O	O
3rd place	O	O	O	O	O
4th place	O	O	O	O	O
5th place	O	O	O	O	O

C. Three objectives test *(*required)*:

Question 2*: Based on the table and the figure below, Rank the solutions (A, B, C, D, E). 1st place represents the best while 5th represent the worst. The goal is to optimize4 objectives that are equally important for the design (1=High Satisfaction, 0=Low Satisfaction)

Solution	Objective 1	Objective 2	Objective 3	Objective 4
Solution A	1	1	1	0,3
Solution B	0,5	0,5	0,5	0,5
Solution C	0,4	0,4	0,4	0,4
Solution D	0,9	0,9	0,2	0,7
Solution E	0,2	0,2	0,5	0,4

The level of satisfaction of the solutions



A parallel coordinate represents solutions' satisfaction of the objectives

Please, select only one circle for each solution

	Solution A	Solution B	Solution C	Solution D	Solution E
1st place	O	O	O	O	O
2nd place	O	O	O	O	O
3rd place	O	O	O	O	O
4th place	O	O	O	O	O
5th place	O	O	O	O	O