Une plateforme open source d'optimisation générique hautement distribuée

Adrien Cabarbaye[1], Aurélien Cabarbaye[1], André Cabarbaye[1 & 2]

[1] CAB INNOVATION, 3 rue de la Coquille 31500 Toulouse France
Adrien.cabarbaye@gmail.com, Aurélien.cabarbaye@gmail.com
[2] Centre National d'Etudes Spatiales (CNES), 18 avenue Edouard Belin 31401 Toulouse France
Andre.cabarbaye@cnes.fr

Mots-clés *: Recherche opérationnelle, Optimisation multidisciplinaire, Optimisation distribuée, Open source*

Résumé :

Cet article présente une plateforme d'optimisation générique (Gencab Indra) en cours de développement (disponible fin 2016) qui cherche à résoudre des problématiques multiples dont celles de conception multidisciplinaire, qui mettent en œuvre différents domaines d'expertises interdépendants. Afin de pouvoir s'adapter à la diversité des problèmes rencontrés, notamment dans le domaine spatial, et faciliter leur formalisation par des ingénieurs non informaticiens, elle offre une large palette de formats de variables de décision ainsi qu'un ensemble d'algorithmes d'optimisation locale et globale pouvant s'imbriquer entre eux selon différents niveaux hiérarchiques. Exploitant les dernières technologies issues d'Internet et du Big Data, cette plateforme open source bénéficie d'une capacité de calculs parallèles massivement distribués. Elle s'inscrit dans un projet collaboratif qui a pour ambition de fédérer une communauté de chercheurs, développeurs, industriels et donneurs d'ordres.

1 Introduction

La plupart des problèmes d'optimisation dans le domaine industriel sont de type boîte noire car des informations du premier ordre (gradients) ne sont pas accessibles et que l'expression analytique des informations d'ordre zéro (valeurs de la fonction) est souvent inconnue pour diverses raisons (résultat de simulation, temps de calcul prohibitif, traitement inconnu ou caché, etc.). Les méthodes d'optimisation utilisées doivent pouvoir s'affranchir de multiples optima locaux et trouver un optimum après un nombre minimal d'évaluations de fonctions rarement linéaires, souvent contraintes et généralement gourmandes en temps de calcul. Ces méthodes doivent, par ailleurs, se révéler robustes face à la diversité des problématiques rencontrées par des spécialistes métiers qui n'ont pas forcément une expertise en recherche opérationnelle et aide à la décision.

Afin de répondre à ce besoin, un outil générique d'optimisation (Gencab) a été développé par la société Cab Innovation au début des années 2000 [7]. Basé sur des algorithmes génétiques couplés à un algorithme de simplexe (Nelder-Mead) pour accélérer la convergence, cet outil est notamment utilisé dans le domaine spatial pour optimiser des architectures de systèmes avec leur soutien logistique, ajuster des modèles probabilistes complexes et planifier des essais (Optimal design). Il a récemment été utilisé dans le cadre d'un projet de drone martien pour optimiser la forme des pales d'un rotor d'hélicoptère caractérisées au moyen de courbes de Bézier.

Fonctionnant sous Excel, cet outil présente des limites en termes de performance et de flexibilité. La conception aérospatiale est en effet gourmande en puissance de calcul et a souvent recours à des logiciels de simulation externes (XFoil pour le design de profil d'aile par exemple). Elle recouvre des disciplines interdépendantes qui ne peuvent véritablement aboutir à un optimum que si elles font l'objet d'un traitement global [1][2]. Aussi une refonte de cet outil s'impose-t-elle à nos yeux, d'autant qu'il nous semble opportun de tirer parti des nouvelles technologies issues d'Internet (web services, cloud computing…) et du Big Data (calculs parallèles et distribués).

En dépit d'une offre foisonnante, l'analyse préalable de l'état de l'art n'a pas permis de trouver la perle rare correspondant à nos attentes. Dans le domaine open source, la plupart des logiciels répondent à des besoins spécifiques, à des fins académiques ou de recherche, et leur capacité d'interfaçage s'avère limitée (Jenetics, Jenes, Watchmaker framework, pyEvolve, pyGMO, GALIB, JGAP, ECJ, EASEA). De même les nouvelles plateformes fondées sur le Big Data (MLib, Mahout, Vowpal Wabbit) ne proposent pas d'algorithme d'optimisation globale. C'est pourquoi nous avons pris la décision de développer une plateforme open source d'optimisation générique déployable sur des architectures hautement distribuées par des ingénieurs non informaticiens. Nous souhaitons inscrire ce développement dans un projet collaboratif, nommé Gencab Indra, afin de bénéficier de l'aide de la communauté scientifique et industrielle tant pour enrichir l'expression du besoin, en cherchant à définir un standard de formalisation des problématiques d'optimisation, que pour améliorer et enrichir les algorithmes retenus.

2 Vers un outil d'optimisation générique

Il n'existe pas de méthode universelle en optimisation mais un ensemble foisonnant de méthodologies dont chacune est efficace sur des problèmes présentant certaines caractéristiques :

- Les méthodes par évaluations et séparations progressives (branch and bound) consistent à énumérer intelligemment des solutions de problèmes combinatoires de manière à pouvoir éliminer des solutions partielles qui ne peuvent pas mener à la solution recherchée.
- les méthodes globales recherchent l'optimum global quand les méthodes locales s'arrêtent à l'optimum local au voisinage de conditions initiales,
- les méthodes de recherche directes impliquent l'examen séquentiel de solutions déterminées par une stratégie qui ne repose pas sur l'exploitation d'un modèle mathématique,
- les méthodes stochastiques prennent des décisions aléatoires dans la recherche contrairement aux méthodes déterministes, etc.

Aussi, l'utilisateur apparait-il comme un acteur incontournable dans le choix des méthodes bien que peu lui importe de choisir la meilleure si son problème se résout dans un délai raisonnable et que le format des variables de décision s'adapte à la diversité de ses problématiques.

Au début des années 2000, ce constat conduisit au développement d'un outil générique d'optimisation (Gencab) basé sur des algorithmes génétiques couplés à un algorithme de simplex (Nelder-Mead). Limité à des variables réelles, entières et binaires, et fonctionnant dans l'environnement Excel, celui-ci permet de traiter des applications diverses mais montre également ses limites face à des problématiques toujours plus complexes.

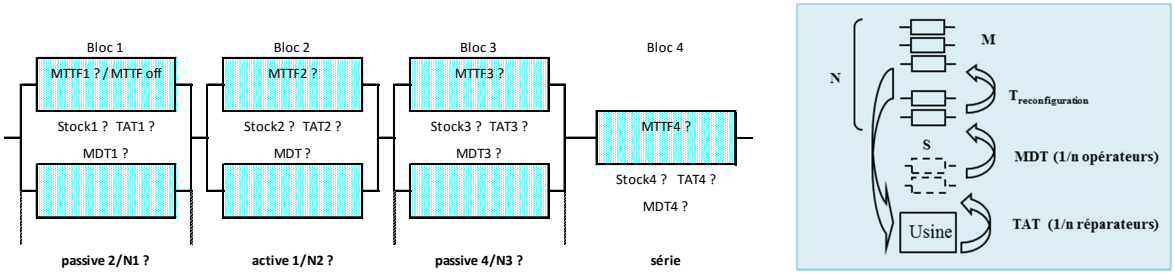
2.1 Apport et limite d'un outil existant

Utilisé dans le domaine de la fiabilité, Gencab peut se coupler à des outils de traitements markoviens, d'arbres de fautes, ou de simulation de systèmes hybrides (à variables aléatoires et continues) à états discrets. Il est ainsi employé pour optimiser globalement l'architecture et les conditions de maintenance et de soutien logistique d'installations au sol de systèmes spatiaux (centre de contrôle, station de réception, etc.) [9]. Contraint par la tenue d'un objectif de disponibilité opérationnelle, le coût global de possession d'un tel système est minimisé dans l'exemple de la figure 1, en jouant sur 19 paramètres entiers ou réels. Ceux-ci portent sur la fiabilité des équipements à travers leur MTTF (Mean time to Failure), les niveaux de redondance m parmi n, les durées d'indisponibilité MDT (Mean Down Time) qui recouvrent les temps de détection, d'intervention, de réparation et remise en service, les stocks de rechange et les durées TAT (Turn Around Time) de réapprovisionnement des stocks, qui ont chacun une influence sur la disponibilité et sur les coûts.

Ce type d'optimisation ne pose pas de difficulté particulière quand les évaluations s'effectuent rapidement (de l'ordre d'une seconde). Mais il n'en est pas de même quand des complexités architecturales, comme par exemple la présence d'un stock de rechange partagé, nécessitent de passer à la simulation de Monte-Carlo. Cette dernière augmente d'un facteur 1000 environ la durée de chacune des évaluations.

La simulation s'impose également pour résoudre des problématiques spatiales complexes comme celles relatives aux constellations de satellites pour lesquelles des choix de satellites sur des orbites multiples et des stratégies de déploiement et de renouvellement avec des lanceurs de tailles diverses sont à définir.

En dépit d'améliorations apportées sur le couplage entre optimisation et simulation de Monte-Carlo consistant à limiter le nombre de simulation des solutions jugées médiocres après évaluation grossière, la parallélisation massive des traitements s'avère incontournable pour résoudre efficacement ce type de problème.



ELEMENTS	MTTF ON (heure)	Nb	Type de redondance	MTTF OFF (heure)	Taux d' utilisation r (%)	Durée de reconfig. (heure)	MDT (heure)	Nombre d'opérateurs (1 ou n)	Stock de rechanges	TAT (heure)	Nombre de réparateurs (1 ou n)	Disponibilité à t infini	Coût (€)
Bloc 1	3836		passive 2/3	115079		50	102		2	502		0,9683	113160
Bloc 2	2936		active 1/3				408		3	525		0,9863	100664
Bloc 3	4479		passive 4/6				162		3	510		0,9883	150689
Bloc 4	6709		série				101		1	671		0,9755	83687
$	ENSEMBLE											0,9207	448200
												≥	↓
											Objectif :	0,92	

FIG. 1 – Optimisation globale d'un système (architecture, maintenance et soutien logistique)

Cet outil est également utilisé comme moyen d'ajustement de modèles probabilistes complexes [8] et de planification optimale d'essais (Optimal design). En effet, l'utilisation d'un outil d'optimisation globale s'avère incontournable pour mettre en œuvre la méthode du maximum de vraisemblance quand cette dernière présente des optima multiples, ou pour maximiser le déterminant de la matrice de Fisher (méthode D-optimal) relative à des processus non linéaires. Ainsi des données hétérogènes de retour d'expériences (REX) issues de conditions d'utilisation ou d'environnement variées (température, vibration, humidité, charge, voltage…) peuvent être exploitées au moyen de divers modèles accélérés de fiabilité (Weibull…) ou de dégradation (Processus gamma non stationnaire…), et des essais coûteux peuvent être réduits en nombre en maximisant a priori la précision de leurs résultats.

Dans le domaine de la conception optimale, Gencab a récemment été utilisé dans le cadre d'un projet de drone martien pour optimiser la forme des pales d'un rotor d'hélicoptère. Illustrée par la figure 2, cette forme est définie au moyen de courbes de Bézier dans l'espace à 3 dimensions constitué de la distance à l'emplanture (position), de la largeur de pale (corde) et de l'incidence. L'optimisation consiste à minimiser la puissance énergétique nécessaire au vol stationnaire, tout en respectant la contrainte de portance qui doit être suffisante pour supporter la masse de l'engin. Développé pour des applications terrestres, le modèle de calcul paramétrique a été associé à des profils aérodynamiques types à très faible coefficient de moment aérodynamique et très faibles Reynolds, mais cependant conçus pour des valeurs supérieures à celles rencontrées sur le sol martien. Aussi, l'approfondissement de cette étude originale impose-t-il l'emploi d'un outil d'optimisation globale ayant la capacité de se coupler à un outil d'évaluation de profils aérodynamiques tel que XFOIL, par exemple. Il nécessite, par ailleurs, un traitement multidisciplinaire pour optimiser globalement les aspects aérodynamiques, structuraux et d'analyse de mission.

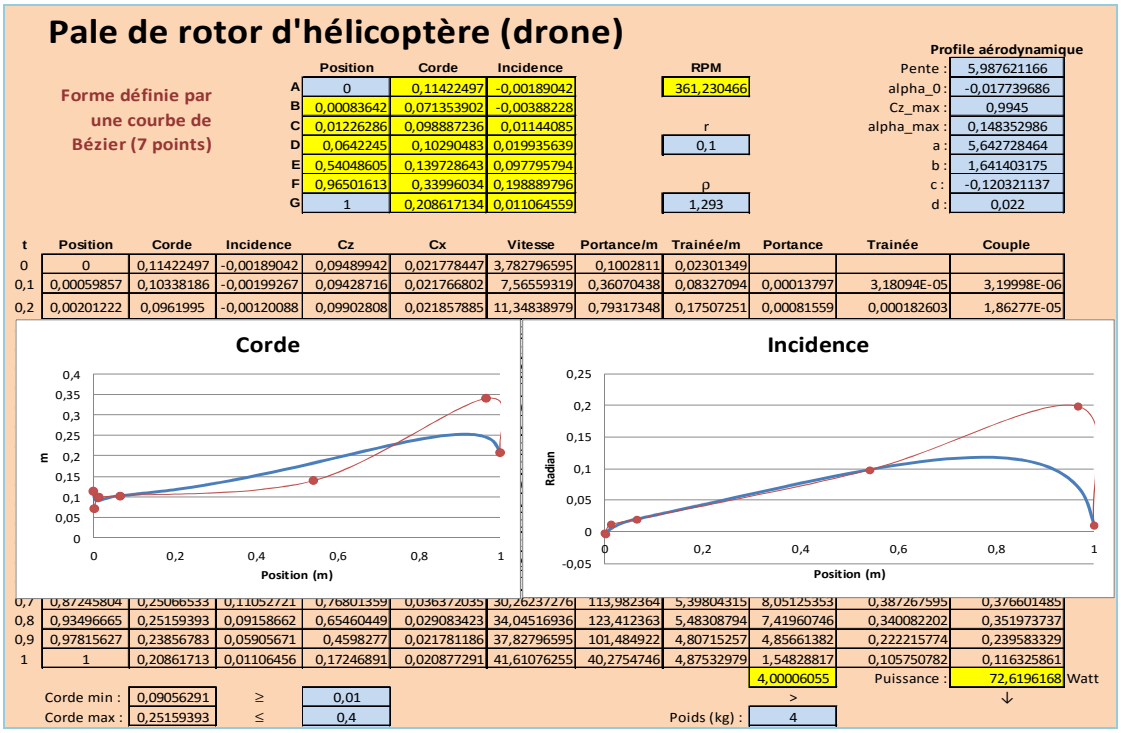


FIG. 2 – Forme et vitesse minimisant la consommation d'énergie

2.2 Optimisation multidisciplinaire (Multidisciplinary Design Optimization)

L'Optimisation multidisciplinaire (OMD ou MDO en anglais) cherche à résoudre des problèmes de conception mettant en œuvre plusieurs disciplines [1][2]. L'optimum global recherché est meilleur que la configuration trouvée en optimisant chacune des disciplines indépendamment des autres, car il prend en compte les interactions entre celles-ci. Illustré par la figure 3, un système multidisciplinaire comprend des variables globales (z), des variables locales propres à chacune des disciplines (x) et des variables de partage (y) qui sont des sorties des différentes disciplines.

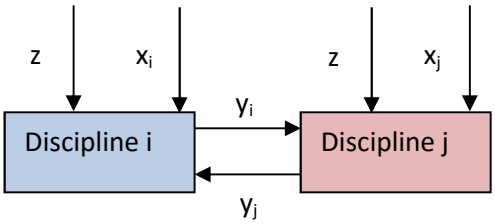


FIG. 3 – Système multidisciplinaire

A titre d'exemple, la conception d'un satellite est effectuée par des équipes ayant des connaissances spécifiques dans un domaine précis telles que l'instrumentation de la charge utile, le contrôle d'attitude et d'orbite, la structure, le système électrique, le système thermique... Chaque métier s'attache à obtenir une conception acceptable dans son domaine en faisant régulièrement des compromis avec les autres spécialités. Le but à atteindre est généralement de minimiser le poids, le volume et le coût du satellite tout en lui assurant des performances acceptables durant toute la durée de la mission. Cependant, quand l'optimisation s'arrête aux frontières de chacune des disciplines, la gestion des interfaces conduit à considérer des pires cas et cumuler des marges cachées qui peuvent avoir des effets de seuil très pénalisants, dont notamment le changement de catégorie de satellite (gros, mini, micro, nano) ou de lanceur (Véga, Soyouz, Ariane).

L'optimisation multidisciplinaire fait appel à des méthodes mono ou multiniveaux. La méthode MDF (Multidisciplinary Feasible Design) met en œuvre un optimiseur unique qui agit sur un

système intégré comprenant un MDA (Multidisciplinary Design Analysis). Ce dernier résout les équations d'état et détermine les sorties des disciplines à chaque itération du processus d'optimisation. Il pénalise néanmoins les temps de calcul.

$$Min\ f(x,y,z) \quad g(x,y,z) \leq 0 \quad f : critère\ d'optimisation \quad g : contraintes\ diverses \qquad (1)$$

La méthode IDF (Individual discipline Feasible) utilise des contraintes d'égalité sur les sorties des disciplines (Y) afin de supprimer le besoin de MDA. La faisabilité multidisciplinaire n'est alors assurée qu'à la convergence de l'algorithme.

$$Min\ f(x,y,z) \quad g(x,y,z) \leq 0\ ;\ Y - y = 0 \qquad (1)$$

Illustrée par la figure 4, la méthode CO (Collaborative optimization) met en œuvre plusieurs optimiseurs qui reproduisent le fonctionnement d'une entreprise où la direction donne des consignes que les différents services tentent de respecter. Au niveau système, un optimiseur joue sur les variables globales et de partage pour minimiser la fonction de coût tout en respectant les contraintes système ainsi que des contraintes d'égalité assurant le respect des consignes au niveau inférieur.

$$Min\ f(y,z) \quad g(y,z) \leq 0 \quad Y - y = 0 \quad zi - z = 0 \quad zj - z = 0 \qquad (1)$$

Au niveau disciplinaire, des optimiseurs cherchent à suivre au mieux les consignes tout en respectant des contraintes propres à la discipline.

$$\text{Min } \|yi - Yi(z_i,x_i,y_j)\|^2 + \|z_i - z\|^2 \quad g(z_i,x_i,y_j) \leq 0 \qquad (1)$$

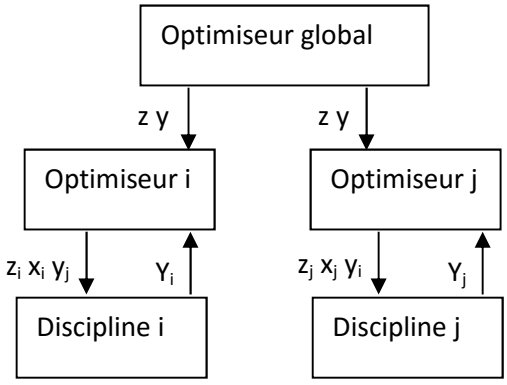


FIG. 4 – Système multidisciplinaire

Cette méthode à 2 niveaux est étendue à des niveaux hiérarchiques multiples dans la méthode ATC (Analytical Target Cascading). D'autres méthodes multiniveaux existent telles que la méthode CSSO (Concurrent Subsapce Optimization) pour laquelle chaque discipline tente d'optimiser le problème global en utilisant des modèles approchés des autres disciplines, ou la méthode BLISS (Bi-Level Integrated System Synthesis method) où les variables de couplage sont gérées par un MDA. Selon la complexité des modèles utilisés dans chacune des disciplines (modèles empiriques, éléments finis, CFD…), des modèles de substitution simplifiés peuvent être utilisés afin de faciliter la recherche de l'optimum global.

3 Caractéristiques d'un outil d'optimisation générique

3.1 Fonctionnalités générales

Afin de pouvoir s'adapter à la diversité des problèmes rencontrés et faciliter leur formalisation par des ingénieurs non informaticiens, un outil d'optimisation générique doit offrir une large palette de formats de variables de décision ainsi qu'un ensemble d'algorithmes d'optimisation locale et globale pouvant s'imbriquer entre eux selon différents niveaux. Outre une capacité de recherche multi-objectifs, il doit pouvoir s'interfacer à divers logiciels d'évaluation ou de simulation externes et permettre la parallélisation de traitements massivement distribués afin de limiter les temps de calcul.

Le format des variables de décision peuvent notamment se choisir parmi les types suivants :

- binaire (0 ou 1),
- réel entre valeurs min et max, incluses ou exclues,
- entier entre valeurs min et max, incluses ou exclues,
- valeur alphanumérique parmi n (choix d'un profil aérodynamique standard par exemple),
- liste de n valeurs ordonnées (problème du voyageur de commerce),
- liste de m valeurs, différentes ou non, ordonnées ou non, parmi n (problème du sac à dos, planification de tâches), etc.

Ces types de variables peuvent s'employer indifféremment dans un problème et le choix d'un ordre parmi les variables de différents types constitue une forme générique de solution, similaire à celle des chromosomes des Algorithmes Génétiques.

La possibilité pour l'utilisateur (ou de manière semi-automatique) d'effectuer des regroupements au sein de cette liste ordonnée permet de la soumettre à des optimisations séquentielles employant des algorithmes adaptés aux formats des variables de chacun des regroupements (recherche arborescente, recherche directe, Algorithmes Génétiques, etc.). Ainsi, le choix de valeurs alphanumériques par une méthode de recherche directe pourra s'effectuer en préalable au traitement de variables réelles par d'autres algorithmes. De même l'évaluation de solutions candidates, au cours de l'optimisation, pourra elle-même faire appel à des optimisations, comme cela s'avère parfois nécessaire pour résoudre des problématiques d'analyse de missions.

Enfin l'ouverture à des plateformes d'évaluation diverses et la parallélisation des traitements sur des architectures multi-cœurs distribuées en clusters apparaissent indispensables pour offrir la flexibilité nécessaire à la résolution des problèmes industriels d'une certaine complexité.

3.1.1. Algorithmes génétiques

Développés en 1970 par John Holland et ses collaborateurs à l'université du Michigan, les algorithmes génétiques sont des algorithmes d'optimisation inspirés des mécanismes de la sélection naturelle dans le monde vivant [3]. Des chromosomes constitués de différents gènes, correspondant à des configurations de paramètres, subissent des opérations de mutation, croisement et sélection au sein d'une population, comme l'illustre la figure 5. La mutation est une modification aléatoire de la valeur d'un gène de chromosome. Le croisement est un échange aléatoire d'informations entre deux chromosomes donnant naissance à deux descendants. La sélection consiste à dupliquer des chromosomes afin de générer une nouvelle population de la taille de celle d'origine. Les chromosomes ayant une performance élevée ont une forte probabilité d'avoir un ou plusieurs descendants identiques à eux-mêmes. Les chromosomes moins performants risquent de disparaitre mais gardent cependant une chance d'être conservés. Ainsi, la performance de la population s'améliore progressivement au cours des générations successives.

Ce principe de base peut s'enrichir d'autres opérateurs qui améliorent singulièrement la performance des algorithmes (voir figure 6).

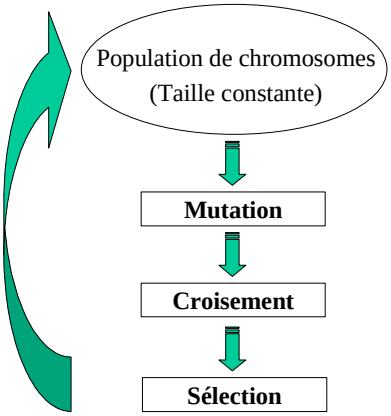


FIG. 5 – Principe de base des algorithmes génétiques

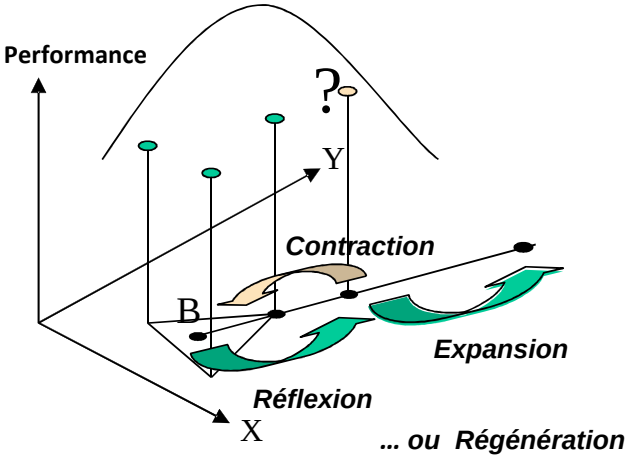


FIG. 7 – Simplexe

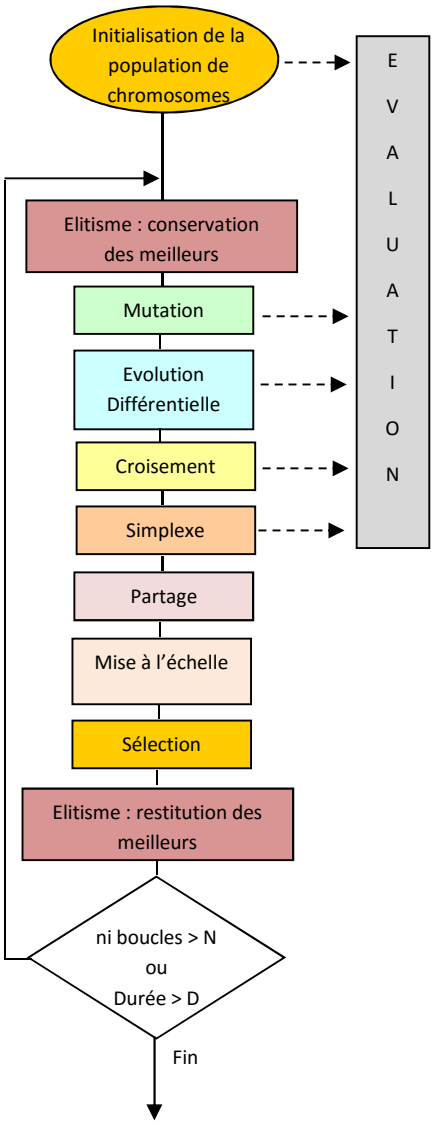


FIG. 6 – Algorithmes génétiques

L'élitisme consiste à conserver à chaque génération un ou plusieurs chromosomes de la population parmi les meilleurs qui pourraient disparaître au cours de la sélection ou d'autres opérations.

Proposée en 1995 par K. Price et R. Storn, l'évolution différentielle consiste à générer un nouveau chromosome en sommant, aux gènes d'un membre de la population, la différence entre les gènes de deux autres chromosomes tirés aléatoirement [4]. A mi-chemin entre la mutation et le croisement, cet opérateur joue simultanément sur l'ensemble des gènes réels ou entiers.

Illustré par la figure 7, le simplexe (ou algorithme de Nelder-Mead) est une méthode d'optimisation locale qui peut être utilisée pour améliorer un certain nombre de chromosomes parmi les meilleurs. S'appliquant uniquement aux gènes réels ou entiers (avec arrondi), cette recherche s'effectue selon un certain nombre de pas.

Le partage (sharing) est une transformation de la valeur de la fonction d'évaluation qui a pour but de maintenir la diversité en pénalisant l'agrégation des chromosomes au moyen d'une notion de distance caractérisant la dissimilarité entre les individus.

La mise à l'échelle (scaling) est également une transformation de la valeur de la fonction d'évaluation ayant pour but de créer un effet de zoom sur les résultats au fur et à mesure de la recherche. Les écarts d'évaluation sont d'abord limités, afin d'éviter que les bons chromosomes deviennent prépondérants, puis amplifiés progressivement pour accélérer la convergence.

Par ailleurs, l'introduction d'un contrôle adaptatif à certains opérateurs s'avère nécessaire pour limiter progressivement l'exploration au fur et à mesure de la recherche. Ainsi, la mutation d'un gène réel (ou entier) se traduira par un saut d'amplitude décroissante au cours du temps au fur et à mesure de l'amélioration progressive de la population.

La population initiale est choisie de manière aléatoire ou déterministe, afin de répartir les chromosomes de manière homogène dans l'espace des solutions ou intégrer des candidats issus d'une expertise ou d'un traitement préalables. Les algorithmes sont alors lancés pour un certain nombre de boucles (N) ou pour une certaine durée (D).

3.1.2. Recherche multi-objectifs

Une recherche multi-objectifs consiste à optimiser une fonction suivant plusieurs critères dont certains peuvent être antagonistes (coût et durée d'un trajet par exemple). La performance d'une configuration se présente alors sous une forme vectorielle. Outre l'éventuelle pondération des critères, il n'est pas possible de définir de manière générale la valeur optimale d'un problème d'optimisation multi-objectifs. Il en revanche possible de définir un ensemble de solutions dominantes qui sont meilleures que les autres suivant l'un des critères et forment un front de Pareto.

3.1.3. Gestion des contraintes

La gestion des contraintes peut s'opérer de deux manières différentes. Soit leur violation conduit à une forte pénalité de performance, en acceptant des incursions transitoires dans les domaines interdits, soit une fonction barrière, croissant très rapidement lorsqu'on s'approche des frontières de contraintes, est ajoutée à la performance afin que la recherche reste toujours en dehors du domaine interdit (méthode du point intérieur).

3.1.4. Parallélisation des traitements

Afin de gagner du temps de calcul, les algorithmes génétiques peuvent fonctionner simultanément sur différents processeurs. Cette parallélisation des traitements peut s'envisager de diverses manières :

- l'un des processeurs gère la population totale et envoie les évaluations à d'autres pour qu'elles se fassent en même temps,
- la population est répartie en sous-populations sur l'ensemble des processeurs,
- la population est répartie en sous-populations sur des processeurs différents qui eux-mêmes renvoient à d'autres le soin d'effectuer les évaluations.

Dans tous les cas, il est cependant nécessaire d'avoir un mécanisme de communication inter-processus permettant à un réseau hétérogène de machines d'apparaître comme une seule entité.

L'évaluation se révèle généralement gourmande en capacité de calcul et est répétée un grand nombre de fois au cours des traitements. Aussi, sa parallélisation s'avère prometteuse.

Le parallélisme par îlots consistant à répartir la population sur différents processeurs introduit des regroupements (clustering) puisque les îlots convergent vers des optima qui leur sont propres. Aussi, un opérateur de migration entre processus peut-il être introduit afin de propager une partie de l'élite vers d'autres unités de traitement.

La parallélisation peut se faire par l'utilisation de tâches (threads) dans un même processus et/ou via l'utilisation simultanée de plusieurs processus sur des architectures distribuées en grappes (clusters) telles que celles proposées par les fournisseurs de cloud (e.g. Amazon AWS, Microsoft Azure, Google Compute Engine, ou les francais CloudWatt, Numergy, Outscale, OVH et Ikoula).

3.1.5. Adaptation des opérateurs aux divers types de gène

L'opérateur de mutation peut être étendu à tous les types de gènes comme indiqué dans la table 1.

Type de gène	Mutation
Réel	Saut aléatoire limité par les bornes min et max
Entier	Identique à celle d'un gène réel suivie d'un arrondi
Binaire	Changement d'état (0 ou 1)
Alphanumérique	Tirage aléatoire d'une valeur alphanumérique parmi n
Liste de n valeurs ordonnées	Inversion de deux valeurs tirées aléatoirement de la liste
Liste de m valeurs non ordonnées parmi n	Remplacement d'une valeur tirée aléatoirement de la liste par une des n valeurs tirée aléatoirement
Liste de m valeurs différentes non ordonnées parmi n	Remplacement d'une valeur tirée aléatoirement de la liste par une des n valeurs tirée aléatoirement non présentes dans la liste
Liste de m valeurs ordonnées parmi n	Identique à celle d'une liste de m valeurs non ordonnées parmi n ou à celle d'une liste de m valeurs ordonnées
Liste de m valeurs différentes ordonnées parmi n	Identique à celle d'une liste de m valeurs différentes non ordonnées parmi n ou à celle d'une liste de m valeurs ordonnées

TAB. 1 – Mutations

L'opérateur de croisement de type enjambement (Crossover), correspondant à l'échange de gènes entre parents, est applicable à tous les types de gènes. En revanche, celui de type barycentrique consistant à remplacer la valeur d'un gène par une pondération entre ceux de deux parents, n'est applicable qu'à des gènes réels ou entiers (avec arrondi).

3.2 Déploiement sur des architectures hautement distribuées

Gencab Indra est une plateforme d'optimisation paramétrique générique hautement distribuée. Illustrée par la figure 8, celle-ci permet de déployer des algorithmes d'optimisation sur des fermes de calcul (cluster) au moyen d'un ensemble cohérent de composants logiciels (e.g. Apache Spark [5]).

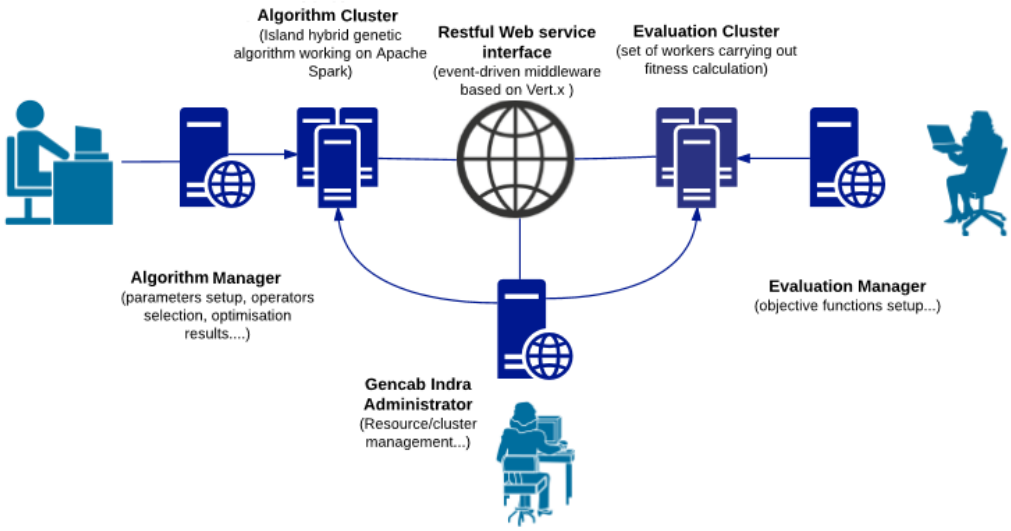


FIG. 8 – Plateforme d'optimisation Gencab Indra

Développés initialement pour le Big Data, ce genre d'architecture est facilement configurable dans le cloud. Les algorithmes d'optimisation font également appel à des ressources externes pour évaluer des configurations de paramètres via des services Web RESTful suivant la philosophie des microservices [6].

Le formalisme très générique des algorithmes génétiques consistant à définir une solution, soit une configuration de paramètres de types divers, sous la forme d'une liste ordonnée de paramètres (pseudo chromosomes) est conservé sur Gencab Indra, et ceci, même si cette solution peut résulter de n'importe quel type d'algorithme d'optimisation. Adaptés aux types de paramètres à traiter, ces algorithmes d'optimisation locale (simplexe, etc.) ou globale (recherche arborescente, algorithmes génétiques, etc.) peuvent s'imbriquer entre eux pour faciliter la formulation des problématiques et accélérer les traitements.

3.3 Imbrication des algorithmes

Une liste ordonnée de paramètres de différents types (chromosome) peut être traitée par plusieurs algorithmes imbriqués qui interviennent chacun sur un certain nombre de paramètres successifs de la liste.

3.3.1. Optimisations successives imbriquées

Les algorithmes imbriqués interviennent successivement sur un ensemble de paramètres de la liste. Cette technique est utilisée afin de pouvoir utiliser des algorithmes adaptés à des types de paramètres formant des sous-groupes dans la liste. Elle peut l'être également pour traiter des problématiques pour lesquelles l'évaluation nécessite elle-même une optimisation.

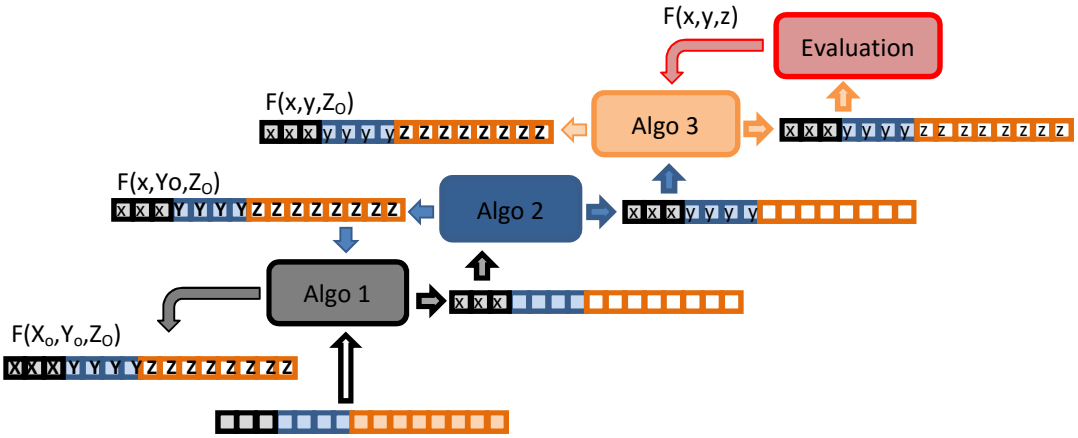


FIG. 9 – Optimisations successives imbriquées

Ainsi, dans l'exemple de la figure 9, l'algorithme de type 1 n'intervient que sur les 3 premiers paramètres. Il transmet à l'algorithme de type 2 des configurations de 3 paramètres à évaluer. L'algorithme de type 2 intervient sur les 4 paramètres suivants. Il transmet à l'algorithme de type 3 des configurations de 7 (3+4) paramètres à évaluer. Des configurations de 15 (3+4+8) paramètres sont alors envoyées au système chargé de l'évaluation par l'algorithme de type 3 en jouant sur les 8 derniers paramètres jusqu'à l'obtention d'un optimum partiel qui est retransmis à l'algorithme de type 2 avec sa performance. Ce dernier réitère l'opération en jouant sur ses 4 paramètres jusqu'à obtenir un optimum partiel retransmis à l'algorithme de type 1 qui lui-même réitère l'opération en jouant sur ses 3 paramètres jusqu'à obtenir l'optimum global. Dans le cadre de l'Optimisation multidisciplinaire, les optimisations successives imbriquées font partie des méthodes mononiveaux (single-level) puisqu'elles font appel à un unique optimiseur.

3.3.2. Optimisations parallèles imbriquées

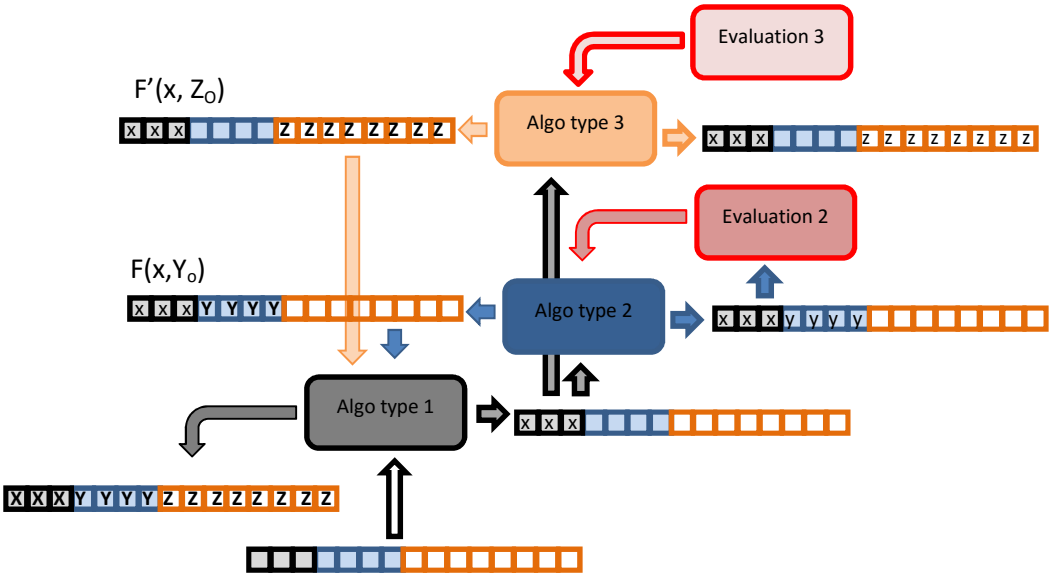


FIG. 10 – Optimisations parallèles imbriquées

Certains des algorithmes imbriqués interviennent parallèlement sur un ensemble de paramètres de la liste. Cette technique est utilisée en recherche multi-objectifs quand les différents critères ne dépendent pas individuellement de tous les paramètres. Elle permet alors de différencier les évaluations pour en limiter le nombre au strict nécessaire. Ainsi, dans l'exemple de la figure 10, l'algorithme 1 transmet aux algorithmes 2 et 3 des configurations de 3 paramètres à évaluer. L'algorithme 2 intervient sur les 4 paramètres suivants et lance des évaluations de type 2 jusqu'à l'obtention d'un optimum qui est retransmis à l'algorithme 1 avec sa performance. L'algorithme 3 intervient de la même manière en jouant sur les 8 derniers paramètres.

Dans le cadre de l'Optimisation multidisciplinaire, les optimisations parallèles imbriquées font partie des méthodes multiniveaux avec des variables (paramètres) partagées ayant une influence sur plusieurs disciplines et des variables locales (ou d'état) qui n'interviennent que dans une seule.

3.3.3. Evaluation stochastique

Les problématiques peuvent recouvrir des variables aléatoires. L'évaluation porte alors sur des résultats statistiques (moyenne, quantile…) issus d'évaluations multiples ayant chacune fait l'objet de tirages aléatoires (simulation de Monte-Carlo).

Ces évaluations, dont le nombre dépend de la précision des résultats recherchée, peuvent être traitées en parallèle par différents processeurs pour ne pas trop pénaliser les temps de calcul, comme l'illustre la figure 11.

Un algorithme de couplage est alors nécessaire pour gérer les évaluations (individuellement ou par échantillon) et élaborer les résultats statistiques à partir des résultats individuels. Après une estimation grossière, celui-ci doit limiter au plus juste le nombre d'évaluations des configurations peu prometteuses afin que les temps de calcul ne soient pas rédhibitoires même en parallélisant les traitements sous incertitude [1].

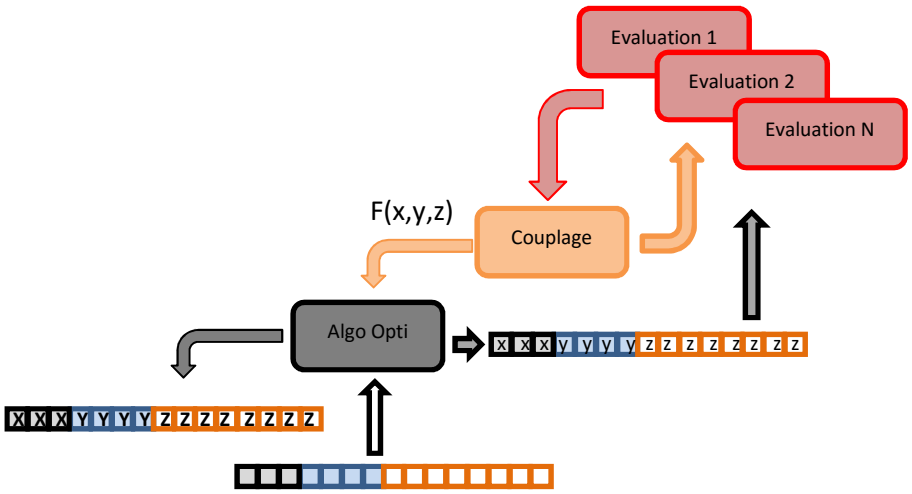


FIG. 11 – Optimisation couplée à la simulation de Monte-Carlo

Afin de simplifier les interfaces et limiter les échanges, les algorithmes imbriqués peuvent être traités globalement par les mêmes processeurs qui déportent cependant les évaluations sur d'autres processeurs.

3.3.4. Structure générique d'entrée/sortie des algorithmes

A des fins de standardisation, la même structure générique d'entrée / sortie, décrit en figure 12, peut être utilisée pour tous les différents types d'algorithmes d'optimisation.

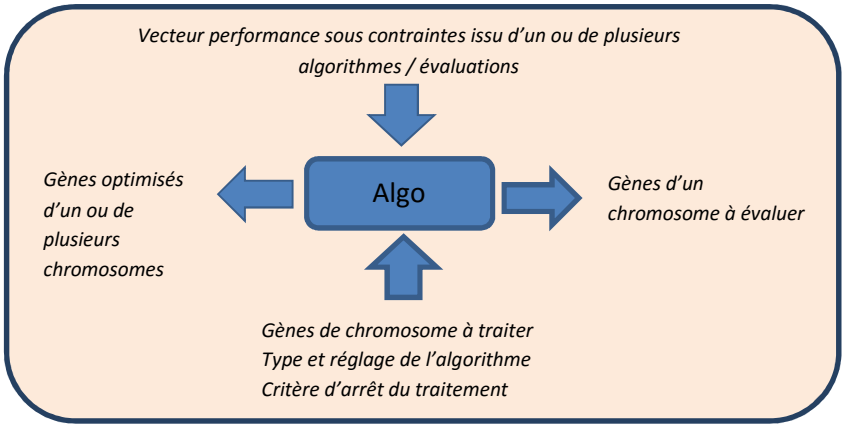


FIG. 12 – Structure générique d'entrée/sortie des algorithmes

A titre d'exemple, l'algorithme du simplexe peut être appliqué sur les 6 derniers gènes réels d'un chromosome dont les 4 premiers, de type alphanumérique, sont fournis en entrée.

La recherche itérative de l'optimum local est menée à partir d'un simplexe, soit 7 (n+1) configurations de paramètres réels tirées aléatoirement, dont on calcule la performance ainsi que le barycentre (voir figure 7). En partant du barycentre dans la direction du meilleur élément du simplexe, la recherche consiste à effectuer une réflexion (déplacement du barycentre vers le meilleur d'une distance double) suivie d'une expansion (déplacement du barycentre vers le meilleur d'une distance quadruple) si la recherche est fructueuse, ou bien d'une contraction (déplacement du barycentre vers le meilleur d'une demi-distance) dans le cas contraire. Si aucune amélioration n'a été observée durant ces trois opérations, une régénération est alors opérée. Celle-ci consiste à repartir d'un simplexe resserré comprenant le meilleur élément parmi les précédents.

Cinquante boucles de simplexe (pas) constituent par exemple le critère d'arrêt. Durant chaque boucle de traitement, les chromosomes obtenus par réflexion, expansion et contraction peuvent être évalués simultanément sur 3 processeurs différents pour diviser le temps de calcul par 2. D'autres

méthodes de recherche directe sont cependant mieux adaptées à une parallélisation massive (évaluations simultanées en amplitude et direction).

Chaque système d'évaluation renvoie une performance recouvrant le critère et les contraintes. Dans le cas d'une optimisation multi-objectifs, chacun des objectifs doit faire l'objet d'une recherche propre par l'algorithme du simplexe, et celles-ci peuvent être parallélisées.

L'algorithme renvoie enfin la solution optimale et sa performance obtenues à l'issue des 50 pas de simplexe.

4 Développement open source dans un cadre collaboratif

Le développement de Gencab Indra s'inscrit dans un projet collaboratif afin de bénéficier de l'aide de la communauté scientifique et industrielle pour enrichir sa spécification et améliorer ses algorithmes. Un site internet lui est dorénavant dédié (hébergé dans un premier temps à l'adresse : http://www.cabinnovation.com/ingenierie/GencabIndra et une liste de membres partenaires mise à jour.

5 Conclusion

En cherchant à fédérer une communauté de chercheurs, développeurs, industriels et donneurs d'ordres pour définir un standard de formalisation des problématiques d'optimisation et développer des outils en open source pour y répondre, ce projet constitue, par ailleurs, un cadre de réflexion collective propice à la diffusion des techniques d'optimisation dans l'industrie où elles s'avèrent souvent absentes bien qu'indispensables à sa survie dans un monde globalisé.

Références

[1] L. JAEGER. *Optimisation multidisciplinaire sous incertitude en phase conceptuelle avion.* PhD thesis, 2013. Université Toulouse 3 Paul Sabatier

[2] J. CLÉMENT. *Optimisation multidisciplinaire : étude théorique et application à la conception des avions en phase d'avant-projet.* PhD thesis, 2009. Institut Supérieur de l'Aéronautique et de l'Espace

[3] D. Goldberg, Algorithmes génétiques, Addison–Wesley France (1994)

[4] R.Storn, K. Price, *Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces*, Journal of Global Optimization, December 1997, Volume 11, Issue 4, pp 341-359 - Springer

[5] M. Zaharia. *Spark: cluster computing with working sets*, Proceedings of the 2nd USENIX conference on Hot topics in cloud computing, p.10-10, June 22-25, 2010, Boston, MA

[6] S. Newman. *Building Microservices*. ISBN 978-1-4919-5035-7.

[7] A. Cabarbaye. *Outil générique d'optimisation dans le domaine discret et/ou continu éventuellement stochastique*. ROADEF'03 – Avignon, 2003.

[8] A. Cabarbaye, A. Tanguy, S. Bosse, *Adjustment of complex probabilistic models and estimation of confidence intervals in a discrete manner*, PSAM11 & ESREL 2012, 25 - 29 June 2012, Helsinki

[9] S. Bosse, A. Cabarbaye, *Modèle markovien générique de redondance avec stock de rechanges*, Qualita 2015, Nancy