THÈSE

en vue de l'obtention du

Doctorat de l'Université de Toulouse

Délivré par l'Institut National Polytechnique de Toulouse (INP Toulouse)
Discipline ou spécialité : Systèmes Industriels

Présentée et soutenue par Aurélien CODET DE BOISSE
Le 5 février 2013

Aide à la décision exploitant de la connaissance générale et contextuelle : application à la maintenance d'hélicoptères.

Rapporteurs
Jean-Yves DANTAN – Professeur, ENSAM Metz
Emmanuel CAILLAUD – Professeur Université Strasbourg

Jury
Lionel ROUCOULES – Professeur, ENSAM Aix-en-provence *(Examinateur)*
Raphaël CHENOUARD – Maître de conférences, EC Nantes *(Examinateur)*
Michel ALDANONDO – Professeur, EM Albi *(Directeur de thèse)*
Laurent GENESTE – Professeur, ENI Tarbes *(CoDirecteur de thèse)*
Élise VAREILLES – Maître-Assistant, EM Albi *(Examinateur)*
Thierry COUDERT – Maître de conférences, ENI Tarbes *(Examinateur)*

École doctorale : *Systèmes (EDSYS)*
Unité de recherche : *CGI - EM Albi, LGP - ENI Tarbes*
Directeurs de thèse : Michel ALDANONDO et Laurent GENESTE

Remerciements

Je tiens à remercier en premier lieu mes encadrants de thèse sans qui ce manuscrit n'existerait pas : Michel Aldanondo, directeur de thèse, Élise Vareilles, encadrante, Paul Gaborit, encadrant, Laurent Geneste co-directeur de thèse et Thierry Coudert, encadrant. Merci à vous cinq de m'avoir encadré, et plus important encore, de m'avoir supporté !

J'adresse aussi mes remerciements aux personnes qui ont accepté de participer à mon jury de thèse :

- Lionel Roucoules, professeur à l'ENSAM d'Aix-en-provence et président de mon jury de thèse,
- Jean-Yves Dantan et Emmanuel Caillaud, professeurs à l'ENSAM de Metz qui m'ont fait l'honneur d'accepter de rapporter ce manuscrit,
- Raphaël Chenouard, maître de conférence à l'École Centrale de Nantes et membre de mon jury de thèse.

Je tiens aussi à remercier l'ensemble des enseignants chercheurs du GI pour leur disponibilité. Fred, Matthieu, Jacques, François, Didier : merci ! Je voudrais aussi remercier Isabelle pour sa disponibilité sans faille et sa gentillesse permanente.

Merci aussi à tous les thésards avec qui j'ai pu passer du temps et avoir bon nombre de conversations enrichissantes ! Thomas, Trup, Aurélie, Guillaume, Frisouille, Christophe, Nico, Antho, Guy, Pauline, Myriam, Laura, chacun d'entre-vous m'avez apporté beaucoup ! Je voudrais aussi remercier les footeux, basketteurs, grimpeurs, volleyeurs, pongistes... avec qui j'ai partagé de très bons moments sportifs !

D'une manière globale, je souhaite adresser mes plus sincères remerciements aux Mines d'Albi, établissement dans lequel j'ai été pendant plus de 7 ans, et dans lequel je me sens désormais un peu comme chez moi...

Plus personnellement, je voudrais remercier mes parents qui m'ont toujours soutenu dans l'ensemble de mes projets, qui ont toujours cru en moi tout au long de mes études et qui continuent de croire en moi malgré ma soif de

nouvelles aventures ! Je voudrais aussi remercier mes deux grands frères qui m'ont toujours poussé à me surpasser pour qu'ils soient fiers de moi... un doctorat suffira-t-il ?

Enfin, je ne saurais jamais comment assez remercier Marina, avec qui je partage ma vie, qui m'a encouragé tout au long de ces trois ans, et qui est toujours à mes côtés, dans les moments difficiles comme les plus heureux !

Table des matières

Introduction

Le contexte industriel et économique est en mutation permanente. Les nouvelles technologies, l'instabilité des marchés, la concurrence sur les coûts, qualités et délais associés à la nécessité de s'inscrire dans un développement durable, font que les décisions sont de plus en plus délicates à prendre et ont des conséquences de plus en plus impactantes.

De manière simultanée, et illustré par les lois énoncées par [Moore 1998], les outils et systèmes informatiques deviennent de plus en plus sophistiqués. La puissance de calcul, l'ergonomie plus intuitive, les volumes de stockage de données et la rapidité d'accès à l'information permettent maintenant de disposer d'outils d'aide à la décision de plus en plus performants.

Á l'intersection des décisions toujours plus délicates et des outils d'aide à la décision toujours plus performants, nos travaux s'intéressent à l'aide à la décision exploitant de la connaissance capitalisée dans le but d'aider les activités de conception. Nous considérons deux types de connaissance, une connaissance dite générale, car valide pour un grand nombre de situations variées, et une connaissance plus contextuelle dans le sens où elle n'est exploitable que sur un petit nombre de situations spécifiques.

Le plus souvent, les travaux scientifiques en aide à la décision à base de connaissance considèrent un seul type de connaissance et proposent des modèles de représentation de ces connaissances, des systèmes de stockages et des outils aidant à la décision. Nous visons cet objectif mais souhaitons pouvoir exploiter les deux types de connaissances simultanément et en bonne complémentarité car, comme nous le verrons par la suite, la seule connaissance générale n'est pas toujours suffisante pour aider à la conception de manière adéquate.

Dans le cadre de nos travaux, nous considérons un type de formalisme pour chaque type de connaissance, permettant de représenter, stocker et exploiter la connaissance pour aider à la décision :

- en ce qui concerne la connaissance générale, nous considérerons que l'aide à la décision peut être associée à un problème de satisfaction de contraintes. L'aide à la décision est alors fournie par un mécanisme de propagation de contraintes qui élimine progressivement des mauvaises solutions et guide le concepteur vers les bonnes,
- en ce qui concerne la connaissance contextuelle, nous considérerons que la décision peut être aidée par des analogies ou comparaisons avec des situations ou cas passés et mémorisés. La connaissance est alors contenue dans ces cas, et l'aide à la décision visera à rechercher des cas similaires au problème considéré pour de même guider le concepteur.

La connaissance générale étant par définition valide sur un grand nombre de situations, notre proposition est d'utiliser cette connaissance pour aider à la décision dans un premier temps (propagation de contraintes). Si elle ne suffit pas ou s'il semble possible d'affiner l'aide à la décision, nous proposons d'exploiter en complément la connaissance contextuelle. Pour cela, nous introduirons le principe de contrainte dite contextuelle, qui est une contrainte dépendante de cas sélectionnés dans une base de cas.

Nos travaux sont illustrés sur des exemples issus de la maintenance d'hélicoptères car ces travaux de thèse se sont déroulés dans le cadre du projet Hélimaintenance (projet FUI du pôle Aerospace-Valley visant à optimiser la maintenance d'hélicoptères). Ce terrain d'application se prête parfaitement à l'exploitation simultanée de connaissance générale et contextuelle. En effet, la connaissance générale de maintenance provient de la réglementation et des documents constructeurs et la connaissance contextuelle résulte de l'enregistrement de toutes les activités effectuées par les opérateurs de maintenance.

Le manuscrit s'articule en conséquence de la manière suivante :

Dans le chapitre 1, nous présenterons l'aide à la décision, et plus particulièrement l'aide à la conception. Nous exposerons ce que recouvrent les notions de connaissance générale et contextuelle ainsi que les outils qui nous permettent d'exploiter ces deux types de connaissances.

Dans le chapitre 2, nous définirons les éléments principaux de l'exemple concernant la maintenance d'hélicoptères qui illustrera tout le mémoire. A l'aide de cet exemple, nous étudierons les travaux du domaine ayant associé connaissance générale et contextuelle. Ceci nous permettra d'affiner et de conforter notre problématique.

Dans le chapitre 3, nous décrirons notre proposition de solution : la contrainte contextuelle. Il s'agit, en fait, d'une contrainte paramétrée dont les valeurs des paramètres sont déterminées à partir d'une requête particulière

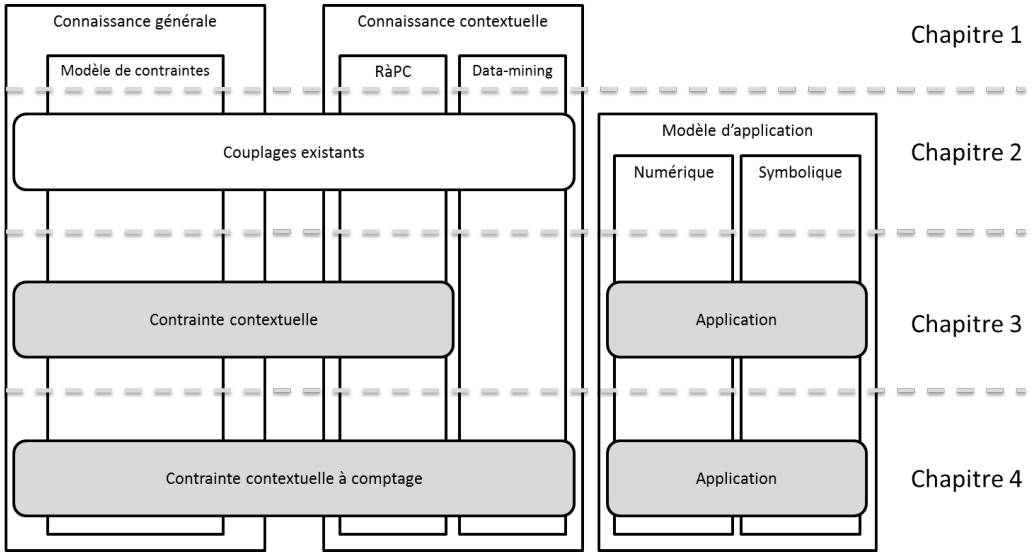


FIGURE 1 : Plan de lecture du manuscrit de thèse

dans une base de cas. Nous illustrerons cette proposition sur deux exemples, l'un portant sur une variable numérique et l'autre sur une variable symbolique.

Le chapitre 4 sera consacré à l'implémentation d'un mécanisme de comptage des valeurs possibles des paramètres. Le but étant de ne conserver que les valeurs les plus fréquentes avec l'idée qu'elles soient les plus plausibles. De même, ce mécanisme sera illustré sur les deux exemples du chapitre 3. La figure 1 schématise ce plan de lecture.

Chapitre 1

Problématique

Dans un contexte global de complexification des activités de conception [Tichkiewitch 2005], les outils informatiques d'aide à la décision sont de plus en plus sollicités [Amable 2006]. Selon [Zouari 2007] ou encore [Gomes 2008], l'utilisation d'outils d'aide à la décision de qualité est stratégique pour les entreprises. Ces outils présentent l'avantage d'utiliser le support informatique et toutes leurs performances en termes de puissance de calcul et de stockage, associés à une ergonomie toujours améliorée et une prise en main de plus en plus aisée. Par contre, leur emploi nécessite de formaliser la connaissance utilisée. Or, depuis maintenant plusieurs décennies, les entreprises conservent leurs données dans l'idée de les réutiliser par ailleurs. Cette réutilisation contribuer à créer de la connaissance [Aussenac et al. 1996].

Lors de la prise de décision en conception, l'être humain s'appuie sur une multitude de processus cognitifs très divers. Que ce soit en résolution de problèmes arithmétiques [Kintsch et al. 1985], en raisonnement s'appuyant sur l'histoire [Wineburg 1991], en pilotage de systèmes industriels [Samurçay et al. 1996] ou encore en conception de logiciels [Rouet et al. 1995], ses choix utilisent deux principaux raisonnements : le raisonnement par analogie [Py 1994] et le raisonnement déductif [Duval 1991]. Le raisonnement par analogie consiste à comparer un problème à des situations passées mémorisées, alors que le raisonnement déductif consiste à inférer des règles préétablies. En aide à la décision, la majorité des travaux proposent d'utiliser une seule de ses approches, alors que l'être humain passe de l'une à l'autre [Minsky 1974]. Nous visons en conséquence à faire fonctionner en bonne complémentarité deux mécanismes d'aide à la décision, l'un fonctionnant par analogie et le second inférant des règles ou des contraintes, afin de s'approcher au maximum du processus de raisonnement humain. En effet, le raisonnement humain s'adapte avec agilité à des sources ou types de connaissances les plus diverses dont la quantité et la qualité peuvent varier.

L'ensemble de nos propositions est illustré par des exemples portant sur la maintenance d'hélicoptères. En effet, nos travaux de thèse se sont déroulés dans le cadre du projet Hélimaintenance qui mobilise des laboratoires universitaires de différentes institutions (ISAE, ONERA, EMAC) et des entreprises de la région Midi-Pyrénées (IXAIRCO, C3EM, SEMIA). Le but du projet Hélimaintenance est l'étude de méthodes permettant une réduction des coûts de maintenance des hélicoptères :

- en amont, lors de la phase de planification, afin d'estimer au mieux l'ensemble des opérations de maintenance à effectuer et la durée d'immobilisation des aéronefs,
- en aval, lors de la réalisation du plan de maintenance, par l'exploitation de connaissances issues du terrain permettant une meilleure adéquation entre le plan, les conditions de maintenance (ressources, matériels, imprévus) et l'état réel de l'appareil (corrosion, pannes, vieillesse).

Dans ce cadre, nos travaux ont pour objectif de proposer la mise en place d'un outil d'aide à la conception de processus de maintenance. En effet, la connaissance issue des constructeurs et de la réglementation permet d'alimenter un raisonnement déductif, alors que la connaissance issue des opérations de maintenance permet d'alimenter quant à elle un raisonnement par analogie.

Dans un premier temps, nous situerons la problématique de nos travaux. Puis nous caractériserons ce que nous appelons « connaissance » et présenterons les approches utilisées tout au long de ce mémoire pour capitaliser et exploiter la connaissance. Un plan de lecture terminera ce premier chapitre.

1.1 Situation de la problématique

Dans cette section, nous allons commencer par introduire les notions de conception et d'aide à la conception. Ensuite, nous verrons qu'il existe différentes approches pour aider à la conception et comment celles-ci exploitent de la connaissance pour aider à la prise de décision. Enfin, nous préciserons notre objectif qui est d'utiliser deux types de connaissances au sein d'un même outil d'aide à la conception.

1.1.1 Conception

Les travaux et approches concernant l'activité de conception relèvent de plusieurs écoles de pensée depuis plus de quatre décennies : [Gero 1990], [Pahl

et al. 1996], [Suh 1998], [Hubka et al. 1988] ou plus récemment [Ulrich et al. 2011]. La conception peut concerner un produit, un service ou un processus et consiste à définir un objet conformément à des spécifications préalablement explicitées. Selon [Lonchampt 2004], concevoir consiste à résoudre un problème incomplet et insuffisamment défini au départ. Les contraintes étant prises en compte au fil du temps réduisent progressivement l'espace de solution. Selon [Pahl et al. 1996], la démarche de conception comporte trois principales étapes :

1. repérer, dans les spécifications, les fonctionnalités attendues pour l'objet à concevoir,
2. étudier, parmi les possibilités, les solutions de conception conformes aux spécifications,
3. évaluer les solutions proposées et sélectionner les plus pertinentes quant à la conception à venir.

La démarche de conception mobilise un certain nombre de connaissances. Selon [Chandrasekaran 1990], il existe trois types de connaissances en conception :

La connaissance de l'objectif qui correspond aux fonctionnalités de l'objet à concevoir. Cela signifie que pour un objet donné, le concepteur a déjà été en situation de concevoir des objets ayant des fonctionnalités similaires.

La connaissance du domaine qui correspond à l'environnement de l'objet et aux technologies à déployer pour concevoir celui-ci. Cela signifie que le concepteur a déjà été en situation de concevoir un objet dans le même environnement et les mêmes conditions technologiques.

La connaissance de la démarche qui correspond au savoir-faire. Cela signifie que le concepteur a déjà conçu cet objet et connait donc la démarche de conception. Ce type de connaissance est directement lié à la culture des entreprises et fortement dépendant du retour d'expérience de ses employés.

[Chandrasekaran 1990] propose de typer les activités de conception en fonction de la disponibilité des connaissances :

- lorsque seule la connaissance sur l'objectif (fonctionnalités attendues du produit) est disponible, nous parlons de conception créative [Huot 2005]. La connaissance sur le domaine et sur la démarche est alors apportée par le concepteur,
- lorsque la connaissance sur l'objectif ainsi que celle sur le domaine sont disponibles, nous parlons de conception innovante [Le Masson et al. 2006]. Seule la démarche de conception est apportée par le concepteur,

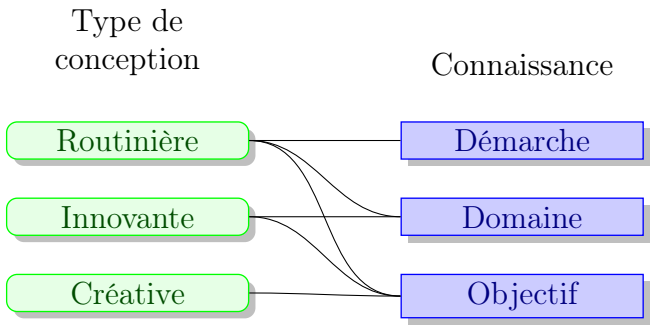


FIGURE 1.1 : Conception ⇔ connaissance

– enfin, lorsque l'ensemble des connaissances est disponible (objectif, domaine et démarche), nous parlons de conception routinière [Toussaint 2010]. Dans cette situation, les concepteurs ont une certaine habitude et expérience sur la démarche de conception d'une même famille d'objets.

La figure 1.1 résume les différents types de conceptions en fonction des connaissances disponibles.

Dans nos travaux, nous nous plaçons dans un contexte de conception routinière car il s'agit d'opérations de maintenance récurrentes. De ce fait, outre la connaissance fournie par le constructeur et la réglementation, de la connaissance opérationnelle peut être accumulée. En conséquence, les trois types de connaissances sont disponibles, soit au sein des manuels d'entretien fournis par chaque constructeur pour chaque famille d'aéronef (connaissance de l'objectif et du domaine), soit auprès des maintenanciers (connaissance de la démarche) qui ont l'habitude de maintenir les aéronefs.

1.1.2 Aide à la décision en conception

De même que la conception, l'aide à la décision est un domaine dans lequel de nombreux travaux ont lieu depuis près de quarante ans. Les premiers travaux identifiés datent de près de trente ans, avec entre autres [Giard et al. 1985] en génie industriel ou encore [Eppinger 1991] en conception. Les recherches en aide à la décision sont toujours très actives de nos jours, avec par exemple des recherches sur la robustesse des systèmes d'aide à la décision [Aissi et al. 2010] ou encore l'utilisation d'aide à la décision en situations d'urgence [Pfaff et al. 2010]. L'un des domaines d'application qui utilise le plus ces travaux est la finance comme le montre la revue bibliographique proposée par [Steuer et al. 2003].

[Zaraté 2005] nous donne une définition de l'aide à la décision basée sur celle de [Roy 1996] comme étant « *l'activité de celui qui, prenant appui sur*

des modèles clairement explicités et plus ou moins complètement formalisés, cherche à obtenir des éléments de réponses aux questions que se pose un intervenant dans un processus de décision, éléments concourant à éclairer la décision et normalement à prescrire un comportement de nature à accroître la cohérence entre l'évolution du processus d'une part, les objectifs et le système de valeurs au service duquel cet intervenant se trouve placé d'autre part. »

Selon [Bensana 1998], « *modéliser un problème de conception revient à identifier quels sont les choix de conception à effectuer, c'est à dire les degrés de liberté dont dispose le concepteur et établir les relations qui lient ces choix entre eux ou avec les spécifications et qui définissent le cadre dans lequel doit s'effectuer la conception.* »

Concevoir un produit consiste à apporter une ou plusieurs solutions à un ou plusieurs besoins précédemment explicités. Un système d'aide à la conception peut alors être vu comme une application d'un système d'aide à la décision, dans lequel la décision portera sur les choix de conception à effectuer.

Les outils d'aide à la conception permettent avant tout d'affranchir le concepteur de tâches répétitives et de répercuter automatiquement des décisions sur l'ensemble de la conception en cours [Gomes 2008]. Le concepteur peut alors se focaliser sur les décisions qui requièrent une mobilisation de connaissances spécifiques difficiles à formaliser et à exploiter dans un outil informatique. En effet, selon [Prasad 1996], 80% du temps passé en entreprise l'est à effectuer des tâches routinières. Nous allons donc proposer des outils permettant de diminuer ce ratio et ainsi laisser plus de temps à des tâches créatives.

Dans nos travaux, nous nous intéressons à l'aide à la conception interactive qui consiste à guider l'utilisateur au fil de ses choix et à converger progressivement vers une solution. Cela revient donc, comme l'a défini [Lonchampt 2004], à prendre en compte, au fur et à mesure des apports d'exigences équivalentes aux besoins dans le processus de conception et ainsi compléter un problème de conception initialement incomplet [Lévine et al. 1989].

1.1.3 Aide à la conception et connaissances

Afin de mettre en œuvre des outils d'aide à la conception, l'utilisation de connaissances est indispensable. Pour cela, il est nécessaire de classer la connaissance dont nous disposons en fonction de sa nature. Dans nos travaux applicatifs sur la maintenance d'hélicoptères, nous disposons de deux connaissances distinctes :

- la connaissance issue des documentations constructeurs. C'est une connaissance décrivant des procédures ou des manières de faire nécessairement respectée afin de conserver le certificat de navigabilité de l'appareil. Cette connaissance est formalisée dans les documentations constructeurs et les réglementations,
- la connaissance issue des expériences des maintenanciers. C'est une connaissance qui s'enrichit au fil du temps. Cette connaissance est enregistrée dans des bases de données sous forme de description de situations s'apparentant à des cas.

Historiquement, la classification proposée par [Polanyi 1958] et [Polanyi et al. 1966], puis reprise par [Nonaka 1994] consiste à différencier connaissance explicite et connaissance tacite.

La connaissance explicite est une connaissance qui peut être formulée, écrite, reproduite et redémontrée, c'est ce que l'on appelle le *savoir*. Nous pouvons prendre comme exemple de savoir, la définition du poids d'un objet P qui correspond au produit de sa masse m par l'accélération de la pesanteur g, soit $P = m.g$.

La connaissance tacite quant à elle, n'est pas encore formulée ou non formulable. Elle correspond souvent à ce que l'on appelle le *savoir-faire*. Par exemple, la différence de temps passé à la maintenance d'un moteur d'hélicoptère entre un maintenancier expérimenté et un maintenancier débutant, bien que le manuel de maintenance suivi soit le même, est principalement due à leur différence de savoir-faire et aux connaissances tacites mobilisées.

Pour nos travaux, nous ne considérons que la connaissance explicite car les connaissances issues des expériences des maintenanciers sont enregistrées dans une base de données. Les deux connaissances dont nous disposons sont alors explicites. De nombreux ouvrages proposent des classifications de connaissances autres que celle initialement introduite par [Polanyi 1958]. Nous avons pu identifier une différenciation entre connaissances « savoir-faire » et connaissances « compétences » proposée par [Grundstein 2000]. Le savoir-faire correspond au fait d'avoir appris une procédure, alors que la compétence demande en plus de cela de l'expérience dans l'application de cette procédure. En 2000, [Eteläpelto 2000] proposait le principe de contextualité de la connaissance, confirmé ensuite par [Pomerol et al. 2001] qui introduisait la notion de profondeur de la connaissance (connaissances profondes en oppositions à connaissances de surface). [Montani 2011] synthétise cela en formalisant le principe de connaissance contextuelle et connaissance générale lors d'une application en médecine. C'est en se basant sur ses travaux que nous classifierons alors la connaissance que nous exploiterons :

La connaissance générale

Il s'agit d'une connaissance applicable dans tous les cas de figure, quel que soit le contexte. L'exemple précédent $P = m.g$ est donc une connaissance générale car, quel que soit le contexte, cette formule est vérifiée. La connaissance issue de la documentation constructeur des aéronefs ainsi que la réglementation sera ainsi exploitée comme une connaissance générale.

La connaissance contextuelle

C'est une connaissance qui dépend du contexte. Si nous reprenons l'exemple précédent en nous plaçant dans un référentiel ou contexte terrestre, la valeur de l'accélération normale de la pesanteur sur Terre est approximée à $9,81 m.s^{-1}$. La connaissance $P = m.9,81 m.s^{-1}$ est donc contextualisée au contexte terrestre. La connaissance issue des expériences des maintenanciers sera alors exploitée comme une connaissance contextuelle.

1.1.4 Systèmes à base de connaissances

Les synthèses sur les systèmes à base de connaissances sont peu fréquentes, nous pouvons néanmoins citer [Leondes 2000]. Afin d'exploiter les connaissances pour l'aide à la conception, celles-ci doivent être enregistrées dans des bases de connaissances [Vancza 1999], [Eppinger 1991]. Un moteur de traitement va utiliser cette base de connaissance afin d'en extraire des informations pouvant être interprétées par l'humain [Elkan et al. 1993]. Le terme de système à base de connaissances regroupe donc une base de connaissances ainsi qu'un moteur de traitement [Le Ber et al. 2006]. La figure 1.2 page suivante représente l'environnement d'un système à base de connaissances, ici utilisé afin de résoudre un problème. Dans notre problématique, cela s'apparente à un outil d'aide à la conception assisté par ordinateur. [Chapman et al. 1999] montre que les systèmes à base de connaissance peuvent contribuer à l'amélioration des systèmes de conception assistée par ordinateur, ce constat est également confirmé par [Trousse 1989] et [Vargas 1995].

Les systèmes à base de connaissances sont utilisés dans divers domaines tels que la cuisine [Blansché et al. 2010], la médecine [Holt et al. 2005], la détection de fraudes [Phua et al. 2010], la gestion de l'hospitalisation à domicile [Costa Filho et al. 2012], la conception automobile [Pargamin 2002] où encore la construction aéronautique [Djefel 2010].

De manière générale, le monde réel apporte des problèmes et des solutions. Ceux-ci sont analysés par un expert pour alimenter la base de connaissances du système. Le moteur de traitement utilise la base de connaissances pour apporter des solutions à l'utilisateur qui lui expose ses problèmes.

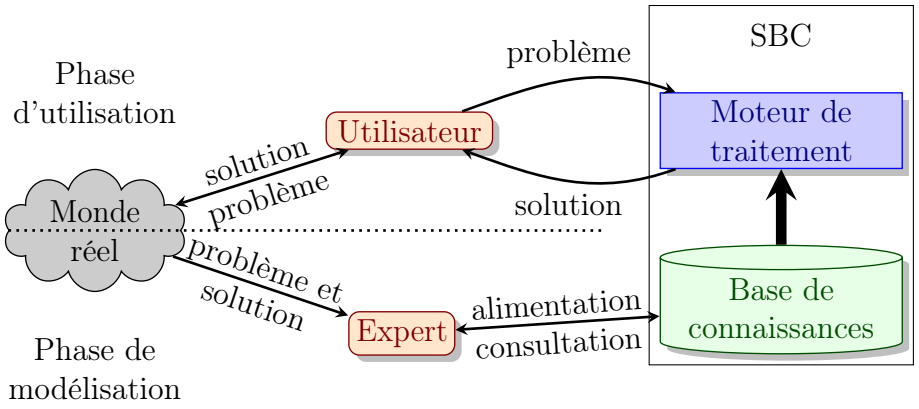


FIGURE 1.2 : Environnement d'un Système à Base de Connaissances

1.1.5 Premier cadrage de la problématique : vers une utilisation conjointe des deux types de connaissances pour de l'aide à la conception

Il est possible de faire un parallèle entre le raisonnement humain et le fonctionnement d'un système à base de connaissances. Comme l'ont proposé [Purvis et al. 1995] et [Didierjean 2001], la prise de décision humaine se base sur deux principales sources de connaissances : les cas passés pour trouver une analogie entre la décision à prendre et des décisions passées (connaissance plutôt contextuelle), et les règles établies ou créées au fil du temps (connaissance plutôt générale). Selon [Lieber 2008], la plupart des systèmes à base de connaissance n'exploite qu'une seule de ces deux sources de connaissances. Dans son mémoire d'habilitation à diriger la recherche, [Lieber 2008] présente le recours à la connaissance contextuelle comme « *une alternative au raisonnement utilisant des règles, coûteux en terme d'acquisition de connaissances* ». Nous nous inscrivons dans cette idée, et proposons d'exploiter connaissances contextuelles et connaissances générales conjointement et non pas en tant que solutions alternatives.

Au niveau de notre terrain d'application, la base de connaissances générale est relativement stable : les manuels d'entretien des aéronefs édités par les constructeurs ainsi que les réglementations, garantissant leur navigabilité et définissant les plans de maintenance, n'évoluent que très rarement. Par contre, la base de connaissances contextuelle s'enrichit régulièrement au fur et à mesure de la réalisation des activités de maintenances : les informations clés concernant chaque intervention de maintenance sont stockées afin d'être exploitées lors de la phase de planification et celle de réalisation pour estimer, au plus juste, la charge de maintenance.

Selon [Kolodner 1993] et [Leondes 2000], les systèmes à base de connaissances basés sur de la connaissance générale demandent un effort de conception plus important que ceux basés sur de la connaissance contextuelle : une phase d'extraction et de formalisation des connaissances est nécessaire (relations, définition des domaines, etc.). D'un point de vue de l'utilisateur, les systèmes à base de connaissances exploitant la connaissance générale ne nécessitent que peu d'apports de connaissances lors de la prise de décision. De plus, par définition, la connaissance générale s'applique dans tous les cas de figure alors que la connaissance contextuelle est limitée aux seules expériences passées.

Dans nos travaux, nous utilisons en bonne complémentarité de la connaissance générale et de la connaissance contextuelle dans un système à base de connaissance afin d'aider à la conception.

1.2 Caractérisation des connaissances

Afin de clarifier nos propos, il est nécessaire de préciser le lien entre données et connaissances. Il existe de nombreuses définition de ces termes et des « étapes » intermédiaires entre données et connaissance, tels que proposés dans [Jarboe et al. 2001] ou [Prax 1997]. Nous avons choisi de nous baser sur la synthèse effectuée dans [Beler 2008] qui propose un résumé très clair de ce lien. La figure 1.3 page suivante représente le lien logique entre données et connaissances, appelé le processus d'apprentissage. Les éléments de ce processus sont :

la donnée qui représente des éléments à l'état brut, non reliés entre eux, atomiques. Les données n'ont aucune valeur informative du fait de leur exclusion du contexte,

l'information qui représente un ensemble de données reliées entre-elles. Les données sont agrégées de manière à pouvoir en extraire une forme de savoir,

l'expérience qui est un ensemble d'informations sur une situation que l'on peut situer entre l'information et la connaissance,

la connaissance qui est une information à forte valeur ajoutée, permettant ainsi de générer de nouvelles informations à partir de celle-ci,

le savoir-faire qui consiste à savoir réutiliser de la connaissance pour résoudre de nouveaux problèmes. On parle de savoir-faire lorsque l'humain s'approprie des connaissances.

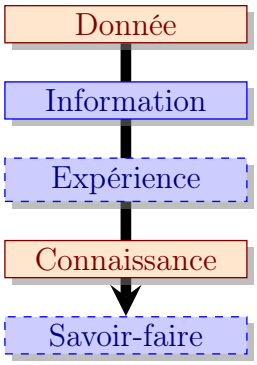


FIGURE 1.3 : Représentation du processus d'apprentissage

Pour représenter les données, nous allons utiliser des variables. Il est utile de différencier les termes de données et de variables. Une variable est une donnée qui n'a pas été valuée. Par exemple, la température est une variable, alors que la donnée est 100. Une donnée est donc l'instanciation d'une variable.

Un problème de conception se décrit en valuant les variables définissant des besoins et des attentes. La solution d'un problème est décrite par des variables valuées définissant des composants et des propriétés, comme l'a proposé [Suh 1998] dans sa théorie de la conception axiomatique des systèmes.

Dans cette section, nous allons commencer par caractériser les variables et les données. Par la suite, comme nous utilisons ces données dans des bases de connaissance, nous allons caractériser les différentes manières de formaliser des connaissances.

1.2.1 Variables et données

1.2.1.1 Types de variables

Les variables utilisées peuvent être continues ou symboliques. Des données continues sont des nombres, leur domaine de définition est défini comme un intervalle de valeurs : *A est compris entre 10 et 20*. À l'inverse, le domaine de définition des données symboliques peut être listé exhaustivement : *A fait partie de la liste {Puma, Gazelle, Dauphin, Tigre}*. Les variables numériques discrétisées peuvent être traitées comme des données symboliques car il est possible de lister l'ensemble de leurs valeurs.

1.2.1.2 Absence d'une donnée dans une base de connaissance

Il peut y avoir deux raisons à l'absence d'une donnée dans une base de connaissance. Nous avons identifié une première raison de son absence par manque d'informations : nous parlons alors de donnée *indéfinie*. Cela peut être dû à l'oubli du relevé de la valeur, la perte de la donnée ou le défaut d'un capteur.

Comme l'a indiqué [Van Oudenhove de Saint Géry 2006], la seconde raison qui peut aboutir à l'absence d'une donnée est la non pertinence de celle-ci dans un contexte particulier. Si nous prenons par exemple la description générique d'un moteur d'hélicoptère, celui-ci peut alors être à explosion ou à turbine. Si nous souhaitons enregistrer la cylindrée du moteur, cette donnée ne sera pas pertinente dans le cas d'un moteur à turbine mais seulement dans le cas d'un moteur à explosion. Nous parlons alors de donnée *inutile* dans ce cas de figure.

1.2.2 Connaissance formalisée par des cas

Nous venons de caractériser les variables supports des bases de données, nous pouvons maintenant aborder différentes manières de formaliser la connaissance afin d'effectuer de l'aide à la conception interactive. Nous nous situons en aide à la conception de processus de maintenance d'hélicoptères.

1.2.2.1 Description

Dans cette section, nous nous intéressons au stockage des couples problèmes/solutions sous forme de cas. Selon [Kolodner 1993] ainsi que [Fuchs 2008], un cas $\mathcal{C}$ est défini par un ensemble de variables valuées $\mathcal{V}$ (aussi appelées selon les références descripteurs ou attributs) dont on a, auparavant, fixé le domaine de définition $\mathbb{D}_{\mathcal{V}}$. L'ensemble des variables d'un cas permet de décrire le problème rencontré et la solution proposée. Lorsqu'un cas est stocké en base, chaque variable de ce cas doit être renseignée. La base de données permettant de stocker des cas est appelée une base de cas.

L'exploitation d'une base de cas pour en extraire des informations peut être faite par divers outils tels que le raisonnement à partir de cas (que nous présentons en section 1.3.1 page 24) ou le *data-mining* (ou exploration de données que nous présentons en section 1.3.2 page 26).

Prenons un exemple simple qui consiste à enregistrer l'utilisation, le type, la puissance et la fréquence de panne d'un hélicoptère. Les variables ainsi que leurs domaines sont les suivantes :

	Problème			Solution
cas	Utilisation	Type	Puissance	Fréquence de panne
1	Secours	Puma	500	Faible
2	Militaire	Puma	500	Moyen
3	Militaire	Gazelle	100	Fort

TABLE 1.1 : Exemple de base de cas

- Utilisation, $\mathbb{D}_{\text{utilisation}} = \{Secours, Militaire, Television, indéfini\}$
- Type, $\mathbb{D}_{\text{type}} = \{Puma, Gazelle, Dauphin, indéfini\}$
- Puissance, $\mathbb{D}_{\text{puissance}} = \{[100; 500], indéfini\}$
- Panne, $\mathbb{D}_{\text{panne}} = \{Faible, Moyen, Fort, indéfini\}$

Nous pouvons remarquer que la valeur *indéfini* est disponible pour chaque variable, ce qui autorise l'absence de la donnée. Il n'existe par contre aucune variable qui ait pour valeur possible la valeur *inutile* car toutes les variables sont pertinentes dans cet exemple. La table 1.1 est un exemple de base de cas. Nous pouvons y voir les variables qui sont l'utilisation, le type, la puissance et la fréquence de panne de l'appareil. Les cas sont décrits par chaque ligne, et les données correspondent aux valeurs enregistrées pour l'ensemble des variables.

Nous considérons dans nos travaux qu'une collection de cas représente de la connaissance contextuelle. En effet, l'ensemble du contexte est nécessaire pour que chaque donnée soit exploitable. Il n'est pas possible de générer automatiquement cette connaissance. Dans notre situation, chaque cas représente un contexte de maintenance différent.

1.2.2.2 Couverture de l'espace de solutions par une base de cas

Pour caractériser une base de cas, nous introduisons ce que nous appelons sa couverture : cela correspond à l'espace couvert par les cas existants sur l'espace couvert par les cas possibles, soit l'espace de solutions. La couverture permet donc d'estimer la quantité de connaissance contenue dans une base de cas. Plus une base de cas a un taux de couverture élevé, plus celle-ci est intéressante à exploiter pour aider à la prise de décision.

Nous considérons trois ensembles distincts, tel que présenté sur la figure 1.4 page suivante :

- $\mathcal{T}$ le cardinal de l'ensemble des domaines de définition du modèle courant,

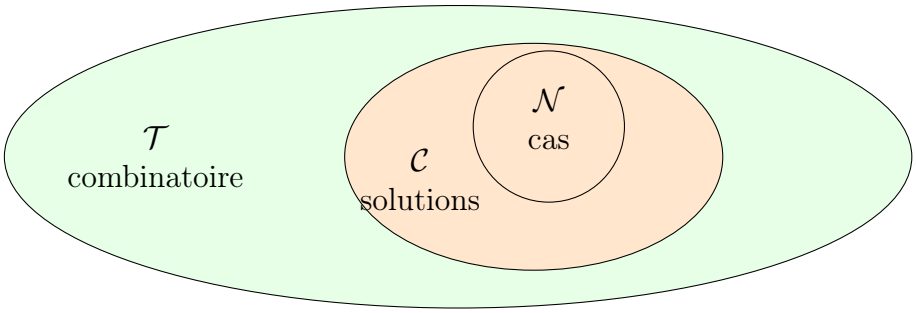


FIGURE 1.4 : Représentation de l'espace de solutions.

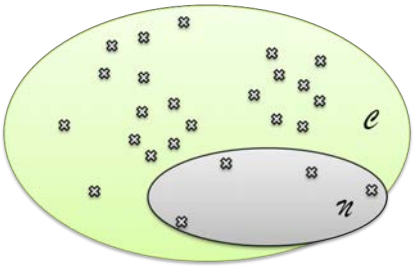
- $\mathcal{C}$ le nombre de cas possibles ou solutions atteignables dans le modèle courant. Ce nombre de cas possibles n'est pas connu à l'avance, mais est obligatoirement inférieur ou égal à $\mathcal{T}$,
- $\mathcal{N}$ le nombre de cas distincts présents dans la base de cas.

Nous voyons que $\mathcal{N}$ est inclus dans $\mathcal{C}$ qui lui-même est inclus dans $\mathcal{T}$.

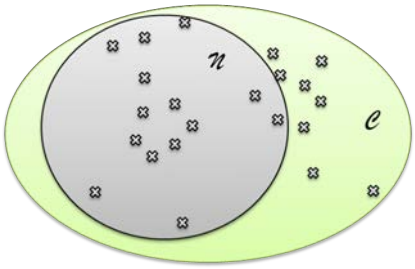
Nous définissons la couverture $\mathcal{R}_{cas}$ comme le rapport entre $\mathcal{N}$ et $\mathcal{C}$. La principale difficulté de ce calcul est que le nombre de cas possibles ou solutions atteignables ne peut pas être connu à l'avance. En effet, les variables d'un problème de conception ne sont pas totalement indépendantes, alors $\mathcal{C} \leq \mathcal{T}$. Nous proposons donc de confondre le nombre de cas possibles $\mathcal{C}$ avec la combinatoire des variables de la base de cas $\mathcal{T}$, $\mathcal{C} = \mathcal{T}$, le cardinal de cet ensemble étant toujours supérieur ou égal au nombre de cas possibles. Plus la base de cas contient de cas différents, plus la couverture va tendre vers une asymptote correspondant à la couverture des cas possibles. Il est possible de généraliser ce principe avec les variables continues en discrétisant leur domaine. Il revient alors à l'expert de décider du pas de discrétisation.

Exemple

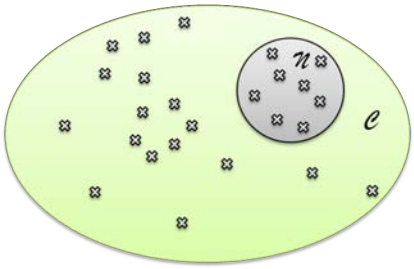
Si nous reprenons l'exemple donné en table 1.1 page ci-contre, le cardinal de $\mathcal{T}$ est de $3 * 3 * 3 * 2 = 54$. Admettons qu'une règle empêche que les hélicoptères de type Dauphin soient utilisés dans un contexte militaire. La couverture des cas possibles $card(\mathcal{C})$ est donc de $54 - 6 = 48$ (l'ensemble de la combinatoire à laquelle on retire {Dauphin, Militaire, {Oui, Non}, {Faible, Moyen, Fort}}. Cela signifie que pour couvrir l'intégralité des cas possibles, il faut 48 cas distincts. La base de cas représentée en table 1.1 page précédente contient 3 cas distincts, $card(\mathcal{N})$. Si nous ignorons la règle précédemment citée, nous trouvons comme valeur de couverture de la base de cas : $\mathcal{R} = \frac{\mathcal{N}}{\mathcal{T}} = \frac{3}{54} = 5.6\%$. Lorsque la base se remplira, la couverture



(a) Base de cas peu couvrante



(b) Base de cas couvrante



(c) Base de cas localement couvrante

FIGURE 1.5 : Exemples de bases de cas

> *augmentera, jusqu'à 48 cas distincts, état à laquelle, malgré les ajouts de nouveaux cas, la couverture stagnera alors à* $\mathcal{R} = \frac{48}{54} = 88.9\%$.

Nous avons représenté sur la figure 1.5 trois bases de cas qui s'appliquent sur une même répartition de l'espace de solutions $\mathcal{C}$. Chaque croix indique une solution possible au problème posé. Nous pouvons remarquer que la base de cas représentée en figure 1.5a est peu couvrante car elle contient peu de cas solutions. La base de cas représentée en figure 1.5b est bien couvrante car composée de nombreux cas appartenant à l'ensemble des solutions. La base de cas représentée en figure 1.5c est localement couvrante car bien centrée sur une zone de l'espace de solution, cela signifie qu'elle couvre correctement certains contextes de solutions.

1.2.2.3 Caractérisation de la connaissance formalisée par des cas

A l'enregistrement ou à l'analyse d'un cas, il est important de s'interroger sur sa complétude et sur sa validité. Un cas est complet lorsque l'ensemble des valeurs des variables qui le composent sont soit définies, soit inutiles. Il est alors possible de calculer la complétude d'un cas comme le rapport du nombre de variables définies sur le nombre total de variables. Un cas possédant 6 variables et dont seulement trois sont définies (ou inutiles) a une

complétude de 50%. Un cas est estimé comme valide lorsqu'un processus de validation (humain ou informatique) l'a validé comme étant un cas relatant une expérience passée (qui peut avoir été un échec ou une réussite) pouvant aider à la prise de décision dans le futur.

Nous pouvons maintenant définir la complétude d'une base de cas comme étant la moyenne de la complétude des cas qui la composent. Une base de cas est dite valide lorsque l'ensemble des cas qui la composent sont valides.

Dans le cadre de nos travaux, nous parlerons de confiance en la connaissance contextuelle supportée par une base de cas. Cette notion de confiance est directement liée aux notions de couverture, de complétude et de validité des bases de cas. Nous accorderons plus de confiance à une base de cas bien couvrante, complète et valide, qu'à une base de cas peu couvrante, incomplète ou partiellement validée.

1.2.3 Connaissance formalisée par des modèles

1.2.3.1 Description

Nous entendons par modèle un ensemble de variables et de relations entre ces variables. Nous verrons en section 1.3.3 page 28 que nous avons choisi d'utiliser un ensemble de contraintes pour modéliser les relations entre les variables.

Si nous reprenons l'exemple précédemment utilisé, nous pouvons expliciter la relation qui empêche aux hélicoptères de type Dauphin d'être militaires par la contrainte logique suivante : $Type = \text{Dauphin} \Rightarrow Utilisation \neq \text{Militaire}$.

Formaliser de la connaissance sous forme de modèle signifie que l'on dispose de connaissances sur les variables et des liens entre ces variables. Dans le cadre de nos travaux, la connaissance générale, issue des manuels d'entretien édités par les constructeurs, est formalisée à l'aide d'un modèle.

1.2.3.2 Couverture de l'espace de non-solution par un modèle

De même que défini en section 1.2.2.2 page 16, le principe de couverture est très important lorsque l'on formalise de la connaissance sous forme de modèle. Nous parlons alors de couverture du modèle : il correspond à l'ensemble des solutions impossibles qui ont été d'ores et déjà supprimées. Soit :

– $\mathcal{T}$ la cardinalité de l'ensemble des domaines de définition du modèle courant,

- $\overline{\mathcal{C}}$ le nombre de combinaisons impossibles. Ce nombre de combinaisons impossibles n'est pas connu à l'avance, mais est obligatoirement inférieur ou égal à $\mathcal{T}$,
- $\overline{\mathcal{N}}$ le nombre de combinaisons interdites par le modèle courant. $\overline{\mathcal{N}}$ est inclut dans $\overline{\mathcal{C}}$.

Nous définissons la couverture, notée $\mathcal{R}_{modele} = \frac{\overline{\mathcal{N}}}{\overline{\mathcal{C}}}$. Il est donc possible de faire le parallèle avec la couverture des cas. Nous pouvons alors remarquer que $\mathcal{T} = \overline{\mathcal{C}} + \mathcal{C}$. De même que pour la couverture des cas, le nombre de combinaisons impossibles ne peut pas être connu d'avance. En effet, les relations sont rarement toutes connues. Nous proposons donc de confondre $\overline{\mathcal{C}}$ avec $\mathcal{T}$, $\overline{\mathcal{C}} = \mathcal{T}$. Plus le nombre de relations formalisées augmente, plus la couverture va tendre vers une asymptote correspondant au maximum de combinaisons impossibles. Il est bien entendu possible de généraliser ce principe avec les variables continues en discrétisant leur domaines. Il revient alors à l'expert de décider du pas de discrétisation.

Exemple

Prenons un modèle simple composé de trois variables ayant chacune deux valeurs possibles :

- *Utilisation = {Militaire, Civil}*
- *Type = {Puma, Gazelle}*
- *Puissance = {500, 200}*

Le produit cartésien de l'ensemble des domaines de définition des variables du modèle donne huit combinaisons possibles. Dans la réalité, il existe une relation qui impose qu'une gazelle est forcément civile. Ces deux relations diminuent alors le nombre de combinaisons possibles à cinq, ce qui correspond aux huit combinaisons auxquelles on retire {Militaire, Gazelle, 500}, {Militaire, Gazelle, 200} et {Civil, Gazelle, 500}. Tant que l'expert ne connait aucune relation, il a donc $\overline{\mathcal{N}} = 0$*, alors que* $\overline{\mathcal{C}} = 8$*, donc* $\mathcal{R}_{modele} = \frac{0}{8} = 0$*. Lorsqu'il aura analysé le modèle et déduit la première relation (une gazelle est forcément civile), il aura une valeur de* $\overline{\mathcal{N}} = 2$ *correspondant aux combinaisons {Militaire, Gazelle, 500} et {Militaire, Gazelle, 200}. La couverture de son modèle sera donc de* $\mathcal{R}_{modele} = \frac{2}{8} = 25\%$.

Afin de faire le parallèle entre la couverture d'une base de cas et la couverture d'un modèle, nous pouvons dire que, face à un nombre de combinaisons $\mathcal{T}$, le peuplement de la base de cas permet d'augmenter la valeur de $\mathcal{N}$ alors que l'écriture de relations permet d'augmenter la valeur de $\overline{\mathcal{N}}$. L'idéal dans la modélisation d'un système est d'aboutir à $\mathcal{T} = \overline{\mathcal{N}} + \mathcal{N}$.

Formalisme	Avantage	Inconvénient
Cas	Évolutif	Contextuel
Modèles	Général	Stable

TABLE 1.2 : Comparatif entre la formalisation sous forme de cas et sous forme de modèles

1.2.3.3 Caractérisation de la connaissance formalisée sous forme de modèle

De manière similaire à la caractérisation de la connaissance formalisée par des cas, nous pouvons caractériser la connaissance formalisée sous forme de modèle par sa complétude et sa validité. Dire qu'un modèle n'est pas complet consiste à dire que certaines relations ne sont pas encore formalisées. Dire qu'un modèle n'est pas valide consiste à dire que l'ensemble des relations liant les variables interdisent des combinaisons de valeurs qui devraient être autorisées, ou en autorisent qui devraient être interdites.

Dans le cadre de nos travaux, nous parlerons de confiance en la connaissance générale supportée par un modèle. Cette notion de confiance est directement liée aux notions de couverture, de complétude et de validité des modèles utilisés. Nous accorderons plus de confiance à un modèle bien couvrant, complet et valide, qu'à un modèle peu couvrant, incomplet ou partiellement validé

1.2.4 Second cadrage de la problématique : association de deux formalismes exploitant de la connaissance

Les deux formalismes que nous avons présentés (le formalisme sous forme de cas (section 1.2.2 page 15) et le formalisme sous forme de modèles (section 1.2.3)) ont chacun leurs avantages et leurs inconvénients. Le tableau 1.2 résume les avantages et inconvénients que nous allons présenter ci-dessous.

Pour la formalisation sous forme de cas :

– son avantage majeur est son évolutivité vis-à-vis de l'acquisition de nouvelles connaissances contextuelles [Silva Garza et al. 1996], [Fuchs 2008]. Cela signifie que toute modification d'une base de cas (ajout, modification ou suppression de cas valides) est directement répercutée via le moteur de traitement qui l'exploite. Cela permet, dans notre situation, d'utiliser le formalisme sous forme de cas dès les premières acquisitions de connaissances,

– le pendant de l'évolutivité du formalisme sous forme de cas est sa contextualité. Chaque cas étant très dépendant du contexte dans lequel il s'est réalisé, un effort supplémentaire d'ajout de connaissance par l'utilisateur est souvent nécessaire pour adapter la solution d'un cas à un problème courant, ayant un contexte similaire (proche mais différent). Il s'agit là de la principale source de travaux réalisés sur la raisonnement à partir de cas ([Li et al. 2009], [Minor et al. 2010] ou encore [Sharaf Eldeen et al. 2012]).

Pour la formalisation sous forme de modèle :

– son avantage majeur est le fait que la connaissance stockée est plus générale que dans un formalisme sous forme de cas (livres, procédures, processus, instructions, etc.). Cela signifie qu'elle est utilisable dans un plus grand nombre de situations. Cela permet, dans notre problématique, d'utiliser cette connaissance dans des situations qui n'ont pas encore été rencontrées,
– l'obligation de définir de manière formelle les relations entre les variables d'un modèle a pour principal inconvénient de rendre celui-ci plus difficilement évolutif [Lieber 2008]. Cela signifie que malgré l'acquisition de nouvelles connaissances, il faut attendre une nouvelle actualisation des connaissances (utilisant par exemple du *data-mining* [Han et al. 2006]) pour mettre à jour le modèle (identification, définition formelle, validation, etc.).

Dans le cadre de nos travaux, nous avons vu dans les sections précédentes que nous souhaitons utiliser simultanément de la connaissance contextuelle formalisée sous forme de cas et de la connaissance générale formalisée sous forme de modèles pour aider à la conception routinière de processus de maintenance. Dans un premier temps, l'unique connaissance utilisée sera la connaissance générale issue des constructeur, des réglementations ou autres règles imposées. Au fil de l'alimentation en cas de maintenances de la base de cas, il s'agira alors d'utiliser les cas passés (connaissance contextuelle) pour garantir une meilleure adéquation entre la solution théorique préconisée (nécessairement élaborée dans un cadre général fourni par la connaissance sous forme de modèles) et la solution effective (celle-ci dépendant de son contexte de mise en œuvre). Par exemple, la connaissance générale nous permettra de calculer, selon plusieurs variables caractérisant le problème de maintenance, une charge de maintenance théorique qui pourra être modulée (augmentée ou réduite) par des connaissances contextuelles relatives aux ressources affectées ou à l'occurrence d'imprévus. À terme, il sera possible d'enrichir la connaissance générale formalisée sous forme de modèles avec la généralisation de connaissances contextuelles.

Les différents types de connaissances que nous souhaitons exploiter étant définis, nous pouvons cadrer plus précisément l'objet de nos travaux. Il s'agit d'utiliser de la connaissance générale afin d'obtenir des valeurs théoriques, puis de les affiner par l'utilisation de connaissance contextuelle afin d'obtenir des valeurs plus proches de la réalité.

Nous allons donc, dans les sections à venir, présenter les outils qui vont permettre l'exploitation des connaissances générales et contextuelles.

1.3 Approches retenues pour exploiter les deux types de connaissances

Nous avons différencié deux types de formalisation de connaissances : la formalisation sous forme de cas pour la connaissance contextuelle (section 1.2.2 page 15) et la formalisation sous forme de modèle pour la connaissance générale (section 1.2.3 page 19).

En ce qui concerne la connaissance générale formalisée sous forme de modèle, les travaux effectués en intelligence artificielle ont débouché sur trois types de modèle et outils : les modèles à base de faits et règles, les représentations ontologiques et les modèles à base de contraintes. Nous n'avons pas retenu les approches basées sur les ontologies, car si elles sont très descriptives [Giaretta 1995 ; Gruber et al. 1995] elles sont plus limitées en ce qui concerne les déductions. L'aspect déductif est d'avantage supporté par les mécanismes à base de règles [Dayal et al. 1988] qui ont donné naissance aux systèmes experts [Harmon et al. 1988]. A la même période, les approches par contraintes sont apparues. L'intérêt des approches par contraintes par rapport aux règles a été montré sur un exemple simple dans [Weigel et al. 1994] où l'apport des CSP pour la séparation des connaissances du processus de traitement est clairement mis en évidence. Complétant cet aspect, les possibilités d'utilisation interactive de la connaissance, via les mécanismes de filtrage de contraintes [Bessiere 1994], nous ont conduits à retenir ce type de formalisme et les outils de filtrage associés.

En ce qui concerne les connaissances contextuelles formalisées par des cas. Le choix du raisonnement à partir de cas [Aamodt et al. 1994] pour l'exploitation de cas est dû au fondement même du mécanisme de raisonnement à partir de cas. En effet, ce mécanisme permet, à partir d'un ensemble de situations mémorisées (stockées sous forme de cas disponibles dans la base de cas) d'effectuer un raisonnement par analogie en s'appuyant sur des notions de similarité et de distance. Cependant, lorsque les cas analogues ou proches sont plus nombreux il devient alors nécessaire d'essayer de dissocier les

plus pertinents ou probables. Cette recherche de cas pertinent ou besoin d'extraction de signification vise alors à extraire du sens d'un ensemble de cas et s'apparente au mécanisme de fouille de donnée [Fayyad et al. 1996]. En conséquence, certains mécanismes issus du raisonnement à partir de cas, identification de cas proche, et de la fouille de données, sélection des cas probables, ont été retenus pour nos travaux.

Le raisonnement à partir de cas et le *data-mining*, utilisés pour l'exploitation de la connaissance contextuelles, sont présentés respectivement en section 1.3.1 et 1.3.2 page 26. Le formalisme des problèmes de satisfaction de contraintes, utilisés pour l'exploitation de la connaissance générale, est présenté dans la section 1.3.3 page 28.

1.3.1 Raisonnement à Partir de Cas

Le Raisonnement à Partir de Cas (RàPC) ou *Case-Based Reasoning* (CBR) a été introduit par Minsky en 1975 ([Minsky 1974]). En 1994, Kolodner a formalisé les bases du RàPC dans son livre ([Kolodner 1993]) et dans le même temps, Aamodt et Plazza ont formalisé le cycle RàPC ([Aamodt et al. 1994]).

Le cycle d'utilisation d'un RàPC est composé de quatre étapes : la recherche, l'adaptation, la révision et l'apprentissage. Le processus vu dans sa globalité consiste à utiliser des situations (appelées cas) passées (cas sources) pour résoudre une situation en cours ou à venir (cas cible). Intéressons-nous maintenant à chaque étape :

La recherche consiste à définir un cas cible afin d'identifier les cas source les plus proches de cette situation. Il s'agit de comparer les valeurs du cas cible aux valeurs des cas sources grâce à un calcul de similarité locale. Un calcul de similarité globale permet ensuite de calculer, pour chaque cas source, sa similarité par rapport au cas cible.

L'adaptation permet, pour les cas sources sélectionnés par l'utilisateur, d'adapter leurs solutions au problème du cas cible.

La révision intervient une fois le cas cible résolu. Il s'agit alors de faire un retour d'expérience sur la solution adaptée en précisant si celle-ci a été une réussite ou un échec, et surtout pour quelles raisons.

L'apprentissage est la dernière phase qui consiste à enregistrer la situation qui vient de passer afin d'enrichir la base de connaissance.

Concevoir un processus RàPC revient à déterminer des variables $\mathcal{V}$ descriptives d'un cas, décrire des similarités locales pour chaque variable $sim(\mathcal{V})$ et décrire une similarité globale $\mathcal{G}$:

La détermination des variables consiste à identifier les variables qui permettent de caractériser de manière représentatives le couple problème/solution ciblé. Cela signifie que l'on estime que les données correspondant à ces variables sont porteuses de sens. Chaque variable se voit attribuer un type (symbolique ou continue) ainsi qu'un domaine.

La description des similarités locales est l'étape la plus complexe car il s'agit de définir, pour chaque variable, quelle sera la similarité entre les valeurs de son domaine. La similarité est un chiffre compris entre 0 et 1, la valeur 1 signifiant que les valeurs sont totalement similaires, et 0 qu'elles ne le sont pas du tout. [Bergmann 2002] propose un ensemble de méthodes de calcul de similarités locales. Parmi celles-ci, il en existe trois principales formes :

- la similarité binaire : si les deux valeurs sont égales leur similarité est de 1, sinon elle est de 0,
- la similarité par formule qui s'applique sur les variables continues, l'idée est de déterminer une fonction dépendant de la différence entre deux valeurs : la source et la cible. Nous pouvons par exemple avoir $sim(\mathcal{V}_{source}, \mathcal{V}_{cible}) = max(\frac{-|\mathcal{V}_{source}-\mathcal{V}_{cible}|+100}{100}, 0)$. Cette formule indique que lorsque la différence entre la source et la cible est supérieure à 100, la similarité est de 0, sinon, la similarité est égale à l'inverse de la différence entre la source et la cible sur 100,
- la matrice de similarité qui s'applique sur les variables discrètes. Il s'agit ici de représenter une matrice des valeurs possibles pour une variable avec en colonne la donnée source et en ligne la donnée cible. La similarité entre une valeur source et une valeur cible est inscrite à l'intersection des lignes et colonnes correspondantes. Un exemple de matrice de similarité pour des types d'hélicoptères est présenté table 1.3 page suivante.

Il existe de nombreuses autres similarités qui dépendent de l'utilisation du RàPC (cas le plus similaire, cas le plus adaptable, etc.), mais les trois citées ci-dessus sont celles que nous utiliserons dans nos travaux. Par exemple, [Mille 2006] propose d'utiliser des ontologies pour calculer des similarités entre valeurs.

La similarité globale permet d'agréger les similarités locales afin de déterminer la similarité entre deux cas. [Bergmann 2002] propose un certain nombre de calculs de similarités globales. La similarité globale la plus couramment utilisée est la moyenne des similarités locales. Un autre type de similarité globale est une moyenne pondérée, cela signifie que l'on donne plus d'importance à la similarité locale d'une ou plusieurs variables par rapport à une ou plusieurs autres.

Type	Puma	Tigre	Dauphin	Gazelle
Puma	1	0,8	0,3	0,2
Tigre		1	0,4	0,4
Dauphin			1	0,8
Gazelle				1

TABLE 1.3 : Exemple de matrice de similarité portant sur le type d'hélicoptère

Il existe de très nombreuses applications du raisonnement à partir de cas en conception tels que ceux présentés par [Guo et al. 2012], [Lee et al. 2002], [Mileman et al. 2002], [Mok et al. 2008] ou encore [Woon et al. 2005]. Il arrive aussi que certains mécanismes du raisonnement à partir de cas soient utilisés dans des outils spécifiques, par exemple en ingénierie pharmaceutique comme le présente [Avramenko et al. 2006].

Le processus RàPC exploite une base de cas, mais il est aussi possible d'exploiter une base de cas en utilisant du *data-mining*, ce que nous allons présenter dans la section à venir.

1.3.2 Data-mining

La fouille de donnée ou *data-mining* est un procédé qui découle du besoin d'extraire des connaissances de bases de données de plus en plus fournies. Ce procédé a été formalisé et décrit de manière très précise dans l'article de Fayyad en 1996 ([Fayyad et al. 1996]) ainsi que dans [ShuHsien et al. 2012] qui effectue une revue des applications du *data-mining* 2000 à 2011.

De manière générale, le *data-mining* est un procédé qui consiste à analyser de grandes bases de données pour en tirer des groupes de données non repérables par l'humain. Il existe de très nombreux algorithmes permettant de faire du *data-mining*, en 1998, le livre [Westphal et al. 1998] proposait une classification de ces algorithmes en quatre grand groupes, et ceux-ci sont toujours d'actualité : la description, la structuration, l'explication, et l'association.

La description consiste à trouver un résumé des données qui soit intelligible pour l'humain. La description fait entre autre usage de statistiques descriptives et d'analyse factorielle. *Exemple : moyenne d'age des hélicoptères ayant eu un taux de panne supérieur à 20%.*

La structuration consiste à faire ressurgir des groupes « naturels » qui représentent des entités particulières. La structuration fait usage des techniques de *clustering* ou d'apprentissage non supervisé. *Exemple :*

découvrir une typologie de comportements d'hélicoptères d'un même type.

L'explication consiste à prédire les valeurs d'un attribut à partir d'autres attributs. L'explication utilise des techniques de régression, ou d'apprentissage supervisé. *Exemple : prédire la probabilité pour un hélicoptère donné, de tomber en panne en fonction de ses caractéristiques (type, puissance, utilisation, etc.).*

L'association consiste à trouver des ensembles de descripteurs qui sont corrélés. L'association utilise des techniques de règles d'associations. *Exemple : les hélicoptères de type Puma ont fréquemment des problèmes de stabilité du rotor.*

Il n'existe pas à proprement parler d'étape de modélisation d'un procédé de fouille de données. La principale activité à effectuer se situe au niveau de son utilisation. Les difficultés de mise en place d'une fouille de données sont : l'agglomération de l'information qui peut nécessiter de créer des bases de données de taille très importante, le choix et le paramétrage de l'algorithme de fouille qui est un point particulièrement délicat et demande une connaissance experte des différents algorithmes, et enfin l'interprétation des données, car comme dans chaque système d'analyse de données, l'interprétation joue un rôle clé dans la qualité des données restituées.

Dans nos travaux, nous cherchons à identifier des paramètres de règles en s'appuyant sur une base de cas existante : nous nous situons donc dans un contexte de *data-mining* d'association. L'algorithme le plus usité dans ce domaine est l'algorithme *APRIORI* décrit par [Agrawal et al. 1993]. Nous nous sommes principalement intéressés aux termes utilisés par cet algorithme. Cet algorithme permet, à partir d'une base de cas, de déterminer des relations d'implications. Ces relations sont du type de l'équation 1.1.

$$\text{Antécédent} \Rightarrow \text{Conséquent}(\text{Support}; \text{Confiance}) \tag{1.1}$$

L'antécédent est un n-uplet (qui peut aussi être un singleton) de données qui est la condition de déclenchement de la règle. Par exemple Type = Puma et Couleur = Noir.

Le conséquent est un n-uplet (qui peut aussi être un singleton) de données qui est le résultat de la condition de déclenchement liée. Par exemple, Etat = Bon.

Le support permet de connaitre la récurrence de cette règle. Il s'agit en fait du pourcentage d'apparition du couple antécédent ⇒ conséquent. Un support faible signifie que la règle peut être une coïncidence.

La confiance est le pourcentage d'apparition du conséquent lorsque l'antécédent est présent. Si la confiance est de 100%, cela signifie que pour chaque apparition de l'antécédent, le conséquent est toujours apparu. Si la confiance est de 33%, cela signifie que le conséquent est apparu dans un tiers des cas où l'antécédent est apparu.

L'utilisation que nous prévoyons de faire de l'algorithme *APRIORI* nous permettra de déduire des paramètres de règles à partir d'une base de cas. Ces règles pourront alors être exploitées par un moteur de filtrage de contraintes, ce que nous présentons dans la section suivante.

De nombreux travaux ont été réalisés en aide à la conception utilisant du *data-mining*. [BaudLavigne et al. 2011] ou encore [Da Cunha et al. 2006] proposent de concevoir conjointement une famille de produit et sa chaine logistique en utilisant du *data-mining*. On trouve aussi des travaux tels que [Kim et al. 2005] qui proposent d'utiliser le *data-mining* pour optimiser des problèmes de conception. [Braha 2001] propose un état de l'art des méthodes et des applications du *data-mining* en conception. [Agard et al. 2004] propose des méthodologies afin d'obtenir des familles de produit en conception.

1.3.3 Problèmes de satisfaction de contraintes

Les problèmes de satisfaction de contraintes ou *Constraint Satisfaction Problem* (CSP) ont été définis par Montanari en 1974 ([Montanari 1974]). Ils permettent de modéliser de la connaissance générale, et de raisonner sur celle-ci afin de trouver des solutions à un problème donné.

Un CSP est composé d'un ensemble de variables $\mathcal{V}$ définies sur un domaine $\mathbb{D}_{\mathcal{V}}$ et d'un ensemble de contraintes $\mathcal{C}$ liant ces variables. L'idée générale d'un CSP est de faire en sorte que l'ensemble des contraintes portant sur l'ensemble des variables soient toujours vérifiées. L'étape de modélisation d'un CSP consiste en trois principales phases : la détermination des variables descriptives, la détermination des domaines de ces variables, et enfin la définition des contraintes liant les variables entre elles.

Trouver une solution d'un CSP revient à valuer chacune des variables tout en vérifiant qu'elles respectent les contraintes du modèle.

Ce principe se décline alors en deux principales utilisations :

La résolution qui consiste à explorer de manière systématique l'espace de recherche et à fournir une ou plusieurs solutions d'un problème. L'algorithme de résolution le plus souvent utilisé est l'algorithme de retour arrière [Golomb et al. 1965].

Type	Couleur
Puma	Noir
Puma	Rouge
Gazelle	Noir
Gazelle	Blanc
Gazelle	Rouge
Libellule	Noir
Libellule	Rouge

TABLE 1.4 : Exemple de table de compatibilité

Le filtrage consiste, quant à lui, à réduire l'espace de solutions de manière itérative en ne conservant que les valeurs autorisées par les contraintes. Cette réduction s'effectue par exemple par méthode de 2b-cohérence pour les variables continues [Lhomme 1993], et par arc-cohérence pour les variables discrètes [Bessiere 1994].

Nous différencions deux groupes de contraintes : les contraintes unaires et les contraintes portant sur plusieurs variables. Les contraintes unaires sont en fait les réductions utilisateur. Il s'agit ici de valuer une variable en ne sélectionnant qu'une valeur pour celle-ci. Concernant les contraintes portant sur plusieurs variables, elles peuvent prendre trois principales formes : une formule arithmétique, une formule logique ou une table de compatibilité.

- les formules arithmétiques s'appliquent sur les variables numériques. Nous pouvons par exemple avoir $\mathcal{V}_1 = 2 \otimes e^{\mathcal{V}_2 \ominus \mathcal{V}_3}$. Nous pouvons remarquer dans la formule précédente que les opérateurs mathématiques sont des opérateurs sur des intervalles [Moore 1966] car les CSP permettent de travailler sur des données sous forme d'intervalles et non seulement sur des valeurs scalaires,
- les formules logiques s'appliquent sur tous les types de données. Un exemple de formule logique est $\mathcal{V}_1 = A \& \mathcal{V}_2 < 10 \Rightarrow \mathcal{V}_3 = Rouge$. Nous pouvons remarquer que les variables peuvent être indifféremment de type symbolique ou numérique.
- les tables de compatibilité s'appliquent sur tous les types de données. Chaque colonne représente une variable, et chaque ligne représente une combinaison possible de valeurs pour ces variables. Lors de l'étape de propagation de contraintes, le CSP vérifie que la combinaison souhaitée est présente dans l'une des tables de compatibilité. A l'inverse, il existe des tables d'incompatibilité qui recensent l'ensemble des combinaisons interdites. Un exemple de table de compatibilité peut être observé en table 1.4.

Nous retrouvons de nombreux travaux concernant l'aide à la conception utilisant des problèmes de satisfactions de contraintes. Entres autres, [Gelle 1998], [Mulyanto 2002], [Vareilles 2005] puis [Chenouard 2007] ont effectués leurs travaux de thèses sur l'aide à la conception utilisant des contraintes.

1.3.4 Utilisation des outils dans notre proposition

Comme nous l'avons indiqué dans la section 1.1.5 page 12, nous souhaitons utiliser simultanément de la connaissance contextuelle et de la connaissance générale afin d'aider à la décision en maintenance d'hélicoptères. La connaissance générale nous permet une première définition du plan de maintenance à réaliser pour un aéronef particulier et la connaissance contextuelle nous permet d'adapter, au plus juste, ce plan en prenant en compte le savoir-faire des maintenanciers dans des contextes similaires (conditions d'utilisation de l'aéronef ou disponibilité et compétences des ressources). La connaissance contextuelle sera exploitée par deux outils : le RàPC et le *data-mining*. La connaissance générale sera exploitée quant à elle par des algorithmes de filtrage de contraintes. Les sections suivantes expliquent comment ses outils participent à notre approche d'aide interactive à la conception.

1.3.4.1 RàPC : la recherche

Nous avons vu que le cycle RàPC est composé de quatre principales étapes que sont la recherche, l'adaptation, la révision et l'apprentissage. Pour nos travaux, nous n'utiliserons que l'étape de recherche. En effet, l'identification dans la base de cas des cas proches du cas courant est primordiale pour nous afin d'utiliser la connaissance contextuelle.

La base de cas du RàPC que nous utilisons est mixte, cela signifie que les variables qui décrivent les cas peuvent être indifféremment continues ou symboliques. De plus, nous utilisons des similarités sur des domaines, ce qui signifie qu'une cible peut être un domaine ($[3; 12]$ par exemple) et non pas seulement un singleton ($\{5\}$ par exemple).

À notre connaissance, il n'existe qu'un seul logiciel libre et correctement documenté permettant de faire du RàPC, celui-ci est myCBR [1], mais il ne répond pas aux critères suivants :

- recherche sur des intervalles et non uniquement sur des singletons,
- interfaces possibles avec des bibliothèques extérieures,

1. Disponible à l'adresse http://www.mycbr-poroject.net

– utilisation en *stand-alone* (myCBR requiert jColibri et Protégé).

Nous avons donc développé en interne au Centre Génie Industriel des Mines d'Albi un outil de recherche de cas similaires pour les besoins de la thèse.

1.3.4.2 Data-mining : l'association

Dans les quatre grands domaines du *data-mining* que sont la description, la structuration, l'explication et l'association, nous avons utilisé la description dans nos travaux. Nous utiliserons un algorithme se basant sur les principes de l'algorithme *APRIORI*, utilisant les termes d'antécédent et de conséquent ainsi que le terme de confiance pour caractériser la fréquence d'occurrence d'une valeur d'une variable.

1.3.4.3 CSP : le filtrage

Nous utilisons la programmation par contraintes pour effectuer du filtrage de contraintes. Ce choix est dû au besoin d'assistance interactive que nous souhaitons mettre en œuvre. En effet, étant donné que nous nous situons dans une problématique d'aide à la conception interactif, notre outil doit interagir avec l'utilisateur, lui permettant ainsi de faire des choix, d'observer ses conséquences, de revenir en arrière, de modifier des valeurs.

Nous effectuons du filtrage sur des variables mixtes, cela signifie que les variables manipulées peuvent être indifféremment numériques ou symboliques.

Différents outils tels que ILOG Solver [2], Choco [3] ou Eclipse [4]étaient envisageables, mais après analyse et travaux de [Djefel 2010], nous avons décidé d'utiliser pour nos travaux CoFiADe, logiciel développé par le Centre Génie Industriel des Mines d'Albi [5]. Les avantages de CoFiADe par rapport à nos besoins sont :

– la maîtrise du langage utilisé et la parfaite connaissance de l'architecture du logiciel,
– un moteur de filtrage de contraintes contrairement à la plupart des moteurs de CSP qui sont centrés sur de la résolution,
– la possibilité de travailler sur des CSP mixtes (variables continues et discrètes dans un même modèle),
– la possibilité d'utiliser des variables continues sur des multi-intervalles.

2. Disponible à l'adresse : http://ibm.com/software/integration/optimization/cplex-cp-optimizer/
3. Disponible à l'adresse : http://www.emn.fr/z-info/choco-solver/
4. Disponible à l'adresse : http://eclipseclp.org/
5. disponible à l'adresse http://cofiade.mines-albi.fr

1.4 Synthèse

Dans ce chapitre, nous avons présenté les différents types de conceptions, et nous sommes arrivés au constat que, dans le cadre de la maintenance d'hélicoptère, nous évoluons dans un contexte de conception routinière. Étant dans ce contexte routinier, nous sommes dans une situation où de la connaissance peut être accumulée au fil du temps. De plus, étant dans un domaine où la réglementation est forte, nous disposons de connaissances formalisées issues des constructeurs et des organismes de régulation.

Ainsi, nous avons différencié deux types de connaissances :

- la connaissance générale qui provient des constructeurs. C'est une connaissance qui est le plus souvent imposée, et qui constitue une première base pour effectuer de l'aide à la décision en maintenance,
- la connaissance contextuelle qui provient de l'expérience. C'est une connaissance qui se construit au fur et à mesure de la réalisation des activités de maintenance et qui apporte des informations additionnelles à la connaissance générale.

La problématique de nos travaux vise en conséquence à exploiter, de manière quasiment simultanée, de la connaissance générale et de la connaissance contextuelle pour aider à la décision en conception. Nous proposons d'exploiter la connaissance générale en utilisant un moteur de filtrage de contraintes. Concernant la connaissance contextuelle, nous proposons d'utiliser d'une part des méthodes de recherche issues du raisonnement à partir de cas et d'autre part du principe de confiance issu du *data-mining* afin de compléter la connaissance générale avec de la connaissance contextuelle.

Notre objectif est de mettre au point un outil d'aide à la décision permettant :

- de propager des contraintes grâce à un moteur de filtrage tant que cela est possible et dans la limite des connaissances générales disponibles,
- lorsque la connaissance générale est insuffisante, d'utiliser la phase de recherche du RàPC afin d'identifier des cas passés proches du cas présent et ainsi déduire des contraintes dites contextuelles (objet d'étude du chapitre 3),
- lorsque les contraintes contextuelles identifient des cas trop nombreux, du fait d'une couverture forte de la base de cas, de déduire des fréquences d'occurrence grâce aux méthodes issues du *data-mining* permettant d'exploiter la connaissance contextuelle la plus adéquate (objet d'étude du chapitre 4).

La suite de ce manuscrit sera divisée en trois parties. Dans le chapitre 2, nous présenterons un cas d'application qui nous permettra d'illustrer les propositions de l'ensemble du manuscrit. Nous présenterons alors différents travaux que nous avons identifiés dans la littérature qui permettent de coupler les approches par contraintes, par raisonnement à partir de cas et par *data-mining*. Dans le chapitre 3, nous proposerons un nouveau type de contraintes que nous appelons contraintes contextuelles. Ces contraintes permettent, à partir d'une base de cas, de créer des contraintes qui dépendent du contexte dans lequel elles sont déclenchées. Nous illustrerons l'utilisation de ces contraintes sur deux exemples : l'un s'appliquant à des variables continues et l'autre s'appliquant à des variables symboliques. Pour terminer, dans le chapitre 4, nous proposerons une fonctionnalité complétant ces contraintes contextuelles qui permet, outre la limitation des domaines des variables impactées par les contraintes, d'estimer les valeurs les plus plausibles ou probables pour le contexte évoqué précédemment. Cette fonctionnalité sera de même illustrée sur les exemples présentés au chapitre 3.

Chapitre 2

Utilisation conjointe de la connaissance générale et contextuelle

Nous avons vu dans le chapitre précédent que nous souhaitons associer deux types de connaissances pour aider à la conception : la connaissance contextuelle et la connaissance générale. Nous avons choisi de formaliser la connaissance générale avec un modèle exploité par un moteur de propagation de contraintes. Concernant la connaissance contextuelle, nous avons décidé de la formaliser grâce à une base de cas et d'extraire de la connaissance contextuelle grâce à des principes issus du raisonnement à partir de cas et du *data-mining*.

Dans ce chapitre, nous allons tout d'abord présenter l'exemple illustratif que nous utiliserons tout au long de ce mémoire. Cet exemple concerne la maintenance d'hélicoptères et est modélisé à l'aide de variables, de contraintes liant les variables et de matrices de similarités portant sur les variables. Il se décompose en deux sous-exemples comportant chacun un modèle, le premier met en œuvre des variables continues, le second des variables symboliques. Ensuite, nous présenterons les différentes approches, identifiées dans la littérature, qui intègrent les méthodes fonctionnant à base de contraintes, de raisonnement à partir de cas et de *data-mining*. Cette intégration sera appelée « couplage » dans la suite de ce mémoire. Nous allons tout d'abord étudier les couplages permettant d'améliorer la construction des modèles et des bases de connaissances en phase d'élaboration du système d'aide à la décision. Ensuite, nous nous intéresserons aux couplages permettant d'exploiter les deux types de connaissances en phase d'utilisation du système d'aide à la décision.

2.1 Support permettant l'illustration de nos propositions

L'exemple que nous allons présenter est volontairement simple. Il ne reflète pas la réalité de la maintenance d'hélicoptères, mais il se base sur des constats et relations entre variables simples à comprendre et à se représenter.

Nous commençons par présenter le problème, puis nous listerons les variables utilisées pour caractériser ce problème. Ensuite, nous expliciterons les contraintes liant les variables puis les fonctions de similarité pour chaque variable du modèle.

2.1.1 Présentation du problème

L'exemple que nous utilisons s'inspire de la maintenance d'hélicoptères tel que nous avons l'identifié lors d'entretiens avec des experts en maintenance d'Hélimaintenance. Une activité de maintenance peut durer de cinq minutes à plusieurs semaines, et la majorité du temps, elle ne concerne qu'une partie de l'hélicoptère (moteur, pâles, rotor, cabine).

Dans notre travail, nous utilisons deux exemples distincts. La différence entre ces deux exemples réside dans la nature des variables sur lesquelles l'aide à la décision intervient. Dans un cas il s'agit d'une variable continue et dans l'autre cas, il s'agit d'une variable symbolique :

– pour l'exemple portant sur une variable continue, nous nous intéressons à l'estimation du temps de remplacement d'un moteur sur un hélicoptère. Lorsqu'un hélicoptère est livré à un atelier pour effectuer une maintenance, les maintenanciers disposent de deux connaissances : la connaissance théorique livrée par le constructeur, et leur propre connaissance acquise au fil du temps. Le principe consiste donc à approcher le temps de maintenance à prévoir en utilisant les données constructeurs, puis à préciser ce temps en utilisant l'expérience issue de cas passés. Le changement de moteur est une étape relativement peu fréquente mais très lourde pour l'hélicoptère. Le principal problème rencontré dans cette maintenance est que le propriétaire de l'hélicoptère peut se voir privé de son aéronef pendant beaucoup plus longtemps que théoriquement prévu à cause des conditions de vol de l'aéronef. La plus-value apporté par notre système d'aide à la décision se situe alors au niveau de l'estimation de se temps d'immobilisation : une meilleure estimation permet une meilleure prévision de charge pour le maintenancier et ainsi pour le client final.

– pour l'exemple portant sur une variable symbolique, nous nous plaçons dans le cadre d'une maintenance sur les patins d'atterrissage. À chaque vol de l'avion, les patins doivent être vérifiés. La plupart du temps, une simple vérification suffit à obtenir l'habilitation de vol. Il arrive parfois que les patins aient à être revissés, réparés voire remplacés. Cela peut arriver, par exemple, lorsqu'un hélicoptère a subi un choc à l'atterrissage, ou que des conditions de vols difficiles ont ajouté du jeu aux visseries du patin. Il est alors possible, en utilisant des cas passés, d'estimer les risques qu'a le responsable de maintenance d'avoir à effectuer plus qu'une simple révision. Dans cette situation, la variable que nous allons chercher est donc une variable correspondant à la maintenance à prévoir sur le patin.

Chaque aéronef est décrit par son modèle, son utilisation (civil ou militaire) et le constructeur du fuselage. Un aéronef donné est soumis par son utilisateur à des conditions de vol données. Cela signifie que pour un hélicoptère, il est possible de décrire ses conditions d'utilisations telles que l'ambiance de vol (sable, terre, mer, froid) et le pays dans lequel il évolue. L'ensemble de ses informations sont notées dans la base de cas à la réception d'un appareil pour une maintenance. En plus de cela, sont notées les informations concernant le moteur dont il était équipé depuis sa dernière maintenance (puissance, cylindrée, blindage). Enfin, sont notées des informations sur la coopération moteur/hélicoptère, telles que le fait que, dans ces conditions, le moteur ait connu peu ou de nombreuses défaillances, ou encore le taux moyen de sollicitation du moteur.

Chaque activité de maintenance d'hélicoptère aboutit donc à l'ajout d'un cas dans la base de cas. La base de cas est ainsi remplie de cas passés, ces derniers étant le plus complet possible.

Il existe des règles connues dans la maintenance d'hélicoptères auxquelles il est impossible de déroger. Il s'agit, par exemple, du fait qu'un hélicoptère utilisé dans un contexte militaire doit avoir un moteur blindé. D'autres règles existent, certaines sont implicitement prises en compte par les maintenanciers expérimentés, d'autres peuvent demeurer inconnues du fait de leur complexité ou de leur faible fréquence d'apparition.

Dans les sections suivantes, nous allons définir les variables utilisées dans notre modèle, ainsi que les contraintes les liant entre elles et les similarités locales de chaque variable.

2.1.2 Variables de caractérisation du modèle et de la base de cas

Voici l'ensemble des variables utilisées ainsi que leur abréviation, leur domaine et leur description. Ces variables sont présentes à la fois dans le modèle et dans la base de cas. De plus, quelques caractéristiques intéressantes à prendre en compte pour la compréhension du modèle sont données pour chaque variable. Il est utile de noter que la valeur *indéfini* est possible pour chacune des variables. Cela signifie qu'il est possible, dans une situation, que la valeur d'une variable ne soit pas renseignée.

Les variables ainsi que leur domaine de validité permettant la description d'un hélicoptère sont :

- Modèle (*mod*) : {Gazelle, Dauphin, Tigre, Puma}. Il s'agit du modèle de l'hélicoptère dans l'ordre de poids, respectivement 1200kg, 2200kg, 3000kg, 3600kg.
- Utilisation (*uti*) : {Civil, Militaire}. Il s'agit du type d'utilisation de l'appareil. Les hélicoptères utilisés dans un contexte militaire demandent généralement plus de puissance et de fiabilité aux fortes sollicitations, de plus, le moteur doit obligatoirement être blindé.
- Constructeur (*cst*) : {A, B, C}. Il s'agit du constructeur du fuselage de l'aéronef. Certains constructeurs ne sont pas autorisés dans certains pays. De plus, certains constructeurs de fuselage ne sont pas adaptés à certains moteurs et certains constructeurs sont plus fiables que d'autres.

Les variables ainsi que leur domaine de validité permettant la description du moteur sont :

- Puissance (*pui*) : {100, 200, 300, 400, 500}. Il s'agit de la puissance du moteur (en *cv*). Un moteur puissant sur un hélicoptère de petite taille permet d'avoir une faible sollicitation, à l'inverse, un moteur peu puissant sera très sollicité sur un hélicoptère de grande taille.
- Cylindrée (*cyl*) : {1500, 2500, 4000, 5000, 6000}. Il s'agit de la cylindrée du moteur (en cm^3). La cylindrée est directement liée à la puissance du moteur.
- Blindage (*bli*) : {Oui, Non}. Cette variable caractérise la présence ou non d'un blindage. Un moteur blindé est plus lourd qu'un moteur non blindé, mais le blindage est une condition nécessaire pour un hélicoptère militaire et parfois pour le civil.

Les variables ainsi que leur domaine de validité permettant la description des conditions d'utilisation et de l'état du moteur sont :

– Sollicitation (sol) : [0 ; 100]. Il s'agit du pourcentage de sollicitation du moteur. Ce pourcentage est renseigné lorsqu'un appareil est pris en charge. Il s'agit du rapport entre la puissance moyenne utilisée et la puissance maximale disponible du moteur.

– Défaillance (def) : {Très faible, Faible, Moyen, Fort, Très fort}. Il s'agit de la fréquence des défaillances du moteur. Il arrive parfois que les moteurs connaissent des défaillances. Une fréquence faible n'est pas critique, mais une fréquence forte signifie que le moteur a des difficultés à répondre correctement à la sollicitation.

– Ambiance (amb) : {Sable, Terre, Mer, Froid}. Cette variable caractérise l'ambiance dans laquelle l'hélicoptère a eu l'occasion de voler. Chaque ambiance a une influence sur le fonctionnement nominal du moteur.

– Pays (pay) : {France, Suède, Tunisie, Espagne}. Il s'agit du pays dans lequel l'hélicoptère évolue. Chaque pays a ses particularités en termes de climat, ainsi qu'en terme de constructeurs et éventuellement certaines règles internes à prendre en compte.

Maintenant que nous avons connaissance des variables utilisées dans notre modèle, nous allons présenter les relations les liant dans la section suivante.

2.1.3 Contraintes pour la formalisation de la connaissance générale

La liste des contraintes formalisant la connaissance générale est représentée ci-dessous :

C1 : lie le blindage à l'utilisation de l'appareil (table 2.1a page suivante). Cette contrainte signifie qu'un hélicoptère militaire doit obligatoirement posséder un moteur blindé. Cette contrainte est imposée par les autorités afin d'obtenir un certificat de navigabilité.

C2 : lie le constructeur du fuselage de l'appareil et le pays (table 2.1b page suivante). On peut voir que le constructeur A est présent en France, en Espagne et en Suède, le constructeur B en France et en Espagne et le constructeur C en Tunisie et en Espagne. Cette contrainte est issue des faits.

C3 : lie la puissance à la cylindrée du moteur (table 2.1c page suivante). On peut voir qu'à une puissance donnée correspond une cylindrée donnée. Cette contrainte est fournie par le constructeur.

TABLE 2.1 : Tables de compatibilité du modèle illustratif

(a) Contrainte C1

bli	*uti*
Oui	Civil
Oui	Militaire
Non	Civil

(b) Contrainte C2

cst	*pay*
A	France
A	Espagne
A	Suède
B	France
B	Espagne
C	Tunisie
C	Espagne

(c) Contrainte C3

pui	*cyl*
100	1500
200	2500
300	4000
400	5000
500	6000

2.1.4 Matrices de similarités locales associées aux variables

Il est nécessaire de définir une fonction de similarité locale pour chaque variable utilisée dans le modèle. Toutes les similarités sont décrites sous forme de matrices (table 2.2 page ci-contre). Si l'on prend par exemple la matrice de similarité locale des constructeurs (2.2d page suivante), la lecture de celle-ci permet de voir que le constructeur A et le constructeur B sont estimés comme parfaitement similaires, et qu'ils sont eux-mêmes similaires à 70% au constructeur C.

2.1.5 Synthèse

Maintenant que nous avons présenté notre exemple illustratif, nous allons présenter, dans les sections à venir, les différents travaux que nous avons identifiés visant à coupler les systèmes à base de connaissances que nous utilisons. Les exemples que nous donnons afin d'illustrer chaque couplage se basent sur l'exemple que nous venons de présenter.

2.2 Couplages pour l'aide à la constitution de connaissance

Dans cette section, nous allons présenter les différentes approches identifiées dans la littérature exploitant à la fois de la connaissance contextuelle et de la connaissance générale. Cette utilisation conjointe des deux connaissances est effectuée dans un contexte de constitution de connaissance, il s'agit donc

TABLE 2.2 : Matrices de similarité du modèle illustratif

(a) Similarité du modèle d'hélicoptère (*mod*)

mod	Puma	Tigre	Dauphin	Gazelle
Puma	1	0,8	0,3	0,2
Tigre		1	0,4	0,4
Dauphin			1	0,8
Gazelle				1

(b) Similarité de l'ambiance de vol de l'hélicoptère (*amb*)

amb	Sable	Terre	Mer	Froid
Sable	1	0,7	0,9	0,3
Terre		1	0,8	0,6
Mer			1	0,1
Froid				1

(c) Similarité du pays de vol de l'hélicoptère (*pay*)

pay	France	Suède	Tunisie	Espagne
France	1	0,4	0,4	0,8
Suède		1	0,1	0,2
Tunisie			1	0,8
Espagne				1

(d) Similarité du constructeur de l'hélicoptère (*cst*)

cst	A	B	C
A	1	1	0,7
B		1	0,7
C			1

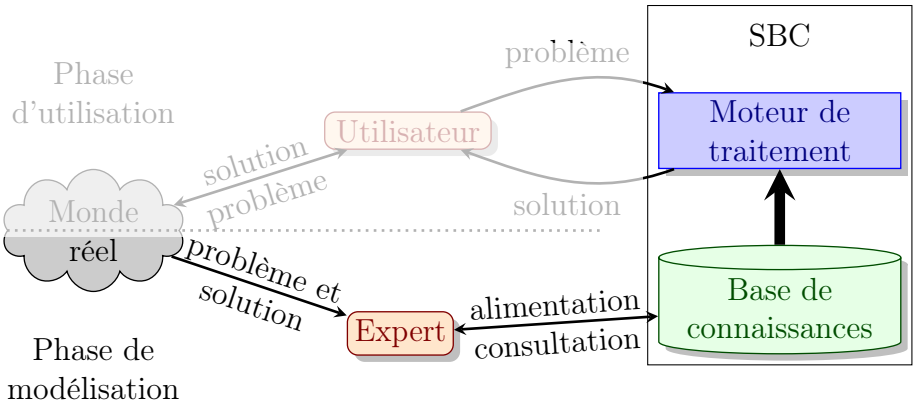


FIGURE 2.1 : Figure illustrative de la phase de conception de la base de connaissance

de constitution du modèle (variables, domaines des variables et contraintes portant sur les variables) ou de la base de cas (descripteurs, domaines des descripteurs, similarités locales et similarité globale) (cf figure 2.1). Ces étapes de complétion ou de validation de la connaissance se déroulent hors du contexte d'utilisation en faisant appel à des experts.

2.2.1 Validation des connaissances

La validation des connaissances permet de s'assurer que les connaissances contenues dans une base de cas ou un modèle sont valides (au sens présenté en sections 1.2.2.3 page 18 et 1.2.3.3 page 21). Deux raisons peuvent mener à une vérification des connaissances. Cela peut être dû à l'import de nouvelles connaissances extérieures, il s'agit alors d'utiliser les connaissances déjà contenues dans le système pour vérifier que la nouvelle connaissance ne va pas entrainer des dysfonctionnements. L'autre raison est la mise à jour des connaissances, cela signifie que l'on souhaite vérifier une connaissance que l'on estime vieillissante en la confrontant à une connaissance à laquelle on apporte plus de confiance.

Dans les deux sections à venir, nous nous placerons dans une situation extrême de confiance totale dans l'un ou l'autre des deux formalismes. Il est évident que dans la réalité, cette hypothèse forte n'est pas vérifiée. Le processus de validation de la connaissance est donc souvent un aller-retour entre la validation de la connaissance contextuelle 2.2.1.1 page suivante et la validation de la connaissance générale 2.2.1.2 page 44.

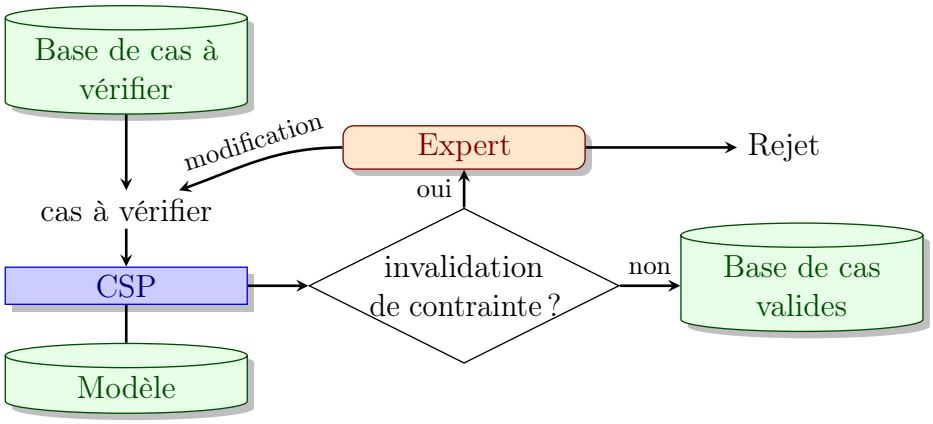


FIGURE 2.2 : La connaissance générale permet de valider la connaissance contextuelle

Cas	...	*cst*	*pay*	...
1	...	A	France	...
2	...	A	Suède	...
3	...	B	Suède	...
4	...	B	Tunisie	...
5	...	B	France	...

TABLE 2.3 : Cas à confronter à la connaissance générale

2.2.1.1 Validation de la connaissance contextuelle

Dans cette partie, nous nous situons dans le cas où la confiance accordée au modèle est totale. Nous cherchons à valider ou invalider de la connaissance contextuelle.

Il s'agit, dans cette situation, de valider la totalité d'une base cas. Les cas sont alors confrontés un à un au modèle auquel on accorde confiance. Tous les cas qui respectent l'ensemble des contraintes du modèle sont alors enregistrés dans une base de cas valides. Les cas non validés par le modèle de contraintes sont soumis à un expert qui devra définir la cause de cette invalidation (erreur de saisie, capteur mal positionné, etc.) et les modifier ou les rejeter au besoin. La figure 2.2 schématise ce processus.

Dans notre exemple, il s'agit de vérifier que les cas représentés table 2.3 sont bien valides par rapport au modèle de connaissance générale décrit en section 2.1 page 36. Cette table comporte les constructeurs et pays de vol de 5 moteurs qui viennent d'être livrés. Il s'agit alors de vérifier la bonne concordance entre les cas et le modèle de contraintes.

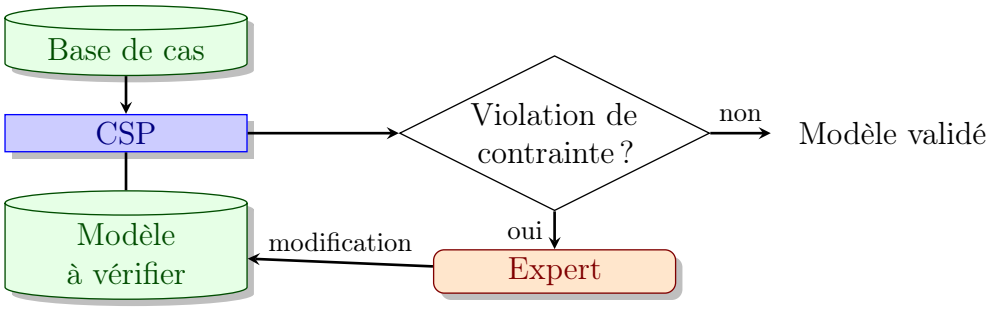


FIGURE 2.3 : La connaissance contextuelle permet de valider la connaissance générale

Exemple

En utilisant la contrainte C2 représentée en table 2.1b page 40, on remarque que les cas 1, 2 et 5 de la base de cas sont valides, alors que les cas 3 et 4 ne le sont pas. Il s'agit alors de soumettre les cas 3 et 4 à un expert afin qu'il en vérifie la cohérence. Il peut ici s'agir d'une erreur de saisie dans la colonne constructeur ou pays. Considérons que le problème du cas 3 est dû à une erreur de saisie concernant le constructeur. L'expert peut alors le modifier. L'origine du problème du cas 4 n'est quant à elle pas décelable, l'expert va alors rejeter ce cas de la base de cas exploitable par le système à base de connaissances.

À notre connaissance, il n'existe pas de travaux portant sur ce couplage.

2.2.1.2 Validation de la connaissance générale

Dans cette partie, nous nous situons dans le cas où la confiance donnée à la base de cas est totale. Nous cherchons à valider ou invalider la connaissance générale.

Il s'agit, dans cette situation, de confronter l'ensemble des cas de la base de cas aux contraintes du modèle. Si aucun cas n'entre en contradiction avec les contraintes, alors on estime que le modèle est valide. Sinon, le modèle doit être modifiée par un expert afin qu'il devienne cohérent avec l'ensemble des cas de la base de cas (modification ou suppression de contraintes). Le processus est alors réitéré.

La figure 2.3 schématise ce processus.

Exemple

Considérons la contrainte C1 présentée en table 2.1a page 40 à laquelle on a retiré le premier tuple. Cela signifie que les hélicoptères blindés sont nécessairement militaires. Cette contrainte C1' est repré-

TABLE 2.4 : Tables utilisées dans l'exemple de la section 2.2.1.2 page précédente

(a) Contrainte C1'

bli	uti
Oui	Militaire
Non	Civil

(b) Base de cas de l'exemple

Cas	...	bli	uti	...
1	...	Oui	Militaire	...
2	...	Non	Civil	...
3	...	Non	Civil	...
4	...	Oui	Civil	...
5	...	Oui	Militaire	...

> *sentée en table 2.4a. On dispose d'une base de 5 cas d'hélicoptères, telle que représentée en table 2.4b. Les cas 1, 2, 3 et 5 ne violent aucunement la contrainte C1', le cas 4 quant à lui vient en contradiction avec la contrainte C1' du fait qu'il autorise le couple Blindage = Oui avec Utilisation = Civil. Cette erreur est soumise à l'expert afin qu'il fasse évoluer le modèle. Dans ce cas simple, nous voyons alors qu'il suffit d'ajouter un tuple à la table de compatibilité pour que cela fonctionne. La contrainte redevient alors telle que présentée en table 2.1a page 40.*

On trouve des applications de ce type de couplage dans la littérature. Par exemple, [Felfernig et al. 2007] présente un système qui permet d'invalider des contraintes d'un CSP grâce à des cas passés.

2.2.2 Complétion de la connaissance

La complétion de connaissances permet d'utiliser un type de connaissances pour en compléter un autre. Pour une base de cas, il s'agit alors de renseigner les valeurs indéfinies de certaines variables dans les cas. Pour un modèle de contraintes, il s'agit d'ajouter ou de préciser certaines contraintes.

2.2.2.1 Complétion de la connaissance contextuelle

Dans cette partie, nous nous situons dans le cas où la connaissance générale est exploitée afin de compléter la connaissance contextuelle. Dans cette situation, nous allons utiliser un CSP afin de compléter certains cas. Ce type de mécanisme est utile lorsque l'on dispose d'une base de cas incomplète à cause d'un défaut de capteur ou d'informations non saisies par exemple, et que l'on souhaite compléter ces cas.

FIGURE 2.4 : La connaissance générale permet de compléter la connaissance contextuelle

Le principe de fonctionnement est de soumettre à un CSP les cas incomplets. Des algorithmes de filtrage de CSP permettent alors de déduire des valeurs possibles pour les variables dont la valeur est indéfinie. Ces informations sont alors analysées par un expert qui va pouvoir compléter les cas. La figure 2.4 schématise ce processus.

Exemple

On a à disposition une base de cas incomplets représentée en table 2.5a page suivante. Cette base de cas manque de données, la complétude des cas est respectivement de 75%, 50% et 100%, la complétude de la base est donc de 75%. Nous avons à disposition toutes les contraintes décrites dans la section 2.1.3 page 39. L'expert va alors confronter un à un les cas incomplets au moteur de filtrage de contraintes. Les données obtenues permettront de compléter les cas. Pour le cas n° 1, le moteur de filtrage de contraintes lui retournera une valeur possible de 300cv pour la puissance du moteur, il ne reste donc à l'expert qu'à saisir cette valeur après analyse du cas ainsi complété. Pour le cas n° 2, le moteur de filtrage de contraintes retournera une valeur possible de 2500 pour la cylindrée du moteur, ainsi que le panel de valeurs possibles suivant : {Puma, Gazelle, Libellule, Fenec} pour le type d'hélicoptère car aucune contrainte ne vient imposer ce choix. L'expert va alors saisir la valeur pour la cylindrée et le cas atteindra alors une complétude de 75%. Le cas n° 3 quant à lui n'est pas concerné par cette étape de complétion de l'information. La base de cas complétée par l'expert est représentée en table 2.5b page suivante et a désormais 92% de complétude.

À notre connaissance, il n'existe pas de travaux sur ce couplage.

TABLE 2.5 : Bases de cas de l'exemple en section 2.2.2.1 page 45

(a) Cas incomplets

Cas	...	mod	pui	cyl	...
1	...	Puma	*indéfini*	4000	...
2	...	*indéfini*	200	*indéfini*	...
3	...	Libellule	100	1500	...

(b) Cas complétés

Cas	...	mod	pui	cyl	...
1	...	Puma	**300**	4000	...
2	...	*indéfini*	200	**2500**	...
3	...	Libellule	100	1500	...

FIGURE 2.5 : La connaissance contextuelle permet de compléter la connaissance générale

2.2.2.2 Complétion de la connaissance générale

Dans cette partie, nous nous situons dans le cas où la connaissance contextuelle est exploitée afin de compléter la connaissance générale. Dans notre situation, nous allons utiliser une base de cas pour ajouter ou préciser des contraintes. Ce type d'utilisation est fait lorsqu'une grande quantité de cas est disponible, et que l'on souhaite en extraire de la connaissance générale.

Le principe de fonctionnement est d'effectuer l'analyse d'une base de cas par un expert, celui-ci va alors déterminer de nouvelles règles afin de les injecter dans le modèle. L'étape d'analyse est souvent assistée par un système informatique reposant sur des méthodes d'analyse de données, et notamment le *data-mining*. [Maimon et al. 2010] présente une revue des diverses méthodes de *data-mining* utilisées afin d'effectuer de l'analyse de données. La figure 2.5 schématise ce processus.

Exemple

On dispose d'un extrait d'une base de cas dans laquelle on a enregistré la puissance du moteur, son taux de sollicitation, le type d'hélicoptère sur lequel il est installé et la fréquence des défaillances. Cette base de cas est valide (table 2.6 page suivante) et analysée par un expert. L'intervention humaine est indispensable pour cette étape, car

Cas	...	pui	sol	mod	def	...
1	...	100	100	Puma	Très fort	...
2	...	200	98	Puma	Faible	...
3	...	500	25	Puma	Très faible	...
4	...	100	100	Puma	Très fort	...
5	...	100	60	Libellule	Moyen	...
6	...	500	32	Puma	Fort	...

TABLE 2.6 : Extrait d'une base de cas contenant des données sur la puissance du moteur, sa sollicitation, le type d'hélicoptère sur lequel il a été monté et la probabilité d'occurrence de pannes.

> *la base de cas ne peut pas être considérée comme totalement couvrante. Ici, les règles déduites sont par exemple les suivantes :*
>
> – $pui = 100 \wedge mod = Puma \rightarrow sol = 100$
>
> – $sol = 100 \rightarrow def =$ Très fort
>
> – $pui = 500 \wedge mod = Puma \rightarrow sol < 40$
>
> *Bien entendu, ces règles sont discutables étant donné que la base de cas n'est pas exhaustive, c'est la raison pour laquelle ce processus ne peut être effectué que par un expert, qui a de la connaissance tacite, et ce lors de la phase de formalisation de la connaissance.*

Le *data-mining* fait partie des méthodes possibles permettant d'obtenir de la connaissance générale en exploitant de la connaissance contextuelle. [Maimon et al. 2010] propose un large aperçu des techniques de *data-mining* utilisées depuis une décennie.

2.3 Couplages pour l'aide à l'utilisation d'un système à base de connaissance

Dans cette section, nous nous situons en phase d'exploitation, cela signifie que l'expert n'intervient pas sur la collaboration entre les différents outils exploitant de la connaissance. Ces couplages doivent être robustes car l'utilisateur n'a pas la connaissance du modèle que peut avoir l'expert.

Nous nous plaçons dans la phase d'exploitation des connaissances pour l'aide à la décision (figure 2.6). Nous distinguons deux principes de couplage des outils à base de connaissances : le couplage séquentiel qui consiste à utiliser un outil, puis un autre, et à retourner le résultat ; et le principe de

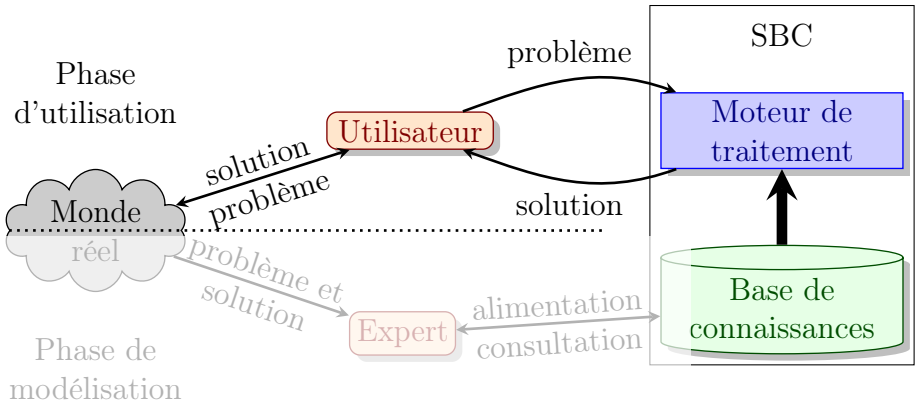


FIGURE 2.6 : Illustration de la phase d'utilisation

support qui consiste à n'utiliser qu'un outil qui, au besoin, peut faire appel lui-même à un second système.

2.3.1 Séquence CSP puis RàPC

Comme l'ont proposé [Boutilier et al. 1997] ou encore [Ha et al. 1998], il est possible d'utiliser un moteur de filtrage de contraintes pour restreindre l'espace de recherche d'un RàPC. En effet, lors de recherches sur des bases de cas de grandes dimensions, l'utilisation préalable d'un moteur de filtrage de contraintes pour filtrer les cas potentiellement utiles permet un gain de temps et de saisie notable pour l'utilisateur.

Exemple

Prenons une base de cas, avec une répartition égale entre Puma, Dauphin, Gazelle et Tigre. Lorsque l'utilisateur aura sélectionné un type d'hélicoptère alors l'étape de recherche du RàPC ne se fera plus sur la totalité des cas mais sur seulement un quart de ceux-ci grâce au moteur de filtrage de contraintes qui sélectionnera les cas pertinents par rapport à cette réduction du domaine des types d'hélicoptères.

2.3.2 Séquence RàPC puis CSP

On rencontre aussi des travaux qui présentent l'idée d'utiliser le CSP pour aider à l'adaptation de cas. Comme nous l'avons présenté en section 1.3.1 page 24, la phase d'adaptation est la phase la plus complexe d'un processus de raisonnement à partir de cas car il s'agit d'adapter un cas passé à un cas

cible. Le CSP va assister l'utilisateur dans cette phase. [Purvis et al. 1995] ou [Ruet et al. 2002] proposent d'utiliser des contraintes afin d'adapter un cas de la base de cas à un cas courant.

Exemple

Prenons le cas d'un utilisateur qui réalise la phase de recherche d'un processus de RàPC. La phase de recherche du RàPC lui propose un cas proche de sa requête, mais l'hélicoptère retenu est de type civil alors que l'utilisateur est en possession d'un hélicoptère militaire. Le CSP forcera donc le fait que l'hélicoptère soit blindé lors de la phase d'adaptation de la solution, facilitant ainsi l'adaptation à l'utilisateur.

2.3.3 CSP en support au data-mining

Parmi les travaux que nous avons identifiés, [Khiari et al. 2010] propose un système de *Constraint-based mining*. Il s'agit alors d'utiliser un CSP pour retrouver des règles avec le *data-mining*. Le principe général est de créer des « méta-contraintes » puis d'effectuer une étape de recherche de règles avec des techniques de *data-mining*. Les règles trouvées qui correspondent aux « méta-contraintes » sont retenues, les autres sont mises de côté. Cette méthode permet principalement de diminuer le temps de traitement des résultats obtenus après *data-mining* car celui-ci ne va effectuer la recherche de règle que sur les règles dont la forme a été prédéfinie dans la « méta-contraintes ».

Exemple

Si l'on définit une méta-contrainte telle que $\forall(X, Y) \rightarrow X \leq Y$ *et que le* data-mining *nous renvoie une liste de règles dont certaines respectent cette contrainte, et d'autres non, seules les règles respectant la contrainte seront retenues.*

2.3.4 Data-mining en support au CSP

Il existe actuellement de nombreux outils permettant d'utiliser le *data-mining* afin de générer des contraintes pour le CSP, mais tous ces outils sont utilisés en phase de conception des modèles ou des bases de cas. [Brin et al. 1997] ainsi que [Nortet et al. 2005] proposent de nombreuses méthodes permettant de déduire des contraintes de grandes bases de cas grâce à des techniques de *data-mining*. On voit dans ces articles que les principales utilisations du *data-mining* pour la déduction de contraintes se situent en

médecine (un grand nombre de patients permettent de déduire de nouvelles relations entre les symptômes et les maladies) ou dans le commerce, et plus particulièrement le e-commerce (les clients ayant acheté l'article X ont aussi acheté l'article Y par exemple).

Concernant la phase d'exploitation, nous allons, dans le chapitre 4, présenter une proposition d'utilisation simultanée de *data-mining* et de CSP.

Exemple

Un grand nombre de cas de maintenance d'hélicoptères pourra permettre de déduire des règles sur les tendances aux pannes en fonction du type d'hélicoptère.

[Buza et al. 2010] propose de récupérer un ensemble de résultats obtenus par résolution de contraintes, puis d'obtenir le résultat optimal en utilisant des algorithmes de *data-mining*.

2.3.5 Data-mining en support au RàPC

Les travaux de [Zhuang et al. 2009] proposent d'utiliser le *data-mining* pour créer des *clusters* sur lesquels sont effectuées les recherches de cas similaires en RàPC. Les *clusters* étant des regroupements de données ayant le même contexte, la recherche ne porte que sur quelques milliers de cas lorsque la base de cas initiale contenait plus d'un million d'enregistrements.

Exemple

Si l'on possède une base de cas de plusieurs millions d'enregistrements et que l'on effectue un processus de raisonnement à partir de cas dessus, les durées de traitement seront importantes. Il est alors possible d'effectuer une étape de data-mining *afin de réduire le nombre de cas en créant des clusters représentatifs d'ensembles de cas. Le processus de raisonnement à partir de cas sera alors effectué sur ces clusters, et donc sur une base de cas de plus petite taille, permettant ainsi une durée de traitement réduite.*

[Chougule et al. 2011] proposent d'utiliser une méthode exploitant à la fois le *data-mining* et plus particulièrement le *rule-mining* associé au raisonnement à partir de cas. Ils proposent de repérer grâce au *data-mining* les causes d'insatisfactions des clients et de les enregistrer en base de cas. Il s'agit donc ici d'utiliser le *data-mining* afin de peupler une base de cas exploitée par un processus de raisonnement à partir de cas.

2.4 Synthèse

Dans ce chapitre, nous avons présenté l'exemple qui nous a permis d'illustrer les différents couplages d'aide à la décision exploitant de la connaissance générale et de la connaissance contextuelle identifiés dans la littérature. Un premier état de l'art de ces couplages avait été effectué dans [Codet de Boisse et al. 2010].

Ces différents travaux permettent de différencier deux principes d'aide ou d'assistance. Le premier vise à aider l'élaboration des bases de connaissance durant la phase de conception du système d'aide à la décision. Le second vise à aider l'exploitation de ces connaissances lors de la phase d'utilisation du système d'aide à la décision.

Concernant la phase de conception, nous avons montré comment il était possible de valider des connaissances (section 2.2.1 page 42) et de compléter des connaissances (section 2.2.2 page 45). De manière synthétique, suivant les niveaux de confiance accordés aux connaissances générale et contextuelle, la couverture et la validité de l'une permet d'améliorer la couverture et la validité de l'autre.

Concernant la phase d'utilisation, nous disposons de trois outils et nous avons montré comment certaines interactions permettent d'améliorer l'aide à la décision. Nous avons pu identifier cinq des six interactions possibles. À notre connaissance, il n'existe aucuns travaux où le raisonnement à partir de cas fournit un support au *data-mining*.

Nous avons donc vu que la complétion et la validation des connaissances se situent uniquement dans une phase de conception des outils d'aide à la décision. Lors de l'utilisation, seuls des échanges de données entre les outils sont effectués. L'ensemble de ces travaux montre donc clairement qu'il n'y a pas d'approche permettant d'exploiter en bonne complémentarité connaissance générale et contextuelles en phase d'utilisation du système d'aide à la décision. Nos travaux visent en conséquence à combler ce manque, et la méthode que nous allons proposer complètera la connaissance générale par de la connaissance contextuelle en phase d'utilisation des outils d'aide à la décision.

L'identification des couplages que nous avons effectuée dans ce chapitre à fait l'objet d'une publication dans le journal *Engineering Applications of Artificial Intelligence* ([Vareilles et al. 2011]). Nous avions alors identifié les couplages entre les différents outils que nous avons précédemment cités. De plus, nous avions introduit le principe de contrainte contextuelle afin d'utiliser de la connaissance contextuelle pour compléter de la connaissance générale

en phase d'utilisation d'un système à base de connaissance. Nous allons, dans le chapitre suivant, définir ce principe de contrainte contextuelle qui permet d'utiliser de la connaissance contextuelle pour combler des manques de connaissance générale.

Chapitre 3

Prise en compte des connaissances contextuelles par le biais d'une contrainte contextuelle

Dans le chapitre précédent, nous avons identifié plusieurs travaux visant à exploiter à la fois des connaissances générales et des connaissances contextuelles. Ces travaux proposent pour la plupart une utilisation séquentielle des outils et en conséquence de chaque type de connaissance. Dans ce chapitre, nous nous intéressons à l'utilisation simultanée des deux types de connaissances.

Un modèle de connaissances générales se compose d'un ensemble de variables liées les unes aux autres. Il arrive fréquemment que nous ne soyons pas en mesure d'expliciter et de formaliser toutes les relations existantes entre les variables. Cela est dû à la difficulté d'extraire et de formaliser cette connaissance générale. La couverture de la connaissance générale n'est donc pas totale.

Dans ce chapitre, nous allons nous intéresser à ce qu'il est possible de faire lorsqu'il manque des relations entre variables dans un modèle de connaissance générale. Ce manque de connaissance ayant pour conséquence de rendre indépendantes certaines variables ou groupes de variables, l'idée est alors d'utiliser de la connaissance contextuelle afin de déterminer (ou de préciser) les relations manquantes. La figure 3.1 page suivante représente de manière schématisée cette situation. Les lignes pleines représentent des relations connues entre les variables, alors que les lignes en pointillés représentent les relations existantes mais non explicitées. Nos travaux visent à expliciter ces relations en tenant compte de leur contexte.

Dans ce chapitre, nous allons commencer par situer notre proposition : la problématique, l'idée proposée et les exemples illustrant notre contribution.

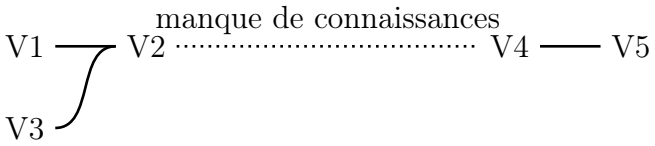


FIGURE 3.1 : Relations formalisées ou connues entre les variables d'un modèle

Par la suite, nous introduirons le principe de contrainte contextuelle, que nous décrirons en détail. Pour finir, nous illustrerons l'utilisation de cette contrainte contextuelle sur deux exemples : le premier portant sur des variables continues et le second sur des variables symboliques.

3.1 Situation d'utilisation

3.1.1 Problématique

Avec de l'expérience en aide à la décision, il arrive que les experts sachent tacitement qu'il existe une relation entre certaines variables, sans pour autant être en mesure de l'expliciter. La connaissance générale disponible ne couvre pas l'ensemble des relations entre ces variables. La connaissance générale étant insuffisante pour cette partie du modèle, nous proposons d'exploiter la connaissance contextuelle afin de combler ce manque de connaissances.

Nous nous plaçons donc dans la situation suivante :

- la connaissance générale est incomplète sur certaines relations entre variables : le modèle fait apparaître un manque de connaissances et n'est donc pas assez couvrant (au sens donné en section 1.2.3.2 page 19),
- l'expert a identifié un ensemble de variables sur lesquelles la connaissance contextuelle permet d'apporter des connaissances pour les relier et combler en partie ce manque.

3.1.2 Notre proposition de solution

Dans le cadre de nos travaux, nous proposons d'introduire la notion de contrainte contextuelle afin d'exploiter la connaissance contextuelle lorsque la connaissance générale n'est pas assez couvrante. Nous définissons une contrainte contextuelle comme étant une contrainte paramétrique ([Vareilles et al. 2011]). Les valeurs de ses paramètres sont fixées à partir de la connaissance contextuelle. Une contrainte contextuelle est donc une contrainte dont les

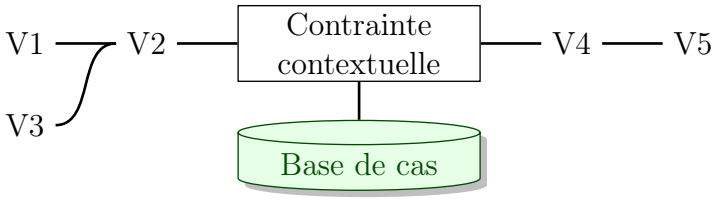


FIGURE 3.2 : Relations formalisées ou connues entre les variables d'un modèle

valeurs des paramètres peuvent changer en fonction du contexte dans lequel elle est déclenchée.

Pour présenter brièvement le principe de contrainte contextuelle, il s'agit d'expliciter les variables qui sont reliées entre elles et les variables qui peuvent influencer les valeurs des paramètres. Le moteur de traitement des contraintes contextuelles va alors chercher dans une base de cas des valeurs possibles pour les paramètres et proposer une contrainte paramétrée. Les valeurs possibles des paramètres sont directement issues des cas similaires à la situation présente en termes de contexte.

Si l'on se base sur le schéma 3.1 page précédente, il s'agit en fait de remplacer la relation en pointillés par l'utilisation d'une contrainte contextuelle utilisant une base de cas. Cette proposition de fonctionnement est représentée sur le schéma 3.2 où l'on voit que la contrainte contextuelle va venir combler le manque de connaissances entre la variable V2 et la variable V4.

3.1.3 Exemple illustratif

Afin d'illustrer nos propos, nous allons nous appuyer sur deux exemples issus de la maintenance d'hélicoptères. Ces exemples étendent le modèle de la section 2.1 page 36 auquel nous allons ajouter quelques variables et contraintes.

Les deux exemples utilisés permettent d'illustrer d'une part notre proposition sur des variables continues et, d'autre part, sur des variables symboliques.

Exemple appliqué aux variables continues. Notre exemple concerne l'estimation de la durée de montage d'un moteur. Nous disposons d'informations livrées par les constructeurs d'hélicoptères précisant les différents temps de montage en fonction des couples moteur / hélicoptère. Ces durées sont des valeurs théoriques, mais il se trouve qu'en réalité, elles peuvent varier de manière importante en fonction des conditions de vol effectives d'un hélicoptère ou des conditions dans l'atelier.

Notre proposition permet alors de déduire une durée estimée du temps de montage en fonction du temps de montage théorique et des informations sur le contexte de la maintenance.

Exemple appliqué aux variables symboliques. Notre exemple concerne la prévision des opérations à effectuer sur des patins d'atterrissage. Selon le constructeur, avant chaque vol, les patins d'atterrissage doivent être vérifiés, et réparés si besoin. De plus, tous les vingt vols, ils doivent être totalement démontés et révisés. Cela implique donc que pendant les 19 vols qui précèdent la dépose des patins, les maintenanciers n'ont aucune visibilité sur l'opération qu'ils vont devoir effectuer. Pourtant, il apparait que selon les conditions de vol de l'hélicoptère, certains patins aient à être réparés entre deux déposes, et d'autres non.

Notre proposition permet donc de déduire les opérations à prévoir sur l'hélicoptère en fonction des données constructeurs et du contexte de vol.

3.1.4 Synthèse

Dans nos travaux, nous cherchons à allier connaissances générales et connaissances contextuelles afin d'améliorer la prise de décision en conception. Nous introduisons pour cela dans ce chapitre le principe de contrainte contextuelle. Ce type de contrainte permet d'intégrer, dans un modèle de connaissances générales, de la connaissance qui dépend du contexte courant.

3.2 Description d'une contrainte contextuelle

Dans cette section, nous présentons le concept de contraintes contextuelles. Dans un premier temps nous allons présenter les différents éléments qui composent la contrainte contextuelle, puis nous présenterons son fonctionnement.

3.2.1 Élements constitutifs de la contrainte contextuelle

Une contrainte contextuelle (représentée dans l'expression 3.1) est composée de huit éléments :

$$cc(LVR, sim_g, ms, LVC, LVP, FP, LVI, FCC) \tag{3.1}$$

– LVR la liste des variables sur lesquelles la recherche de cas similaires est effectuée dans la base de cas, il s'agit donc de variables obligatoirement présentes dans la base de cas,
– sim_g la fonction de similarité globale utilisée pour calculer la similarité entre le cas cible et les cas sources. sim_g porte uniquement sur des variables contenues dans LVR car la recherche est effectuée uniquement sur ces variables,
– ms le seuil au-delà duquel un cas est estimé comme suffisamment similaire au cas cible pour être porteur de connaissances dans le contexte présent décrit par LVR,
– LVC la liste des variables que l'on souhaite retenir suite à la sélection des cas les plus similaires afin de calculer les valeurs des paramètres,
– LVP la liste des paramètres de la contrainte,
– FP la fonction permettant de calculer, pour chaque cas, la valeur des paramètres de LVP en fonction des variables de LVC,
– LVI la liste des variables qui seront impactées dans le CSP par cette contrainte contextuelle,
– FCC la contrainte elle-même, composée des variables de LVI des paramètres de LVP.

3.2.2 Fonctionnement de la contrainte contextuelle

Lors du déclenchement d'une contrainte contextuelle, le mécanisme qui suit se déroule :

1. La recherche en fonction des variables de LVR est effectuée dans la base de cas. La similarité globale de chaque cas par rapport aux valeurs des variables de LVR est calculée grâce à l'expression de sim_g. Seuls les cas dont la similarité globale est supérieure ou égale à ms sont retenus dans la base de cas filtrée,
2. pour chaque cas de la base de cas filtrée, la formule FP est appliquée sur les variables de LVC afin d'obtenir la valeur des paramètres présents dans LVP,
3. les valeurs de LVI sont alors obtenues avec la contrainte FCC et les valeurs de LVP. En fonction de la configuration du moteur de filtrage, deux variantes sont alors envisageables :
   (a) les valeurs calculées des variables de LVI sont affichées à l'utilisateur, qui peut les considérer comme une forme de conseil,

Alg. 1 FONCTIONNEMENT DE LA RECHERCHE DANS LE RÀPC

– ∴ – *Dcible est le tableau des domaines cibles fournis par l'utilisateur à la contrainte contextuelle*
– ∴ – *Dcible*[*i*] *est le domaine cible de la variable i*
– ∴ – *base est un tableau de n cas*
– ∴ – *base*[*n*] *est le n-ième cas*
– ∴ – *simiglob*[*n*] *est un tableau contenant les similarités globales* sim_g *des cas par rapport au cas cible*
– ∴ – *cas est le tableau associé au cas source en cours de calcul*
– ∴ – *vsource*[*i*] *est la valeur source de la variable i*
– ∴ – *simi est un tableau temporaire de similarités locales*

Début
  Pour c allant de 1 à n **Faire**
    $cas \leftarrow base[c]$
    Pour i allant de 1 à Taille(cas) **Faire**
      Si ($vsource[i] = indéfini$OU $vsource[i] = inutile$) **Alors**
        $simi[i] \leftarrow 1$
      Sinon
        Si (Type ($vsource[i]$) = Symbolique) **Alors**
          $simi[i] \leftarrow \max_{\forall j \in Dcible[i]}(sim(vsource[i], Dcible[i])$
        Sinon
          Si (($vsource[i] \in Dcible[i]$) **Alors**
            $simi[i] \leftarrow 1$
          Sinon
            $simi[i] \leftarrow \max_{\forall j \in Dcible[i]}(sim(vsource[i], Dcible[i])$
          Fin Si
        Fin Si
      Fin Pour
      $simiglob[n] \leftarrow \sum_{i=1}^{Taille(cas)} [\omega_i * (simi[i])^p]^{1/p}$
  Fin Pour
  Retourner($simiglob$)
Fin

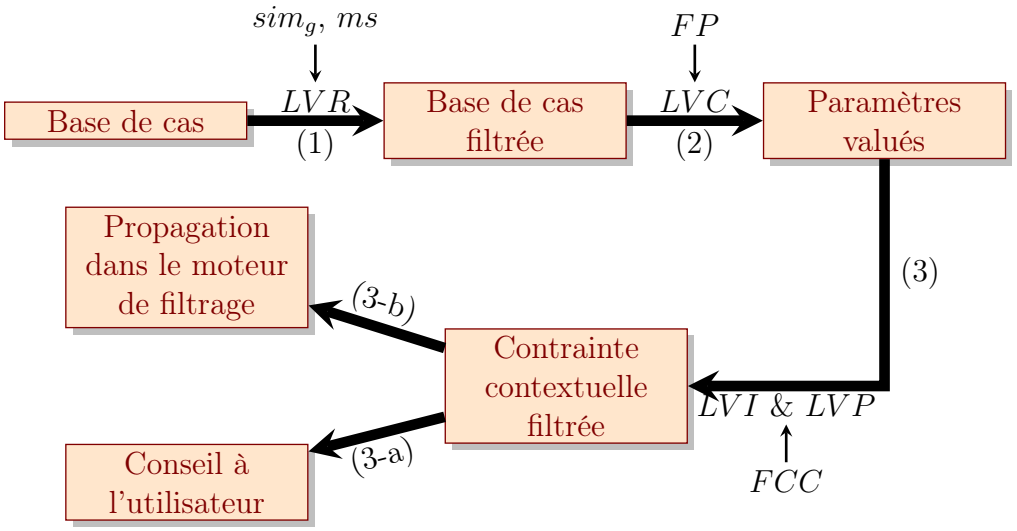


FIGURE 3.3 : Schéma du fonctionnement d'une contrainte contextuelle

(b) la contrainte est automatiquement propagée. Ce fonctionnement est susceptible d'entraîner une instabilité dans le moteur de filtrage.

La figure 3.3 représente l'enchaînement des actions jusqu'à l'étape 5. Ce fonctionnement sera illustré en section 3.3 page 64 pour une variable continue et en section 3.4 page 73 pour une variable discrète.

3.2.2.1 Identification et sélection des cas pertinents pour la constitution de connaissances contextuelles

La phase de recherche du RàPC permet d'identifier l'ensemble des cas présents dans la base de cas avec comme information supplémentaire leur similarité avec le cas cible dans le contexte décrit par LVR. Dans notre situation, les valeurs des cas source ($\mathcal{V}_{source}$) sont obligatoirement des singletons (valeur connue, *indéfinie* ou *inutile*) car l'enregistrement d'une valeur de variable dans la base de cas est obligatoirement un singleton. Les valeurs du cas cible sont par contre des domaines ($\mathbb{D}_{\mathcal{V}_{cible}}$) car issues d'un filtrage par un moteur de filtrage de contraintes, qui fournit des ensembles de domaines réduits.

Calcul de la similarité locale Pour ce calcul de similarité locale, nous distinguons trois cas de figure en fonction de la nature de la valeur source $\mathcal{V}_{source}$ et du domaine cible $\mathbb{D}_{\mathcal{V}_{cible}}$:

- si $\mathcal{V}_{source}$ est *indéfinie* (c'est-à-dire que sa valeur est manquante mais nécessaire) ou *inutile* (c'est-à-dire que sa valeur est manquante mais non nécessaire), alors la similarité locale vaut 1. Ce choix est un choix optimiste, ce qui signifie que l'on estime que quand on ne connait pas une valeur, il y a des chances que celle-ci soit celle recherchée,
- si les valeurs du domaine $\mathbb{D}_{\mathcal{V}_{cible}}$ et la valeur de $\mathcal{V}_{source}$ sont définies et de type symbolique, la similarité locale est la similarité maximale entre les valeurs de $\mathbb{D}_{\mathcal{V}_{cible}}$ et la valeur de $\mathcal{V}_{source}$. Ce calcul de similarité est représenté par l'équation 3.2,
- si les valeurs de $\mathbb{D}_{\mathcal{V}_{cible}}$ et la valeur de $\mathcal{V}_{source}$ sont définies et de type numériques, nous distinguons deux cas de figure. Soit $\mathcal{V}_{source} \in \mathbb{D}_{\mathcal{V}_{cible}}$, dans ce cas la similarité locale vaut 1. Sinon, la similarité locale est égale à la similarité maximale entre $\mathcal{V}_{source}$ et chacune des bornes de $\mathbb{D}_{\mathcal{V}_{cible}}$. Ce calcul de similarité est représenté par l'équation 3.3.

$$sim(\mathcal{V}_{source}, \mathbb{D}_{\mathcal{V}_{cible}}) = \max_{\mathcal{V}_i \in \mathbb{D}_{\mathcal{V}_{cible}}} (sim(\mathcal{V}_{source}, \mathcal{V}_i)) \tag{3.2}$$

$$sim(\mathcal{V}_{source}, \mathbb{D}_{\mathcal{V}_{cible}}) = \begin{cases} 1 & \text{si} \quad \mathcal{V}_{source} \in \mathbb{D}_{\mathcal{V}_{cible}} \\ \max_{\mathcal{V}_i \subset \mathbb{D}_{\mathcal{V}_{cible}}} (sim(\mathcal{V}_{source}, \mathcal{V}_i)) & \text{si} \quad \mathcal{V}_{source} \notin \mathbb{D}_{\mathcal{V}_{cible}} \end{cases} \tag{3.3}$$

Calcul de la similarité globale Pour calculer la similarité globale (notée sim_g) d'un cas source par rapport au cas cible, nous utilisons une fonction de Minkowski présentée entre autres dans [Avramenko et al. 2006] ou [Nunez et al. 2004] et représentée par l'équation 3.4 avec :

- $Nbvar$ le nombre de variables décrivant le cas n,
- $\mathcal{V}^i_{source}$ la valeur de la variable i dans le cas n,
- $\mathbb{D}_{\mathcal{V}^i_{cible}}$ le domaine ciblé de la variable i,
- ω_i le poids attribué à la variable i.

Cette fonction permet, en faisant varier le paramètre p, d'obtenir différents types de moyennes. Si $p = 1$, il s'agit d'une moyenne des similarités locales pondérées par ω_i. Si $p = 2$, il s'agit alors d'un calcul de distance Euclidienne. De manière générale, plus p augmente, plus l'influence des attributs ayant une similarité locale élevée est importante.

$$sim_g = \sum_{i=1}^{Nbvar} \left[\omega_i * \left(sim(\mathcal{V}^i_{source}, \mathbb{D}_{\mathcal{V}^i_{cible}}) \right)^p \right]^{1/p} \tag{3.4}$$

L'algorithme 1 page 60 représente le fonctionnement de la recherche, du déclenchement de celle-ci jusqu'à l'association des cas et de leur similarité par rapport à la cible.

L'intérêt de cet algorithme est double car d'une part nous proposons de comparer une plage de valeurs (ici $\mathbb{D}_{\mathcal{V}_{cible}}$) avec une valeur unique (ici $\mathcal{V}_{source}$) et, d'autre part, nous effectuons un calcul de similarité y compris sur des valeurs qui peuvent ne pas être définies.

3.2.2.2 Sélection des cas à retenir

Une fois l'ensemble des cas présents dans la base de cas analysés et leurs similarités globales calculées par rapport à la cible, il reste à sélectionner ceux qui peuvent s'avérer utiles pour alimenter le calcul des paramètres de la contrainte. Nous avons donc défini un seuil de similarité minimal (ms) qui permet de faire la sélection. Cela consiste à conserver les cas dont la similarité globale est supérieure ou égale à ce seuil, et à ne pas utiliser ceux dont la similarité est inférieure.

Si aucun cas n'est retenu, deux possibilités peuvent être envisagées :

- la contrainte contextuelle devient alors non pertinente, elle n'est donc plus prise en compte dans le modèle de connaissances générales,
- la contrainte contextuelle est encore prise en compte, mais en conservant la dernière valeur connue pour chaque paramètre.

3.2.2.3 Détermination des paramètres de la contrainte

Pour chaque cas de la base de cas retenue, nous appliquons la formule FP afin de calculer les paramètres. Une fois les paramètres calculés pour chaque cas, nous proposons de les agréger pour les appliquer à FCC :

- si le paramètre est de type numérique, nous proposons de retenir l'intervalle englobant de l'ensemble des paramètres calculés pour chaque cas car nous considérons que toutes les valeurs se situant entre la valeur minimale et la valeur maximale sont possibles. Cela constitue une solution simple à mettre en place mais qui peut s'avérer problématique s'il y a une forte variabilité au niveau des paramètres calculés sur chaque cas. Une autre solution peut donc être de remonter l'union des valeurs.

Exemple *Sur 10 cas retenus, 9 cas font apparaitre un paramètre compris entre 0.5 et 0.8, et un dixième fait apparaitre un paramètre à 4. Retenir l'intervalle englobant donnerait donc pour résultat*

> $[0.5; 4]$, *ce qui semble assez peu réaliste. Nous pourrions donc par exemple retenir une valeur du paramètre de* $[0.5; 0.8] \cup 4$.

Cet exemple illustre bien l'importance de la validation des cas par un expert lors de la phase de conception du modèle comme nous l'avons montré en section 2.2.1.1 page 43,

– si le paramètre est de type symbolique, nous proposons de retenir l'union des valeurs identifiées dans les cas pour ce paramètre.

FCC est alors propagée afin de réduire les domaines des variables de LVI. Ces domaines réduits sont alors utilisés comme des conseils pour l'utilisateur ou sont propagés automatiquement par le moteur de filtrage de contraintes.

3.2.3 Synthèse

Dans cette section, nous avons proposé une manière de relier plusieurs variables (LVI) en utilisant de la connaissance contextuelle (LVR) comme nous l'avions schématisé figure 3.2 page 57. Nous allons désormais présenter deux exemples d'application de la contrainte contextuelle. L'un permet d'illustrer notre proposition avec une contrainte contextuelle portant sur des paramètres continus et l'autre sur des paramètres symboliques.

3.3 Application avec un paramètre continu

Comme nous l'avons indiqué en section 3.1.3 page 57, nous cherchons à estimer au mieux le temps de montage d'un moteur sur un hélicoptère particulier dans des conditions données. Les opérateurs de maintenance possèdent des données très précises fournies par les constructeurs des moteurs et des aéronefs.

Les constructeurs fournissent des informations concernant le temps théorique de montage d'un moteur donné sur un hélicoptère donné. Ces connaissances générales sont détenues par les maintenanciers.

En plus de cela, avec l'expérience, les maintenanciers se sont rendu compte qu'en fonction de différents critères, ce temps de montage peut varier de manière significative. Ils ont remarqué qu'en fonction du pays et de l'ambiance de vol, le temps de montage peut être diminué ou augmenté.

Nous allons, dans cette section, présenter les contraintes que nous introduisons en plus de celles déjà présentes dans le modèle donné en section 2.1, puis nous introduirons la contrainte contextuelle telle que l'expert l'a décrite. Enfin, nous allons dérouler un scénario afin d'illustrer nos propos.

3.3.1 Définition du modèle d'application

3.3.1.1 Définition des variables utilisées

Pour illustrer nos propositions, nous n'utiliserons qu'une partie des variables présentées en section 2.1.2 page 38. Les variables retenues sont le modèle d'hélicoptère (*mod*), le constructeur du moteur (*cst*), la puissance du moteur (*pui*), l'ambiance de vol (*amb*) et le pays de vol (*pay*). De plus, nous allons introduire de nouvelles variables dans le modèle, celles-ci sont des variables numériques permettant d'estimer le temps de maintenance (leur domaine est également précisé) :

- durée théorique de montage (*dth*) en minutes : $[0\,;\,+\infty[$. Il s'agit de la durée théorique de montage d'un couple moteur / hélicoptère. Cette valeur est donnée par le constructeur,
- durée estimée de montage (*des*) en minutes : $[0\,;\,+\infty[$. Il s'agit de la durée estimée de montage d'un couple moteur / hélicoptère. Cette durée est une estimation déterminée par le filtrage de la contrainte contextuelle,
- durée réelle de montage (*dre*) en minutes : $[0\,;\,+\infty[$. Il s'agit du temps réellement passé au montage d'un couple moteur / hélicoptère.

Dans un cas idéal, $des = dre$. Cela signifie que la valeur estimée par la contrainte contextuelle est parfaite. Toutes les variables, hormis *des*, sont enregistrées dans la base de cas de maintenance des hélicoptères afin de pouvoir être utilisées. Nous ne préciserons pas les formules de calcul de la similarité locale pour ces variables car celles-ci ne seront pas utilisées dans l'exemple qui va suivre.

3.3.1.2 Données constructeur

Les constructeurs fournissent des données théoriques concernant la maintenance de leurs appareils. Ces données sont exploitées par notre outil sous forme de contraintes. Ici, nous allons nous intéresser à la contrainte permettant d'estimer la durée théorique de maintenance. Cette contrainte permet d'obtenir la durée théorique de montage (*dth*) en fonction de la puissance du moteur (*pui*), du modèle de l'hélicoptère (*mod*) et de son constructeur (*cst*). Nous nommerons cette contrainte **C-3**. La contrainte est représentée dans la table 3.1 page suivante et correspond à de la connaissance générale.

TABLE 3.1 : Contrainte C-3

(a)

pui	*mod*	*cst*	*dth*
100	Gazelle	A	280
100	Gazelle	B	320
200	Gazelle	A	336
100	Gazelle	C	360
100	Dauphin	A	360
200	Gazelle	B	384
300	Gazelle	A	392
100	Dauphin	B	400
200	Gazelle	C	432
200	Dauphin	A	432
100	Dauphin	C	440
100	Tigre	A	440
300	Gazelle	B	448
400	Gazelle	A	448
100	Tigre	B	480
200	Dauphin	B	480
300	Gazelle	C	504
300	Dauphin	A	504
500	Gazelle	A	504
400	Gazelle	B	512
100	Tigre	C	520
100	Puma	A	520
200	Dauphin	C	528
200	Tigre	A	528
100	Puma	B	560
300	Dauphin	B	560
200	Tigre	B	576
400	Gazelle	C	576
400	Dauphin	A	576
500	Gazelle	B	576

(b)

pui	*mod*	*cst*	*dth*
100	Puma	C	600
300	Dauphin	C	616
300	Tigre	A	616
200	Tigre	C	624
200	Puma	A	624
400	Dauphin	B	640
500	Gazelle	C	648
500	Dauphin	A	648
200	Puma	B	672
300	Tigre	B	672
400	Dauphin	C	704
400	Tigre	A	704
200	Puma	C	720
500	Dauphin	B	720
300	Tigre	C	728
300	Puma	A	728
400	Tigre	B	768
300	Puma	B	784
500	Dauphin	C	792
500	Tigre	A	792
400	Tigre	C	832
400	Puma	A	832
300	Puma	C	840
500	Tigre	B	864
400	Puma	B	896
500	Tigre	C	936
500	Puma	A	936
400	Puma	C	960
500	Puma	B	1008
500	Puma	C	1080

3.3.1.3 Base de cas

La table 3.2 page suivante représente la base de cas disponibles pour faire fonctionner la contrainte contextuelle. L'ensemble des cas de la base est complet (pas de valeurs indéfinies), et n'invalide aucune des contraintes précédemment définies (connaissance générales). Il s'agit donc ici d'une base de cas valide, et donc utilisable dans notre contexte. Malgré cela, la base de cas est relativement peu couvrante ($\frac{30}{96000}$ soit 0.03%). En effet, si l'on calcule le cardinal du produit cartésien des domaines de définition des variables présentes dans le modèle, on trouve qu'il existe 96 000 combinaisons possibles pour la description du problème (en estimant qu'une variable continue possède 10 intervalles comme représenté dans l'équation 3.5), or nous n'avons que 30 cas. Nous allons tout de même voir qu'une base de cas peu couvrante peut s'avérer utile à certaines déductions pour une contrainte contextuelle.

$$\begin{aligned} \mathcal{C} &= \mathcal{C}_{mod} * \mathcal{C}_{con} * \mathcal{C}_{pui} * \mathcal{C}_{amb} * \mathcal{C}_{pay} * \mathcal{C}_{dth} * \mathcal{C}_{dre}, \\ &= 4 * 3 * 5 * 4 * 4 * 10 * 10, \\ &= 96000 \end{aligned} \tag{3.5}$$

TABLE 3.2 : Base de cas de l'exemple avec variables continues du Chapitre 3

	mod	*cst*	*pui*	*amb*	*pay*	*dth*	*dre*
1	Puma	C	300	Mer	Espagne	840	1008
2	Puma	B	500	Sable	Tunisie	1008	1441,44
3	Dauphin	A	100	Sable	Tunisie	360	514,8
4	Dauphin	B	500	Terre	Suède	720	518,4
5	Tigre	A	400	Froid	Suède	704	563,2
6	Dauphin	B	200	Mer	Espagne	480	576
7	Gazelle	B	400	Mer	Suède	512	491,52
8	Tigre	C	200	Froid	Suède	624	499,2
9	Puma	B	100	Froid	Suède	560	448
10	Puma	B	200	Sable	Tunisie	672	960,96
11	Gazelle	B	300	Terre	France	448	524,16
12	Gazelle	B	300	Terre	France	448	524,16
13	Dauphin	A	400	Terre	France	576	673,92
14	Gazelle	C	500	Froid	Suède	648	518,4
15	Dauphin	A	500	Sable	Espagne	648	842,4
16	Gazelle	B	500	Mer	France	576	898,56
17	Dauphin	C	500	Froid	Suède	792	633,6
18	Puma	C	100	Terre	Espagne	600	540
19	Tigre	A	200	Mer	Suède	528	506,88
20	Tigre	C	100	Sable	Tunisie	520	743,6
21	Puma	A	500	Terre	Tunisie	936	926,64
22	Dauphin	C	200	Terre	Suède	528	380,16
23	Puma	A	400	Froid	Suède	832	665,6
24	Gazelle	B	300	Mer	Espagne	448	537,6
25	Puma	C	400	Mer	Tunisie	960	1267,2
26	Dauphin	A	200	Sable	Tunisie	432	617,76
27	Tigre	B	200	Mer	Suède	576	552,96
28	Tigre	A	300	Terre	Espagne	616	554,4
29	Tigre	A	500	Sable	Tunisie	792	1132,56
30	Dauphin	A	500	Sable	Tunisie	648	926,64

3.3.1.4 Définition de la contrainte contextuelle

Il reste maintenant à définir la contrainte contextuelle. Pour cela, nous allons expliciter un à un ses paramètres :

- concernant LVR, l'expert sait que la durée réelle de changement du moteur est principalement affectée par l'ambiance et le pays de vol de l'hélicoptère. Il va donc chercher l'ensemble des cas qui se rapprochent d'une ambiance et d'un pays équivalent au cas rencontré. On a donc $LVR = \{amb, pay\}$,
- sim_g est la moyenne des similarités locales de l'ambiance et du pays. Cela signifie que l'expert estime que la similarité globale entre deux cas est autant pondérée par l'ambiance de vol que par le pays,
- ms est choisi par l'expert pour que les cas identifiés par la recherche soient fortement similaires au cas rencontré par l'utilisateur. Une valeur de $ms = 0.9$ est donc choisie,
- concernant LVC, l'expert souhaite que le système identifie, pour l'ensemble des cas passés similaires, le rapport qu'il y avait entre la durée réelle et la durée théorique. Il est donc utile d'avoir $LVC = \{dth, dre\}$,
- LVP ne contient qu'un seul paramètre qui est le rapport entre la durée réelle et la durée théorique nous le nommerons α,
- FP est une équation permettant de déterminer α. Ici, il s'agit du rapport entre la durée réelle et la durée théorique, nous avons donc $\alpha = \frac{dre}{dth}$. α étant numérique, nous retiendrons l'intervalle englobant de ses valeurs pour l'agrégation comme nous l'avons proposé dans la section 3.2.2.3 page 63,
- Concernant LVI, la contrainte permet de calculer la durée estimée de maintenance des en fonction de dth et du paramètre α déduit du contexte. Les deux variables impactées dans le CSP sont donc des et dth. On aura donc $LVI = \{des, dth\}$,
- Enfin FCC est de la forme $des = dth \otimes \alpha$. Cela signifie que l'on multiplie la durée théorique déduite par le moteur de filtrage par le paramètre α que l'on a calculé grâce à FP.

Pour résumer, la contrainte contextuelle est donc la formule représentée en équation 3.6 page ci-contre.

$$
\begin{aligned}
cc_1 \quad (\quad & LVR = \{amb, pay\}, \\
& sim_g = \frac{sim_{amb} + sim_{pay}}{2}, \\
& ms = 0.9, \\
& LVC = \{dth, dre\}, \\
& LVP = \{\alpha\}, \\
& FP = \{\alpha = \frac{dre}{dth}\}, \\
& LVI = \{des, dth\}, \\
& FCC : des = dth \otimes \alpha \\
) \quad &
\end{aligned}
\tag{3.6}
$$

Maintenant que nous avons explicité l'ensemble des variables, des contraintes et des tables de similarités du modèle, la base de cas utilisée, la contrainte contextuelle, ainsi que la formule permettant son obtention, nous allons dérouler un scénario afin de mettre en avant les intérêts de notre proposition.

3.3.2 Scénario de déroulement

Nous nous plaçons dans la situation d'un utilisateur qui, au fil des réductions des domaines des variables du modèle de connaissances générales, sélectionne les valeurs qu'il souhaite à propos d'un couple hélicoptère / moteur afin de pouvoir estimer la durée de montage du moteur sur l'aéronef. Nous allons présenter, pour chaque étape, la variable réduite par l'utilisateur, les variables réduites par les contraintes du CSP et le déclenchement de la contrainte contextuelle si cela s'avère utile.

Initialisation du système. À son état initial, les domaines des variables du CSP sont tels que présentées en section 2.1.2 page 38. Les variables *des* et *dre* ont toujours pour domaine $[0\,;+\infty[$ et *dth* a un domaine réduit du fait de l'application de la contrainte C-3. Le nouveau domaine de *dth* est donc {280, ..., 1080} [1]

Il s'agit tout d'abord de mettre en œuvre la contrainte contextuelle afin d'avoir une première estimation du paramètre α et donc de FCC. La

1. Pour des raisons de lisbilité, nous ne listerons pas l'ensemble des valeurs de *dth* et de *des* lorsque le nombre d'éléments qui les composent sera supérieur à cinq.

recherche dans la base de cas est donc faite avec amb = {Sable, Terre, Mer, Froid} et pay = {France, Suède, Espagne, Tunisie}. Les mesures de similarités prenant la similarité la plus forte, et chaque cas étant renseigné, la similarité de chaque cas sera donc de 1. L'ensemble des cas sera donc considéré et nous aurons donc les valeurs de dre, de dth et de α telles que représentées dans le tableau 3.3a page ci-contre. On obtient donc pour le paramètre α le domaine [0.72 ; 1.56]. On a donc une contrainte FCC de la forme $des = dth \otimes \alpha = \{280, ..., 1080\} \otimes [0.72; 1.56] = \{201.6, ..., 1684.8\}$.

On voit d'ores et déjà qu'avant même d'avoir saisi les informations dans le moteur de filtrage, la contrainte contextuelle propose une estimation du temps de montage plus réaliste que le domaine de départ.

Réduction de mod = Puma. L'utilisateur ayant noté qu'il allait intervenir sur un Puma, la contrainte C-3 sera déclenchée, réduisant ainsi le domaine de dth à dth = {520, ..., 1080}. Aucune variable de LVR n'ayant été modifiée, la contrainte contextuelle n'est pas re-calculée. Par contre, des est ré-estimé en fonction de la nouvelle valeur de dth, entrainant donc $des = dth \otimes \alpha = \{520, ..., 1080\} \otimes [0.72; 1.56] = \{374.4, ..., 1684.8\}$.

Réduction de amb = Sable. Il n'existe aucune contrainte connue impactant l'ambiance de vol de l'hélicoptère, par contre, amb fait partie des variables de recherche de la contrainte contextuelle ($amb \in LVR$). Il faut donc effectuer à nouveau la recherche dans la base de cas. En se basant sur la matrice de similarité donnée en table 2.2b page 41, il est possible de calculer la similarité de chaque cas par rapport à cette cible. Le pays n'étant toujours pas défini, sa similarité locale reste de 1. Le résultat du calcul de chaque cas est visible en table 3.3b page ci-contre.

On peut alors voir que les cas numéros 4, 5, 8, 9, 11, 12, 13, 14, 17, 18, 21, 22, 23 et 28 ne seront pas agrégés car leur similarité globale est inférieure au seuil ms fixé par l'expert ($ms = 0.9$). Le moteur d'agrégation va donc calculer un rapport différent entre dth et dre. On aura donc désormais un paramètre α réduit de [0.72 ; 1.56] à [0.96 ; 1.56] et donc FCC qui sera de $des = dth \otimes \alpha = \{520, ..., 1080\} \otimes [0.96; 1.56] = \{499.2, ..., 1684.8\}$.

Réduction de pui = 400. L'utilisateur ayant noté que la puissance de l'hélicoptère à maintenir est de 400cv, la contrainte C-3 va alors réduire la valeur de dth à {832, 896, 960}. Le domaine de des est alors de nouveau réduit en fonction des nouvelles valeurs de dth, ce qui donne $des = dth \otimes \alpha = \{832, 896, 960\} \otimes [0.96; 1.56] = \{798.72, ..., 1497.6\}$.

TABLE 3.3 : Calculs sur la base de cas de l'exemple d'utilisation d'une contrainte contextuelle avec des variables continues

(a) Initialisation

	dth	dre	α
1	1020	1224	1,2
2	1008	1441,44	1,43
3	360	514,8	1,43
4	720	518,4	0,72
5	836	668,8	0,8
6	480	576	1,2
7	608	583,68	0,96
8	780	624	0,8
9	728	582,4	0,8
10	672	960,96	1,43
11	448	524,16	1,17
12	544	636,48	1,17
13	576	673,92	1,17
14	756	604,8	0,8
15	756	982,8	1,3
16	672	1048,32	1,56
17	792	633,6	0,8
18	600	540	0,9
19	660	633,6	0,96
20	676	966,68	1,43
21	1092	1081,08	0,99
22	660	475,2	0,72
23	832	665,6	0,8
24	448	537,6	1,2
25	960	1267,2	1,32
26	540	772,2	1,43
27	576	552,96	0,96
28	748	673,2	0,9
29	792	1132,56	1,43
30	648	926,64	1,43

(b) amb = Sable

	$sim(amb)$	$sim(pay)$	Similarité globale
1	0,9	1	0,95
2	1	1	1
3	1	1	1
4	0,7	1	0,85
5	0,3	1	0,65
6	0,9	1	0,95
7	0,9	1	0,95
8	0,3	1	0,65
9	0,3	1	0,65
10	1	1	1
11	0,7	1	0,85
12	0,7	1	0,85
13	0,7	1	0,85
14	0,3	1	0,65
15	1	1	1
16	0,9	1	0,95
17	0,3	1	0,65
18	0,7	1	0,85
19	0,9	1	0,95
20	1	1	1
21	0,7	1	0,85
22	0,7	1	0,85
23	0,3	1	0,65
24	0,9	1	0,95
25	0,9	1	0,95
26	1	1	1
27	0,9	1	0,95
28	0,7	1	0,85
29	1	1	1
30	1	1	1

(c) pay = {Espagne, Tunisie}

	$sim(amb)$	$sim(pay)$	Similarité globale
1	0,9	1	0,95
2	1	1	1
3	1	1	1
4	0,7	0,2	0,45
5	0,3	0,2	0,25
6	0,9	1	0,95
7	0,9	0,2	0,55
8	0,3	0,2	0,25
9	0,3	0,2	0,25
10	1	1	1
11	0,7	0,8	0,75
12	0,7	0,8	0,75
13	0,7	0,8	0,75
14	0,3	0,2	0,25
15	1	1	1
16	0,9	0,8	0,85
17	0,3	0,2	0,25
18	0,7	1	0,85
19	0,9	0,2	0,55
20	1	1	1
21	0,7	1	0,85
22	0,7	0,2	0,45
23	0,3	0,2	0,25
24	0,9	1	0,95
25	0,9	1	0,95
26	1	1	1
27	0,9	0,2	0,55
28	0,7	1	0,85
29	1	1	1
30	1	1	1

Réduction de cst = C. Enfin, l'utilisateur fixe la valeur correspondant au constructeur à C, la contrainte C-3 va alors réduire le domaine de dth à $\{960\}$. On aura donc $des = dth \otimes \alpha = \{960\} \otimes [0.96; 1.56] = [921.6; 1497.6]$.

La contrainte C2 va alors réduire le domaine possible pour le pays dans lequel l'hélicoptère vole à $pay = \{\text{Tunisie, Espagne}\}$. La variable pay faisant partie de LVR, la recherche dans la base de cas va être effectuée à nouveau. La table 3.3c page précédente donne les valeurs des similarités globales pour chaque cas. On peut voir que le nombre de cas dont la similarité globale est supérieure à 0.9 a largement diminué. Désormais seuls les cas 1, 2, 3, 6, 10, 15, 20, 24, 25, 26, 29 et 30 sont conservés. L'agrégation des valeurs de dre et dth retenues réduit la valeur du paramètre α de [0.96 ; 1.56] à [1.2 ; 1.43] aboutissant ainsi à une modification de FCC telle que $des = dth \otimes \alpha = \{960\} \otimes [1.2; 1.43] = [1152; 1372.8]$.

3.3.3 Bilan du scénario de déroulement

Nous pouvons remarquer sur cet exemple que la contrainte contextuelle permet d'avoir une idée plus proche de la réalité du temps de montage, par rapport aux données indiquées par les constructeurs. En effet, dans cet exemple, une fois que l'utilisateur a saisi toutes les données, sans connaissance contextuelle, il obtient un temps de montage du moteur de 960 minutes. Or, il se trouve qu'étant données les conditions de vol de l'hélicoptère, le temps de maintenance à prévoir est plutôt situé entre 1152 et 1372.8 minutes. Bien entendu, ces valeurs, provenant des expériences passées, doivent être traitées avec prudence, mais elles permettent tout de même ici de voir que le temps de montage à prévoir est très probablement supérieur d'environ 200 minutes au temps de montage théorique.

Au fil des interventions de maintenance et donc de l'ajout de cas dans la base de cas, les valeurs du rapport entre la durée théorique et la durée réelle (ici α) doivent se stabiliser pour chaque contexte décrit par LVR. Cela signifie alors que la base de cas devient assez couvrante pour prendre en compte l'ensemble des combinaisons possibles. Une approche complémentaire consiste à déterminer les probabilités d'occurrence pour chaque valeur du temps de montage. Nous proposerons, dans le chapitre suivant, un système permettant, en plus du filtrage décrit ici, de déterminer les probabilités d'occurrence pour chaque valeur.

3.4 Application avec une variable symbolique

Comme nous l'avons indiqué en section 3.1.3 page 57, nous cherchons à identifier au mieux le type d'intervention qui peut avoir lieu sur les patins d'atterrissage d'un hélicoptère. Il s'agit donc, pour l'utilisateur de notre outil, de saisir la description de l'hélicoptère ainsi que ses conditions de vol. L'outil permet alors de définir les interventions de maintenance possibles en fonction de la description saisie. Cela permettra ainsi à l'utilisateur d'avoir une idée des interventions à effectuer, et ainsi de planifier au mieux la maintenance à venir.

Dans cet exemple, la connaissance générale impose une dépose en fonction de l'ancienneté des patins, cette connaissance est non négociable. La connaissance contextuelle va apporter des informations permettant de suggérer des anticipations sur les actions de maintenance à prévoir.

3.4.1 Définition du modèle d'application

3.4.1.1 Définition des variables utilisées

Pour cet exemple, nous n'utilisons que la variable définissant le modèle de l'hélicoptère (*mod*) parmi les variables présentes en section 2.1.2 page 38. Par contre, nous allons introduire de nouvelles variables dans le modèle (avec leur domaine de validité). Celles-ci sont des variables permettant de décrire les conditions de vol d'un hélicoptère ainsi que de définir le type d'intervention ayant lieu sur les patins d'atterrissage :

- conditions climatiques de vol (*cdt*) : {Calmes, Normales, Secousses, Violentes}. Il s'agit de décrire au mieux les conditions dans lesquelles le vol s'est effectué,
- soin du pilote (*pil*) : {Soigneux, Normal, Non soigneux, Agressif}. Les pilotes ont des manières de piloter les hélicoptères qui différent en fonction de leur caractère. Un pilote soigneux effectue des atterrissages qui usent peu les patins. À l'inverse, un pilote agressif a tendance à effectuer des atterrissages relativement brutaux, usant de manière prématurée les patins,
- ancienneté des patins (*anc*) : $[0\,;\,20]$ ($anc \in \mathbb{N}$). Il s'agit ici de noter le nombre de vols depuis la dernière dépose des patins. Une dépose signifie que l'on retire les patins de l'aéronef, qu'ils sont intégralement vérifiés, remplacés au besoin, et remontés. Le constructeur estime qu'une dépose revient à installer des patins neufs,

anc	pat_th
$[0; 19]$	$\{R1, R2, R3, R4\}$
20	$R4$

TABLE 3.4 : Données constructeur concernant la relation entre l'ancienneté des patins et la maintenance à effectuer

- l'intervention théorique issue des données constructeur à venir sur les patins (pat_th) : {R1, R2, R3, R4}. Cette variable permet de définir l'intervention qui a lieu sur les patins, avec :

  R1 : aucune autre intervention,

  R2 : obligation de revisser les points de fixation à l'hélicoptère,

  R3 : réparation *in situ*,

  R4 : dépose des patins.

  Il est obligatoire, avant chaque vol ou lors de chaque immobilisation de l'appareil, pour le maintenancier d'effectuer une vérification visuelle. Cette vérification peut aboutir à l'une des quatre interventions citées ci-dessus en fonction de l'état des patins,
- l'intervention estimée à venir sur les patins (pat_es) : {R1, R2, R3, R4}. Il s'agit là de l'intervention déduite par la contrainte contextuelle,
- l'intervention réellement effectuée sur l'hélicoptère (pat_re) : {R1, R2, R3, R4}. Il s'agit de l'intervention enregistrée dans la base de cas telle qu'elle a réellement été effectuée.

Les variables pat_th et pat_es ne sont pas enregistrées dans la base de cas car elles ne sont pas révélatrices du contexte. Par contre, la variable pat_re est enregistrée car elle apporte une connaissance basée sur les cas passés.

3.4.1.2 Données constructeur

Nous disposons de données constructeur qui imposent certaines interventions de maintenance en fonction de l'ancienneté des patins. Ces données sont écrites sous la forme d'une table de comptabilité que l'on peut voir table 3.4. Si l'on regarde en détail cette contrainte, nous pouvons voir qu'il est obligatoire de faire une maintenance de type R4 lorsque l'ancienneté des patins atteint vingt vols sans dépose (remettant ainsi à zéro l'ancienneté des patins). Autrement, l'ensemble des opérations peut être effectué. Ces données constructeur permettent donc d'obtenir une intervention théorique (pat_th).

TABLE 3.5 : Tables de similarité des variables *cdt* et *pil*

(a) *cdt*

cdt	Calmes	Normales	Secousses	Violentes
Calmes	1	0.5	0	0
Normales		1	0.5	0
Secousses			1	0.5
Violentes				1

(b) *pil*

pil	Soigneux	Normal	Non soigneux	Agressif
Soigneux	1	0.5	0	0
Normal		1	0.5	0
Non soigneux			1	0.5
Agressif				1

3.4.1.3 Matrice de similarité pour les variables de contexte

Les variables *cdt*, *pil* et *anc* font partie du contexte d'utilisation de l'hélicoptère. Cela signifie que des calculs de similarité vont être effectués sur ces variables. De fait, il est utile de définir leurs similarités locales. Les tables 3.5a et 3.5b présentent les matrices de similarité locale de *cdt* et *pil*. La formule représentée par l'équation 3.7 permet de calculer la similarité locale de *anc*. Cette formule revient à estimer que plus l'ancienneté cible est éloignée de l'ancienneté source, plus la similarité est faible. Une différence d'ancienneté de 20 implique une similarité locale de 0, une différence d'ancienneté de 10 implique une similarité locale de 0.5.

$$sim_{anc} = \frac{20 - |\mathcal{V}_{source} - \mathcal{V}_{cible}|}{20} \tag{3.7}$$

3.4.1.4 Base de cas

Le système dispose d'une base de cas qui est remplie au fil des maintenances sur les patins. Chaque cas de la base est complet, et n'invalide aucune contrainte précédemment définie. Il s'agit donc d'une base de cas valide, et donc utilisable dans notre contexte. De même que dans la section 3.3 page 64, nous disposons d'une base de 30 cas que nous pouvons voir table 3.6 page 77. Cette base est faiblement couvrante ($\frac{30}{5120}$ soit 0.58%) car comme nous l'avons calculé à l'aide de l'équation 3.8 page suivante, le produit cartésien de notre base de cas est de 5120.

$$\begin{aligned} Comb &= \mathcal{C}_{cdt} * \mathcal{C}_{mod} * \mathcal{C}_{pil} * \mathcal{C}_{anc} * \mathcal{C}_{pat\_re} \\ &= 4 * 4 * 4 * 20 * 4 \\ &= 5120 \end{aligned} \tag{3.8}$$

3.4.1.5 Définition de la contrainte contextuelle

Il reste maintenant à définir un à un les éléments de la contrainte contextuelle à entrer dans le système :

- Concernant LVR, l'expert sait que l'intervention sur les patins est affectée par les conditions de vol, le modèle de l'hélicoptère, le pilote ainsi que l'ancienneté des patins. On a donc $LVR = \{cdt, mod, pil, anc\}$,
- sim_g est la moyenne des similarités locales des variables de LVR,
- ms sera choisi par l'expert pour que les cas sélectionnés soient fortement similaires au cas rencontré par l'utilisateur. Une valeur de $ms = 0.8$ est choisie,
- concernant LVC, l'expert souhaite que le système sélectionne, pour l'ensemble des cas passés similaires, l'intervention ayant eu lieu sur les patins (pat_re),
- LVP ne comporte qu'une seule valeur qui est l'ensemble des valeurs de pat_re sélectionnées par la recherche dans la base de cas. La variable pat_re étant une variable symbolique, il s'agit en fait de faire l'union des valeurs de LVC,
- FP est inutilisée car la contrainte contextuelle ne requiert pas de calcul pour obtenir LVP,
- Concernant LVI, la contrainte est chargée de déduire les interventions possibles (pat_es) en fonction des interventions théoriques données par le constructeur (pat_th) et des informations obtenues à partir de la base de cas (pat_re). On aura donc $LVI = \{pat\_th, pat\_es\}$,
- enfin, FC est une formule logique permettant d'obtenir pat_es en fonction de pat_th et de pat_re. Si nous détaillons la formule que nous avons, alors il s'agit soit d'imposer $pat\_es = R4$ si pat_th est réduit à la seule valeur $R4$ ou de faire l'intersection des valeurs des variables pat_re et pat_th.

TABLE 3.6 : Base de cas de l'exemple avec variables symboliques du Chapitre 3

	cdt	*mod*	*pil*	*anc*	*pat_re*
1	Normales	Dauphin	Non soigneux	12	R2
2	Calmes	Gazelle	Agressif	16	R2
3	Calmes	Gazelle	Soigneux	2	R1
4	Calmes	Gazelle	Agressif	15	R2
5	Violentes	Gazelle	Normal	10	R1
6	Violentes	Tigre	Agressif	0	R1
7	Secousses	Gazelle	Agressif	16	R1
8	Normales	Gazelle	Soigneux	6	R1
9	Violentes	Puma	Normal	1	R1
10	Calmes	Tigre	Normal	10	R1
11	Violentes	Tigre	Normal	16	R4
12	Secousses	Puma	Normal	0	R1
13	Calmes	Tigre	Soigneux	2	R1
14	Normales	Puma	Soigneux	5	R1
15	Violentes	Tigre	Normal	3	R1
16	Violentes	Puma	Non soigneux	19	R2
17	Violentes	Puma	Normal	17	R4
18	Calmes	Dauphin	Soigneux	2	R1
19	Calmes	Gazelle	Normal	11	R1
20	Secousses	Dauphin	Normal	13	R1
21	Secousses	Puma	Agressif	13	R2
22	Calmes	Tigre	Agressif	1	R1
23	Violentes	Puma	Soigneux	3	R1
24	Violentes	Dauphin	Agressif	10	R1
25	Secousses	Tigre	Normal	18	R1
26	Secousses	Puma	Normal	17	R1
27	Normales	Dauphin	Non soigneux	8	R3
28	Calmes	Puma	Agressif	12	R1
29	Normales	Tigre	Soigneux	19	R4
30	Normales	Puma	Soigneux	8	R1

Pour résumer, la contrainte contextuelle est donc la formule représentée en équation 3.9.

$$
\begin{aligned}
cc_2 \quad (\quad & LVR = \{cdt, mod, pil, anc\}, \\
& sim_g = \frac{sim_{cdt} + sim_{mod} + sim_{pil} + sim_{anc}}{4}, \\
& ms = 0.8, \\
& LVC = \{pat\_re\}, \\
& LVP = \{pat\_re\}, \\
& FP = \emptyset, \\
& LVI = \{pat\_th, pat\_es\}, \\
& FCC : pat\_es = \begin{cases} R4 & \text{si } path\_th = R4 \\ pat\_th \cap pat\_re & \text{dans les autres cas} \end{cases} \\
) & \qquad (3.9)
\end{aligned}
$$

Maintenant que nous avons explicité l'ensemble des variables, des contraintes, la base de cas utilisée, ainsi que la contrainte contextuelle, nous allons dérouler un scénario afin de mettre en avant les intérêts de notre proposition.

3.4.2 Scénario de déroulement

Nous nous plaçons dans la situation d'un utilisateur qui, au fil de ses choix, sélectionne les valeurs qu'il souhaite à propos d'une intervention sur des patins d'hélicoptère. Nous allons présenter, pour chaque étape, la variable sélectionnée par l'utilisateur, les variables réduites par les contraintes du CSP et les valeurs déduites pour *pat_es* par la contrainte contextuelle.

Initialisation du système. À son état initial, le domaine de *mod* est complet de même que les domaines des variables présentées 3.4.1.1 page 73.

Il s'agit tout d'abord de faire fonctionner la contrainte contextuelle afin d'avoir une première valorisation du paramètre *pat_re* de *FCC*. La recherche dans la base de cas est donc faite avec *cdt* = {Calmes, Normales, Secousses, Violentes}, *mod* = {Gazelle, Dauphin, Tigre, Puma}, *pil* = {Soigneux, Normal, Non soigneux, Agressif} et *anc* = [0 ; 20]. Les mesures de similarités prenant la similarité la plus forte, et chaque cas étant renseigné, la similarité de chaque cas vaut 1. L'ensemble des cas est donc sélectionné après filtrage par *ms*. Nous avons donc pour valeurs de *pat_re* retenues {R1, R2, R3, R4}. Nous avons donc pour résultat $pat\_es = pat\_th \cap pat\_re$ = {R1, R2, R3, R4}.

Réduction de anc = 15. L'hélicoptère a effectué 15 vols depuis le dernier changement de patins. L'utilisateur entre donc cette information dans le système. De par l'utilisation de la fonction de similarité locale sur l'ancienneté de montage des patins (disponible en équation 3.7 page 75), le système en déduit les similarités locales (colonne « $sim(anc)$ » table 3.7a page 81). Le calcul de la similarité globale représenté dans la définition de la contrainte contextuelle (équation 3.9 page ci-contre) est effectué. Les résultats sont donnés dans la colonne « Similarité globale » de la table 3.7a page 81. Nous pouvons alors voir qu'étant donné le seuil de similarité fixé à 0.8, aucun cas de la base de cas n'est estimé trop lointain pour être écarté. La valeur retenue pour pat_re reste inchangée ce qui n'entraine aucune réduction supplémentaire pour pat_es : {R1, R2, R3, R4}.

Réduction de pil = Agressif. La variable pil faisant partie des variables de LVR, alors la contrainte contextuelle est déclenchée. En se basant sur la matrice de similarité disponible table 3.5b page 75, nous voyons que les valeurs *Soigneux* et *Normaux* ont une similarité locale de 0 par rapport à la valeur $Agressif$, que la valeur *Non soigneux* à une similarité locale de 0.75 par rapport à la valeur $Agressif$, et que la valeur $Agressif$ a une similarité locale de 1 par rapport à la valeur $Agressif$.

Nous pouvons ainsi voir dans la table 3.7b page 81 la nouvelle valeur de similarité globale avec cette nouvelle réduction. Le nombre de cas conservés dans LVC est ainsi réduit. Nous obtenons alors pat_re = {R1, R2} ce qui entraîne donc une valeur de $pat\_es = pat\_th \cap pat\_re$ = {R1, R2}.

Réduction de mod = Puma. La variable mod faisant partie des variables de LVR, la contrainte contextuelle est déclenchée. En se basant sur la matrice de similarité disponible en table 2.2a page 41, nous pouvons calculer la similarité locale de chaque cas.

Nous pouvons voir dans la table 3.7c page 81 la similarité globale des cas. Le nombre de cas conservés selon le seuil $ms \geq 0.8$ dans LVC est ainsi réduit. Nous obtenons alors pat_re = {R1, R2} ce qui entraîne donc une valeur de $pat\_es = pat\_th \cap pat\_re$ = {R1, R2}.

Réduction de cdt = Secousses. Enfin, l'utilisateur saisit les conditions de vol dans le système d'aide à la décision. La contrainte contextuelle va à nouveau être déclenchée. Le calcul de similarité sur cdt est effectué en fonction de la matrice de similarité donnée en table 3.5a page 75. La similarité globale est alors recalculée comme indiqué dans la table 3.7d page 81. La valeur

retenue de pat_re est {R2}. La valeur de pat_es indiquée à l'utilisateur est donc pat_es = {R2}.

3.4.3 Bilan du scénario de déroulement

Il est important de re-préciser que l'information donnée par la contrainte contextuelle ne reflète pas forcément la réalité mais plutôt une tendance issue des cas passés. Dans l'exemple que nous venons de dérouler, le peu de cas présents dans la base de cas implique une réduction trop forte du domaine de pat_es, mais cela sera comblé au fil de l'alimentation en cas de la base de cas et donc de l'augmentation de la couverture de la connaissance contextuelle. Très rapidement, au fur et à mesure du remplissage de la base de cas, très peu de réductions ne pourront avoir lieu. En effet, toutes les interventions R1, R2, R3 et R4 sont possibles à tout moment (exception faite de la vingtième visite). C'est pourquoi nous allons proposer dans le chapitre suivant d'effectuer un comptage sur les cas afin d'ajouter à la contrainte contextuelle des fréquences d'occurrence des différentes interventions en fonction du contexte.

3.5 Synthèse

Dans ce chapitre, nous avons introduit le principe de contrainte contextuelle. Une contrainte contextuelle est une contrainte paramétrique dont les paramètres dépendent du contexte dans lequel elle doit être utilisée. L'utilisation d'une contrainte contextuelle est nécessaire lorsque la connaissance générale est insuffisante pour couvrir un problème d'aide à la décision. La connaissance contextuelle permet alors d'apporter, sur un ensemble de variables données, des connaissances qui s'accumulent au fil du temps. Plus la couverture de la base de cas est importante, plus la contrainte contextuelle apporte des informations proches de la réalité.

La principale limite d'une contrainte contextuelle est par définition sa contextualité. Cela signifie qu'une contrainte contextuelle peut, par manque de couverture de la base de cas, interdire un certain nombre de valeurs de variables alors que celles-ci sont en réalité possibles. Il est donc nécessaire pour l'expert en charge de la modélisation des connaissances de l'outil utilisant une contrainte contextuelle, de décider si cette contrainte doit être traitée comme une contrainte classique (avec propagation) ou comme une contrainte de conseil (affichage sans propagation).

Lorsque la couverture de la base de cas augmente, des cas identiques (suivant leur description sur la liste des variables de LVR) se rencontrent. La

TABLE 3.7 : Calculs sur la base de cas de l'exemple d'utilisation d'une contrainte contextuelle avec des variables symboliques

(a) anc = 15

	sim_{anc}	sim_g
1	0,85	0,9625
2	0,95	0,9875
3	0,35	0,8375
4	1	1
5	0,75	0,9375
6	0,25	0,8125
7	0,95	0,9875
8	0,55	0,8875
9	0,3	0,825
10	0,75	0,9375
11	0,95	0,9875
12	0,25	0,8125
13	0,35	0,8375
14	0,5	0,875
15	0,4	0,85
16	0,8	0,95
17	0,9	0,975
18	0,35	0,8375
19	0,8	0,95
20	0,9	0,975
21	0,9	0,975
22	0,3	0,825
23	0,4	0,85
24	0,75	0,9375
25	0,85	0,9625
26	0,9	0,975
27	0,65	0,9125
28	0,85	0,9625
29	0,8	0,95
30	0,65	0,9125

(b) pil = Agressif

	sim_{anc}	sim_{pil}	sim_g
1	0,85	0,5	0,8375
2	0,95	1	0,9875
3	0,35	0	0,5875
4	1	1	1
5	0,75	0	0,6875
6	0,25	1	0,8125
7	0,95	1	0,9875
8	0,55	0	0,6375
9	0,3	0	0,575
10	0,75	0	0,6875
11	0,95	0	0,7375
12	0,25	0	0,5625
13	0,35	0	0,5875
14	0,5	0	0,625
15	0,4	0	0,6
16	0,8	0,5	0,825
17	0,9	0	0,725
18	0,35	0	0,5875
19	0,8	0	0,7
20	0,9	0	0,725
21	0,9	1	0,975
22	0,3	1	0,825
23	0,4	0	0,6
24	0,75	1	0,9375
25	0,85	0	0,7125
26	0,9	0	0,725
27	0,65	0,5	0,7875
28	0,85	1	0,9625
29	0,8	0	0,7
30	0,65	0	0,6625

(c) mod = Puma

	sim_{anc}	sim_{pil}	sim_{mod}	sim_g
1	0,85	0,5	0,3	0,6625
2	0,95	1	0,2	0,7875
3	0,35	0	0,2	0,3875
4	1	1	0,2	0,8
5	0,75	0	0,2	0,4875
6	0,25	1	0,8	0,7625
7	0,95	1	0,2	0,7875
8	0,55	0	0,2	0,4375
9	0,3	0	1	0,575
10	0,75	0	0,8	0,6375
11	0,95	0	0,8	0,6875
12	0,25	0	1	0,5625
13	0,35	0	0,8	0,5375
14	0,5	0	1	0,625
15	0,4	0	0,8	0,55
16	0,8	0,5	1	0,825
17	0,9	0	1	0,725
18	0,35	0	0,3	0,4125
19	0,8	0	0,2	0,5
20	0,9	0	0,3	0,55
21	0,9	1	1	0,975
22	0,3	1	0,8	0,775
23	0,4	0	1	0,6
24	0,75	1	0,3	0,7625
25	0,85	0	0,8	0,6625
26	0,9	0	1	0,725
27	0,65	0,5	0,3	0,6125
28	0,85	1	1	0,9625
29	0,8	0	0,8	0,65
30	0,65	0	1	0,6625

(d) cdt = Secousses

	sim_{anc}	sim_{pil}	sim_{mod}	sim_{cdt}	sim_g
1	0,85	0,5	0,3	0,5	0,5375
2	0,95	1	0,2	0	0,5375
3	0,35	0	0,2	0	0,1375
4	1	1	0,2	0	0,55
5	0,75	0	0,2	0,5	0,3625
6	0,25	1	0,8	0,5	0,6375
7	0,95	1	0,2	1	0,7875
8	0,55	0	0,2	0,5	0,3125
9	0,3	0	1	0,5	0,45
10	0,75	0	0,8	0	0,3875
11	0,95	0	0,8	0,5	0,5625
12	0,25	0	1	1	0,5625
13	0,35	0	0,8	0	0,2875
14	0,5	0	1	0,5	0,5
15	0,4	0	0,8	0,5	0,425
16	0,8	0,5	1	0,5	0,7
17	0,9	0	1	0,5	0,6
18	0,35	0	0,3	0	0,1625
19	0,8	0	0,2	0	0,25
20	0,9	0	0,3	1	0,55
21	0,9	1	1	1	0,975
22	0,3	1	0,8	0	0,525
23	0,4	0	1	0,5	0,475
24	0,75	1	0,3	0,5	0,6375
25	0,85	0	0,8	1	0,6625
26	0,9	0	1	1	0,725
27	0,65	0,5	0,3	0,5	0,4875
28	0,85	1	1	0	0,7125
29	0,8	0	0,8	0,5	0,525
30	0,65	0	1	0,5	0,5375

contrainte contextuelle proposée n'exploite pas cette forme de redondance. Il semble pourtant logique que les cas les plus fréquents doivent être pris en compte avec plus d'intérêt que les cas exceptionnels. Nous allons proposer une méthode prenant ces aspects en compte dans le chapitre suivant.

Chapitre 4

Prise en compte des connaissances contextuelles par le biais d'une contrainte contextuelle à comptage

Dans le chapitre précédent, nous avons montré comment un nouveau type de contrainte permettait d'utiliser de la connaissance contextuelle pour compléter des déductions effectuées avec de la connaissance générale dans le cas où cette dernière ne s'avérait pas assez couvrante. Cette nouvelle contrainte, dite contextuelle, a été exploitée sur des problèmes mettant en œuvre des paramètres continus ou des paramètres symboliques.

Lorsque la couverture de la base de cas augmente, le nombre de cas sélectionnés lors de la recherche augmente. Le fonctionnement de la contrainte contextuelle exploite alors des valeurs de paramètres plus nombreuses ou à occurrences multiples. Il semble alors intéressant de considérer que les valeurs les plus fréquemment proposées pour les paramètres sont les plus plausibles et en conséquence de ne considérer que ces valeurs plus fréquentes dans le processus de filtrage de la contrainte contextuelle. Les valeurs autorisées du paramètre de la contrainte étant moins nombreuses, son filtrage réduira alors les valeurs possibles des autres variables de la contrainte et en conséquence l'espace de solutions.

L'objet de ce chapitre est d'étudier cette amélioration. Nous présentons le principe du comptage dans un premier temps et illustrons ensuite son fonctionnement sur les deux exemples du chapitre 3.

4.1 Situation d'utilisation

4.1.1 Problématique

Conformément au chapitre 3 (section 3.2.2.3) le fonctionnement de la contrainte contextuelle détermine un ensemble de valeurs possibles pour le paramètre de la contrainte. Ces valeurs peuvent être :

- une liste de valeurs possibles pour les variables de type symbolique,
- un intervalle de valeurs possibles pour les variables de type continu.

Ces valeurs sont déterminées par une recherche effectuée en fonction des valeurs des variables de la liste LVR dans la base de cas supportant la contrainte contextuelle. Lorsque la base de cas grandit et devient plus couvrante, une recherche pour un contexte donné (c'est-à-dire un ensemble de variables valuées de LVR) fournira des valeurs (symboliques ou numériques) en plus grand nombre avec des distributions variées. Une distribution uniforme correspond à des valeurs de paramètres équiprobables, tandis qu'une distribution non uniforme révèlera des valeurs de paramètres plus ou moins plausibles pour un contexte donné bien sûr.

La contrainte contextuelle n'exploite pas cette information de distribution et cette notion de valeur de paramètre plus ou moins plausible. Le problème que nous considérons en conséquence consiste à élaborer et évaluer un principe de comptage et de filtrage permettant de déterminer une fréquence d'occurrence de valeurs de paramètres et de ne garder que les plus plausibles lors du filtrage de la contrainte contextuelle.

4.1.2 Proposition de solution

Nous proposons, afin d'améliorer le principe de contrainte contextuelle, d'apporter des informations supplémentaires à l'utilisateur avec l'ajout du comptage contextuel. Il s'agit alors :

- d'une fréquence d'occurrence pour chaque valeur possible des variables discrètes,
- d'une répartition discrétisée de la fréquence d'occurrence sur les intervalles des variables continues.

Nous verrons dans la section 4.2 page suivante que le comptage contextuel est une extension d'une contrainte contextuelle. Pour décrire brièvement le mécanisme, il s'agit ici de définir les variables permettant la description du contexte, la fonction de similarité globale permettant de calculer la similarité entre deux cas de la base de cas, la similarité minimale pour laquelle un cas de la base de cas est estimé comme assez proche du cas courant, les variables retenues dans la base de cas, les paramètres du comptage contextuel et enfin les formules permettant d'obtenir la valeur des paramètres en fonction des variables conséquentes.

4.1.3 Exemple illustratif

Nous allons utiliser dans ce chapitre les mêmes exemples illustratifs que dans le chapitre précédent. Voici un rappel de leurs cadres respectifs :

- Concernant l'exemple s'appliquant aux variables continues, il s'agit d'estimer au mieux le temps de montage d'un moteur sur un hélicoptère. Nous avons pu voir dans le chapitre précédent que nous obtenons une valeur du paramètre α qui est le rapport entre les durées réelles et théoriques enregistrées dans la base de cas. Ce paramètre α module alors la durée théorique du problème courant, permettant ainsi d'obtenir une durée estimée de montage. Le principal problème dans ce fonctionnement est la prise en compte équitable des valeurs cibles et des valeurs marginales. Cela signifie, par exemple, que si la durée de montage est, de manière rarissime, deux fois plus importante qu'à l'accoutumée, cette valeur sera tout autant prise en compte par la contrainte contextuelle, que les valeurs moyennes. Nous proposons donc d'associer aux valeurs du paramètre α discrétisé une fréquence d'occurrence dans un contexte similaire au problème courant.
- Concernant l'exemple s'appliquant aux variables symboliques, il s'agit d'estimer au mieux la maintenance à prévoir sur les patins d'atterrissage d'un hélicoptère (pat_es). Nous avons vu dans le chapitre précédent que tous les vingt vols au maximum, les patins d'atterrissage doivent être systématiquement déposés, mais qu'entre temps, quatre niveaux de révisions sont possibles. Aucune connaissance générale ne permet de savoir quelle révision est à prévoir, et la contrainte contextuelle permet seulement de montrer ce qui s'est déjà passé. Nous proposons donc d'associer aux valeurs de pat_es une fréquence d'occurrence dans un contexte similaire au problème courant. Cette fréquence d'occurrence sera nommée comptage contextuel.

4.2 Description d'une contrainte contextuelle à comptage

Dans cette section, nous allons présenter le mécanisme de comptage contextuel. Le fonctionnement de la recherche dans la base de cas est analogue à celui des contraintes contextuelles présenté au chapitre 3. La principale différence se situe au niveau de la forme des informations fournies par la contrainte contextuelle.

4.2.1 Définition d'une contrainte contextuelle à comptage

Le comptage contextuel complète la contrainte contextuelle telle que décrite dans le chapitre précédent. Une contrainte contextuelle sur laquelle on ajoute le comptage contextuel est une contrainte contextuelle à comptage. Une contrainte contextuelle à comptage (*ccc*) comprend quatre éléments tel que présenté en expression 4.1 page suivante, avec :

- *cc* la contrainte contextuelle sur laquelle s'appuie le comptage contextuel. Pour rappel, une contrainte contextuelle se formalise par un octuplet de la forme $csc(LVR, sim_g, ms, LVC, LVP, FP, LVI, FCC)$ avec LVR la liste des variables descriptives du contexte, sim_g la formule de similarité globale, ms le seuil de similarité minimale, LVC la liste des variables pertinentes pour le calcul des paramètres, LVP la liste des paramètres que l'on souhaite calculer, FP la formule d'obtention des paramètres de LVP en fonction des variables de LVC, LVI la liste des variables impactées par la contrainte contextuelle dans le moteur de filtrage et FCC la contrainte paramétrique permettant de lier LVC, LVP et LVI.
- fm la fréquence d'occurrence minimale pour laquelle les valeurs des paramètres de LVP sont conservées dans la contrainte contextuelle. Cet élément peut être fixé à *indéfini* dans le cas où l'on ne souhaite pas filtrer les fréquences d'occurrence mais obtenir une forme de « conseil ». Nous verrons en section 4.2.3 page 89 ce que cela implique.
- $LVPnb$ est une liste qui définit, pour chaque paramètre numérique continu de LVP, le nombre d'intervalles à créer dans le cas d'une discrétisation du paramètre. Cette valeur est fixée à *inutile* dans le cas d'un paramètre de type symbolique.

– $LVPdom$ est une liste qui définit, pour chaque paramètre numérique continu de LVP, son intervalle de définition si l'on souhaite effectuer une discrétisation à pas fixe. Cette valeur est fixée à *inutile* dans le cas d'un paramètre symbolique ou si l'on souhaite effectuer une discrétisation à pas mobile. Nous verrons plus en détail l'effet et le fonctionnement de la discrétisation en section 4.2.3 page 89.

$$ccc(cc, fm, LVPnb, LVPdom) \tag{4.1}$$

Le comptage contextuel porte sur la liste des variables retenues dans la base de cas (LVC), la liste des paramètres (LVP) et la formule permettant d'obtenir les paramètres (FP). Il s'agit de calculer la fréquence d'occurrence de chaque valeur de LVC et d'exploiter cette nouvelle connaissance.

4.2.2 Description du comptage contextuel

Le comptage contextuel utilise une technique issue du *data-mining* et plus particulièrement de l'algorithme $APRIORI$ présenté en section 1.3.2 page 26. Cet algorithme permet de retrouver des règles dans une base de données qui sont de la forme Antécédent $\Rightarrow$ Conséquent et auxquelles sont associées un support et une confiance (Support; Confiance). Nous notons l'association des règles et du couple support-confiance de la manière suivante : Antécédent $\Rightarrow$ Conséquent(Support; Confiance). L'algorithme $APRIORI$ part du principe que l'on ne connait pas l'ensemble des variables présentes dans les antécédents et les conséquents, ni leur valeur. La valeur du support des règles est donc importante pour savoir si la règle obtenue s'applique uniquement à quelques cas isolés ou à une grande part de la base de cas.

Dans notre situation, nous connaissons la liste des variables de l'antécédent de notre règle car elle correspond aux variables par lesquelles s'effectue la recherche dans la base de cas (LVR). Le support n'apporte pas d'information supplémentaire pour l'exploitation du comptage contextuel car il s'apparente à la couverture locale de la base de cas dans le contexte courant. Les conséquents de notre règle seront les variables de LVC. Nous connaissons donc déjà les variables conséquentes, mais pas leurs valeurs. Étant donné que dans notre situation, l'antécédent est LVR, que le conséquent est LVC et que le support n'apporte pas d'information utile, l'expression Antécédent $\Rightarrow$ Conséquent(Support; Confiance) employée en *data-mining* devient pour notre problème de comptage $LVR \Rightarrow LVC$(Confiance).

Pour chaque cas de la base de cas filtrée par la recherche de cas similaires du processus de raisonnement à partir de cas et par l'utilisation du seuil ms

de la contrainte contextuelle, nous appliquons FP sur les variables de LVC pour obtenir les valeurs des paramètres de LVP. Le comptage est ensuite effectué sur les valeurs obtenues pour chaque paramètre de LVP.

Le calcul de la fréquence d'occurrence ne peut se faire que sur des valeurs discrètes. Cela signifie qu'il faut effectuer un prétraitement sur les variables continues pour effectuer le comptage contextuel. Pour cela, nous proposons de les discrétiser de deux manières :

- la discrétisation du paramètre continu à pas fixe. Cette méthode permet une meilleure lisibilité des évolutions des réductions successives opérées par l'utilisateur. Il s'agit ici de définir le nombre d'intervalles que l'on souhaite obtenir ($LVPnb$) et l'intervalle de définition du paramètre ($LVPdom$). Les intervalles seront alors fixes quelles que soient les valeurs prises par le paramètre à discrétiser. Cette méthode possède néanmoins un inconvénient : l'obligation de connaitre à l'avance le domaine de définition du paramètre à discrétiser,
- la discrétisation du paramètre continu à pas mobile. Il s'agit alors, après le calcul des valeurs des paramètres de LVP pour chaque cas, de considérer la valeur minimale et la valeur maximale du paramètre à discrétiser. L'expert aura préalablement fixé le nombre d'intervalles sur lequel il souhaite effectuer la discrétisation ($LVPnb$). La discrétisation à pas mobile consiste alors à créer autant d'intervalles que l'expert l'a demandé en partant de la valeur minimale de la variable à discrétiser pour arriver à sa valeur maximale. Cette méthode est utilisable lorsqu'il est impossible de connaitre à l'avance le domaine de définition de la variable sur laquelle on effectue le comptage.

L'intérêt de la discrétisation à pas fixe est la lisibilité des résultats pour l'utilisateur. Cependant, nous devons souligner le risque de perte de précision, au fil des réductions, lié à la potentielle concentration des valeurs des paramètres sur un unique intervalle. L'exemple suivant illustre le comptage contextuel sur une variable continue discrétisée, d'une part avec un pas fixe et, d'autre part avec un pas mobile, et met en lumière les limites identifiées.

Exemple

Un paramètre prend pour valeur après une première recherche dans la base de cas les valeurs suivantes : {0.02, 0.03, 0.08, 0.13, 0.16, 0.17}. Dans l'exemple, la valeur de LVPnb est fixée à 3.

Les fréquences obtenues avec un intervalle prédéfini à [0 ; 0.3] sont représentées dans le tableau de gauche ci-dessous et les fréquences d'occurrence obtenues avec un intervalle variable (ici [0.02 ; 0.17])) sont représentées dans le tableau de droite ci-dessous.

Intervalle	*Fréquence*
$[0; 0.1[$	*50%*
$[0.1; 0.2[$	*50%*
$[0.2; 0.3]$	*0%.*

Intervalle	*Fréquence*
$[0.02; 0.07[$	*33%*
$[0.07; 0.12[$	*16%*
$[0.12; 0.17]$	*50%.*

Si la réduction utilisateur amène à obtenir seulement les valeurs {0.02, 0.03, 0.08} pour le paramètre, nous obtenons alors les résultats suivants : le tableau de gauche représente les fréquences avec un intervalle prédéfini (qui reste donc [0 ; 0.3]) et le tableau de droite donne les fréquences d'occurrence avec un intervalle variable (ici [0.02 ; 0.08]).

Intervalle	*Fréquence*
$[0; 0.1[$	*100%*
$[0.1; 0.2[$	*0%*
$[0.2; 0.3]$	*0%.*

Intervalle	*Fréquence*
$[0.02; 0.04[$	*66%*
$[0.04; 0.06[$	*0%*
$[0.06; 0.08]$	*33%.*

Nous voyons dans l'exemple ci-avant que la discrétisation à pas mobile permet d'obtenir des informations plus précises que pour une discrétisation à pas fixe mais qu'il est plus délicat d'interpréter les résultats pour un utilisateur.

Une fois la discrétisation de la variable effectuée, il faut effectuer le comptage des occurrences en passant les valeurs des paramètres présents dans LVP une à une et en incrémentant le compteur d'occurrence des intervalles correspondants.

Il faut ensuite diviser le nombre d'occurrences relevées par intervalles (variables continues discrétisées) ou par valeur (variables discrètes) par la taille de la base de cas filtrée par rapport aux valeurs de LVR afin de connaitre la fréquence d'occurrence.

L'algorithme 2 page suivante représente ce fonctionnement.

Suite à cette étape, l'ensemble des valeurs des paramètres présents dans LVP est alors déterminé avec la fréquence d'occurrence associée.

4.2.3 Fonctionnement de la contrainte contextuelle à comptage

Comme nous l'avons précédemment présenté, le comptage contextuel est un complément de la contrainte contextuelle. Nous allons décrire son fonctionnement dans cette section où nous faisons le parallèle avec la section 3.2.2 page 59 qui présentait le fonctionnement de la contrainte contextuelle. Le comptage contextuel se situe alors à l'étape 4 de ce fonctionnement. Le filtrage d'une contrainte contextuelle à comptage suit alors le déroulement suivant :

Alg. 2 Algorithme de calcul de la confiance pour chaque valeur de LVC

– $\therefore$ – *$BdCF$ est la liste des cas retenus en fonction de LVR, sim_g et ms*
– $\therefore$ – *LVP est la liste des paramètres à calculer*
– $\therefore$ – *FP est la formule permettant d'obtenir les paramètres en fonction des valeurs de LVC*
– $\therefore$ – *$Comptage$ est un tableau permettant le comptage des valeurs*
– $\therefore$ – *$Confiance$ est une matrice permettant le calcul de la fréquence d'occurrence pour chaque valeur*

Début
  Pour chaque p de LVP **Faire**
    Pour chaque *cas* de $BdCF$ **Faire**
      $Comptage[FP(LVC)] \leftarrow Comptage[FP(LVC)] + 1$
    Fin Pour
    Pour chaque *valeur* de $Comptage$ **Faire**
      $Confiance[p][valeur] \leftarrow \frac{Comptage[valeur]}{Taille(BdCF)}$
    Fin Pour
  Fin Pour
  Retourner($Confiance$)
Fin

1. la recherche en fonction des variables de LVR est effectuée dans la base de cas. La similarité globale de chaque cas par rapport aux valeurs des variables de LVR est calculée grâce à l'expression de sim_g. Seuls les cas dont la similarité globale est supérieure ou égale à ms sont retenus dans la base de cas filtrée,
2. pour chaque cas de la base de cas filtrée, la formule FP est appliquée sur les variables de LVC afin d'obtenir la valeur des paramètres présents dans LVP,
3. pour chaque paramètre présent dans LVP, un comptage est effectué afin de comptabiliser la fréquence d'occurrence de chaque valeur. Nous obtenons alors un ensemble de valeurs (pour les paramètres de type symbolique) ou un ensemble d'intervalles (pour les paramètres de type continu discrétisé) accompagné d'une fréquence d'occurrence pour chaque élément de cet ensemble,
4. les valeurs de LVI sont alors obtenues avec la contrainte FCC et les valeurs de LVP. Quatre situations peuvent être rencontrées :
   - fm est défini et les paramètres sont de type symbolique : les valeurs des paramètres de LVP dont la fréquence d'occurrence est inférieure ou égale à fm sont retirées des valeurs possibles pour ces paramètres. La contrainte contextuelle est alors filtrée avec les valeurs autorisées des paramètres de LVP,
   - fm est défini et les paramètres sont de type continu : les intervalles des paramètres de LVP dont la fréquence d'occurrence est inférieure ou égale à fm sont retirés des intervalles possibles pour ces paramètres. La contrainte contextuelle est alors filtrée avec les intervalles autorisés des paramètres de LVP,
   - fm est *indéfini* et les paramètres sont de type symbolique : la contrainte est filtrée en gardant pour information les fréquences d'occurrence des paramètres. L'utilisateur obtient alors un « conseil » sur les valeurs de LVI accompagnées de leur fréquence d'occurrence,
   - fm est *indéfini* et les paramètres sont de type continu : la contrainte est filtrée en gardant pour information les fréquences d'occurrence des intervalles des paramètres. L'utilisateur obtient alors un conseil sur les intervalles de LVI accompagnés de leur fréquence d'occurrence. Nous présenterons dans la section 4.2.4 page 93 les calculs permettant la propagation des fréquences d'occurrence sur des variables de type continu.

La figure 4.1 page 93 représente l'enchaînement des actions précédemment décrites.

Concernant la valeur de fm choisie par l'utilisateur, quatre grandes tendances se distinguent. Nous nommerons n le nombre d'éléments du paramètre, un élément étant une valeur lorsque le paramètre est de type symbolique ou un intervalle lorsque le paramètre est de type continu discrétisé.

- Si l'on fixe $fm = \frac{1}{n}$, cela signifie que dans le cas où les fréquences d'occurrence des éléments du paramètre sont toutes égales, alors ceux-ci sont tous pris en compte. Ce choix permet d'assurer la conservation d'au moins un élément pour le paramètre,
- si l'on fixe $fm > \frac{1}{n}$, cela signifie que l'on souhaite conserver les éléments dont la fréquence d'occurrence est forte. Ce choix est délicat car il est possible, dans le cas où la distribution des fréquences d'occurrence est proche d'une distribution uniforme, de ne retenir aucun élément pour le paramètre. Le cas échéant, la contrainte contextuelle n'apportera aucune information à l'utilisateur,
- si l'on fixe $fm < \frac{1}{n}$, cela signifie que même en cas de déséquilibre de la distribution des fréquences d'occurrence des éléments, il est possible que tous les éléments du paramètre soient conservés. Ce choix permet d'être sûr de conserver des éléments du paramètre, mais au risque d'en conserver certains qui sont assez peu pertinents dans le contexte courant,
- si l'on fixe $fm = 0$, cela signifie que tous les éléments du paramètre sont conservés, quelle que soit leur fréquence d'occurrence. Ce choix aboutit au même résultat que l'utilisation d'une contrainte contextuelle simple.

En fonction du contexte défini par l'utilisateur au fil de ses choix, les fréquences d'occurrence des éléments du paramètre évoluent, en augmentant ou en diminuant. Cela signifie donc que, pour chaque élément, sa fréquence peut être parfois supérieure, parfois inférieure, et parfois oscillante autour de fm. Le fonctionnement d'un moteur de filtrage de contraintes ne permet pas d'agrandir le domaine de validité d'une variable mais seulement de le réduire au fil des filtrages successifs. Les variables présentes dans LVI étant des variables du modèle de contraintes, elles ne peuvent pas voir leur domaine de validité augmenter. Malgré les oscillations possibles autour de la valeur de fm de la fréquence d'occurrence des éléments du paramètre, le moteur de filtrage n'autorisera pas, pour une variable de LVI, l'augmentation de son domaine de validité au fil des réductions utilisateur. Ce mécanisme s'effectue en ne conservant, pour un nouveau domaine de LVI, que son intersection entre le domaine précédent et le domaine qui vient d'être obtenu.

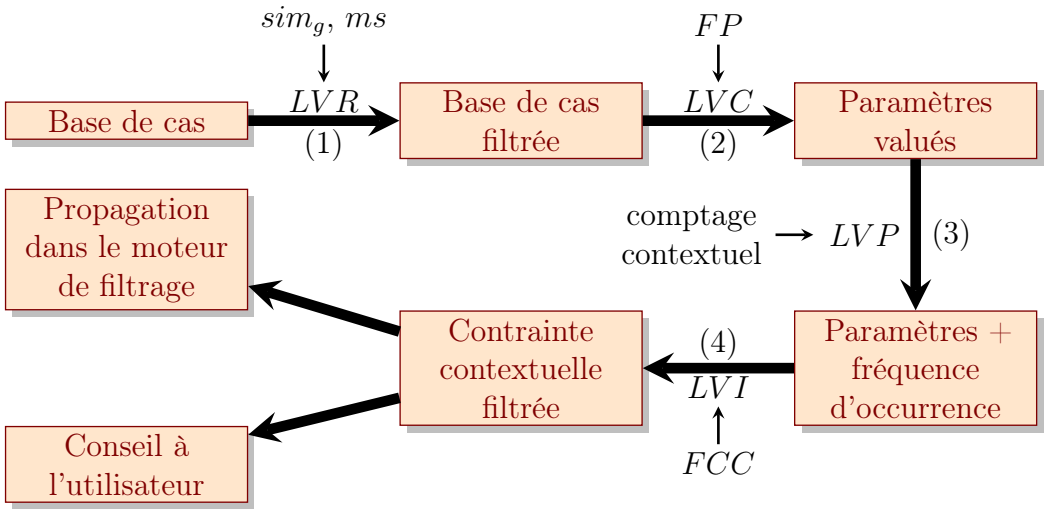


FIGURE 4.1 : Schéma du fonctionnement d'une contrainte contextuelle avec comptage

4.2.4 Opérations arithmétiques sur des intervalles pour fournir du conseil à l'utilisateur

Nous allons voir, dans cette section, notre proposition quant aux opérations arithmétiques sur des intervalles entre un paramètre discrétisé et une variable numérique continue. Ces opérations arithmétiques sont utilisées lorsque l'on souhaite fournir du conseil grâce au comptage contextuel. En effet, la contrainte contextuelle FCC doit être filtrée en prenant en compte les paramètres de LVP et les variables de LVI, les fréquences d'occurrence doivent alors être transposées des paramètres aux variables de LVI. La taille des intervalles étant inégale, il s'agit ensuite d'appliquer la fréquence d'occurrence à des intervalles de même taille.

Nous utiliserons les notations suivantes :

- V_1 la variable numérique discrétisée que l'on souhaite obtenir, accompagnée des fréquences d'occurrence de chacun de ses intervalles.
- p le paramètre numérique discrétisé accompagné des fréquences d'occurrence de ses intervalles.
- V_2 la variable numérique continue sur laquelle nous souhaitons effectuer l'opération arithmétique.
- $f([A;B])$ la fréquence d'occurrence d'un intervalle $[A;B]$.

La démarche d'obtention de V_1 est alors composée de 5 étapes :

1. Effectuer l'opération avec l'arithmétique des intervalles ([Moore 1966]) tout en conservant, pour chaque intervalle obtenu, la fréquence d'occurrence du paramètre.
2. Discrétiser les intervalles résultants de l'opération arithmétique de sorte que leurs intersections soient nulles et répartir les fréquences d'occurrence des intervalles en faisant l'hypothèse qu'au sein d'un intervalle la répartition est équiprobable.
3. Obtenir, pour chaque intervalle, la somme des fréquences d'occurrence auxquelles il est soumis.
4. Définir la taille d'intervalle adéquate afin de discrétiser en intervalles de même taille.
5. Calculer la fréquence d'occurrence pour chaque intervalle alors obtenu.

Ce cheminement permet d'obtenir la fréquence d'occurrence pour chaque intervalle du domaine de la variable V_1.

Exemple

Afin d'illustrer nos propos, nous utilisons un court exemple. Plaçons nous dans la situation où nous avons la formule $V_1 = p \otimes V_2$, avec :

- *p le paramètre obtenu suite au comptage contextuel. Nous nous situons dans le cas où $p = [0; 2]$ avec pour répartition des fréquences d'occurrences les valeurs suivantes :*

p	*fréquence*
$[0; 0,5[$	*10%*
$[0,5; 1[$	*20%*
$[1; 1,5[$	*60%*
$[1.5; 2]$	*10%*

- *V_2 la variable de LVI descriptive du problème sur lequel s'applique le comptage contextuel. Son domaine de définition est [50 ; 100].*

Nous allons alors dérouler les étapes précédemment décrites pour illustrer notre proposition :

1. *On effectue la multiplication arithmétique sur les intervalles ($V_1 = [0; 2] \otimes [50; 100]$) et l'on obtient les valeurs suivantes :*

$f(R)$	*Fréquence*
$[0; 50[$	*10%*
$[25; 100[$	*20%*
$[50; 150[$	*60%*
$[75; 200]$	*10%*

2. *On discrétise l'ensemble des intervalles obtenus de sorte que leurs intersections soient nulles et on répartit la fréquence d'occurrence :*

$$\begin{aligned}
f([0;50[) = 10\% \quad &\Rightarrow \quad f([0;25[) = 5\% \\
&\qquad\quad f([25;50[) = 5\% \\
f([25;100[) = 20\% \quad &\Rightarrow \quad f([25;50[) = 6.66\% \\
&\qquad\quad f([50;75[) = 6.66\% \\
&\qquad\quad f([75;100[) = 6.66\% \\
f([50;150[) = 60\% \quad &\Rightarrow \quad f([50;75[) = 15\% \\
&\qquad\quad f([75;100[) = 15\% \\
&\qquad\quad f([100;150[) = 30\% \\
f([75;200]) = 10\% \quad &\Rightarrow \quad f([75;100[) = 2\% \\
&\qquad\quad f([100;150[) = 4\% \\
&\qquad\quad f([150;200]) = 4\%
\end{aligned}$$

3. *Il reste à sommer les fréquences sur chaque intervalle et l'on obtient la fréquence d'occurrence de la variable V_1 sur chacun de ses intervalles :*

$f(V_1)$	*Fréquence*
$[0;25[$	*5%*
$[25;50[$	*5 + 6.66 = 11.66%*
$[50;75[$	*6.66 + 15 = 21.66%*
$[75;100[$	*6.66 + 15 + 2 = 23.66%*
$[100;150[$	*30 + 4 = 34%*
$[150;200]$	*4%*

4. *Ici, la taille d'intervalle qui permet d'obtenir des intervalles de même taille est 25. Nous aurons alors 8 intervalles d'une taille de 25.*
5. *La redistribution des fréquences d'occurrences donne alors les valeurs suivantes :*

$f(V_1)$	*Fréquence*
$[0; 25[$	*5%*
$[25; 50[$	*11.66%*
$[50; 75[$	*21.66%*
$[75; 100[$	*23.66%*
$[100; 125[$	*17%*
$[125; 150[$	*17%*
$[150; 175[$	*2%*
$[150; 200]$	*2%*

Nous avons désormais la possibilité de fournir à l'utilisateur ces fréquences et ainsi de lui prodiguer du conseil sur les variables continues.

Dans les sections à venir, nous allons appliquer une contrainte contextuelle à comptage sur deux exemples, l'un s'appliquant à des variables numériques et l'autre à des variables symboliques.

4.3 Exemple illustrant la contrainte contextuelle à comptage avec un paramètre continu

De même que dans le chapitre précédent, nous allons utiliser l'exemple du montage d'un moteur sur un hélicoptère. Le principe est d'estimer au mieux la durée de montage d'un couple moteur / hélicoptère en utilisant de la connaissance contextuelle en complément de la connaissance générale donnée par le constructeur.

Nous rappelons ici l'ensemble des variables de description ainsi que leurs domaines tels que définis en section 2.1.2 page 38 et 3.3.1.1 page 65 :

- *mod* est le modèle de l'hélicoptère : {Puma, Dauphin, Tigre, Gazelle}.
- *cst* est le constructeur de l'appareil : {A, B, C}.
- *pui* est la puissance du moteur : {100, 200, 300, 400, 500}.
- *amb* est l'ambiance de vol de l'hélicoptère : {Terre, Mer, Sable, Froid}.
- *pay* est le pays de vol de l'hélicoptère : {Espagne, Tunisie, Suède, France}.
- *dth* est la durée théorique de montage donnée par le constructeur, elle est calculée en fonction de *mod*, *cst* et *pui*.
- *dre* est la durée réelle passée au montage pour les cas passés.
- *des* est la durée de montage estimée par notre outil.

4.3.1 Définition de la contrainte contextuelle à comptage

Nous souhaitons dans notre situation estimer au mieux la durée de montage à prévoir du couple moteur / hélicoptère. Cette durée n'étant pas enregistrée dans la base de cas, nous souhaitons obtenir des informations sur le rapport entre les durées réelles et les durées théoriques. Il s'agit donc de la variable α telle que nous l'avons définie dans le chapitre précédent. α est alors appliquée à dth afin d'avoir la meilleure estimation possible de des.

Afin de garder au maximum le parallèle entre le chapitre précédent et ce chapitre, nous allons garder la contrainte contextuelle cc_1 définie en section 3.6 page 69, pour rappel, nous la recopions dans l'expression 4.2.

Nous allons définir deux comptages contextuels, l'un permettant de fournir du conseil sur des, et l'autre effectuant un filtrage automatique sur α afin de permettre la propagation automatique de la contrainte contextuelle. Les deux formulations du comptage contextuel sont données dans l'expression 4.4. Le comptage contextuel permettant de fournir du conseil est noté ccc_{1-1}, celui effectuant le filtrage automatique est noté ccc_{1-2}, nous pouvons voir que nous avons choisi $fm = 5\%$. Étant donné que le nombre de pas de α est de vingt, nous avons décidé de fixer le seuil à l'inverse du nombre d'éléments du paramètre comme nous l'avions présenté en section 4.2.3 page 89.

$$
\begin{aligned}
cc_1 \quad (\quad & LVR = \{amb, pay\}, \\
& sim_g = \frac{sim_{amb} + sim_{pay}}{2}, \\
& ms = 0.9, \\
& LVC = \{dth, dre\}, \\
& LVP = \{\alpha\}, \\
& FP = \{\alpha = \frac{dre}{dth}\}, \\
& LVI = \{des, dth\}, \\
& FCC : des = dth \otimes \alpha)
\end{aligned}
\tag{4.2}
$$

(4.3)

$$
\begin{array}{lll}
ccc_{1-1} & (& cc = cc_1, \\
 & & fm = \textit{indéfini}, \\
 & & LVPnb = \{20\}, \\
 & & LVPdom = \{[0;2]\}) \\
ccc_{1-2} & (& cc = cc_1, \\
 & & fm = 5\%, \\
 & & LVPnb = \{20\}, \\
 & & LVPdom = \{[0;2]\})
\end{array}
\tag{4.4}
$$

4.3.2 Base de cas

Pour cet exemple, nous avons utilisé une base de cas valide et complète au sens défini en section 1.2.3.3 page 21 et constituée de 3000 cas. Cela représente donc une couverture théorique de 3.1% ($\frac{3000}{96000}$). Cela représente une couverture faible, mais bien plus importante que celle que nous avions dans le chapitre 3 (0.03%). Cette base de cas au format csv est disponible à l'adresse `http://perso.mines-albi.fr/~codetdeb/these/bdc-continue-3000.csv`.

4.3.3 Scénario de déroulement d'un comptage contextuel avec conseil

Dans ce scénario, nous nous situons exactement dans la même situation que dans le scénario déroulé dans le chapitre précédent (section 3.3.2 page 69) permettant d'illustrer les contraintes contextuelles sur une variable continue. Les réductions utilisateurs sont les mêmes, les résultats ne seront pas les mêmes du fait de la base de cas différente, et le conseil apporté sera bien plus important grâce aux calculs de fréquence d'occurrence de valeurs sur les paramètres.

Initialisation du système À l'initialisation du système, les domaines des variables du moteur de filtrage sont tels que présentés en section 2.1.2 page 38. *des* et *dre* ont pour domaines $[0\,;+\infty[$ et *dth* a pour domaine {280, ..., 1080}.

La contrainte contextuelle à comptage va alors être déclenchée une première fois afin d'obtenir une valeur de α ainsi qu'une fréquence d'occurrence des valeurs de son domaine. Ces fréquences sont données dans la première colonne de la table 4.1 page 100. Chaque fréquence d'occurrence de α est alors

appliquée aux valeurs de *dth* afin d'obtenir des fréquences d'occurrences de *des* avec le cheminement que nous avons présenté en section 4.2.4 page 93.

La fréquence d'occurrence de chaque valeur de *des* est alors affichée à l'utilisateur sous forme d'une graphique tel que présenté en figure 4.2 page suivante.

1. **Réduction de mod = Puma**

   La réduction du modèle d'hélicoptère au type Puma entraine une réduction de *dth*. α n'est pas modifié car aucune variable de la contrainte contextuelle à comptage n'est impliquée par la réduction. *des* va alors être recalculé en fonction de la nouvelle valeur de *dth* et affiché à l'utilisateur tel que nous le voyons dans le graphique 4.2 page suivante.

2. **Réduction de amb = Sable**

   La réduction de l'ambiance de vol à sable entraîne un nouveau calcul de α dont on peut observer le résultat en table 4.1 page suivante. Ces nouvelles fréquences d'occurrence permettent de recalculer *des* en fonction des valeurs de *dth*. Ces valeurs sont alors affichées à l'utilisateur comme représenté dans le graphique 4.2 page suivante (courbe non visible car recouverte intégralement par la courbe correspondant à *pui* = 400).

3. **Réduction de pui = 400**

   La réduction de la puissance du moteur à 400 n'entraîne pas de nouveau calcul de α, par contre, la valeur de *dth* est à nouveau calculée, ce qui permet au système de proposer de nouvelles fréquences d'occurrence de *des* tel que nous le voyons dans le graphique 4.2 page suivante.

4. **Réduction de cst = C**

   Le constructeur étant le constructeur C, l'intervention ne peut avoir lieu qu'en Tunisie ou en Espagne. La variable *pay* appartenant à LVR, la valeur de α est à nouveau calculée, on obtient alors de nouvelles fréquences d'occurrence pour α. La répartition des occurrences de α sur l'ensemble de son domaine discrétisé est disponible en table 4.1 page suivante. De plus, *dth* est désormais fixé à 960 minutes. Cela permet donc de calculer une probabilité pour *des* et de l'afficher à l'utilisateur tel que nous le voyons dans le graphique 4.2 page suivante.

TABLE 4.1 : Liste des valeurs prises par α au fil des réductions utilisateur

α	Initialisation	mod = Puma	amb = Sable	pui = 400	cst = C
[0; 0.1[	0	0	0	0	0
[0.1; 0.2[	0	0	0	0	0
[0.2; 0.3[	0	0	0	0	0
[0.3; 0.4[	0	0	0	0	0
[0.4; 0.5[	0	0	0	0	0
[0.5; 0.6[	0	0	0	0	0
[0.6; 0.7[	2.03	2.03	0	0	0
[0.7; 0.8[	7.03	7.03	0	0	0
[0.8; 0.9[	6.6	6.6	1.38	1.38	4.24
[0.9; 1[	15.47	15.47	10.91	10.91	19.38
[1; 1.1[	14.97	14.97	11.7	11.7	15.56
[1.1; 1.2[	11.73	11.73	8.74	8.74	15.56
[1.2; 1.3[	13.33	13.33	16.43	16.43	19.24
[1.3; 1.4[	11.4	11.4	16.69	16.69	18.25
[1.4; 1.5[	5.4	5.4	10.45	10.45	7.21
[1.5; 1.6[	4.6	4.6	9.07	9.07	0.57
[1.6; 1.7[	4.7	4.7	9.26	9.26	0
[1.7; 1.8[	2.67	2.67	5.26	5.26	0
[1.8; 1.9[	0.07	0.07	0.13	0.13	0
[1.9; 2[	0	0	0	0	0

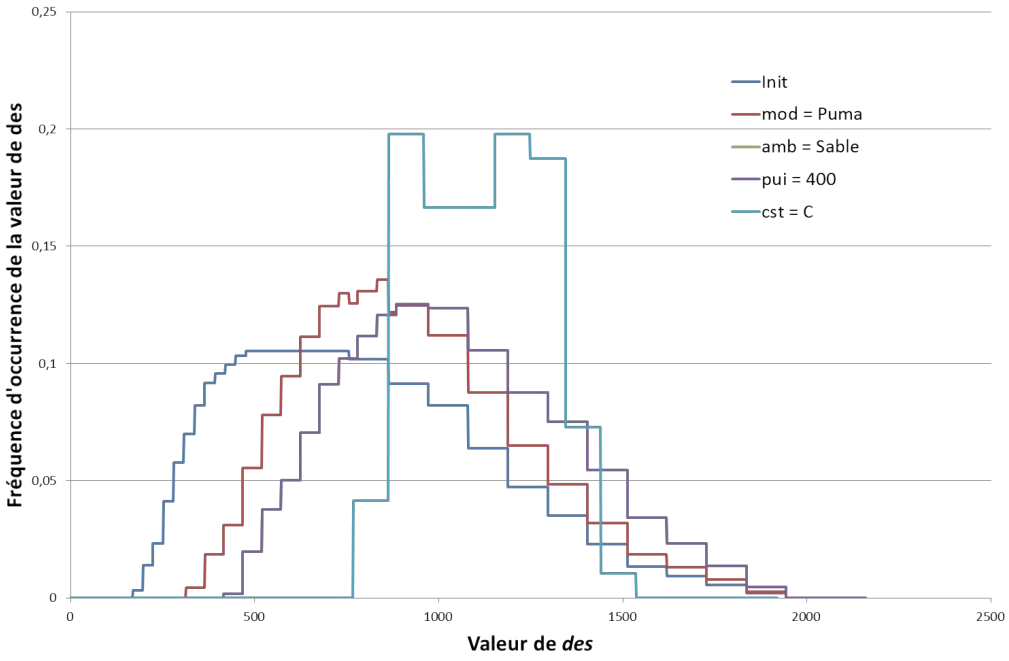


FIGURE 4.2 : Fréquence d'occurrence des valeurs de *des* au fil des réductions

TABLE 4.2 : Liste des valeurs prises par α au fil des réductions utilisateur

α	Initialisation	mod = Puma	amb = Sable	pui = 400	cst = C
$[0; 0.1[$	0	0	0	0	0
$[0.1; 0.2[$	0	0	0	0	0
$[0.2; 0.3[$	0	0	0	0	0
$[0.3; 0.4[$	0	0	0	0	0
$[0.4; 0.5[$	0	0	0	0	0
$[0.5; 0.6[$	0	0	0	0	0
$[0.6; 0.7[$	2.03	2.03	0	0	0
$[0.7; 0.8[$	7.03	7.03	0	0	0
$[0.8; 0.9[$	6.6	6.6	1.38	1.38	4.24
$[0.9; 1[$	15.47	15.47	10.91	10.91	19.38
$[1; 1.1[$	14.97	14.97	11.7	11.7	15.56
$[1.1; 1.2[$	11.73	11.73	8.74	8.74	15.56
$[1.2; 1.3[$	13.33	13.33	16.43	16.43	19.24
$[1.3; 1.4[$	11.4	11.4	16.69	16.69	18.25
$[1.4; 1.5[$	5.4	5.4	10.45	10.45	7.21
$[1.5; 1.6[$	4.6	4.6	9.07	9.07	0.57
$[1.6; 1.7[$	4.7	4.7	9.26	9.26	0
$[1.7; 1.8[$	2.67	2.67	5.26	5.26	0
$[1.8; 1.9[$	0.07	0.07	0.13	0.13	0
$[1.9; 2[$	0	0	0	0	0

4.3.4 Scénario de déroulement d'une contrainte contextuelle à comptage avec filtrage automatique du paramètre

Dans ce scénario, l'utilisateur va effecteur exactement les mêmes filtrages que dans le scénario précédent, la différence réside dans le fait que la contrainte contextuelle va propager automatiquement les résultats dans le moteur de filtrage. Nous utiliserons alors le comptage contextuel ccc_{1-2} défini en équation 4.4 page 98.

Les valeurs de α inférieures à 5% seront alors automatiquement retirées des valeurs possibles pour le paramètre de la contrainte contextuelle. Nous allons donc effectuer les réductions une à une et visualiser les résultats qui sont donc, cette fois-ci, imposés à l'utilisateur du système d'aide à la décision. Les valeurs de α obtenues sont les mêmes que dans le scénario précédent. Dans le tableau 4.2 nous résumons ces valeurs, les cases grisées correspondent aux valeurs de α qui sont retirées automatiquement par le comptage contextuel.

1. **Initialisation**

   Les valeurs de α conservées sont $[0.7; 1.5[$. La valeur de dth étant $\{280, ..., 1080\}$, la valeur de des est alors réduite à $[196; 1620]$.

2. **Réduction de mod = Puma**

   Les valeurs de α conservées ne sont pas altérées, il s'agit toujours de $[0.7; 1.5[$. La valeur de dth étant cette fois de {520, ..., 1080}, la valeur de des imposée à l'utilisateur devient $[364; 1620[$.

3. **Réduction de amb = Sable**

   α est recalculé en fonction de l'ambiance, les valeurs $[0.7; 0.9[$ sont alors retirées des valeurs possibles pour α. Nous pouvons voir ici que, bien qu'à nouveau supérieures à 5%, les valeurs $[1.5; 1.7[$ ne sont pas remises dans le domaine de validité de α. Cette singularité est due au fait que l'on ne peut pas, dans un moteur de filtrage de contrainte, augmenter le domaine d'une variable au fil des réductions comme nous l'avons expliqué en section 4.2.3 page 89. La valeur de dth est toujours de {520, ..., 1080}, la valeur de des propagée sera alors $[468; 1620[$.

4. **Réduction de pui = 400**

   α n'est pas modifié par cette réduction, par contre, la valeur de dth est réduite à {832, 896, 960}. La valeur calculée pour des sera donc de $[748.8; 1440[$.

5. **Réduction de cst = C**

   La réduction du constructeur à la valeur C entraîne une réduction de dth à {960} ainsi qu'une modification de α mais nous pouvons voir dans le tableau 4.2 page précédente qu'aucune valeur précédemment au-dessus de 5% ne tombe en-dessous. La valeur de des donnée à l'utilisateur sera donc de $[864; 1440[$.

Cet exemple nous permet donc de voir que la propagation automatique de la contrainte contextuelle avec comptage contextuel fournit des valeurs plus strictes à l'utilisateur que le fait de lui donner du conseil sur les valeurs possible de des. Le principal risque de cette propagation automatique est de mettre de côté des valeurs qui auraient pu être pertinentes, il est donc important de fixer un seuil assez faible pour ne pas écarter de solutions pertinentes.

4.3.5 Bilan des scénarios de déroulement

Nous pouvons voir que l'utilisation de la contrainte contextuelle à comptage avec conseil apporte des informations supplémentaires à l'utilisation de la contrainte contextuelle. En effet, la contrainte contextuelle permettait d'obtenir les valeurs possibles de des, mais sans aucune information sur la fréquence d'occurrence de chacune. Grâce à la contrainte contextuelle à comptage, il est

désormais possible de connaitre plus précisément la fréquence d'occurrence des valeurs de *des* grâce au comptage contextuel sur les valeurs de α.

L'utilisation de la contrainte contextuelle à comptage avec filtrage automatique permet de réduire les valeurs de *des* du modèle de contraintes. Ce fonctionnement permet alors, lors de l'intégration d'une contrainte contextuelle à comptage dans un moteur de filtrage, de réduire des domaines plus efficacement qu'avec une contrainte contextuelle sans comptage.

4.4 Exemple illustrant le comptage contextuel avec un paramètre symbolique

De même que dans le chapitre précédent, nous allons utiliser l'exemple de la révision des patins d'atterrissage des hélicoptères. Le principe est d'estimer au mieux la maintenance à prévoir sur les patins en utilisant de la connaissance contextuelle en complément de la connaissance générale donnée par le constructeur.

Nous rappelons ici l'ensemble des variables de description ainsi que leurs domaines tels que définis en section 2.1.2 page 38 et 3.3.1.1 page 65 :

- *cdt* sont les conditions de vol de l'hélicoptère : {Calmes, Normales, Secousses, Violentes}.
- *mod* est le modèle de l'hélicoptère : {Puma, Dauphin, Tigre, Gazelle}.
- *pil* est le comportement du pilote de l'hélicoptère : {Soigneux, Normal, Non soigneux, Agressif}.
- *anc* est le nombre de vols depuis lequel les patins n'ont pas été déposés : [0 ;20].
- *pat_th* est l'intervention théorique prévue par le constructeur, elle est définie en fonction de *anc*.
- *pat_re* est l'intervention réellement effectuée sur l'hélicoptère.
- *pat_es* est l'intervention prévue à effectuer sur l'hélicoptère.

4.4.1 Définition de la contrainte contextuelle à comptage

Nous souhaitons dans notre situation estimer au mieux la probabilité qu'a le maintenancier de réaliser une certaine intervention sur les patins. Il s'agit

ici de comptabiliser la fréquence des interventions passées dans un même contexte que le contexte courant.

Afin de conserver au maximum le parallèle entre le chapitre précédent et ce chapitre, nous nous plaçons exactement dans la même configuration pour les éléments du comptage contextuel. L'expression 4.5 rappelle la contrainte contextuelle que nous utilisions dans le chapitre 3. Nous pouvons voir les deux configurations du comptage contextuel dans l'expression 4.6, le comptage contextuel ccc_{2-1} illustre un comptage contextuel sans filtrage automatique et le comptage contextuel ccc_{2-2} avec un filtrage automatique pour les paramètres dont la fréquence d'occurrence est inférieur à 5%.

$$
\begin{aligned}
cc_2 \quad (\quad & LVR = \{cdt, mod, pil, anc\}, \\
& sim_g = \frac{sim_{cdt} + sim_{mod} + sim_{pil} + sim_{anc}}{4}, \\
& ms = 0.8, \\
& LVC = \{pat\_re\}, \\
& LVP = \{pat\_re\}, \\
& FP = \varnothing \\
& LVI = \{pat\_th, pat\_es\}, \\
& FCC : pat\_es = \begin{cases} R4 & \text{si } path\_th = R4 \\ pat\_th \cap pat\_re & \text{dans les autres cas} \end{cases} \\
&)
\end{aligned}
\tag{4.5}
$$

$$
\begin{aligned}
ccc_{2-1} \quad (\quad & cc = cc_2, \\
& fm = \textit{indéfini}, \\
& LVPnb = \varnothing, \\
& LVPdom = \varnothing) \\
ccc_{2-2} \quad (\quad & cc = cc_2, \\
& fm = 5\%, \\
& LVPnb = \varnothing, \\
& LVPdom = \varnothing)
\end{aligned}
\tag{4.6}
$$

4.4.2 Base de cas

Pour cet exemple, nous avons utilisé une base de cas valide et complète dans le sens défini en section 1.2.3.3 page 21 disposant de 3000 cas. Cela représente

donc une couverture théorique de 58.6% ($\frac{3000}{5120}$). Cela représente une couverture moyenne bien plus importante que celle que nous avions dans le chapitre 3 (0.58%). Cette base de cas au format csv est disponible à l'adresse http://perso.mines-albi.fr/~codetdeb/these/bdc-continue-3000.csv.

4.4.3 Scénario de déroulement d'un comptage contextuel avec conseil

Dans ce scénario, la situation est identique à celle du chapitre précédent (section 3.4.2 page 78) permettant d'illustrer les contraintes contextuelles sur une variable symbolique.

1. **Initialisation du système**
   À son état initial, aucune contrainte ne vient réduire de domaines. Il s'agit alors de déclencher le comptage contextuel afin d'obtenir une première fréquence d'occurrence de chaque valeur de pat_re. La similarité globale de chaque cas par rapport à la source est de 1 car la source autorise l'intégralité du domaine de définition des variables, tous les cas sont donc pris en considération.
   Le comptage contextuel va donc fournir pour chaque valeur possible de pat_re ({R1, R2, R3, R4}) sa fréquence d'occurrence. Nous obtenons les résultats donnés dans le tableau 4.3a page suivante. Il est donc très clair qu'il y a de fortes chances (68.73%) de n'avoir rien à faire (R1).
2. **Réduction de $anc = 15$**
   L'utilisateur étant en présence d'un hélicoptère ayant réalisé quinze vols depuis la dernière dépose, il s'agit alors de calculer les nouvelles similarités en connaissance de cause. Il se trouve que ce calcul ne va rien changer aux valeurs retenues dans la base de cas. En effet, la différence maximale se situe par rapport à un hélicoptère n'ayant réalisé aucun vol depuis sa dernière dépose. La similarité locale de l'ancienneté est alors de 0.25 ($\frac{20-|15-0|}{20} = \frac{5}{20}$). La similarité globale est donc au minimum de 0.8125 ($\frac{0.25+1+1+1}{4}$). Tous les cas sont donc conservés car $ms = 0.8$.
   Les fréquences d'occurrence restent alors les mêmes, comme il est possible de le voir dans le tableau 4.3b page suivante.
3. **Réduction de pil = Non soigneux**
   L'application de pil = Non soigneux va commencer à effectuer la sélection dans la base de cas. En effet, les cas dont le pilote est soigneux seront retirés, ainsi que certains dont le pilote est normal ou agressif (en fonction de l'ancienneté de dépose des patins).

TABLE 4.3 : Confiances de valeurs de pat_re après chaque réduction utilisateur

(a) Initialisation

pat_re	Confiance
R1	68.73%
R2	14.97%
R3	4.93%
R4	11.37%

(b) anc = 15

pat_re	Confiance
R1	68.73%
R2	14.97%
R3	4.93%
R4	11.37%

(c) pil = Non soigneux

pat_re	Confiance
R1	61.65%
R2	18.89%
R3	6.37%
R4	13.09%

(d) mod = Tigre

pat_re	Confiance
R1	60.3%
R2	19.92%
R3	5.91%
R4	13.87%

(e) cdt = Secousses

pat_re	Confiance
R1	59.82%
R2	17.79%
R3	6.13%
R4	16.26%

Les nouvelles fréquences d'occurrence sont alors calculées, nous obtenons les résultats présentés dans le tableau 4.3c. Nous voyons alors que la probabilité de n'avoir rien à faire (R1) diminue (passage de 68.73 à 61.65) et que la fréquence des autres interventions augmente donc.

4. **Réduction de mod = Tigre**
   Le Tigre est un hélicoptère dans la moyenne d'un point de vue de la fiabilité de ses équipements. Nous le voyons lorsque nous effectuons la réduction du modèle d'hélicoptère. La fréquence d'occurrence des différentes interventions est relativement peu affectée par cette réduction comme nous pouvons le voir dans la table 4.3d.
5. **Réduction de cdt = Secousses**
   L'hélicoptère ayant évolué dans des conditions de secousses, cela a tendance à fortement dégrader l'état des patins d'atterrissage. La sélection des cas étant effectuée à nouveau, ainsi que le calcul de fréquence d'occurrence, nous voyons dans le tableau 4.3e qu'une fois de plus, les probabilités qu'a le maintenancier de n'avoir rien à faire s'amoindrissent alors que la fréquence de R4 augmente de manière importante.

4.4.4 Scénarios de déroulement d'une contrainte contextuelle à comptage avec conseil

Nous allons maintenant dérouler quatre scénarios qui prennent en compte différents choix de l'utilisateur. Nous conservons le même ordre de choix afin

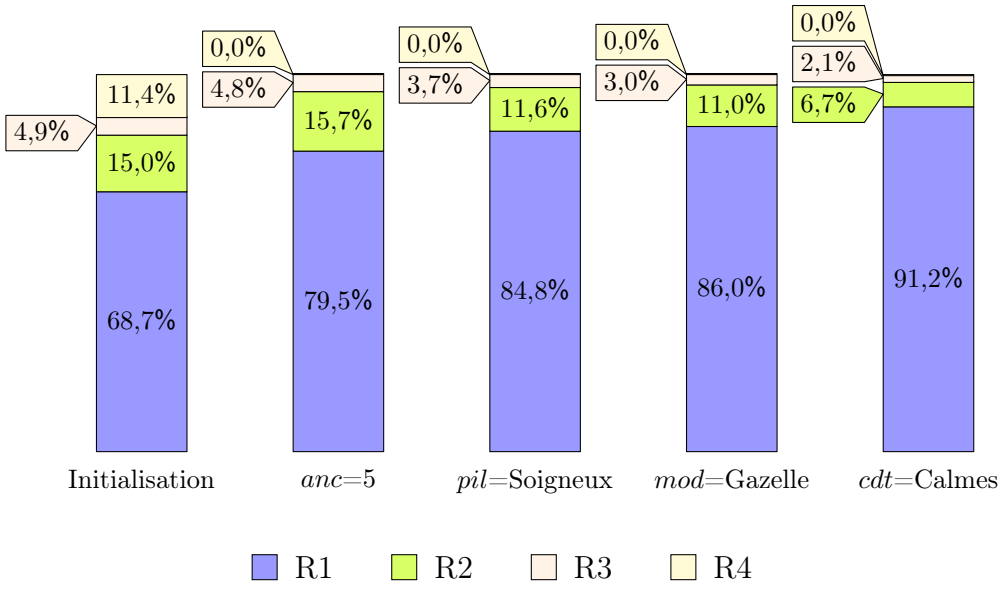


FIGURE 4.3 : Répartition des probabilités d'occurrence des interventions pour le cas très favorable

de garder un maximum de clarté (choix de l'ancienneté, puis du type de pilote, puis du modèle d'hélicoptère, et enfin des conditions de vol).

Nous avons réalisé quatre scénarios dont le troisième correspond à celui illustré en détail ci-avant :

1. Un scénario très favorable, avec une ancienneté de cinq vols, un pilote soigneux, un hélicoptère de type Gazelle (hélicoptère très fiable) et des conditions calmes. La figure 4.3 représente l'évolution des répartitions au fil des réductions utilisateur.
2. Un scénario plutôt favorable, avec une ancienneté de dix vols, un pilote normal, un hélicoptère de type Dauphin (hélicoptère plutôt fiable) et des conditions de vol normales. La figure 4.4 page suivante représente l'évolution des répartitions au fil des réductions utilisateur.
3. Un scénario plus défavorable, avec une ancienneté de quinze vols, un pilote non soigneux, un hélicoptère de type Tigre (hélicoptère plutôt peu fiable), et des conditions de vols avec secousses. La figure 4.5 représente l'évolution des répartitions au fil des réductions utilisateur.
4. Un scénario très défavorable, avec une ancienneté de dix-neuf vols, un pilote agressif, un hélicoptère de type Puma (hélicoptère très peu fiable) et des conditions de vol violentes. La figure 4.6 page 109 représente l'évolution des répartitions au fil des réductions utilisateur.

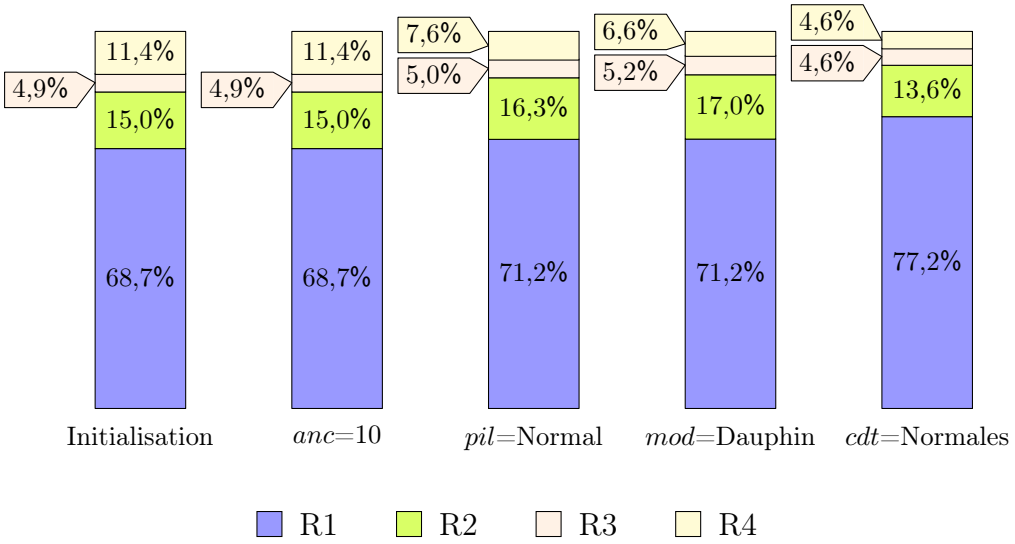


FIGURE 4.4 : Répartition des probabilités d'occurrence des interventions pour le cas favorable

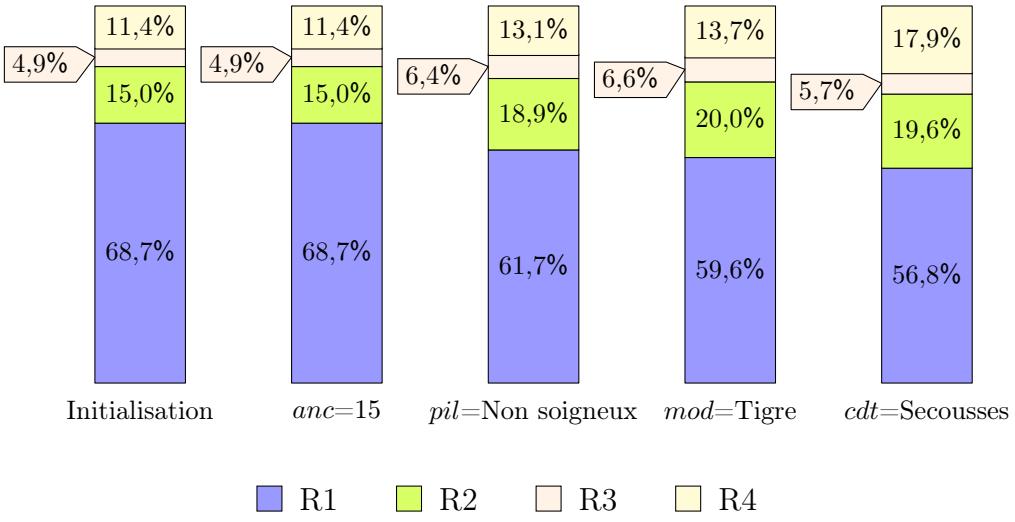


FIGURE 4.5 : Répartition des probabilités d'occurrence des interventions pour le cas défavorable

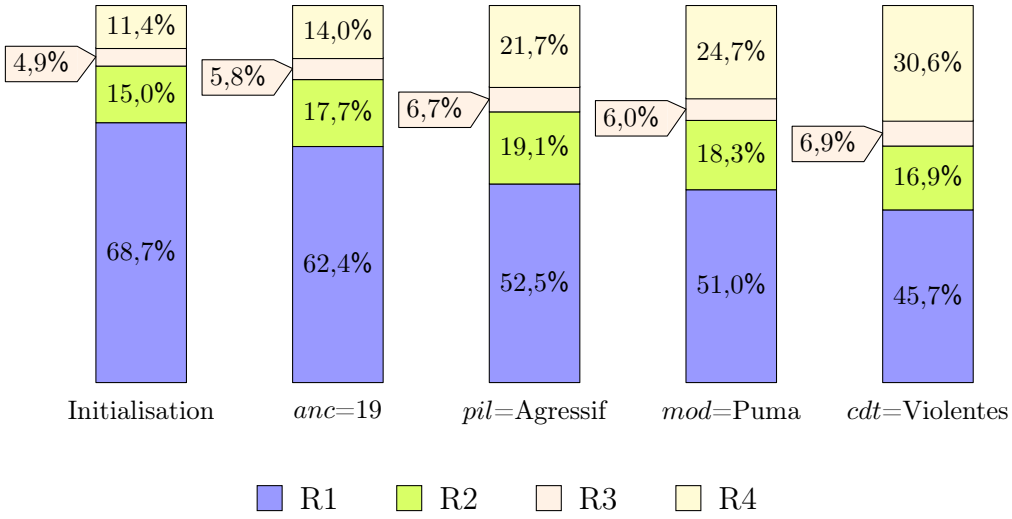


FIGURE 4.6 : Répartition des probabilités d'occurrence des interventions pour le cas très défavorable

4.4.5 Scénario de déroulement d'une contrainte contextuelle à comptage avec filtrage automatique du paramètre

Dans ce scénario, nous effectuons les réductions utilisateur correspondant au scénario de déroulement présenté en section 4.4.3 page 105 en utilisant le comptage contextuel ccc_{2-2} défini en équation 4.6 page 104. Il s'agit donc, dans l'ordre, d'effectuer le comptage contextuel sur *pat_re* après chaque étape :

1. Initialisation,
2. $anc = 15$,
3. $pil =$ Non soigneux,
4. $mod =$ Tigre,
5. $cdt =$ Secousses.

La table 4.4 page suivante donne les valeurs des fréquences d'occurrence de *pat_re* au fur et à mesure des réductions. De même que dans la section 4.3.4 page 101, les cases grisées correspondent aux valeurs de *pat_re* qui sont écartées par la propagation automatique de la contrainte contextuelle, ici il s'agit des fréquences d'occurrences inférieures à 5%.

Nous pouvons remarquer que la valeur R3 est retirée dès l'initialisation du système des valeurs proposées à l'utilisateur pour *pet_es*. Bien que sa fréquence d'occurrence soit supérieure à 5% à la fin de l'ensemble des réductions

TABLE 4.4 : Confiances dans les valeurs de *pat_re* après chaque réduction utilisateur

(a) Initialisation

pat_re	Confiance
R1	68.73%
R2	14.97%
R3	4.93%
R4	11.37%

(b) *anc* = 15

pat_re	Confiance
R1	68.73%
R2	14.97%
R3	4.93%
R4	11.37%

(c) *pil* = Non soigneux

pat_re	Confiance
R1	61.65%
R2	18.89%
R3	6.37%
R4	13.09%

(d) *mod* = Tigre

pat_re	Confiance
R1	60.3%
R2	19.92%
R3	5.91%
R4	13.87%

(e) *cdt* = Secousses

pat_re	Confiance
R1	59.82%
R2	17.79%
R3	6.13%
R4	16.26%

utilisateur, le fait qu'elle soit passé en-dessous de cette valeur fait qu'elle n'est plus proposée à l'utilisateur. Les autres valeurs (R1, R2 et R4) sont donc proposées à l'utilisateur.

Étant donné le peu de valeurs possibles pour *pat_es*, la suppression automatique des valeurs ayant une fréquence d'occurrence inférieure au seuil peut paraitre peu utile, mais cela pourrait s'avérer beaucoup plus utile dans le cas d'un système d'aide à la décision portant sur une variable dont le nombre de valeurs possibles est beaucoup plus important.

4.4.6 Bilan des scénarios de déroulement

Nous pouvons voir que la contrainte contextuelle à comptage sur des variables symboliques apporte de nombreuses informations utiles à l'utilisateur. En effet dans le cas de l'utilisation d'un comptage contextuel avec conseil, il s'agit d'afficher à l'utilisateur quelles sont les fréquences d'occurrence pour chaque valeurs des variables sur lesquelles il attend un conseil. Les différentes réponses du comptage contextuel en fonction des scénarios que nous avons choisi illustrent parfaitement le fait que la connaissance contextuelle apporte des informations auxquelles nous n'aurions pas eu accès avec des contraintes non contextuelles. Dans le cas d'un filtrage automatique sur les paramètres, cela permet, dans le cas de nombreuses valeurs symboliques possibles, de réduire efficacement les possibilités quant aux différentes valeurs d'une variable.

4.5 Synthèse

Dans ce chapitre, nous avons proposé le principe de contrainte contextuelle à comptage. La contrainte contextuelle à comptage est une évolution de la contrainte contextuelle présentée dans le chapitre 3.

L'apport majeur du comptage contextuel par rapport à la contrainte contextuelle est le fait de pouvoir disposer d'une estimation des fréquences d'occurrences des valeurs proposées pour les paramètres. Les cas présents dans la base étant validés par un ou plusieurs experts, tous ces cas encapsulent de la connaissance intéressante pour la prise de décision. L'intérêt du comptage est d'identifier les cas aboutissant aux valeurs des paramètres les plus plausibles.

En situation de conseil, le comptage présente à l'utilisateur une distribution de valeurs possibles. Cette information doit être principalement utilisée pour éliminer les valeurs les moins probables. La distribution de la figure 4.2 illustre bien cela, car il serait hasardeux lors du premier filtrage de ne considérer que les fréquences d'occurrence les plus fortes. La situation de filtrage automatique confirme bien ce constat, avec un seuil de rejet très bas (5%) conduisant au rejet de valeurs très peu probables.

Ce principe de comptage a été illustré sur deux exemples comportant un paramètre contextuel numérique continu et un paramètre symbolique discret. Si l'intérêt du comptage apparaît plus nettement sur l'exemple numérique, cela résulte en fait du nombre d'intervalles caractérisés qui est plus grand (20) que celui des types d'opération de maintenance (4). En effet, le nombre de valeurs peu plausibles est en conséquence plus grand en numérique et permet donc une exploitation plus fructueuse du comptage contextuel.

Conclusion et perspectives

Nous synthétisons dans cette section nos contributions et soulevons des perspectives d'amélioration des éléments proposés.

Conclusion

Nos travaux concernent l'aide à la décision exploitant de la connaissance en aide à la conception interactive. Nous nous situons en conception routinière et avons considéré deux types de connaissance : les connaissances générales et les connaissances contextuelles. Nous avons exploité, respectivement pour ces deux types de connaissances, les approches par propagation de contraintes et les approches de raisonnement à partir de cas pour outiller l'aide à la décision.

Nous nous sommes situés sur une problématique applicative d'aide à la maintenance d'hélicoptères et avons défini deux situations d'aide à la décision avec un modèle de connaissances associé pour chacune. La première situation d'aide à la décision vise à estimer au mieux la durée de maintenance à prévoir pour un hélicoptère qui a évolué dans un contexte donné. La seconde consiste à déterminer au plus juste un type de maintenance à prévoir sur des patins d'atterrissage en fonction des conditions d'emploi de l'hélicoptère. La première situation relève d'une problématique de variables numériques continues tandis que la seconde relève de variables symboliques discrètes. Pour ces deux situations, la connaissance générale issue des normes et données constructeurs impose des opérations de maintenance non négociables. La connaissance contextuelle permet alors d'affiner ces préconisations en proposant, par exemple, d'anticiper des opérations de maintenance ou de corriger des durées voire sélectionner des ressources à utiliser.

Ceci étant posé, nous avons recherché et identifié dans la littérature scientifique deux types de travaux associant ces deux types de connaissance pour aider à la décision. Le premier type vise avant tout, à valider ou compléter de la connaissance, c'est-à-dire que le type de connaissance (générale

ou contextuelle) dans laquelle la confiance et/ou la couverture est la plus forte, est exploitée pour valider ou compléter l'autre type de connaissance (respectivement contextuelle ou générale). Le second type vise à aider à la décision en employant dans divers séquencements la propagation de contraintes, la recherche de cas et certains mécanismes de fouille de données. Ces résultats nous ont conduit à statuer sur le manque d'approche utilisant simultanément (et non pas séquentiellement) les deux types de connaissance pour aider à la décision en conception et à orienter nos travaux dans cette direction.

Nous avons ainsi proposé le principe de contrainte contextuelle qui est une contrainte paramétrique dont la valeur des paramètres dépend du contexte dans lequel ceux-ci sont calculés. Cette contrainte contextuelle nous permet ainsi de prendre en compte de la connaissance contextuelle dans un outil de propagation de contraintes exploitant habituellement de la connaissance générale. Le processus d'utilisation d'une contrainte contextuelle se compose de trois étapes : (i) recherche et sélection des cas passés proches du cas courant dans la base de cas, (ii) calcul des valeurs des paramètres de la contrainte en fonction de la connaissance contextuelle retenue et (iii) obtention des domaines réduits pour les variables du modèle impactées par la contrainte contextuelle. L'exploitation de cette contrainte contextuelle, en situation d'aide à l'estimation de durée de maintenance d'un moteur, a montré comment le contexte d'utilisation permettait d'obtenir des estimations de durée de maintenance plus réalistes. En ce qui concerne l'aide à la détermination de type de maintenance à effectuer sur un patin, l'utilisation de la contrainte contextuelle a fait apparaître des préconisations de maintenance anticipées que la connaissance générale ne pouvait pas proposer.

De manière synthétique, la contrainte contextuelle puise dans les cas les valeurs des paramètres que la connaissance générale ne peut pas fournir. Le mécanisme de recherche de cas, suivant ses réglages et la couverture de la base de cas, fournit un nombre plus ou moins grand de cas. Lorsque les cas fournis deviennent nombreux, certaines valeurs de paramètre peuvent être fournies par plusieurs cas différents ayant alors tendance à souligner leur validité. A l'opposé, une valeur de paramètre issue d'un seul cas sera peu plausible. Ce constat nous a conduits à compléter la contrainte contextuelle avec un mécanisme de comptage issu du *data-mining* fournissant une information supplémentaire sur la fréquence d'occurrence de chacune des valeurs des paramètres. La contrainte contextuelle à comptage permet alors, lors de l'acquisition des valeurs des paramètres, d'effectuer le comptage des fréquences d'occurrence de ces valeurs. L'exploitation de ce mécanisme sur nos deux situations a montré l'intérêt de pouvoir identifier les valeurs les plus fréquentes.

Il a été également montré que ces deux mécanismes, contrainte contextuelle et contrainte contextuelle à comptage, peuvent être utilisés aussi bien

comme une contrainte incluse dans un moteur de filtrage que sous forme de conseil proposé à l'utilisateur du système d'aide à la décision. Le premier fonctionnement autonome permet une aide à la décision beaucoup plus guidée mais peut suivant son réglage retirer des valeurs de paramètre issues de cas pourtant possibles.

Perspectives

Trois types de perspectives sont ouverts par ses travaux. Le premier type visera à améliorer les éléments proposés. Le second consistera en deux manières d'approfondir la notion de contrainte contextuelle. Le dernier est plus orienté vers le domaine applicatif.

Amélioration des éléments proposés Dans le chapitre 3 où la contrainte contextuelle est formalisée, il est indiqué qu'en présence d'un paramètre numérique, les différentes valeurs du paramètre fournies par les cas identifiés sont agrégées dans un intervalle englobant toutes les valeurs. En présence d'un espace de solutions faisant apparaître deux ensembles de cas compacts mais nettement disjoints, le fait de prendre un intervalle englobant fera considérer que tous les cas entre ces deux ensembles compacts sont possibles alors qu'en fait, aucun cas dans la base ne l'indique. Une piste de travail consisterait alors à étudier un principe d'agrégation de valeur autorisant le multi-intervalles qui dans le cas précédent fournirait alors deux intervalles de valeurs pour le paramètre. Ce principe d'agrégation entrerait alors dans la définition de la contrainte contextuelle.

Le comptage est un complément de la contrainte contextuelle qui trouve son intérêt quand la couverture de la base de cas est forte. Dans les chapitres 3 et 4 nous nous sommes placés dans des situations soit de couverture faible (comptage peu intéressant) soit de couverture forte (comptage intéressant). La perspective d'amélioration serait alors de formaliser une forme de seuil de couverture à partir duquel le comptage doit être déclenché. Il est en effet très probable que pour un même problème de maintenance certaines zones de l'espace de solutions sont bien couvertes par des cas (hélicoptère ancien avec un nombre conséquent de maintenances mémorisées) tandis que d'autres ont une couverture très faible (hélicoptère neuf avec peu de maintenances mémorisées).

Une troisième amélioration que nous souhaitons étudier serait la prise en compte de cas dits "négatifs" dans la base de cas. En effet, actuellement, nous nous intéressons seulement aux cas ayant entraîné une bonne solution dite

"positive", mais les échecs apportent également de la connaissance. Il nous semble donc intéressant d'analyser dans quelle mesure il est possible d'étendre la notion de contraintes contextuelles aux cas négatifs d'une base de cas.

Approfondissement de la contrainte contextuelle Dans le chapitre 3, la formulation de la contrainte contextuelle proposée suppose l'existence possible de plusieurs paramètres. Or, dans les deux situations d'aide à la maintenance, nous n'avons illustré nos propositions qu'avec des contraintes contextuelles ne comportant qu'un seul paramètre. Chaque cas fournit une valeur pour un paramètre puis, l'ensemble de ces valeurs est agrégé (intervalle englobant pour le numérique continu, union des valeurs pour le symbolique discret). Il est en conséquence nécessaire d'approfondir les principes d'agrégation en présence de plusieurs paramètres. Ce besoin est probablement plus significatif pour les paramètres numériques, par exemple une contrainte contextuelle : $des = \alpha * dth + \beta$ avec deux paramètres α et β.

Dans tous les problèmes évoqués, la forme de la contrainte contextuelle est connue. C'est à dire que l'ensemble des variables concernées, la formule ou règle d'obtention du paramètre sont fournis par l'expert. Seules les valeurs des paramètres sont inconnues et issues de la connaissance contextuelle via la base de cas. Nous souhaiterions également approfondir le principe de la contrainte contextuelle en sollicitant la connaissance encapsulée dans les cas pour définir les variables concernées et/ou la formule ou règle permettant l'obtention du/des paramètre(s). Des algorithmes de *data-mining*, de régression ou d'analyse de données pourraient être testés dans ce but.

Perspectives applicatives L'ensemble des situations ou exemples sur lequel nous nous sommes appuyés provient de la maintenance d'hélicoptères. Il serait intéressant d'obtenir des jeux de données ou des problématiques d'aide issues d'autres domaines applicatifs. La grande distribution ou les systèmes de santé pour lesquels la connaissance contextuelle est abondante sont sur notre liste de terrains d'études.

Enfin, les algorithmes et programmes que nous avons développés et utilisés ne sont pas intégrés, ils sont écrits dans des langages de programmation différents, et ont été développés dans une optique de défrichage du sujet, au détriment de l'ergonomie et de la généricité. Il reste donc à intégrer ces éléments dans un ensemble cohérent afin de pouvoir effectuer des essais et tests à plus grande échelle sur des modèles plus divers.

Bibliographie

[Aamodt et al. 1994] A. Aamodt et E. Plaza. « Case-based reasoning: Foundational issues, methodological variations, and system approaches ». Dans : *AI communications* 7.1 (1994), p. 39–59 (cf. p. 23, 24).

[Agard et al. 2004] B. Agard et A. Kusiak. « Data-mining-based methodology for the design of product families ». Dans : *International Journal of Production Research* 42.15 (2004), p. 2955–2969. ISSN : 0020-7543. DOI : 10.1080/00207540410001691929 (cf. p. 28).

[Agrawal et al. 1993] R. Agrawal, T. Imieliński et A. Swami. « Mining association rules between sets of items in large databases ». Dans : *ACM SIGMOD Record* 22 (1993), p. 207–216 (cf. p. 27).

[Aissi et al. 2010] Hassene Aissi et Bernard Roy. « Robustness in Multi-criteria Decision Aiding ». Dans : *Trends in Multiple Criteria Decision Analysis*. Sous la dir. de Matthias Ehrgott, José Rui Figueira et Salvatore Greco. International Series in Operations Research & Management Science 142. Springer US, jan. 2010, p. 87–121. ISBN : 978-1-4419-5903-4, 978-1-4419-5904-1 (cf. p. 8).

[Amable 2006] Bruno Amable. « Innovation et compétitivité en Europe ». Dans : *Reflets et perspectives de la vie économique* 1 (2006), 15–30 (cf. p. 5).

[Aussenac et al. 1996] N. Aussenac, P. Laublet et C. Reynaud. « Acquisition et ingénierie des connaissances, tendances actuelles ». Dans : *Cepadues, Toulouse* (1996) (cf. p. 5).

[Avramenko et al. 2006] Yuri Avramenko et Andrzej Kraslawski. « Similarity concept for case-based design in process engineering ». Dans : *Computers & chemical engineering* 30.3 (2006), 548–557 (cf. p. 26, 62).

[BaudLavigne et al. 2011] Bertrand Baud-Lavigne, Bruno Agard et Bernard Penz. « Un nouveau modèle pour la conception conjointe d'une famille de produits et de sa chaîne logistique avec standardisation, externalisation et gammes alternatives ». Français. Dans : *Congrès International de Génie Industriel*. Saint-Sauveur, Canada, oct. 2011 (cf. p. 28).

[Beler 2008] C. Beler. « Modélisation générique d'un retour d'expérience cognitif ». Thèse de doct. INP Toulouse, 2008 (cf. p. 13).

[Bensana 1998] E. Bensana. « Programmation par contraintes sur les intervalles pour la conception préliminaire d'un système de perception. » Thèse de doct. 1998 (cf. p. 9).

[Bergmann 2002] R. Bergmann. « Experience Management: Foundations, Development methodology and, Internet-based Applications ». Dans : *Springer-Verlag Berlin Heidelberg* (2002) (cf. p. 25).

[Bessiere 1994] Christian Bessiere. « Arc-consistency and arc-consistency again ». Dans : *Artificial intelligence* 65.1 (1994), 179–190 (cf. p. 23, 29).

[Blansché et al. 2010] Alexandre Blansché, Julien Cojan, Valmi Dufour-Lussier, Jean Lieber, Pascal Molli, Emmanuel Nauer, Hala Skaf-Molli et Yannick Toussaint. « Taaable 3: Adaptation of ingredient quantities and of textual preparations ». Dans : *18th Int. Conf. on Case-Based Reasoning Workshop Procs.* 2010, 189–198 (cf. p. 11).

[Boutilier et al. 1997] C. Boutilier, R. Brafman, C. Geib et D. Poole. « A constraint-based approach to preference elicitation and decision making ». Dans : *AAAI Spring Symposium on Qualitative Decision Theory* (1997), p. 14 (cf. p. 49).

[Braha 2001] D. Braha. *Data mining for design and manufacturing: methods and applications*. Kluwer academic publishers, 2001 (cf. p. 28).

[Brin et al. 1997] S. Brin, R. Motwani et C. Silverstein. « Beyond market baskets: generalizing association rules to correlations ». Dans : SIGMOD '97 (1997). ACM ID: 253327, p. 265–276. doi : 10.1145/253260.253327 (cf. p. 50).

[Buza et al. 2010] K. Buza, A. Buza et P. B. Kis. « Towards better modeling of supermarkets ». Dans : *2010 International Joint Conference on Computational Cybernetics and Technical Informatics (ICCC-CONTI)*. Mai 2010, p. 499–503. doi : 10.1109/ICCCYB.2010.5491220 (cf. p. 51).

[Chandrasekaran 1990] B. Chandrasekaran. « Design problem solving: A task analysis ». Dans : *AI magazine* 11.4 (1990), p. 59 (cf. p. 7).

[Chapman et al. 1999] Craig B. Chapman et Martyn Pinfold. « Design engineering – a need to rethink the solution using knowledge based engineering ». Dans : *Knowledge-Based Systems* 12.5 (1999), 257–267 (cf. p. 11).

[Chenouard 2007] Raphael Chenouard. « Résolution par satisfaction de contraintes appliquée à l'aide à la décision en conception architecturale ». Thèse de doct. Arts et Métiers ParisTech, déc. 2007 (cf. p. 30).

[Chougule et al. 2011] R. Chougule, D. Rajpathak et P. Bandyopadhyay. « An integrated framework for effective service and repair in the automotive domain: An application of association mining and case-based-reasoning ». Dans : *Computers in Industry* 62.7 (2011), p. 742–754 (cf. p. 51).

[Codet de Boisse et al. 2010] A. Codet de Boisse, E. Vareilles, M. Aldanondo, P. Gaborit, T. Coudert et L. Geneste. « Couplage csp et cbr: premières identifications des modes de couplage ». Dans : 2010 (cf. p. 52).

[Costa Filho et al. 2012] Cicero Ferreira Fernandes Costa Filho, Dayse Aparecida Rivera Rocha, Marly Guimarães Fernandes Costa et Wagner Coelho de Albuquerque Pereira. « Using Constraint Satisfaction Problem approach to solve human resource allocation problems in cooperative health services ». Dans : *Expert Systems with Applications* 39.1 (jan. 2012), p. 385–394. ISSN : 0957-4174. DOI : 10.1016/j.eswa.2011.07.027 (cf. p. 11).

[Da Cunha et al. 2006] Catherine Da Cunha, Bruno Agard et Andrew Kusiak. « Data mining for improvement of product quality ». Dans : *International Journal of Production Research* 44.18-19 (2006), 4027–4041 (cf. p. 28).

[Dayal et al. 1988] Umeshwar Dayal, Alejandro P Buchmann et Dennis R McCarthy. « Rules are objects too: a knowledge model for an active, object-oriented database system ». Dans : *Advances in Object-Oriented Database Systems*. Springer, 1988, p. 129–143 (cf. p. 23).

[Didierjean 2001] A. Didierjean. « Apprendre à partir d'exemples : abstraction de règles et/ou mémoire d'exemplaires ? » Dans : *L'année psychologique* 101.2 (2001), p. 325–348. ISSN : 0003-5033. DOI : 10.3406/psy.2001.29560 (cf. p. 12).

[Djefel 2010] Mériem Djefel. « Couplage de la configuration de produit et de projet de réalisation: exploitation des approches par contraintes et des algorithmes évolutionnaires ». Thèse de doct. Institut National Polytechnique de Toulouse, 2010 (cf. p. 11, 31).

[Duval 1991] Raymond Duval. « Structure du raisonnement déductif et apprentissage de la démonstration ». Dans : *Educational studies in mathematics* 22.3 (1991), 233–261 (cf. p. 5).

[Elkan et al. 1993] Charles Elkan et Russell Greiner. « Building large knowledge-based systems: Representation and inference in the cyc project: DB Lenat and RV Guha ». Dans : *Artificial Intelligence* 61.1 (1993), 41–52 (cf. p. 11).

[Eppinger 1991] Steven D. Eppinger. « Model-based Approaches to Managing Concurrent Engineering ». Dans : *Journal of Engineering Design* 2.4 (1991), p. 283–290. ISSN : 0954-4828. DOI : 10.1080/09544829108901686 (cf. p. 8, 11).

[Eteläpelto 2000] A. Eteläpelto. « Contextual and strategic knowledge in the acquisition of design expertise ». Dans : *Learning and Instruction* 10.2 (2000), p. 113–136. ISSN : 0959-4752. DOI : 10.1016/S0959-4752(99)00014-6 (cf. p. 10).

[Fayyad et al. 1996] U. M Fayyad, G. Piatetsky-Shapiro, P. Smyth et R. Uthurusamy. *Advances in knowledge discovery and data mining*. 1996 (cf. p. 24, 26).

[Felfernig et al. 2007] A. Felfernig, G. Friedrich, K. Isak, K. Shchekotykhin, E. Teppan et D. Jannach. « Automated debugging of recommender user interface descriptions ». Dans : *Applied Intelligence* 31.1 (déc. 2007), p. 1–14. ISSN : 0924-669X. DOI : 10.1007/s10489-007-0105-8 (cf. p. 45).

[Fuchs 2008] Béatrice Fuchs. « Retour et Capitalisation d'expérience. Outils et démarches. » Dans : Jean Renaud, Eric Bonjour, Brigitte Chebel-Morello, Béatrice Fuchs et Nadia Matta. *Retour et capitalisation d'expérience*. Sous la dir. d'AFNOR. 2008. Chap. 4, p. 99–135. ISBN : 978-2-12-465117-7 (cf. p. 15, 21).

[Gelle 1998] Esther Gelle. « On the generation of locally consistent solution spaces in mixed dynamic constraint problems ». Thèse de doct. École Polytechnique Fédérale de Lausanne, 1998 (cf. p. 30).

[Gero 1990] John S. Gero. « Design prototypes: a knowledge representation schema for design ». Dans : *AI magazine* 11.4 (1990), p. 26 (cf. p. 6).

[Giard et al. 1985] Vincent Editeur Giard et Bernard Roy. *Méthodologie multicritère d'aide à la décision*. Editions Economica, 1985 (cf. p. 8).

[Giaretta 1995] Pierdaniele Giaretta. « Ontologies and knowledge bases towards a terminological clarification ». Dans : *Towards very large knowledge bases: knowledge building & knowledge sharing 1995* (1995), p. 25 (cf. p. 23).

[Golomb et al. 1965] S. W. Golomb et L. D. Baumert. « Backtrack Programming ». Dans : *Journal of the ACM* 12.4 (oct. 1965), p. 516–524. ISSN : 00045411. DOI : 10.1145/321296.321300 (cf. p. 28).

[Gomes 2008] S. Gomes. « Ingénierie à base de connaissances pour une conception, productive, optimisée, collaborative et innovante du système Projet-Produit-Process-Usage ». Dans : *Habilitation à diriger les recherches* (oct. 2008) (cf. p. 5, 9).

[Gruber et al. 1995] Thomas R Gruber et al. « Toward principles for the design of ontologies used for knowledge sharing ». Dans : *International journal of human computer studies* 43.5 (1995), p. 907–928 (cf. p. 23).

[Grundstein 2000] M. Grundstein. « from capitalizing on Company's Knowledge to Knowledge Management ». Dans : *Knowledge Management, Classic and Contemporary Works* 12 (2000), 261–287 (cf. p. 10).

[Guo et al. 2012] Yuan Guo, Jie Hu et Yinghong Peng. « A CBR system for injection mould design based on ontology: A case study ». Dans : *Computer-Aided Design* (2012) (cf. p. 26).

[Ha et al. 1998] V. Ha et P. Haddawy. « Toward case-based preference elicitation: Similarity measures on preference structures ». Dans : *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence* (1998), p. 193–201 (cf. p. 49).

[Han et al. 2006] Jiawei Han, Micheline Kamber et Jian Pei. *Data Mining, Second Edition: Concepts and Techniques*. Morgan Kaufmann, avr. 2006. ISBN : 9780080475585 (cf. p. 22).

[Harmon et al. 1988] Paul Harmon, Rex Maus et William Morrissey. *Expert systems: tools and applications*. John Wiley & Sons, Inc., 1988 (cf. p. 23).

[Holt et al. 2005] Alec Holt, Isabelle Bichindaritz, Rainer Schmidt et Petra Perner. « Medical applications in case-based reasoning ». Dans : *The Knowledge Engineering Review* 20.03 (2005), p. 289–292. DOI : 10.1017/S0269888906000622 (cf. p. 11).

[Hubka et al. 1988] Vladimir Hubka et W. Ernst Eder. « Theory of technical systems: a total concept theory for engineering design ». Dans : *Berlin and New York, Springer-Verlag, 1988, 291 p.* 1 (1988) (cf. p. 7).

[Huot 2005] Stéphane Huot. « Une nouvelle approche pour la conception créative: De l'interprétation du dessin à main levée au prototypage d'interactions non-standard ». Thèse de doct. Université de Nantes, 2005 (cf. p. 7).

[Jarboe et al. 2001] Kenan Patrick JARBOE et Athena ALLIANCE. « Knowledge Management as an Economic Development Strategy ». Dans : *Reviews of Economic Development Literature and Practice* 7 (2001) (cf. p. 13).

[Khiari et al. 2010] M. KHIARI, P. BOIZUMAULT et B. CRÉMILLEUX. « Combining CSP and constraint-based mining for pattern discovery ». Dans : *Computational Science and Its Applications–ICCSA 2010* (2010), p. 432–447 (cf. p. 50).

[Kim et al. 2005] Pansoo KIM et Yu DING. « Optimal Engineering System Design Guided by Data-Mining Methods ». Dans : *Technometrics* 47.3 (2005), p. 336–348. ISSN : 0040-1706. DOI : 10.1198/004017005000000157 (cf. p. 28).

[Kintsch et al. 1985] Walter KINTSCH et James G. GREENO. « Understanding and solving word arithmetic problems ». Dans : *Psychological Review* 92.1 (1985), p. 109–129. ISSN : 1939-1471(Electronic);0033-295X(Print). DOI : 10.1037/0033-295X.92.1.109 (cf. p. 5).

[Kolodner 1993] J. L KOLODNER. *Case-based reasoning*. Morgan Kaufmann Publishers, Inc. 1993. ISBN : 1-55860-237-2 (cf. p. 13, 15, 24).

[Le Ber et al. 2006] F. LE BER, J. LIEBER et A. NAPOLI. « Les systèmes à base de connaissances ». Dans : *Encyclopédie de l'informatique et des systèmes d'information* (2006), p. 1197–1208 (cf. p. 11).

[Le Masson et al. 2006] Pascal LE MASSON, Benoit WEIL et Armand HATCHUEL. *Les processus d'innovation: Conception innovante et croissance des entreprises*. Hermes science publ., 2006 (cf. p. 7).

[Lee et al. 2002] K. S. LEE et C. LUO. « Application of case-based reasoning in die-casting die design ». Dans : *The International Journal of Advanced Manufacturing Technology* 20.4 (2002), 284–295 (cf. p. 26).

[Leondes 2000] Cornelius T. LEONDES. *Knowledge-based Systems: Systems and applications*. T. 4. Academic Pr, 2000 (cf. p. 11, 13).

[Lévine et al. 1989] Pierre LÉVINE et Jean-Charles POMEROL. *Systèmes interactifs d'aide à la décision et systèmes experts*. Hermès Paris, 1989 (cf. p. 9).

[Lhomme 1993] O. LHOMME. « Consistency techniques for numeric CSPs ». Dans : *International Joint Conference on Artificial Intelligence*. T. 13. 1993, p. 232 (cf. p. 29).

[Li et al. 2009] Huan Li, Xin Li, Dawei Hu, Tianyong Hao, Liu Wenyin et Xiaoping Chen. « Adaptation rule learning for case-based reasoning ». Dans : *Concurrency and Computation: Practice and Experience* 21.5 (2009), 673–689. ISSN : 1532-0634. DOI : 10.1002/cpe.1368 (cf. p. 22).

[Lieber 2008] J. Lieber. « Contributions à la conception de systèmes de raisonnement à partir de cas ». Thèse de doct. Université Henri Poincaré - Nancy I, jan. 2008 (cf. p. 12, 22).

[Lonchampt 2004] P. Lonchampt. « Co-évolution et processus de conception intégrée de produits: Modèle et support de l'activité de conception ». Thèse de doct. Institut National Polytechnique de Grenoble-INPG, 2004 (cf. p. 7, 9).

[Maimon et al. 2010] O. Maimon et L. Rokach. *Data mining and knowledge discovery handbook*. T. 14. Springer, 2010 (cf. p. 47, 48).

[Mileman et al. 2002] Tony Mileman, Brian Knight, Miltos Petridis, Don Cowell et J. Ewer. « Case-based retrieval of 3-dimensional shapes for the design of metal castings ». Dans : *Journal of Intelligent Manufacturing* 13.1 (2002), 39–45 (cf. p. 26).

[Mille 2006] A. Mille. « From case-based reasoning to traces-based reasoning ». Dans : *Annual Reviews in Control* 30.2 (2006), p. 223–232. ISSN : 1367-5788. DOI : 10.1016/j.arcontrol.2006.09.003 (cf. p. 25).

[Minor et al. 2010] Mirjam Minor, Ralph Bergmann, Sebastian Görg et Kirstin Walter. « Towards Case-Based Adaptation of Workflows ». Dans : *Case-Based Reasoning. Research and Development*. Sous la dir. d'Isabelle Bichindaritz et Stefania Montani. Lecture Notes in Computer Science 6176. Springer Berlin Heidelberg, jan. 2010, p. 421–435. ISBN : 978-3-642-14273-4, 978-3-642-14274-1 (cf. p. 22).

[Minsky 1974] M. Minsky. « A Framework for Representing Knowledge. » Dans : *Psychology of Computer Vision* (1974), p. 211–277 (cf. p. 5, 24).

[Mok et al. 2008] C. K. Mok, M. Hua et S. Y. Wong. « A hybrid case-based reasoning CAD system for injection mould design ». Dans : *International Journal of Production Research* 46.14 (2008), 3783–3800 (cf. p. 26).

[Montanari 1974] U. Montanari. « Networks of constraints: Fundamental properties and applications to picture processing ». Dans : *Information sciences* 7 (1974), p. 95–132 (cf. p. 28).

[Montani 2011] S. Montani. « How to use contextual knowledge in medical case-based reasoning systems: A survey on very recent trends ». Dans : *Artificial Intelligence in Medicine* 51.2 (2011). Advances in Case-Based Reasoning in the Health Sciences, p. 125–131. ISSN : 0933-3657. DOI : 10.1016/j.artmed.2010.09.004 (cf. p. 10).

[Moore 1998] G. E. Moore. « Cramming more components onto integrated circuits ». Dans : *Proceedings of the IEEE* 86.1 (1998), 82–85 (cf. p. 1).

[Moore 1966] R. E. Moore. « Interval analysis. » Dans : *Prince-Hall, Englewood Cliffs, NJ* (1966) (cf. p. 29, 94).

[Mulyanto 2002] Taufiq Mulyanto. « Utilisation des Techniques de Programmation par Contraintes pour la Conception d'Avions ». Thèse de doct. École Nationale Supérieure de l'Aéronautique et de l'Espace, 2002 (cf. p. 30).

[Nonaka 1994] I. Nonaka. « A dynamic theory of organizational knowledge creation ». Dans : *Organization science* (1994), p. 14–37 (cf. p. 10).

[Nortet et al. 2005] C. Nortet, A. Salleb, T. Turmeaux et C. Vrain. « Extraction de Règles d'Association Quantitatives ». Dans : *Mémoire de Master, BRGM et LIFO Université d'Orléans* (2005) (cf. p. 50).

[Nunez et al. 2004] H. Nunez, M. Sanchez-Marre, U. Cortes, J. Comas, M. Martnez, I. Rodriguez-Roda et M. Poch. « A comparative study on the use of similarity measures in casebased reasoning to improve the classification of environmental system situations ». Dans : *Elsevier - Environmental Modelling and Software* 19 (2004), p. 809–819 (cf. p. 62).

[Pahl et al. 1996] G. Pahl, W. Beitz et K. Wallace. *Engineering design: a systematic approach*. Springer Verlag, 1996 (cf. p. 6, 7).

[Pargamin 2002] Bernard Pargamin. « Vehicle sales configuration: the cluster tree approach ». Dans : *ECAI 2002 Configuration Workshop*. 2002, 35–40 (cf. p. 11).

[Pfaff et al. 2010] M.S. Pfaff, J.L. Drury, G.L. Klein, L. More, Sung Pil Moon et Yikun Liu. « Weighing decisions: Aiding emergency response decision making via option awareness ». Dans : *2010 IEEE International Conference on Technologies for Homeland Security (HST)*. Nov. 2010, p. 251–257. doi : 10.1109/THS.2010.5655051 (cf. p. 8).

[Phua et al. 2010] Clifton Phua, Vincent Lee, Kate Smith et Ross Gayler. « A Comprehensive Survey of Data Mining-based Fraud Detection Research ». Dans : *arXiv:1009.6119* (sept. 2010). doi : 10.1016/j.chb.2012.01.002 (cf. p. 11).

[Polanyi 1958] M. Polanyi. *Personal knowledge: Towards a post-critical philosophy*. Psychology Press, 1958 (cf. p. 10).

[Polanyi et al. 1966] M. Polanyi et A. Sen. *The tacit dimension*. Peter Smith Gloucester, MA, 1966 (cf. p. 10).

[Pomerol et al. 2001] J. C. Pomerol et P. Brézillon. « About some relationships between knowledge and context ». Dans : *Modeling and Using Context* (2001), 461–464 (cf. p. 10).

[Prasad 1996] B. Prasad. « Concurrent engineering fundamentals- Integrated product and process organization(Book) ». Dans : *Upper Saddle River, NJ: Prentice Hall PTR, 1996.* (1996) (cf. p. 9).

[Prax 1997] Jean-Yves Prax. « Manager la connaissance dans l'entreprise ». Dans : *Les nouvelles technologies* (1997) (cf. p. 13).

[Purvis et al. 1995] L. Purvis et P. Pu. « Adaptation using constraint satisfaction techniques ». Dans : *Case-Based Reasoning Research and Development* (1995), p. 289–300 (cf. p. 12, 50).

[Py 1994] Michel Py. « Un modèle conceptuel de raisonnement par analogie ». Dans : *Revue d'intelligence artificielle* 8.1 (1994), 63–99 (cf. p. 5).

[Rouet et al. 1995] Jean-François Rouet et André Tricot. « Recherche d'informations dans les systèmes hypertextes : des représentations de la tâche à un modèle de l'activité cognitive ». Dans : *Revue Sciences et techniques éducatives* 2.3 (1995), p. 307–331 (cf. p. 5).

[Roy 1996] B. Roy. *Multicriteria Methodology for Decision Aiding*. Springer, août 1996. ISBN : 9780792341666 (cf. p. 8).

[Ruet et al. 2002] M. Ruet et L. Geneste. « Search and adaptation in a fuzzy object oriented case base ». Dans : *Advances in Case-Based Reasoning* (2002), p. 723–730 (cf. p. 50).

[Samurçay et al. 1996] R. Samurçay et J.-M. Hoc. « Causal versus topographical support for diagnosis in a dynamic situation ». Dans : *Le Travail Humain* (1996), 45–68 (cf. p. 5).

[Sharaf Eldeen et al. 2012] Dina A. Sharaf Eldeen, Ibrahim F. Moawad, Khalid El Bahnasy et M. E. Khalifa. « Learning and Applying Range Adaptation Rules in Case-Based Reasoning Systems ». Dans : *Advanced Machine Learning Technologies and Applications*. Sous la dir. d'Aboul Ella Hassanien, Abdel-Badeeh M. Salem, Rabie Ramadan et Tai-hoon Kim. Communications in Computer and Information Science 322. Springer Berlin Heidelberg, jan. 2012, p. 487–495. ISBN : 978-3-642-35325-3, 978-3-642-35326-0 (cf. p. 22).

[ShuHsien et al. 2012] L. Shu-Hsien, C. Pei-Hui et H. Pei-Yuan. « Data mining techniques and applications – A decade review from 2000 to 2011 ». Dans : *Expert Systems with Applications* 39.12 (2012), p. 11303–11311. ISSN : 0957-4174. DOI : 10.1016/j.eswa.2012.02.063 (cf. p. 26).

[Silva Garza et al. 1996] Andrés Gómez de Silva Garza et Mary Lou Maher. « Design by interactive exploration using memory-based techniques ». Dans : *Knowledge-Based Systems* 9.3 (mai 1996), p. 151–161. ISSN : 0950-7051. DOI : 10.1016/0950-7051(95)01016-5 (cf. p. 21).

[Steuer et al. 2003] Ralph E Steuer et Paul Na. « Multiple criteria decision making combined with finance: A categorized bibliographic study ». Dans : *European Journal of Operational Research* 150.3 (nov. 2003), p. 496–515. ISSN : 0377-2217. DOI : 10.1016/S0377-2217(02)00774-9 (cf. p. 8).

[Suh 1998] Nam P. Suh. « Axiomatic design theory for systems ». Dans : *Research in Engineering Design* 10.4 (1998), 189–209 (cf. p. 7, 14).

[Tichkiewitch 2005] Serge Tichkiewitch. « Complexity in design using collaborative network ». Dans : *Proceedings of 2nd International Conference Virtual Design and Automation, New trends in collaborative product design Automation*. Sous la dir. de Publishing House of Poznan University of Technology. Poznan, Pologne, nov. 2005, p. 55–62. ISBN : 83-7143-228-3 (cf. p. 5).

[Toussaint 2010] Luis Cécilio Toussaint. « Modèles et méthodes pour une conception hautement productive orientée vers la fabrication : application à l'ingénierie routinière de pièces plastiques ». Thèse de doct. Université de Technologie de Belfort-Montbeliard, déc. 2010 (cf. p. 8).

[Trousse 1989] Brigitte Trousse. *Coopération entre systèmes à base de connaissances et outils de CAO: l'environnement multi-agent Anaxagore*. 1989 (cf. p. 11).

[Ulrich et al. 2011] Karl T. Ulrich, Steven D. Eppinger et Anita Goyal. *Product design and development*. Irwin/McGraw-Hill, 2011 (cf. p. 7).

[Van Oudenhove de Saint Géry 2006] T. Van Oudenhove de Saint Géry. « Contribution à l'élaboration d'un formalisme gérant la pertinence pour les problèmes d'aide à la conception à base de contraintes ». Thèse de doct. INP Toulouse, 2006 (cf. p. 15).

[Vancza 1999] Jozsef Vancza. « Artificial intelligence support in design: a survey ». Dans : *Integration of Process Knowledge into Design Support Systems* (1999), p. 57 (cf. p. 11).

[Vareilles et al. 2011] É Vareilles, M. Aldanondo, A. Codet de Boisse, T. Coudert, P. Gaborit et L. Geneste. « How to take into account general and contextual knowledge for interactive aiding design: Towards the coupling of *CSP* and *CBR* approaches ». Dans : *Engineering Applications of Artificial Intelligence* (2011) (cf. p. 52, 56).

[Vareilles 2005] Elise VAREILLES. « Conception et approches par propagation de contraintes: contribution à la mise en øeuvre d'un outil d'aide interactif ». Thèse de doct. Institut National Polytechnique de Toulouse, 2005 (cf. p. 30).

[Vargas 1995] Catalina VARGAS. « Modélisation du processus de conception en ingénierie des systèmes mécaniques: Mise en oeuvre basée sur la propagation de contraintes ». Thèse de doct. 1995 (cf. p. 11).

[Weigel et al. 1994] Rainer WEIGEL et Boi FALTINGS. *Constraint-based knowledge representation for configuration systems*. Rap. tech. Citeseer, 1994 (cf. p. 23).

[Westphal et al. 1998] C. WESTPHAL et T. BLAXTON. *Data mining solutions*. Wiley, 1998 (cf. p. 26).

[Wineburg 1991] Samuel S. WINEBURG. « Historical problem solving: A study of the cognitive processes used in the evaluation of documentary and pictorial evidence ». Dans : *Journal of Educational Psychology* 83.1 (1991), p. 73–87. ISSN : 1939-2176(Electronic);0022-0663(Print). DOI : 10.1037/0022-0663.83.1.73 (cf. p. 5).

[Woon et al. 2005] Fei WOON, Brian KNIGHT, Miltos PETRIDIS et Mayur PATEL. « CBE-conveyor: a case-based reasoning system to assist engineers in designing conveyor systems ». Dans : *Case-Based Reasoning Research and Development* (2005), 640–651 (cf. p. 26).

[Zaraté 2005] P. ZARATÉ. « Des Systèmes Interactifs d'Aide à la Décision aux Systèmes Coopératifs d'Aide à la Décision : Contributions conceptuelles et fonctionnelles ». Habilitation à diriger des recherches. Toulouse, France : Institut National Polytechnique de Toulouse, déc. 2005 (cf. p. 8).

[Zhuang et al. 2009] Z. Y ZHUANG, L. CHURILOV, F. BURSTEIN et K. SIKARIS. « Combining data mining and case-based reasoning for intelligent decision support for pathology ordering by general practitioners ». Dans : *European Journal of Operational Research* 195.3 (2009), p. 662–675 (cf. p. 51).

[Zouari 2007] A. ZOUARI. « Proposition de mécanismes de versionnement et d'agrégation des connaissances de domaine en conception collaborative de produit industriels ». Thèse de doct. Thèse de Doctorat en cotutelle INPG et ENIS, Sfax mars, 2007 (cf. p. 5).

Résumé

Les travaux présentés dans cette thèse ont pour objectif de contribuer à l'élaboration d'un outil d'aide à la décision en conception exploitant de la connaissance générale et de la connaissance contextuelle. L'exploitation de la connaissance générale est effectuée à l'aide d'un moteur de filtrage de contraintes et l'exploitation de la connaissance contextuelle repose sur des principes issus du raisonnement à partir de cas et du *data-mining*.

Le résultat principal est basé sur une notion de contrainte dite « contextuelle ». L'idée forte revient à paramétrer une contrainte en fonction du contexte dans lequel celle-ci doit être filtrée. Suivant le niveau de confiance caractérisant la connaissance contextuelle, cette contrainte pourra soit être propagée de manière autonome dans le moteur de filtrage soit être utilisée pour fournir une forme de conseil à l'utilisateur du système d'aide. Nous proposons pour identifier le contexte d'emploi de la contrainte d'utiliser le principe de recherche par similarité très largement utilisé dans les travaux portant sur le raisonnement à partir de cas. Afin de compléter ou d'affiner les informations résultant du filtrage de cette contrainte contextuelle, nous utilisons des algorithmes de comptage issus du *data-mining* pour fournir des fréquences d'apparition caractérisant une forme de confiance dans le résultat.

Nos travaux s'inscrivent dans le cadre d'un projet FUI portant sur la maintenance d'hélicoptère. Le but de notre outil est d'estimer au mieux la charge, le cycle, les coûts des activités opérationnelles de maintenance d'un hélicoptère. L'originalité des travaux est de considérer d'une part les connaissances constructeurs (connaissance générale) pour déterminer une première estimation puis de la corriger suivant les conditions d'utilisation effective de l'hélicoptère (connaissance contextuelle).

Summary

The works presented in this thesis aims to contribute to the development of a tool for design decision support exploiting general knowledge and contextual knowledge. The use of general knowledge is performed using a constraint filtering engine the exploitation of contextual knowledge is based on principles derived from case-based reasoning and data mining.

The main result relies on a constraint notion called ”contextual constraint”. The principal idea consists to parameterize a constraint with respect to a context where it must be filtered. Depending on the level of confidence characterizing contextual knowledge, this constraint may either be propagated independently by a filtering engine or being used to provide help to the user. We propose in order to identify the context of the constraint to use the principle of similarity search widely used in case-based reasoning. To complete or refine the information resulting from this constraint contextual filtering, we use counting algorithms developed for data-mining to quantify the occurrence characterizing a kind of confidence in the result.

Our work is based on a FUI project relating to helicopter maintenance. The aim of this tool is to better estimate the charge, the cycle, the operational costs of maintenance of an helicopter. The originality of this work is to consider both the manufacturers knowledge (general knowledge) to determine an initial estimate and then to correct it according to the conditions of effective use of the helicopter (contextual knowledge).