Romain Wallon
10 Chemin des Riots
62800 LIÉVIN
FRANCE
@ wallon@lix.polytechnique.fr
www.lix.polytechnique.fr/~wallon/
14/04/1994

Docteur en Informatique

Depuis Octobre 2020 - Postdoc au LIX

Contrat Postdoctoral sur l'intégration de techniques de programmation spéculative dans différents types de solveurs au sein de l'équipe OptimiX du Laboratoire d'Informatique de l'Ecole Polytechnique (LIX), encadré par Claudia D'Ambrosio, Youssef Hamadi et Rémi Delmas.

Précédentes Expériences et Activités de Recherche

Octobre 2017 – Décembre 2020 : Doctorant

— Doctorat intitulé *Raisonnement à partir de contraintes pseudo-booléennes et compilation*
— Financé sur contrat doctoral de l'Université d'Artois
— Préparé au Centre de Recherche en Informatique de Lens (CRIL UMR 8188, Université d'Artois et CNRS)
— Sous la direction de Daniel Le Berre (Université d'Artois) et de Pierre Marquis (Université d'Artois) et encadré par Stefan Mengel (CNRS)
— Jury composé de Nadia Creignou (présidente, Université d'Aix-Marseille), de João Marques-Silva (rapporteur, CNRS) et de Jakob Nordström (rapporteur, Universités de Lund et de Copenhague)
— Mémoire : `https://www.lix.polytechnique.fr/~wallon/reports/phdthesis.pdf`

Avril 2017 – Septembre 2017 : Stage de Master

— Stage de Master intitulé *Raisonnement à partir de contraintes pseudo-booléennes et compilation*
— Réalisé au Centre de Recherche en Informatique de Lens (CRIL UMR 8188, Université d'Artois et CNRS)
— Encadré par Daniel Le Berre (Université d'Artois), Pierre Marquis (Université d'Artois) et Stefan Mengel (CNRS)
— Mémoire : `https://www.lix.polytechnique.fr/~wallon/reports/stage-m2.pdf`

Avril 2016 – Juin 2016 : Travail d'Etude et de Recherche (TER) de Master

— Travail d'Étude et de Recherche sur les *Heuristiques de décomposition de formules CNF*
— Réalisé au Centre de Recherche en Informatique de Lens (CRIL UMR 8188, Université d'Artois et CNRS)
— Encadré par Pierre Marquis (Université d'Artois), Jean-Marie Lagniez (Université d'Artois) et Stefan Mengel (CNRS)
— Mémoire : `https://www.lix.polytechnique.fr/~wallon/reports/ter-m1.pdf`

Formation

Octobre 2017 – Décembre 2020 : Doctorat en Informatique et Applications

— Soutenu le 14 décembre 2020
— CRIL, Université d'Artois et CNRS - LENS

Septembre 2015 – Septembre 2017 : Master d'Informatique, Parcours Intelligence Artificielle

— Mention Très Bien (Major de Promotion)
— Université d'Artois (UFR des Sciences Jean Perrin) - LENS

Septembre 2012 – Septembre 2015 : Licence d'Informatique

— Mention Très Bien (Major de Promotion)
— Université d'Artois (UFR des Sciences Jean Perrin) - LENS

Septembre 2012 — Septembre 2015 : Licence de Mathématiques

— Mention Très Bien (Major de Promotion)
— Université d'Artois (UFR des Sciences Jean Perrin) - LENS

Juin 2012 : Baccalauréat Scientifique

— Spécialité Mathématiques
— Mention Très Bien
— Lycée Henri Darras - LIÉVIN

Domaine de Recherche

Mots-clefs

Intelligence artificielle, raisonnement automatique, algorithmes pour l'inférence et contraintes, cohérence propositionnelle, raisonnement pseudo-booléen, compilation de connaissances

Thématique de Recherche

Les dernières décennies ont vu des avancées importantes dans la résolution du problème de la cohérence propositionnelle. Les solveurs SAT modernes sont ainsi capables de résoudre efficacement des problèmes industriels comportant des millions de variables et de clauses. Néanmoins, certains problèmes sont intrinsèquement hors de leur portée. C'est notamment le cas des problèmes pour lesquels un nombre exponentiel d'étapes de résolution est nécessaire pour démontrer l'incohérence de la formule considérée, comme c'est par exemple le cas pour le *principe du pigeonnier* (aussi appelé *principe des tiroirs* en français). Dans ce contexte, il peut être intéressant d'utiliser un système de preuve plus puissant que

la résolution, comme celui des plans-coupes. Ce dernier *p-simule* le premier, c'est-à-dire que toute preuve par résolution peut être simulée en une preuve utilisant les plans-coupes ayant une taille polynomiale par rapport à la preuve originale. Cependant, les solveurs pseudo-booléens utilisant ce système de preuve théoriquement plus puissant ne parviennent pas à atteindre en pratique les performances des solveurs SAT utilisant la résolution. Cela est en grande partie dû au fait que les contraintes pseudo-booléennes et les règles des plans-coupes nécessitent l'utilisation de structures de données plus complexes à implanter efficacement. De plus, l'extension de l'architecture CDCL des solveurs SAT classiques au cadre pseudo-booléens soulève un certain nombre de problèmes qui ne sont pas encore complètement résolus.

Dans ce contexte, j'ai travaillé à améliorer l'efficacité des solveurs pseudo-booléens en considérant différents facteurs ayant un impact sur leurs performances. En particulier, j'ai étudié l'impact des littéraux non-pertinents [3, 6, 9] produits lors des analyses de conflit réalisées par ces solveurs sur leurs performances. Ces littéraux n'ont pas d'effet sur les contraintes dans lesquelles ils apparaissent, mais peuvent conduire à l'inférence de contraintes plus faibles que lorsque ces littéraux sont supprimés. Il est donc important d'être capable d'éliminer efficacement ces littéraux pour produire des contraintes plus fortes, malgré le fait que la détection desdits littéraux soit NP-difficile. Dans ce but, j'ai proposé une solution *ad hoc* permettant d'identifier ces littéraux pour ensuite pouvoir les retirer. Cette solution, trop coûteuse en pratique pour être considérée comme une contre-mesure, a toutefois mis en évidence l'impact pratique des littéraux non-pertinents dans les contraintes apprises par le solveur. Plus précisément, la taille des preuves produites en leur présence peut être exponentiellement plus grande que lorsque ces littéraux sont supprimés. Pour obtenir de meilleures performances, l'idéal serait donc de pouvoir éviter de les produire, à l'aide de stratégies d'application des règles du système des plans-coupes conçues dans cette optique.

Une approche possible est de considérer la stratégie d'affaiblissement appelée *weaken-ineffective* [2], qui consiste à éliminer les littéraux n'ayant (localement) pas d'effet sur le conflit en cours d'analyse. En effet, cette stratégie, qui tire parti de la règle d'affaiblissement déjà utilisée par les solveurs pseudo-booléens, permet d'éliminer tous les littéraux non-pertinents. Il s'agit cependant d'une stratégie aggressive, pouvant également éliminer des littéraux pertinents. J'ai par ailleurs étudié d'autres stratégies d'affaiblissement [2], ayant pour but de produire des contraintes fortes tout en restant efficace, comme par exemple la stratégie *PartialRoundingSat* qui utilise la règle d'affaiblissement partiel, peu utilisée jusque-là dans les solveurs pseudo-booléens.

Toujours dans le but d'améliorer l'efficacité pratique des solveurs pseudo-booléens, j'ai développé un certain nombre d'heuristiques pour choisir les variables, effacer des contraintes apprises et réaliser des redémarrages pendant l'exécution du solveur. En effet, la plupart des heuristiques implantées dans les solveurs pseudo-booléens sont directement héritées des solveurs SAT, et ne considèrent pas les propriétés des contraintes pseudo-booléennes. Des travaux préliminaires [7] ont montré que ces propriétés peuvent jouer un rôle important dans les performances du solveur.

J'ai implanté toutes ces fonctionnalités dans le solveur *open-source Sat4j*, et la plupart d'entre elles sont disponibles dans la version 2.3.6 de ce solveur. Afin d'évaluer les apports de ces nouvelles fonctionnalités, j'ai également réalisé de nombreuses expérimentations sur le cluster de calcul du CRIL, et analysé leurs résultats à l'aide de scripts R et Python. En particulier, j'ai participé à la réalisation de la bibliothèque *open-source Metrics* [8] permettant l'automatisation de l'analyse de résultats expérimentaux et favorisant leur reproductibilité.

En plus de l'amélioration des approches citées plus haut, plusieurs pistes de recherche existent pour permettre d'obtenir de meilleures performances dans la résolution pratique de problèmes pseudo-booléens. En particulier, il est apparu que certaines approches connues pour être optimales dans les solveurs SAT classiques ne le sont plus une fois implantées dans un contexte pseudo-booléen (par exemple, pour propager des littéraux, ou détecter le niveau du retour-arrière à l'issue de l'analyse de conflit). Ces problèmes récemment identifiés n'ont pour l'instant que peu été étudiés, et une solution possible pour les résoudre serait de considérer une approche parallèle. Dans le cadre de mon contrat postdoctoral, j'étudie actuellement comment une approche *spéculative* peut être envisagée dans ce contexte. De manière générale, cette solution consiste à faire des *suppositions* sur un résultat à venir (soit à l'aide d'heuristiques, soit à l'aide de techniques d'apprentissage automatique) pour pouvoir continuer à exécuter les instructions suivantes *pendant* que le résultat effectif est calculé (notamment lorsque ce calcul est coûteux à réaliser). Si la prédiction du résultat est correcte, alors l'exécution des instructions est validée, sinon elle est ignorée, et les instructions sont de nouveaux exécutées avec le bon résultat. Dans le contexte pseudo-booléen, il serait possible d'émettre des suppositions sur le niveau du retour-arrière ou les littéraux à propager, ces opérations étant à la fois coûteuses et sous-optimales. *In fine*, les solutions spéculatives pourraient permettre d'identifier des approches séquentielles (par exemple, fondées sur de nouvelles heuristiques) capables d'améliorer les performances des solveurs pseudo-booléens.

Du point de vue de la représentation des connaissances, les langages à base de contraintes pseudo-booléennes n'apportent pas beaucoup plus d'avantages que celui des formules CNF dès lors que des opérations (transformations ou requêtes) doivent être réalisées [5]. Le principal avantage de ces langages est leur concision : la représentation d'une formule pseudo-booléenne peut, dans certains cas, demander un espace exponentiel pour être représentée en une formule CNF équivalente. Si la production d'une formule équivalente n'est pas nécessaire, l'utilisation d'encodages

CNF peut permettre de diminuer fortement la taille de la formule CNF à manipuler. Dans ce contexte, il est fréquent de considérer des représentations sous forme de graphes, et d'imposer des restrictions sur la largeur de ces graphes. Cependant, nous montrons que, si la largeur des encodages est bornée, alors l'expressivité de ces encodages devient limitée, de sorte que ces encodages ne peuvent plus être utilisés que pour représenter des formules ayant une faible complexité de communication [1, 4].

De plus, lorsque des garanties de temps de réponse sont attendues pour réaliser des opérations sur les formules considérées (par exemple, dans le cas d'interactions avec l'utilisateur), l'utilisation d'encodages peut ne pas suffire, voire ne pas être applicable suivant les opérations à réaliser sur la base de connaissances. Il peut alors être intéressant de *compiler* la formule considérée, c'est-à-dire de la réécrire sous une forme dans laquelle les opérations que l'on souhaite réaliser peuvent se faire efficacement (généralement, en temps polynomial). Actuellement, la plupart des compilateurs prennent en entrée des formules CNF, et les avantages des langages pseudo-booléens n'ont pas encore été exploités dans ce domaine. En effet, la compilation se faisant généralement en suivant la trace de la preuve produite par un solveur, utiliser un système de preuve plus puissant – en l'occurence, le système des plans-coupes – devrait permettre de produire des preuves plus courtes, et donc des formes compilées de taille plus réduite. De plus, les représentations pseudo-booléennes étant en général plus compactes que les formules CNF, manipuler de telles représentations pourrait être plus simple pour des compilateurs. L'utilisation d'une représentation pseudo-booléenne des formules à compiler pourrait alors rendre envisageable la compilation de formules qui peuvent être naturellement représentées sous cette forme et requièrent une représentation sous forme CNF de taille exponentielle, comme par exemple les encodages de réseaux de neurones binaires. Une fois compilés, de tels encodages pourraient permettre de vérifier efficacement la robustesse du réseau encodé, et d'expliquer les prédictions qu'il réalise.

Enseignements

Au cours des trois dernières années, j'ai dispensé un certain nombre de TD et de TP en Informatique à l'Université d'Artois (IUT de Lens et UFR des Sciences Jean Perrin). De manière générale, je suis à même de dispenser des cours dans tous les domaines de l'informatique, à condition de disposer de suffisamment de temps pour préparer ces enseignements.

Statut	Année Universitaire	CM	TD	TP	Total (Équivalent TD)
Supplément doctoral d'enseignement	2017/2018	0	16.5	47.5	64
Supplément doctoral d'enseignement	2018/2019	0	21	43.5	64.5
Vacataire d'enseignement	2019/2020	0	0	85	56.7

Les détails concernant les enseignements sus-mentionnés sont donnés dans les sections suivantes. Par ailleurs, des lettres justifiant ces détails sont jointes dans les pièces complémentaires du dossier.

DUT1 - Introduction aux Systèmes Informatiques

Public. Première année de DUT Informatique à l'IUT de Lens (Université d'Artois), avec un groupe de 15 étudiants (TP) et un groupe de 30 étudiants (TD).

Contenu pédagogique. Travaux Dirigés (21h) sur le système de gestion de fichiers UNIX, la représentation des entiers signés et non signés, la représentation des nombres réels (norme IEEE-754) et la représentation des caractères. Travaux Pratiques (21h) sur les commandes UNIX de base et l'écriture de scripts Bash.

Période. Années universitaires 2017/2018 et 2018/2019.

DUT1 - Structures de Données et Algorithmes Fondamentaux

Public. Première année de DUT Informatique à l'IUT de Lens (Université d'Artois), avec un groupe de 15 étudiants.

Contenu pédagogique. Travaux Pratiques (19.5h) sur la programmation de fonctions en Python, les structures de données classiques (piles, files), les algorithmes de tri, les entrées/sorties et les enregistrements.

Responsabilités. Réalisation d'un sujet de projet (bataille navale) que les étudiants devaient implanter en trois semaines lors de l'année universitaire 2017/2018.

Période. Années universitaires 2017/2018 et 2018/2019.

DUT1 - Architecture et Programmation des Mécanismes de Base d'un Système Informatique

Public. Première année de DUT Informatique à l'IUT de Lens (Université d'Artois), avec un groupe de 15 étudiants.

Contenu pédagogique. Travaux Pratiques (11.5h) sur l'initiation à la gestion de la mémoire en C++ et en Java.

Période. Année universitaire 2017/2018.

L2 - Assembleur

Public. Deuxième année de Licence Informatique à l'UFR des Sciences Jean Perrin (Université d'Artois) avec un groupe de 20 étudiants.

Contenu pédagogique. Travaux Pratiques (15h) sur la structure d'un programme en Assembleur, les opérations élémentaires, les opérations bit-à-bit, les sauts conditionnels et inconditionnels, les différents types d'adressages, les entrées/sorties et les procédures.

Période. Année universitaire 2019/2020.

L3 - Programmation C Avancée

Public. Troisième année de Licence Informatique à l'UFR des Sciences Jean Perrin (Université d'Artois) avec un groupe de 20 étudiants.

Contenu pédagogique. Travaux Pratiques (18h) sur les structures de données, les fonctions (pointeurs de fonction, nombre d'arguments variable), la gestion de la mémoire (allocation dynamique, pointeurs génériques), le préprocesseur (constantes, macros, compilation conditionnelle), et la compilation à l'aide d'un fichier Makefile (gestion des dépendances, compilation incrémentale, règles génériques).

Responsabilités. Réalisation d'un sujet de projet en deux parties (inspiré du jeu *Leek Wars*), que les étudiants devaient implanter en deux mois, et conception du sujet d'examen de seconde session.

Période. Année universitaire 2019/2020.

L3 - Conception Orientée Objet

Public. Troisième année de Licence Informatique à l'UFR des Sciences Jean Perrin (Université d'Artois), avec deux groupes de 20 étudiants.

Contenu pédagogique. Travaux Pratiques (18h) sur les bases de la programmation avec Java 11 (notions élémentaires du langage, modules), la réalisation de diagrammes UML, les bonnes pratiques de la programmation objet (SonarQube), l'utilisation de patrons de conception (initiation) et la réalisation d'interfaces graphiques avec JavaFX (modèle MVC, FXML, programmation événementielle). Utilisation d'outils de travail collaboratif (GitLab, Mattermost).

Responsabilités. Préparation des sujets de TP hebdomadaires, conception d'un sujet de projet implanté par les étudiants au cours du semestre (jeu du Labyrinthe).

Période. Année universitaire 2019/2020.

L3 - Lambda Calcul et Programmation Fonctionnelle

Public. Troisième année de Licence Informatique à l'UFR des Sciences Jean Perrin (Université d'Artois) avec un groupe de 20 étudiants.

Contenu pédagogique. Travaux Pratiques (16h) sur l'initiation à la programmation fonctionnelle en Haskell, l'écriture de fonctions, la récursivité, la manipulation de listes, le *pattern matching* et les fonctions d'ordre supérieur (*map*, *fold* et *filter*).

Période. Année universitaire 2019/2020.

Encadrement de Stages de Master

Avril 2019 – Juin 2019. Encadrement d'un Travail d'Etude et de Recherche (TER) d'un stagiaire en première année de Master sur l'implantation d'un logiciel permettant d'encoder des problèmes de programmation par contraintes à l'aide de formules pseudo-booléennes. Mise en place de réunions hebdomadaires pendant les dix semaines du stage, relecture du rapport et répétition de la soutenance.

Avril 2019 – Juin 2019. Encadrement d'un Travail d'Etude et de Recherche (TER) d'un stagiaire en première année de Master sur l'implantation d'un outil permettant d'automatiser l'analyse de résultats expérimentaux. Mise en place de réunions hebdomadaires pendant les dix semaines du stage, relecture du rapport et répétition de la soutenance.

Animations et Responsabilités Collectives

Janvier 2019 – Décembre 2020. Membre élu du Conseil de Laboratoire du Centre de Recherche en Informatique de Lens (CRIL), représentant des personnels non-permanents : participation aux réunions mensuelles du Conseil de Laboratoire.

Janvier 2018 – Janvier 2020. Coach pour les étudiants participant à des compétitions de programmation (SWERC, Google HashCode, Battle Dev) : organisation de séances d'entraînement chaque jeudi après-midi, comportant des tutoriels d'algorithmique et des simulations de compétitions pour les étudiants en troisième année de Licence Informatique et en Master Informatique à l'UFR des Sciences Jean Perrin (Université d'Artois).

Novembre 2019. Organisation d'une compétition de programmation pour les étudiants en troisième année de Licence Informatique à l'UFR des Sciences Jean Perrin (Université d'Artois) : préparation de la compétition les quinze jours la précédant, mise en place de la compétition sur deux jours, avec une journée dédiée à l'entraînement et une journée dédiée à la compétition, choix de sujets avec implantation de leur correction, organisation d'une remise de diplôme lors d'une cérémonie de clôture.

Mai 2019 – Juin 2019. Organisation des Journées Des Doctorants (JDD) du Centre de Recherche en Informatique de Lens (CRIL) à Ostende (Belgique), à raison d'un à deux jours par semaine pendant un mois : choix et réservation du lieu du séminaire, collecte des sujets présentés par les doctorants, organisation d'un planning sur deux jours avec repas, pauses café et nuit à l'hôtel, en collaboration avec un permanent du laboratoire et un autre doctorant.

Présentations Orales, Séminaires et Invitations

Janvier 2021. Présentation d'un article à l'*International Joint Conference on Artificial Intelligence*, IJCAI 2020 (visioconférence). *On Irrelevant Literals in Pseudo-Boolean Constraint Learning* [3].

Septembre 2020. Deux présentations dans le cadre des séminaires du CRIL. *Metrics : A Unified Library for Experimenting Solvers* [8].

Août 2020. Séminaire invité au groupe de recherche *Mathematical Insights into Algorithms for Optimization* (visioconférence). *Tuning Sat4j PB solvers for decision problems*. `http://www.csc.kth.se/~jakobn/videoseminars/`.

Juillet 2020. Présentation d'un article à l'*International Conference on Theory and Applications of Satisfiability Testing*, SAT 2020 (visioconférence). *On Weakening Strategies for PB Solvers* [2].

Juillet 2020. Présentation d'un article à l'atelier *Pragmatics of SAT*, POS 2020 (visioconférence). *On Adapting CDCL Strategies for PB Solvers* [7].

Juillet 2020. Présentation d'un article dans le cadre des séminaires *France@International* de la *Plateforme Intelligence Artificielle*, PFIA 2020 (visioconférence). *On Irrelevant Literals in Pseudo-Boolean Constraint Learning* [3].

Juillet 2019. Présentation d'un article à l'atelier *Pragmatics of SAT*, POS 2019, Lisbonne, Portugal. *On Irrelevant Literals in Pseudo-Boolean Constraint Learning* [9].

Juin 2019. Présentation d'un article aux *Journées Francophones de Programmation par Contraintes*, JFPC 2019, Albi, France. *De la pertinence des littéraux dans les contraintes pseudo-booléennes apprises* [6].

Août 2018. Invité à l'atelier Banff *Theory and Practice of Satisfiability Solving* (18w5208), Oaxaca, Mexique.

Juillet 2018. Présentation d'un article à l'*International Joint Conference on Artificial Intelligence*, IJCAI 2018, Stockholm, Suède. *Pseudo-Boolean Constraints from a Knowledge Representation Perspective* [5].

Juillet 2018. Présentation d'un article dans le cadre des séminaires *France@IJCAI* de la *Plateforme Intelligence Artificielle*, PFIA 2018, Nancy, France. *Pseudo-Boolean Constraints from a Knowledge Representation Perspective* [5].

Septembre 2017. Séminaire invité à KTH (Institut Royal de Technologie), Stockholm, Suède. *Pseudo-Boolean Constraints : Reasoning and Compilation*.

Publications

Description Générale des Publications

Les publications présentées ici sont celles publiées au cours de ma thèse. La politique du CRIL est de publier dans les conférences les plus reconnues du domaine, comme c'est le cas pour SAT (*Theory and Applications of Satisfiability Testing*), qui est la principale conférence spécialisée sur la résolution du problème de cohérence propositionnelle, et pour IJCAI (*International Joint Conference on Artificial Intelligence*), qui est l'une des plus importantes conférences généralistes en intelligence artificielle.

Par ailleurs, il est important de noter qu'au sein du laboratoire, l'usage est de placer les auteurs par ordre alphabétique dans les publications.

Liste des Publications

Revues Internationales avec Comité de Lecture

1. Stefan Mengel et Romain Wallon. Graph Width Measures for CNF-Encodings with Auxiliary Variables. Dans *Journal of Artificial Intelligence Research (JAIR)*, volume 67, pages 409-436, 2020. `https://doi.org/10.1613/jair.1.11750`.

Conférences Internationales avec Comité de Lecture

2. Daniel Le Berre, Pierre Marquis et Romain Wallon. On Weakening Strategies for PB Solvers. Dans *23rd International Conference on Theory and Applications of Satisfiability Testing (SAT'20)*, pages 322-331 (article court), 2020. `https://doi.org/10.1007/978-3-030-51825-7_23`.
3. Daniel Le Berre, Pierre Marquis, Stefan Mengel et Romain Wallon. On Irrelevant Literals in Pseudo-Boolean Constraint Learning. Dans *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI'20)*, pages 1148-1154, 2020. `https://doi.org/10.24963/ijcai.2020/160`.
4. Stefan Mengel et Romain Wallon. Revisiting Graph Width Measures for CNF-Encodings. Dans *22nd International Conference on Theory and Applications of Satisfiability Testing (SAT'19)*, pages 222-238, 2019. `https://doi.org/10.1007/978-3-030-24258-9_16`.
5. Daniel Le Berre, Pierre Marquis, Stefan Mengel et Romain Wallon. Pseudo-Boolean Constraints from a Knowledge Representation Perspective. Dans *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI'18)*, pages 1891-1897, 2018. `https://doi.org/10.24963/ijcai.2018/261`.

Conférences Nationales avec Comité de Lecture

6. Daniel Le Berre, Pierre Marquis, Stefan Mengel et Romain Wallon. De la pertinence des littéraux dans les contraintes pseudo-booléennes apprises. Dans *Actes des 15es Journées Francophones de Programmation par Contraintes (JFPC'19)*, pages 43-52, 2019.
`https://hal-mines-albi.archives-ouvertes.fr/hal-02159866/document#page=60`.

Workshops Internationaux avec Comité de Lecture

7. Romain Wallon. On Adapting CDCL Strategies to PB Solvers. Dans *11th International Workshop on Pragmatics of SAT (POS'20)*, 16 pages, 2020.
`http://www.pragmaticsofssat.org/2020/`.
8. Thibault Falque, Romain Wallon et Hugues Wattez. Metrics : Towards a Unified Library for Experimenting Solvers. Dans *11th International Workshop on Pragmatics of SAT (POS'20)*, 14 pages, 2020.
`http://www.pragmaticsofssat.org/2020/`.
9. Daniel Le Berre, Pierre Marquis, Stefan Mengel et Romain Wallon. On Irrelevant Literals in Pseudo-Boolean Constraint Learning. Dans *10th International Workshop on Pragmatics of SAT (POS'19)*, 16 pages, 2019.
`http://www.pragmaticsofssat.org/2019/`.

Description des Publications Jointes

J'ai choisi de joindre à ce document les articles *Pseudo-Boolean Constraints from a Knowledge Representation Perspective* [5], *On Irrelevant Literals in Pseudo-Boolean Constraint Learning* [3] et *On Weakening Strategies for PB Solvers* [2].

Le premier de ces articles [5] correspond aux résultats obtenus pendant mon stage de deuxième année de Master, et constitue le socle de mes travaux sur l'étude des contraintes pseudo-booléennes du point de vue de la représentation des connaissances. En particulier, cet article considère les critères de la carte de compilation afin de déterminer quelles *requêtes* et *transformations* peuvent être réalisées efficacement sur des formules pseudo-booléennes, ainsi que la *concision* de ces formules (i.e., leur capacité à représenter de l'information en utilisant peu d'espace). Toutes les preuves présentées dans cet article ont été réalisées par mes soins, à l'exception de l'une d'entre elles (Proposition 5.9), qui a été proposée par Pierre Marquis. Elles permettent notamment de montrer que le principal avantage des formules pseudo-booléennes est leur concision par rapport au langage CNF largement utilisé.

Les deux autres articles s'articulent autour de l'un des principaux résultats de ma thèse : la présence de littéraux non-pertinents dans les contraintes pseudo-booléennes apprises. En effet, les solveurs pseudo-booléens utilisant le système des plans-coupes peuvent dériver des contraintes contenant des littéraux non-pertinents [3]. Il s'agit là d'un talon d'Achille pour ces solveurs : les contraintes qu'ils apprennent peuvent être plus faibles en leur présence, illustrant la nécessité d'identifier des approches permettant d'éviter leur production. Nous proposons donc dans un premier temps une approche pour les éliminer après leur production [3]. Nous avons par la suite conçu des stratégies d'affaiblissement dans le but d'éviter la production de ces littéraux, notamment avec la stratégie *weaken-ineffective* introduite dans [2], tout en étudiant d'autres stratégies permettant de claires améliorations dans la résolution pratique de problèmes pseudo-booléens, notamment avec la stratégie *PartialRoundingSat*. Ces deux articles présentent des travaux dont je suis à l'origine. J'ai également réalisé leur implantation dans le solveur *Sat4j* et analysé les résultats expérimentaux de ces approches.

Production Logicielle

Je suis un contributeur du logiciel *open-source Sat4j* (`https://gitlab.ow2.org/sat4j/sat4j`), dans lequel j'ai procédé à de nombreuses améliorations. J'ai notamment développé au sein de ce solveur un certain nombre de fonctionnalités en lien avec des articles publiés dans différentes conférences, telles que la suppression des littéraux non-pertinents dans les contraintes pseudo-booléennes apprises [3, 6, 9] et différentes stratégies d'affaiblissement [2]. J'ai également développé de nombreuses heuristiques de décision, stratégies de suppressions des contraintes apprises et politiques de redémarrage qui ont permis d'améliorer les performances du solveur de manière significative [7]. La plupart

de ces fonctionnalités sont disponibles sur la branche principale de *Sat4j*. Elles constituent l'essentiel des nouveautés introduites dans la version 2.3.6 de ce solveur (`https://gitlab.ow2.org/sat4j/sat4j/-/releases/2_3_6`).

Par ailleurs, j'ai participé à la conception de la bibliothèque *open-source Metrics*, conçue pour faciliter l'analyse de résultats expérimentaux (`https://github.com/crillab/metrics`). En particulier, cette bibliothèque propose une chaîne d'outils complète, permettant d'extraire les données à partir des fichiers de *log* produits par les solveurs (indépendamment du format utilisé par ceux-ci), puis de les charger en mémoire pour ensuite tracer des figures couramment utilisées dans la communauté pour analyser les performances de solveurs (entre autres : boîtes à moustaches, *scatter plots*, *cactus plots* ou CDF). Une application web est également mise à la diposition du public pour permettre son utilisation sans passer par une installation locale (`http://crillab-metrics.cloud/`).