CNRS-Momentum 2018

PROJET SCIENTIFIQUE

Veuillez enregistrer ce document au format PDF et le nommer :
Momentum2018_Projet_NOM_Prenom.

NOM : FIJALKOW
Prénom : Nathanaël

THÉMATIQUE CHOISIE[1] :

Relever les défis de l'apprentissage automatique

TITRE

Deep Synthesis : Apprentissage pour la Synthèse de Programmes

DOMAINES DE RECHERCHE – MOTS-CLÉS

Apprentissage automatique
Synthèse de programmes
Programmer par l'exemple (Programming by Example)
Intelligence artificielle
Méthodes formelles

RÉSUMÉ

Écrire un programme informatique, tel qu'un logiciel ou un site internet, est une tâche longue, coûteuse, et source d'erreurs. La synthèse de programmes est un idéal où le programme est généré automatiquement à partir de sa spécification. Il s'agit d'un problème très difficile et nécessitant de fortes ressources, en particulier en temps de calcul. Récemment, de nombreux résultats expérimentaux repoussent les limites de ce qu'il est possible de synthétiser, en même temps que progresse notre compréhension théorique.

L'objectif du projet DeepSynth est d'utiliser des techniques d'apprentissage pour la synthèse de programmes. Au cours des années, les méthodes statistiques basées sur l'apprentissage et en particulier l'apprentissage profond ont dépassé d'autres

1 1. Modéliser le vivant / 2. Comprendre les réseaux complexes / 3. Décrypter les algorithmes du cerveau / 4. Appliquer les sciences des données à la Terre et l'Univers / 5. Doper les sciences pour le sport / 6. Evaluer les effets des faibles doses / 7. Etudier les phénomènes aux interfaces physiques / 8. Relever les défis de l'apprentissage automatique / 9. Concevoir des systèmes inspirés de la nature / 10. Faire voir l'invisible / 11. Repenser l'écosystème urbain / 12. Revisiter le cycle du carbone / 13. Explorer l'intelligence collective

approches et ont obtenu des résultats impressionants. Le problème de la synthèse est un candidat idéal pour développer des techniques d'apprentissage, puisqu'il peut être vu comme un problème d'optimisation dans un très grand espace de solutions.

PROJET DE RECHERCHE

Voir ci-dessous

Pour les chercheurs titulaires du CNRS : **Nom, prénom et signature du Directeur d'unité :**

Jean-Philippe DOMENGER

Programme Momentum 2018
DeepSynth : apprentissage pour la synthèse de programmes

Nom du porteur du projet	Nathanaël Fijalkow
Unité de recherche	LaBRI (UMR 5800)
Employeur du porteur	CNRS
Adresse e-mail du porteur	nathanael.fijalkow@labri.fr
Titre long du projet	Apprentissage pour la synthèse de programmes
Mots-clés	Intelligence artificielle, Méthodes formelles, Synthèse de programmes, Apprentissage

Résumé

Écrire un programme informatique, tel qu'un logiciel ou un site internet, est une tâche longue, coûteuse, et source d'erreurs. La synthèse de programmes est un idéal où le programme est généré automatiquement à partir de sa spécification. Il s'agit d'un problème très difficile et nécessitant de fortes ressources, en particulier en temps de calcul. Récemment, de nombreux résultats expérimentaux repoussent les limites de ce qu'il est possible de synthétiser, en même temps que progresse notre compréhension théorique.

L'objectif du projet DeepSynth est d'utiliser des techniques d'apprentissage pour la synthèse de programmes. Au cours des années, les méthodes statistiques basées sur l'apprentissage et en particulier l'apprentissage profond ont dépassé d'autres approches et ont obtenu des résultats impressionants. Le problème de la synthèse est un candidat idéal pour développer des techniques d'apprentissage, puisqu'il peut être vu comme un problème d'optimisation dans un très grand espace de solutions.

1 Court Curriculum Vitæ

Parcours professionnel

— Depuis Janvier 2018 : Chargé de recherche au CNRS, Équipe Méthodes formelles, LaBRI, Université de Bordeaux

— Janvier 2017 – Décembre 2021 : Chercheur associé à l'Institut Alan Turing, Londres. Responsable du groupe "fondations logiques pour la science des données"

— Août 2016 – Décembre 2016 : Chercheur associé au Simons Institute, Université de Californie, Berkeley.

— Novembre 2015 – Juillet 2016 : Post-doctorant à l'Université d'Oxford.

— Septembre 2012 – Octobre 2015 : Thèse dirigée par Mikołaj Bojańczyk et Thomas Colcombet en cotutelle Université Paris 7 et Varsovie.

Évènements scientifiques reliés au projet Deep Synthesis

— Juillet 2018 : co-organisation de l'école FoPSS "Logic and learning" à Oxford, affiliée à LICS.

— Juillet 2018 : co-organisation du workshop "Summit on Machine Learning Meets Formal Methods" à Oxford, affiliée à FLOC.

— Janvier 2018 : co-organisation du workshop "Logic and learning" à Londres.

— Depuis 2018 : organisation du groupe de lecture Théorie de l'apprentissage au LaBRI.

2 État de l'art, problématique et objectifs

État de l'art

Le problème de la synthèse de programmes est étudié dans le cadre des méthodes formelles, basées sur **l'abstraction d'un système dans un modèle mathématique et sa spécification à l'aide de formalismes logiques**. Dans le cadre de la synthèse un système désigne un programme informatique, mais la notion de système est plus générale. Cette approche permet de raisonner de manière mathématique et automatisée sur des systèmes et de formellement prouver qu'ils sont corrects. Ceci a donné naissance en premier lieu au "model checking" (délicat à traduire en français) : le modèle satisfait-il la spécification logique ? Cette idée fondamentale, proposée en particulier par Clark et récompensée d'un prix Turing, a créé un champ de recherche, la vérification, ayant de nombreuses retombées dans l'industrie.

Le principal défaut du model-checking est de donner une réponse booléenne : vrai ou faux. Si le modèle ne satisfait pas la spécification, que faire ? C'est le point de départ de la synthèse, qui renverse cette question et propose de **synthétiser un modèle à partir de la spécification**.

Réaliser la synthèse de programmes est un enjeu technologique important. En effet, écrire un programme informatique, tel qu'un logiciel ou un site internet, est une tâche longue, coûteuse, et source d'erreurs. La synthèse de programmes automatise ce processus, tout en le rendant infaillable : le programme généré satisfait la spécification, puisque c'est exactement pour cela qu'il a été créé.

Malheureusement, **le problème de la synthèse est très difficile** et nécessite de fortes ressources, en particulier en temps de calcul. Récemment, de nombreux résultats expérimentaux repoussent les limites de ce qu'il est possible de synthétiser, en même temps que progresse notre compréhension théorique.

L'objectif du projet DeepSynth, et plus globalement de mon projet de recherche, est **d'intégrer des techniques d'apprentissage pour effectuer la synthèse de programmes**. Ce changement de paradigme peut mener à des résultats théoriques et pratiques, en combinant la puissance des méthodes formelles avec l'efficacité des techniques d'apprentissage.

Problématique

A priori, tout oppose les méthodes formelles et l'apprentissage. Les premières mettent au premier plan les garanties mathématiques, tandis que le second s'appuie sur des méthodes statistiques, efficaces avec forte probabilité. **L'approche des méthodes formelles est systématique**, basée sur l'inférence de règles logiques dont l'objectif est de décrire le système avec précision. Au contraire, **l'apprentissage utilise des techniques d'optimisation impliquant des choix aléatoires** pour se rapprocher du comportement du système.

Au cours des années, les méthodes statistiques basées sur l'apprentissage, et en particulier l'apprentissage profond, ont dépassé d'autres approches, obtenant des résultats impressionants. La force de ces techniques est de trouver de manière automatisée et d'une certaine manière incontrôlée et incontrôlable des structures de plus en plus précises.

Un exemple frappant très relié au projet DeepSynth est le développement de Alpha Go et Alpha Go Zero [SHM+16]. L'équipe de Google DeepMind a créé une intelligence artificielle pour le jeu de Go qui bat n'importe quel joueur professionnel, ce que l'on pensait irréalisable dans un futur proche. Cette idée que le Go était trop compliqué pour l'intelligence artificielle était basée sur l'explosion combinatoire inhérente, le nombre de parties et de positions étant largement supérieur à ce qu'un ordinateur peut avoir en mémoire. La performance réalisée par Alpha Go montre que les algorithmes d'apprentissage ont réussi à trouver des motifs dans le jeu de Go, isolant des séquences de coups et des ressemblances dans les stratégies.

Comment combiner les méthodes formelles et l'apprentissage ? L'idée est d'utiliser les algorithmes de méthodes formelles comme cadre rigide et précis, et les techniques d'apprentissage comme puissance d'exploration. De fait, de nombreux algorithmes de synthèse sont modulables et dépendent de diverses heuristiques influençant les choix à effectuer et directions à prendre. Les heuristiques utilisées sont en général assez simples et locales : elles suggèrent un choix dans une configuration donnée, sans posséder une vue sur l'exécution globale de l'algorithme. L'idée clé du projet DeepSynth est de décrire ces heuristiques avec des techniques d'apprentissage, en leur donnant une place centrale dans la construction des algorithmes.

Objectifs

L'objectif de mon programme de recherche est une **intégration forte entre méthodes formelles et apprentissage**. Le projet DeepSynth sert d'amorçage à ce programme, en se concentrant sur la synthèse de programmes, et sur deux problèmes en particulier réalisant la synthèse de programmes : les jeux de parité et les processus de décisions markoviens.

La réalisation de ce programme s'appuie sur la combinaison de deux compétences : d'une part, une compréhension des problématiques et approches pour la synthèse de programmes, et de manière plus générale de la vérification de programmes, et d'autre part une connaissance théorique et pratique des algorithmes d'apprentissage. Ces deux points de vue sont traditionnellement séparés, **l'innovation du projet DeepSynth est de les rassembler**.

Mon domaine d'expertise couvre le premier domaine, synthèse et vérification. J'ai construit de nombreux outils théoriques en théorie des jeux [Fij18] et pour la vérification probabiliste [FKP17]. J'ai également participé au développement de logiciels pour mettre en pratique ces constructions [FGKK17]. Au cours des années passées en Angleterre j'ai introduit des méthodes d'apprentissage dans ma recherche.

3 Méthodologie

Le cadre théorique pour l'étude de la synthèse de programmes que j'étudie est celui des jeux, issu de la théorie des automates et de la logique. Cette formalisation est utilisée à l'échelle internationale et chaque année l'objet d'articles de recherche dans les conférences les plus prestigieuses du domaine (LICS, ICALP, CAV, STACS).

De manière informelle, le problème de la synthèse de programmes est reformulé comme suit. Deux joueurs s'affrontent : le contrôleur et l'environnement. Le contrôleur est responsable de l'exécution du programme, et l'environnement décrit toutes les interactions du programme avec l'extérieur (par exemple, entrées du programmes, variables partagées avec d'autres programmes). La spécification du programme devient la condition de gain du contrôleur. L'objectif est de construire une stratégie gagnante pour le contrôleur, qui correspond exactement au programme que l'on souhaite synthétiser. Cette terminologie permet d'utiliser un cadre formel précis et de **construire des algorithmes efficaces pour la problème de la synthèse de programmes**.

Premier axe de recherche : jeux de parité

Les jeux de parité sont le modèle de jeux le plus étudié en vérification, et constituent la base sur laquelle de nombreuses approches s'appuient. En effet, résoudre les jeux de parité est équivalent au problème du model-checking du μ-calculus modal, qui est un formalisme logique largement utilisé en vérification. Un second intérêt des jeux de parité est leur statut théorique : résoudre les jeux de parité est dans $\mathbf{NP} \cap \mathrm{co}\mathbf{NP}$, et nous ne connaissons pas d'algorithme en temps polynomial. Les algorithmes les plus efficaces connus à ce jour ont été implantés dans l'outil PGsolver.

Très récemment, deux articles ont permis une avancée majeure [JL17, CJK+17], en proposant deux algorithmes pour résoudre les jeux de parité en temps quasi-polynomial, *i.e.* $n^{\log(n)}$, où n est la taille du jeu. L'intérêt de ces nouveaux algorithmes est de **reformuler la résolution des jeux de parité comme un problème d'optimisation**. C'est à partir de là que vient naturellement l'idée de résoudre ce problème en utilisant des techniques d'apprentissage, capables de résoudre des problèmes d'optimisation plus rapidement que toute autre méthode à ce jour. L'objectif de cet axe du projet DeepSynth est le suivant :

> Construire des algorithmes efficaces pour résoudre les jeux de parité en s'appuyant sur des méthodes d'apprentissage.

Deuxième axe de recherche : processus de décisions markoviens

Les processus de décisions markoviens forment le socle de la vérification de systèmes probabilistes, et sont également étudiés dans beaucoup d'autres domaines. En particulier, le "reinforcement learning" (un second terme que je ne saurais traduire) utilise ce modèle pour apprendre (au sens acquérir) des systèmes, comme c'est le cas de Alpha Go Zero [SHM+16].

D'un point de vue théorique, la résolution des processus de décisions markoviens est bien comprise. Elle se réduit au problème de la programmation linéaire pour lequel il existe un algorithme polynomial ainsi que plusieurs algorithmes exponentiels mais plus efficaces en pratique. L'état de l'art a été implanté dans l'outil PRISM dirigé par Marta Kwiatkowska [KNP11].

Cependant, pour déployer ces techniques à des échelles industrielles, notre connaissance théorique et pratique n'est parfois pas encore suffisante. Résoudre un processus de décision markovien consiste à construire une stratégie optimale. Dans certains cas, quand les instances sont très grandes, ou possèdent une certaine structure, obtenir une stratégie optimale quelconque n'est pas satisfaisant. Par exemple, une telle stratégie peut être trop compliquée à implanter, parce que sa description est trop longue, ou alors ne pas respecter certaines contraintes structurelles. Jan Kretinsky, Marta Kwiatkowska et leurs collaborateurs ont récemment proposé plusieurs approches pour pallier à ces manques en utilisant des techniques d'apprentissage [BCC+15].

L'objectif de cet axe du projet DeepSynth est **d'étudier la notion de stratégie implantable**. Il existe plusieurs moyens de représenter une stratégie dans un processus de décision markovien, de manière arborescente par exemple. Cependant, les algorithmes les plus efficaces développés à ce jour ne prennent pas cet aspect en compte et décrivent les stratégies de la manière la plus naïve possible, comme une liste d'actions. **Les méthodes d'apprentissage permettent de reconnaître des motifs et des structures**, donnant une décomposition structurelle de l'espace des solutions et impliquant une description simple de ces solutions. L'objectif s'énonce comme suit :

> Construire des algorithmes efficaces pour résoudre les processus de décisions markoviens en s'appuyant sur des méthodes d'apprentissage afin de construire des stratégies optimales et implantables.

4 Résultats escomptés

Le projet DeepSynth combine une approche théorique de construction d'algorithmes et sa validation par la pratique par l'implantation des algorithmes obtenus. Son progrès pourra être évalué au cours de la réalisation du projet à la fois par la compréhension et construction de nouveaux algorithmes mais également par leur succès dans des situations industrielles concrètes. Les

résultats obtenus seront publiés dans des conférences et des journaux au cours de la réalisation du projet.

Les retombées attendues du projet DeepSynth sont les suivantes :

— **Publications**. Les résultats obtenus dans chacun des deux axes feront l'objet de publications dans des conférences et journaux internationaux de premier plan.

— **Développement logiciel**. Les algorithmes construits seront implantés et comparés aux implantations existantes dans les outils PGsolver et PRISM. L'objectif est de démontrer le potentiel des techniques d'apprentissage pour la synthèse de programme d'un point de vue pratique, ouvrant de nombreuses perspectives pour ses retombées industrielles.

— **Dépôt de candidature pour un projet européen**. Le programme Momentum est une opportunité pour monter en puissance et à son issue déposer une candidature pour une "ERC Starting Grant".

5 Collaborations envisagées

Collaborateurs à Bordeaux

Plusieurs membres de l'équipe Méthodes Formelles (LaBRI) seront naturellement inclus dans le développement du projet DeepSynth pour leur expertise en apprentissage et en méthodes formelles : Hugo Gimbert, Gabriele Puppis, et Laurent Simon. Je prévois également de collaborer avec Jérémie Bigot (IMB, Institut Mathématiques de Bordeaux), qui s'intéresse aux aspects mathématiques de l'appentissage profond.

Collaborateurs en France

Pierre Ohlmann commence sa thèse en September 2018 en co-direction avec Olivier Serre (IRIF, Paris 7) et moi-même. Le projet DeepSynth serait un excellent tremplin pour développer sa recherche. Je prévois également de continuer de longues et fructueuses collaborations avec Thomas Colcombet (IRIF, Paris 7), Denis Kuperberg (ENS Lyon), Nathalie Bertrand et Blaise Genest (IRISA Rennes).

Collaborateurs à l'étranger

Je prévois de collaborer à l'internationale selon l'avancement du projet, par exemple avec les chercheurs suivants :

— Lukasz Kaiser (Google Brain)

— Borja Balle (Amazon Cambridge)

— Anuj Dawar (Université de Cambridge)

— Marcin Jurdziński et Ranko Lazić (Université de Warwick)

— Jan Kretinsky (Université technique de Munich)

— Marta Kwiatkowska (Université d'Oxford)

— Prakash Panangaden (Université du Québec à Montréal).

Tous sont des chercheurs de premier plan activement impliqués dans la direction de recherche du projet DeepSynth, à savoir combiner méthodes formelles et apprentissage.

Collaboration avec l'Institut Alan Turing of data science à Londres

J'ai établi un contrat de collaboration entre l'Université de Bordeaux et l'Institut Turing, me permettant de conserver ma fonction de Research Fellow en plus d'être Chargé de Recherche au CNRS. Je co-dirige avec David Pym à l'institut Turing un groupe de recherche appelé "fondations logiques de la science des données". Nous organisons de nombreuses activités autour du thème "logique et apprentissage" qui nourrissent ma réflexion autour de ces questions au cœur du projet DeepSynth et de mon programme de recherche.

L'institut Turing a de forts liens scientifiques avec Google DeepMind, qui sera également un partenaire scientifique naturel pour le développement du projet DeepSynth, en particulier Richard Evans.

6 Calendrier et budget

La réalisation du projet DeepSynth se fera en deux phases :

— la première année (2019), développements théoriques et méthodologiques, publications des résultats théoriques, et identification des potentielles applications industrielles,

— les deuxièmes et troisièmes années (2020 et 2021), implantation des algorithmes et tests à différentes échelles. Collaboration avec des partenaires industriels (Google et Amazon).

Je sollicite pour la réalisation du projet DeepSynth :

— Deux ans de salaire pour un post-doctorant ou ingénieur.

— Frais de fonctionnement : 50 000 par ans. Ceci couvre l'organisation de missions pour les membres actifs de projet ainsi que l'organisation de plusieurs évènements permettant de communiquer sur l'avancement du projet.

Références

[BCC+15] Tomás Brázdil, Krishnendu Chatterjee, Martin Chmelik, Andreas Fellner, and Jan Kretínský. Counterexample explanation by learning small strategies in Markov decision processes. In *Computer Aided Verification - 27th International Conference, CAV*, 2015.

[CJK+17] Cristian S. Calude, Sanjay Jain, Bakhadyr Khoussainov, Wei Li, and Frank Stephan. Deciding parity games in quasipolynomial time. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, 2017.

[FGKK17] Nathanaël Fijalkow, Hugo Gimbert, Edon Kelmendi, and Denis Kuperberg. Stamina : Stabilisation monoids in automata theory. In *Implementation and Application of Automata - 22nd International Conference, CIAA*, 2017.

[Fij18] Nathanaël Fijalkow. The state complexity of alternating automata. In *33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS*, 2018.

[FKP17] Nathanaël Fijalkow, Bartek Klin, and Prakash Panangaden. Expressiveness of probabilistic modal logics, revisited. In *44th International Colloquium on Automata, Languages, and Programming, ICALP*, 2017.

[JL17] Marcin Jurdziński and Ranko Lazić. Succinct progress measures for solving parity games. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS*, 2017.

[KNP11] Marta Z. Kwiatkowska, Gethin Norman, and David Parker. PRISM 4.0 : Verification of probabilistic real-time systems. In *Computer Aided Verification - 23rd International Conference, CAV*, 2011.

[SHM+16] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Vedavyas Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy P. Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587) :484–489, 2016.