Preuve de l'algorithme PNDFS avec Pulse-F*

2025

1 Informations générales

Encadrement	Sami Evangelista : evangelista [@] lipn.univ-paris13.fr Micaela Mayero : mayero [@] lipn.univ-paris13.fr Franck Butelle : butelle [@] lipn.univ-paris13.fr
Lieu	LIPN - CNRS UMR 7030 Université Sorbonne Paris Nord (USPN) 99 av. JB Clément ; 93430 VILLETANEUSE
Durée	6 - 8 semaines

2 Contexte

L'algorithme PNDFS [11] (Parallel Nested Depth-First Search) est un algorithme parallèle de parcours de graphe en profondeur d'abord de recherche de cycles acceptants. Cet algorithme résout le test de vacuité des automates de Büchi en lançant plusieurs instances de l'algorithme séquentiel NDFS [8].

Ces algorithmes sont utilisés dans la plupart des *model checkers* (SPIN [4], LTSMIN [10]) et leur correction est primordiale. Prouver formellement cette correction est une des méthodes les plus fiables. Plusieurs outils de preuves formelles [3] et de preuves de programmes [6] existent, mais peu d'entre eux arrivent à traiter des programmes parallèles/concurrents [1, 5].

L'objectif à long terme est de prouver formellement l'algorithme [9] qui est une optimisation de PNDFS. A cette fin, la preuve de l'algorithme NDFS est en cours, en utilisant les outils Vercors et F*.

3 Objectifs

La preuve formelle de l'algorithme PNDFS (Figure 1) a déjà été faite en Vercors [12]. L'objectif de ce stage est de la refaire en Pulse [2], une extension de F* dédié aux programmes concurrents avec des spécifications et des preuves en logique de séparation concurrente [7].

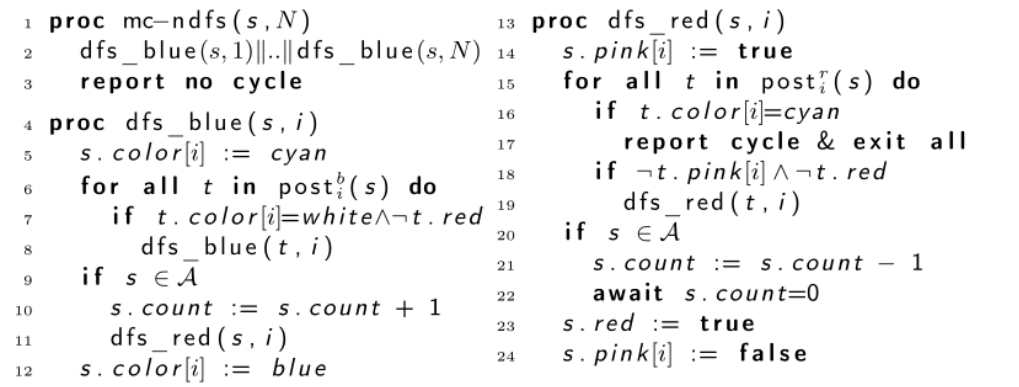

FIGURE 1 – Algorithme PNDFS

Ce travail représentera une avancée importante vers la preuve de l'algorithme [9], plus complexe et non encore prouvé formellement.

Une connaissance en preuves formelles sera appréciée (quelque soit le ou les outils connus).

Références

[1] F* website. `https://fstar-lang.org/`.

[2] Pulse documentation. `https://fstar-lang.org/tutorial/book/pulse/pulse.html`.

[3] Rocq website. `https://rocq-prover.org/`.

[4] SPIN website. `https://spinroot.com/spin/whatispin.html`.

[5] VerCors website. `https://vercors.ewi.utwente.nl/`.

[6] Why3 website. `https://www.why3.org/`.

[7] A. Charguéraud. *Separation Logic Foundations*, volume 6 of *Software Foundations*. Electronic textbook, 2024.

[8] C. Courcoubetis, M. Y. Vardi, P. Wolper, and M. Yannakakis. Memory efficient algorithms for the verification of temporal properties. In *CAV*, volume 531 of *LNCS*, pages 233–242. Springer, 1990.

[9] S. Evangelista, A. Laarman, L. Petrucci, and J. V. D. Pol. Improved multi-core nested depth-first search. In *ATVA*, volume 7561 of *LNCS*, pages 269–283. springer, 2012.

[10] G. Kant, A. Laarman, J. Meijer, J. van de Pol, S. Blom, and T. van Dijk. Ltsmin : High-performance language-independent model checking. In *TACAS*, volume 9035 of *LNCS*, pages 692–707. Springer, 2015.

[11] A. Laarman, R. Langerak, J. van de Pol, M. Weber, and A. Wijs. Multi-core nested depth-first search. In *ATVA*, volume 6996 of *LNCS*, pages 321–335. Springer, 2011.

[12] W. Oortwijn, M. Huisman, S. J. C. Joosten, and J. van de Pol. Automated verification of parallel nested DFS. In *TACAS*, volume 12078 of *LNCS*, pages 247–265. Springer, 2020.