Université Grenoble Alpes

08/11/2024

MCTS planning

Rapport

Auteurs

DURET Laura 12309182

PHILIPPE Aurore 11900982

Sommaire

1. Introduction.....2
2. Comparaison des planificateurs MCTS et ASP.....3
    a. Blocksworld.....3
    b. Depot.....4
    c. Gripper.....5
    d. Logistics.....6
3. Conclusion.....7
4. Références.....8

1. Introduction

Dans le cadre du cours de planification automatique de la deuxième année du master MIASHS, parcours informatique et cognition, de l'UGA nous avions un réaliser un planificateur Monte-Carlo tree search (MCTS) avec des marches aléatoires pures en java et pddl4j.

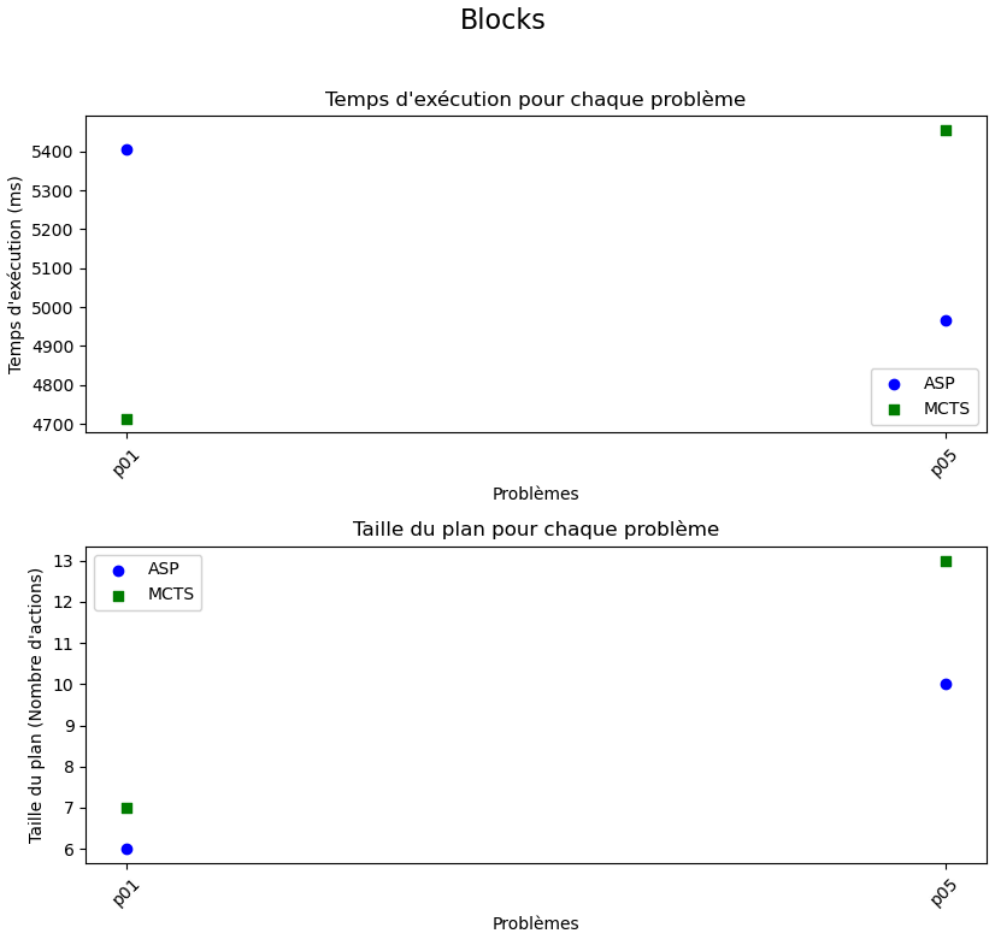
Veuillez retrouver notre répertoire GitHub via le lien suivant : https://github.com/AurorePhl/TP_MCTS.git.

Nous avons finalement pu réaliser le planificateur, mais nous n'avons pas eu le temps d'adapter les paramètres LENGTH_WALK et NUM_WALK aux problèmes afin d'améliorer les performances du planificateur.

Ce document a pour objectif de comparer les performances d'un planificateur A* avec celles de notre planificateur. Pour cela, nous comparons les temps d'exécution et la taille des plans solution trouvés pour des problèmes traités par les deux planificateurs.

2. Comparaison des planificateurs MCTS et ASP

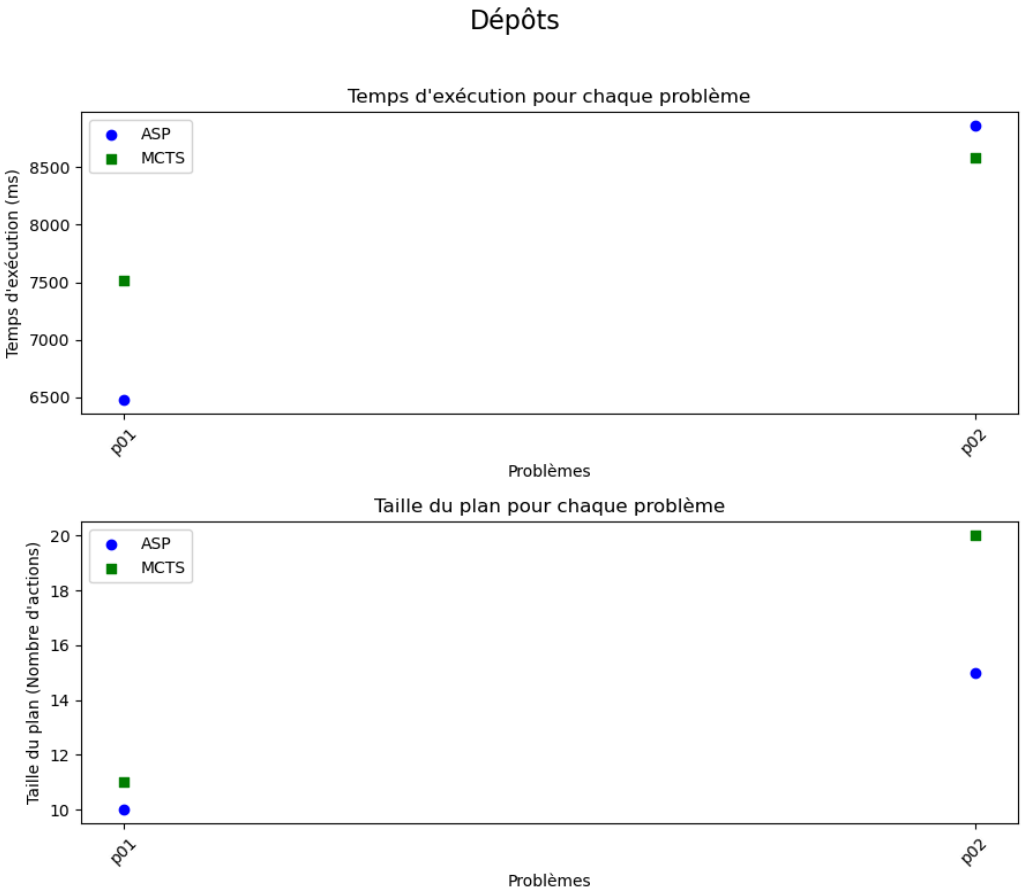
a. Blocksworld



Pour les problèmes blocks, les deux planificateurs, MCTS et ASP, trouvent des plans solutions pour les problèmes 1 et 5. Toutefois, même si les tailles des plans solution trouvés par ASP sont un peu plus petits que ceux trouvés par MCTS. De plus, le temps d'exécution du planificateur ASP est plus faible que celui du planificateur MCTS pour trois des quatres problèmes qui ont été testés.

Le planificateur ASP est donc légèrement meilleur pour résoudre les problèmes blocks que le planificateur MCTS, même si ce dernier a globalement de bonnes performances.

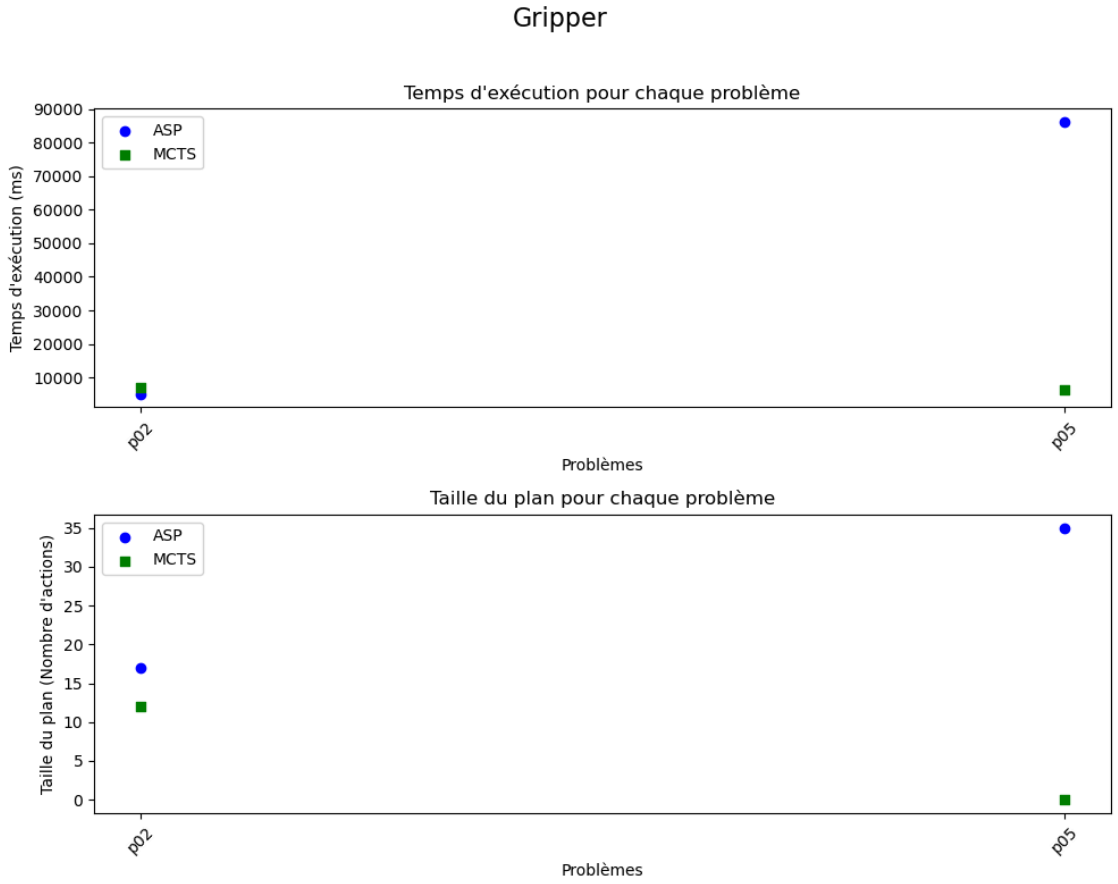
b. Depot



Pour les problèmes depots, l'ASP réalise de meilleures performances que le planificateur MCTS pour le problème 1 puisqu'il trouve un plan plus rapidement et avec moins d'actions.

Tandis que pour le second problème, le temps d'exécution est meilleur pour le MCTS mais il trouve un plan avec plus d'action que l'ASP. On peut ainsi dire que le meilleur planificateur serait l'ASP.

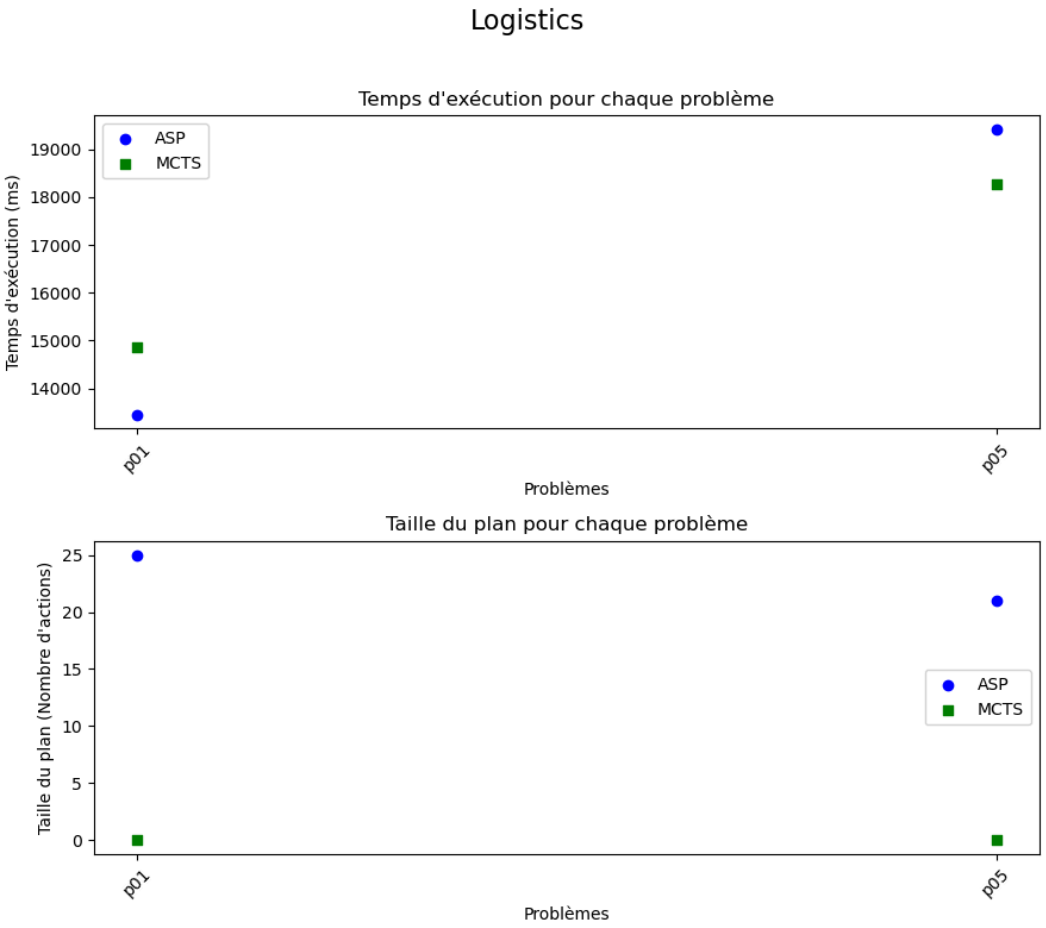
c. Gripper



Pour les problèmes Gripper, le planificateur MCTS ne trouve pas de plan solution, alors que le planificateur ASP en trouve. En revanche, MCTS trouve un plan solution plus petit que ASP pour le problème 2 Gripper.

Pour chaque problème, les temps d'exécution des deux planificateurs semblent similaires.

d. Logistics



Lors des problèmes logistics, le planificateur MCTS ne trouve pas de solution, ce qui peut être dû à un nombre trop faible de marches. Nous avons fixé le nombre de marches par défaut à 50.

Le planificateur ASP trouve des solutions pour chaque problème, ce qui fait de lui le meilleur planificateur dans ce contexte pour résoudre les problèmes logistics.

3. Conclusion

Avec des paramètres par défaut (LENGTH_WALK et NUM_WALK), le planificateur MCTS peut résoudre les problèmes plus rapidement que le planificateur ASP. Toutefois, cela limite sa recherche de solutions. En effet, il arrive souvent que le planificateur MCTS ne parvienne pas à trouver de solution pour des problèmes complexes.

La taille des plans de solution du planificateur MCTS varie pour un même problème en raison, essentiellement, du choix aléatoire des actions. En revanche, la taille des plans de solution du planificateur ASP ne varie pas pour un même problème.

Globalement, le planificateur ASP semble meilleur que le planificateur MCTS avec les marches aléatoires pures, d'autant plus si le problème est complexe. Toutefois, il faudrait tester à nouveau les performances du planificateur MCTS en adaptant les paramètres LENGTH_WALK et NUM_WALK aux problèmes, ce que nous n'avons pas eu le temps de faire dans le temps imparti.

4. Références

Pellier, D. (2021). Monte Carlo Tree Search Planning — PDDL4J Exercises 0.1 documentation. http://pddl4j.imag.fr/repository/exercices/monte_carlo_tree_search_planning.html

Pellier, D. (2021). Writing your own Planner — PDDL4J 4.0 documentation. http://pddl4j.imag.fr/writing_your_own_planner.html

Mackie, C., Schafer, H., Yazdani, N., & Jamieson, K. (2020). CSE599I : Online and Adaptive Machine Learning : Lecture 19 : Monte Carlo Tree Search. https://courses.cs.washington.edu/courses/cse599i/18wi/resources/lecture19/lecture19.pdf