Rapport de Mini-Projet Informatique

Titre de votre Projet

Nom Prénom

19 septembre 2025

Table des matières

1 Introduction

L'objectif de ce projet est la conception et la réalisation d'un système expert pour résoudre le problème du "pic-vide". Notre travail se concentre sur la mise en place d'une base de connaissances robuste et d'un moteur d'inférence en chaînage avant, afin d'explorer et de valider une solution algorithmique.

—

2 Mise en place de la Base de Connaissance

Nous avons structuré notre base de connaissances autour de la **situation de jeu**, qui stocke l'état du plateau à un instant donné. La **base de faits** contient l'ensemble des règles et des situations rencontrées.

2.1 Fonctions de base

— **nombre_situation(jeu)** : Convertit l'état du jeu en un nombre entier unique pour identifier rapidement les situations déjà vues. Cela permet d'éviter les boucles infinies.
— **pic_vide(), regle_jeu(), effectue_deplacement()** : Ces fonctions élémentaires modélisent les actions possibles sur le plateau, l'application des règles et les déplacements du pic.
— **situation_non_vue()** : Vérifie si un état de jeu a déjà été visité. C'est crucial pour l'efficacité de l'algorithme.

— **activation_des_règles()** : Gère le déclenchement des règles en fonction de la situation actuelle.
—

3 Moteur d'Inférence (Chaînage Avant)

Notre moteur d'inférence est basé sur un **chaînage avant**. À partir de la situation de départ (base de fait), il applique de manière itérative les règles pour atteindre l'état final.

3.1 Processus

1. Le moteur d'inférence examine l'état actuel du jeu. 2. Il parcourt les règles disponibles et choisit celle à appliquer. 3. Il exécute la règle et met à jour la base de faits. 4. Le nombre de coups est incrémenté. 5. Les informations sur le coup joué et la nouvelle situation sont affichées à chaque itération. Ce processus se répète jusqu'à ce que la situation cible soit atteinte.
—

4 Analyse et Évaluation

L'évaluation de notre algorithme a montré sa capacité à résoudre le problème du pic-vide. L'analyse de l'évolution de la situation de jeu a révélé que notre approche évite les chemins redondants grâce à la fonction situation_non_vue().

4.1 Nombre de Coups

Le nombre de coups nécessaires varie en fonction de la configuration initiale. Pour une configuration standard, l'algorithme a résolu le problème en **X** coups. Nous avons observé que l'algorithme trouve une solution, mais pas nécessairement la plus courte.

4.2 Discussions et Améliorations

* **Optimalité** : L'algorithme ne garantit pas la solution optimale (le moins de coups possible), car il s'arrête dès qu'une solution est trouvée. Une approche par chaînage arrière ou une recherche plus exhaustive pourrait être explorée pour l'optimisation. * **Bonus** : L'ajout de nouvelles règles ou

la modification du plateau de jeu pourrait être intégré. Cela nécessiterait d'adapter la fonction regle_jeu et la structure de situation_etudiee.

—

5 Conclusion

Ce mini-projet a permis de mettre en pratique les concepts fondamentaux des systèmes experts. La création d'une base de connaissances et d'un moteur d'inférence nous a permis de résoudre le problème posé de manière efficace, tout en validant les choix techniques.