Rapport du Projet PontuXL

Introduction

Le projet PontuXL est une implémentation numérique du jeu de société Pontu, développé dans le cadre du cours INFO B317 - IA et programmation symbolique. Cette version du jeu a été adaptée pour quatre joueurs, avec une particularité majeure : deux des joueurs sont contrôlés par une intelligence artificielle avancée implémentée en Prolog.

Objectifs du Projet

1. Implémenter une version fonctionnelle du jeu Pontu pour quatre joueurs
2. Développer une intelligence artificielle en Prolog utilisant l'algorithme Max^n avec élagage superficiel
3. Créer une interface graphique intuitive et réactive
4. Intégrer un assistant conversationnel (bot) pour aider les joueurs à comprendre les règles
5. Assurer une communication fluide entre JavaScript (frontend) et Prolog (IA)

Structure du Projet

Architecture du projet

Le projet est organisé selon l'architecture suivante :

```
Pontu-Game/
├── index.html            # Page principale du jeu
├── game.js               # Logique principale du jeu en JavaScript
├── styles.css            # Styles CSS pour l'interface
├── pontu_ai.pl           # Intelligence artificielle en Prolog
├── prolog_interface.js   # Interface entre JavaScript et Prolog
├── pbot-elm.pl           # Bot conversationnel en Prolog
├── bot.js                # Interface JavaScript pour le bot
├── README.md             # Documentation du projet
└── favicon.ico           # Icône du site
```

Technologies Utilisées

- **HTML5/CSS3** : Structure et mise en forme de l'interface utilisateur
- **JavaScript** : Logique du jeu et interactions utilisateur
- **Prolog** : Implémentation de l'intelligence artificielle et du bot conversationnel
- **Tau Prolog** : Bibliothèque permettant d'exécuter du code Prolog dans le navigateur

Composants Principaux

1. Interface Utilisateur (`index.html`, `styles.css`)

Interface Utilisateur

Interface Utilisateur

L'interface utilisateur est conçue pour être intuitive et visuellement attrayante. Elle comprend : - Un plateau de jeu 6x6 avec des cases et des ponts - Des lutins de quatre couleurs différentes (vert, bleu, jaune, rouge) - Un panneau d'information indiquant le joueur actuel et l'état du jeu - Un système de chat pour interagir avec le bot explicateur - Des boutons de contrôle (nouvelle partie, aide)

2. Logique du Jeu (`game.js`)

Le fichier `game.js` contient toute la logique du jeu : - Initialisation du plateau et placement des lutins - Gestion des tours de jeu - Validation des mouvements - Détection des conditions de victoire/défaite - Animation des actions (déplacement des lutins, suppression des ponts)

Des effets visuels spécifiques ont été implémentés pour améliorer l'expérience utilisateur, notamment : - Animation de pulsation en rouge pour les ponts avant leur suppression - Message textuel "Pont supprimé" apparaissant au-dessus du pont retiré - Bordure rouge autour du pont pour le rendre plus visible - Mise à jour garantie de l'interface après chaque action

3. Intelligence Artificielle (`pontu_ai.pl`)

Intelligence Artificielle

L'IA du jeu est implémentée en Prolog et utilise l'algorithme Max^n avec élagage superficiel, spécialement adapté aux jeux à plus de deux joueurs.

Algorithme Max^n

Contrairement à l'algorithme Minimax traditionnel qui est conçu pour les jeux à deux joueurs (un joueur maximise son score, l'autre le minimise), l'algorithme Max^n est adapté aux jeux multi-joueurs. Chaque joueur tente de maximiser son propre score, sans nécessairement minimiser celui des autres.

L'algorithme fonctionne comme suit : 1. À chaque état du jeu, on évalue un n-uplet de scores (un score pour chaque joueur) 2. Chaque joueur choisit le coup qui maximise son propre score 3. L'élagage superficiel (Shallow Pruning) est utilisé pour optimiser la recherche

Les principales fonctions Prolog implémentées sont : - `trouver_meilleur_coup/2` : Point d'entrée pour trouver le meilleur coup - `maxn_shallow/4` : Implémentation de l'algorithme Max^n avec élagage superficiel - `evaluer_etat_tous_joueurs/2` : Évalue l'état du jeu pour tous les joueurs - `obtenir_coup_ia_maxn/5` : Interface pour obtenir le coup de l'IA - `obtenir_coup_ia/5` : Version standard utilisée comme fallback

4. Interface JavaScript-Prolog (`prolog_interface.js`)

Ce fichier assure la communication entre le code JavaScript du jeu et l'IA implémentée en Prolog. Il utilise la bibliothèque Tau Prolog pour exécuter le code Prolog directement dans le navigateur.

Le processus d'intégration fonctionne comme suit : 1. Le code Prolog est chargé dans une session Tau Prolog 2. L'état du jeu JavaScript est converti en format Prolog 3. Une requête Prolog est exécutée pour trouver le meilleur coup 4. Le résultat est converti en format JavaScript et utilisé par le jeu

5. Bot Conversationnel (`pbot-elm.pl`, `bot.js`)

Bot Conversationnel

Bot Conversationnel

Le bot conversationnel, nommé PBot, est implémenté en Prolog et permet aux joueurs de poser des questions sur les règles du jeu, les stratégies, etc. Il utilise un système de reconnaissance de mots-clés et de patterns pour comprendre les questions des utilisateurs.

Le fichier `bot.js` sert d'interface entre le chat HTML et le code Prolog du bot, permettant une interaction fluide avec l'utilisateur.

Processus de Développement

Le développement du projet s'est déroulé en plusieurs phases :

1. **Phase de conception** : Définition des règles du jeu, de l'architecture du projet et des technologies à utiliser
2. **Implémentation de base** : Création de l'interface utilisateur et de la logique de jeu en JavaScript
3. **Développement de l'IA** : Implémentation de l'algorithme Max^n en Prolog
4. **Intégration** : Création de l'interface JavaScript-Prolog pour permettre la communication entre les deux langages
5. **Développement du bot** : Création du bot conversationnel en Prolog
6. **Tests et optimisations** : Correction des bugs, amélioration des performances et de l'expérience utilisateur
7. **Finalisation** : Documentation et préparation du livrable final

Défis Rencontrés et Solutions

1. Intégration JavaScript-Prolog

Défi : Faire communiquer efficacement le code JavaScript du jeu avec l'IA implémentée en Prolog.

Solution : Utilisation de la bibliothèque Tau Prolog qui permet d'exécuter du code Prolog directement dans le navigateur. Création d'une interface dédiée (`prolog_interface.js`) pour convertir les données entre les deux formats.

2. Algorithme Max^n pour Jeux Multi-joueurs

Défi : Adapter l'algorithme Minimax traditionnel pour un jeu à quatre joueurs.

Solution : Implémentation de l'algorithme Max^n avec élagage superficiel, qui évalue un n-uplet de scores (un pour chaque joueur) et permet à chaque joueur de maximiser son propre score.

3. Feedback Visuel pour les Actions de l'IA

Défi : Les joueurs avaient du mal à suivre les actions de l'IA, notamment lors de la suppression des ponts.

Solution : Implémentation d'effets visuels spécifiques (animation de pulsation, message textuel, bordure rouge) pour rendre les actions de l'IA plus visibles et compréhensibles.

4. Performance de l'IA

Défi : L'algorithme Max^n peut être très coûteux en termes de calcul, surtout avec une profondeur de recherche élevée.

Solution : Implémentation de l'élagage superficiel (Shallow Pruning) pour réduire l'espace de recherche et optimisation des fonctions d'évaluation pour améliorer la prise de décision.

Fonctionnalités Principales

1. **Jeu à Quatre Joueurs** : Deux joueurs humains (vert et jaune) et deux IA (bleu et rouge)
2. **IA Avancée** : Utilisation de l'algorithme Max^n avec élagage superficiel
3. **Interface Intuitive** : Plateau de jeu interactif avec animations et feedback visuel
4. **Bot Conversationnel** : Assistant pour aider les joueurs à comprendre les règles et stratégies
5. **Effets Visuels** : Animations pour rendre les actions plus compréhensibles

Comment Exécuter le Projet

1. Cloner le dépôt ou télécharger les fichiers
2. Ouvrir un terminal dans le dossier du projet
3. Lancer un serveur web local :

   ```
   python3 -m http.server 9000
   ```

4. Ouvrir un navigateur et accéder à `http://localhost:9000`

Effets Visuels et Feedback Utilisateur

Effets Visuels

Effets Visuels

Un aspect particulier du projet a été l'attention portée aux effets visuels pour améliorer l'expérience utilisateur, notamment lors des actions de l'IA. Ces effets comprennent :

1. **Animation de pulsation en rouge** : Les ponts qui vont être retirés pulsent en rouge avant de disparaître, attirant l'attention du joueur
2. **Message textuel "Pont supprimé"** : Un message apparaît au-dessus du pont pour indiquer clairement l'action en cours
3. **Bordure rouge** : Une bordure rouge est ajoutée autour du pont pour le rendre plus visible

4. **Mise à jour garantie de l’interface** : Après chaque action, l’interface est mise à jour pour refléter l’état actuel du jeu

Ces améliorations visuelles ont résolu un problème important où l’IA indiquait qu’elle supprimait un pont à un endroit mais en supprimait un autre. Grâce à ces effets, les actions de l’IA sont maintenant parfaitement claires pour les joueurs humains.

Conclusion

Conclusion

Le projet PontuXL est une implémentation réussie du jeu de société Pontu, enrichie par une intelligence artificielle avancée en Prolog. L’utilisation de l’algorithme Max^n avec élagage superficiel permet une prise de décision stratégique adaptée aux jeux multi-joueurs.

L’intégration entre JavaScript et Prolog, bien que complexe, a été réalisée avec succès grâce à la bibliothèque Tau Prolog, permettant d’exécuter le code Prolog directement dans le navigateur.

Les effets visuels et le feedback utilisateur ont été particulièrement soignés pour améliorer l’expérience de jeu, notamment en rendant les actions de l’IA plus compréhensibles.

Le bot conversationnel ajoute une dimension pédagogique au projet, permettant aux joueurs de s’informer sur les règles et stratégies du jeu.

Ce projet démontre la puissance de Prolog pour l’implémentation d’algorithmes d’IA complexes et son intégration réussie avec des technologies web modernes.

Perspectives d’Amélioration

Perspectives

1. **Optimisation de l’IA** : Augmenter la profondeur de recherche et améliorer les fonctions d’évaluation

2. **Mode Multijoueur en Ligne** : Permettre à plusieurs joueurs de jouer ensemble via Internet
3. **IA Adaptative** : Développer une IA qui apprend des parties précédentes
4. **Personnalisation** : Ajouter des options de personnalisation (thèmes, niveaux de difficulté)

Équipe de Développement

Team

Ce projet a été développé dans le cadre du cours INFO B317 - IA et programmation symbolique.

PontuXL