UNIVERSITAT POLITÈCNICA DE CATALUNYA

ESCOLA POLITÈCNICA SUPERIOR

D'ENGINYERIA DE VILANOVA I LA GELTRÚ

PROJECTE DE PROGRAMACIÓ

Lab 5: Parallel Data Decomposition Implementation and Analysis

Profesor/a: Bernat Orellana Bech

Estudiantes:
Nima Ghaffarzadeh
Jordi Calatayud Álvarez

25/12/2024

Introducció del problema

L'objectiu de la pràctica consisteix en desarrollar dos codis que implementen un minimax limitat per profunditat i un altre limitat per temps amb IDF per al joc de taula Hex amb un tauler de 11 x 11

Heurística

Tenim múltiples heuristiques:

1 - dijkstraDistanceToWin:

Aquesta heurística es la principal. I consisteix en aplicar l'algoritme amb el mateix nom, Djikstra, al tauler del Hex. Aquest algoritme calcula la distància mínima entre dos punts d'un graf, per això transformem el tauler en una matriu a partir de ahí comprovem les connexions indicant que no pot anar per una fitxa que no siga del seu color i que li costarà uno arribar a un node adjacent. Per tant retornarem la mínima distància per a arribar d'una banda a l'altra del jugador que crida la funció. Aquesta heurística te el pes més alt entre totes les heuristiques(15).

2 - conectividadHeuristic:

Aquesta heurística pren els veïns del mateix color d'una fitxa i augmenta el valor de l'heurística si aquesta fitxa té més veïns del mateix color. L'usem de manera inversa per evitar que les fitxes quedin enganxades a la jugada i per poder marcar el camí general abans de col·locar les fitxes al mig. Aquesta heurística té un pes de 4.

3 - calculateCentralControl:

Calcula la distància de cada fitxa nostra al centre i suma totes les distàncies. Sense aquesta heurística, les fitxes solen quedar enganxades a les cantonades del tauler, perquè són les posicions avaluades primer. Volem les fitxes al centre per poder canviar el camí si l'oponent tanca el nostre camí.

4 - opponentWin:

Aquesta heurística la fem servir només quan l'enemic està molt a prop de guanyar (a una distància de menys o igual a 3 fitxes), i li assignem un valor heurístic molt baix per avisar del perill a l'algorisme.

5 - myWin:

Aquesta heurística és l'oposada a *opponentWin*; dona un valor heurístic molt alt quan estem molt a prop de guanyar (a una distància de menys o igual a 3 fitxes).

Comparar els temps d'execució

Comparar els temps d'execució de *minimax* i *minimax* amb *iterative deepening* per a un nombre fixat de nivells.

Minimax:

- Nivell2: Instantaneo.
- Nivell3:16 segons
- Nivell4: Molta temps

Minimax amb iterative deepening:

- Nivell2: Instantaneo
- Nivell3: Mes que 1 minut
- Nivell4: Molta temps

Nivells baixats i Nodos explorats

Gràfica del nombre de nivells explorats segons el temps disponible:

5 segons:

```
Profunditat màxima:3
Node explorats:    20576
```

10 segons:

```
Profunditat màxima:3
Node explorats:    28578
```

15 segons:

```
Profunditat màxima:3
Node explorats:    35337
```

20 segons:

```
Profunditat màxima:3
Node explorats:    42078
```

Com veiem, amb més temps arribem al nivell 3, però explorem més nodes.

Ordre d'exploració

Explicació de les estratègies d'optimització utilitzades:

- **Poda alfa-beta:**

Sense poda alfa-beta:

```
Profunditat màxima:3
Node explorats:    180678
```

Amb poda alfa-beta:

```
Profunditat màxima:3
Node explorats:    28744
```

Com veiem, tenim un 84% menys de nodes per explorar gràcies a la poda alfa-beta.

- **Ordenació de nodes:**

A les funcions *min* i *max*, ordenem els nodes abans d'arribar a l'últim nivell. Ho fem passant el node a la heurística que tenim i, després, ordenant els resultats. Això ajuda la poda alfa-beta a eliminar més nodes.

Sense ordenació de nodes:

```
Profunditat màxima:3
Node explorats:    30996
```

Amb ordenació de nodes:

```
Profunditat màxima:3
Node explorats:    28331
```

Com veiem, l'ordenació de nodes ens ajuda a explorar 3000 nodes menys.

- **Guardar el millor moviment utilitzant zobrist hashing:**

Utilitzem el *Zobrist hashing* per guardar el millor moviment de la iteració anterior. El *Zobrist* utilitza un codi hash per emmagatzemar l'estat del tauler, i quan troba aquest estat altra vegada en la iteració següent, ja ha guardat el millor moviment com a valor a la taula hash. Llavors, executa aquest moviment primer i, després, explora la resta de moviments. Aquest mètode no redueix el nombre de nodes a explorar, però millora els resultats de *minimax*.

Informació Addicional:

Repositori github: https://github.com/nghaffar21/Hex-PROP.git

Participació de Membres de l'equip: Totes les parts del projecte les hem fet en trucades de Discord. Per tant, el percentatge de participació és gairebé un 50% per a cada membre.