Progetto ASD2022

Pietro Di Lena

Dipartimento di Informatica – Scienza e Ingegneria
Università di Bologna

Algoritmi e Strutture di Dati
Anno Accademico 2021/2022

(M, N, K)-game

- Il (M, N, K)-game è una versione generalizzata del classico tris
  - Il tris è un $(3, 3, 3)$-game
  - La partita viene giocata su una matrice di dimensione $M \times N$
  - Ogni giocatore ha un proprio simbolo
  - Ad ogni turno un giocatore posiziona il proprio simbolo su una casella libera della matrice di gioco
  - Per poter vincere bisogna allineare (verticalmente, diagonalmente oppure orizzontalmente) K simboli consecutivi

- Lo scopo del progetto è quello di sviluppare un giocatore software in grado di giocare nel modo migliore possibile su tutte le istanze (ragionevolmente) possibili di un (M, N, K)-game

- Vincoli
  - Implementazione in Java di un'interfaccia Java pre-definita
  - Vietato usare Thread e, più in generale, parallelizzazione

Proprietà note dei (M, N, K)-game

- Tramite una dimostrazione per furto di strategia si può dimostrare che il secondo giocatore non può avere una strategia che gli assicuri la vittoria se il primo giocatore ha una strategia ottima
  - Il secondo giocatore non può mai vincere se il primo ha una strategia ottima, al massimo può pareggiare (draw)

- E' un gioco *risolto* per alcune configurazioni. Ad esempio:
  - $(3, 3, 3)$ e $(4, 4, 4) \Rightarrow$ draw
  - $(4, 5, 4)$ e $(5, 4, 4) \Rightarrow$ draw
  - $(4, 3, 3)$, $(3, 4, 3)$ e $(6, 5, 4) \Rightarrow$ win (primo giocatore)
  - $(m, 4, 4)$ con $m \leq 8 \Rightarrow$ draw
  - $(m, 4, 4)$ con $m \geq 30 \Rightarrow$ win (primo giocatore)
  - $(m, 4, 4)$ con $8 < m < 30 \Rightarrow$??

Informazioni generali

- Può essere svolto in gruppo (massimo tre persone)
- Si consegna una sola volta
  - Se necessario, saranno richieste correzioni specifiche
- Il voto del progetto resta valido per un tempo *illimitato*
  - A meno che non cambino i docenti del corso
- E' possibile consegnarlo entro Febbraio 2023
  - Oltre Febbraio 2023, attendere le nuove specifiche
- Il voto del progetto pesa per 1/3 sul voto finale
  - Media pesata tra voto dello scritto + voto del progetto

In cosa consiste la prova di progetto

- Sviluppo
  - Il linguaggio di programmazione è Java
  - Vi viene fornita l'interfaccia Java da implementare
  - Vi viene fornito tutto il software necessario per effettuare test
  - Non è ammesso l'uso di Thread e parallelizzazione
  - Suggerimenti più avanti
- Relazione
  - Breve relazione sulle scelte progettuali adottate
  - Suggerimenti più avanti
- Discussione orale
  - Discussione orale sul lavoro svolto
  - Non è un orale sugli argomenti del corso

Modalità di consegna e discussione

- Codici sorgenti e relazione: tramite virtuale
  - Slot di consegna a partire da fine Maggio
  - Possibile consegnare solo in periodi d'esame
  - E' sufficiente che consegni un solo componente del gruppo
  - I componenti del gruppo devono essere indicati alla consegna
- Discussione del progetto: appelli su AlmaEsami+TEAMS
  - Necessario iscriversi su AlmaEsami
  - Tutti i componenti del gruppo devono iscriversi su AlmaEsami
  - Discussione con tutti i componenti del gruppo insieme
  - La discussione deve essere fatta a ridosso della consegna
  - Per ogni slot di consegna sarà indicato anche il giorno della discussione del progetto

Valutazione

- Valutazione delle prestazioni
  - Confronto con 4 algoritmi di riferimento di livello crescente
    - Prestazioni peggiori del Livello 0 $\Rightarrow$ progetto da sistemare
    - Prestazioni tra Livello 0 e 1 $\Rightarrow$ voto basso
    - Prestazioni tra Livello 1 e 2 $\Rightarrow$ voto medio-basso
    - Prestazioni tra Livello 2 e 3 $\Rightarrow$ voto medio-alto
    - Prestazioni migliori del Livello 3 $\Rightarrow$ possibile lode
- Valutazione della relazione
  - Se ben scritta e dettagliata $\Rightarrow$ punti aggiuntivi
  - Se scarna $\Rightarrow$ nessun punto aggiuntivo
  - Se scritta male $\Rightarrow$ punti in meno
- Discussione orale
  - Serve per capire dettagli non chiari da relazione e codice sorgente
  - Serve per capire se tutti hanno contribuito al progetto
  - Partecipazione nulla o scarsa $\Rightarrow$ punti in meno (individualmente)

Valutazione delle prestazioni

- Lista delle 25 configurazioni di gioco testate

M	N	K	Risultato (assumendo due giocatori con strategia ottima)
3	3	3	Patta
4	3	3	Vittoria (Primo giocatore)
4	4	3	Vittoria (Primo giocatore)
4	4	4	Patta
5	4	4	Patta
5	5	4	Patta
5	5	5	Patta
6	4	4	Patta
6	5	4	Vittoria (Primo giocatore)
6	6	4	Vittoria (Primo giocatore)
6	6	5	Patta
6	6	6	Patta
7	4	4	Patta
7	5	4	Vittoria (Primo giocatore)
7	6	4	Vittoria (Primo giocatore)
7	7	4	Vittoria (Primo giocatore)
7	5	5	Patta
7	6	5	Patta
7	7	5	Patta
7	7	6	Patta
7	7	7	?
8	8	4	Vittoria (Primo giocatore)
10	10	5	?
50	50	10	?
70	70	10	?

- Ogni implementazione viene fatta giocare sia come primo giocatore che come secondo (50 partite in totale per coppia di giocatori)

Punteggi assegnati durante una partita

- Si utilizza il seguente schema di punteggi per ogni partita giocata:
  - Vittoria come primo giocatore o a tavolino: 2 punti
  - Vittoria come secondo giocatore: 3 punti
  - Patta: 1 punto
  - Sconfitta: 0 punti
- Il punteggio scelto per la vittoria come secondo giocatore bilancia lo svantaggio di gioco del secondo giocatore
  - Pur avendo una strategia ottima, il secondo giocatore non può mai vincere se il primo giocatore ha una strategia ottima
- La vittoria a tavolino avviene in presenza di errori dell'avversario
  - Ad esempio: timeout, mossa non legale, ecc

Interfaccia MNKPlayer

```
public interface MNKPlayer {
  // Inizializza il giocatore software
  // M = numero di righe nella matrice
  // N = numero di colonne nella matrice
  // K = numero di simboli da allineare
  // first = true se è il primo a giocare
  // timeout_in_secs = numero massimo di secondi per una mossa
  public void initPlayer(int M, int N, int K, boolean first,
                                              int timeout_in_secs);
  // Seleziona e ritorna una mossa tra quelle in FC
  // FC = array di celle libere (giocabili) nella matrice
  // MC = array di celle già occupate (history del gioco)
  public MNKCell selectCell(MNKCell[] FC, MNKCell[] MC);
  // Ritorna il nome del giocatore
  public String playerName();
}
```

O(1)

unica interessante

O(1)

Attenzione: il costruttore non deve prendere argomenti

Package mnkgame

- Pacchetto Java contente l'interfaccia grafica per giocare e testare la propria implementazione
- Nel pacchetto sono anche disponibili due implementazioni (semplici) di MNKPlayer a scopo illustrativo e per effettuare test
- Da riga di comando il gioco può essere avviato nel seguente modo

java MNKGame <M> <N> <K> [MNKPlayer class] [MNKPlayer class]

- Gli ultimi due argomenti sono opzionali:
  - Se usati entrambi parte in modalità Computer vs Computer
  - Se uno dei due è omesso parte in modalità Human vs Computer
  - Se sono omessi entrambi parte in modalità Human vs Human
- Disponibile anche tool per effettuare test senza interfaccia grafica

Suggerimenti per lo sviluppo

- I metodo selectCell(...) deve effettuare una scelta *intelligente* della mossa tra tutte quelle possibili in FC[]
- Il numero di possibili partite giocabili cresce esponenzialmente al crescere della dimensione della matrice di gioco ed il numero di simboli da allineare (impossibile valutarle tutte)
- L'implementazione deve cercare di trovare sempre una soluzione (mossa) accettabile in poco tempo (fissato a 10 secondi)
- L'implementazione non deve essere tarata su 10 secondi ma deve cercare di sfruttare al meglio il tempo specificato tramite il parametro *timeout_in_secs* passato al metodo selectCell(...)
- Per quanto non ci siano vincoli sulle strategie da adottare per la scelta della mossa, il suggerimento è di partire dalle tecniche algoritmiche già studiate per questo tipo di problema (che vedremo)

Suggerimenti per la relazione

- Intestazione
  - Indicare il nome del progetto, nomi, cognomi e numero di matricola dei componenti del gruppo
- Descrizione del problema
  - Descrivere il problema computazionale affrontato
  - Introdurre il problema ed indicare i punti salienti che hanno portato alle scelte progettuali adottate
- Scelte progettuali
  - Descrivere ad alto livello le scelte implementative adottate per il metodo selectCell
  - Citare esplicitamente le strutture dati e gli algoritmi noti utilizzati (se applicabile) e sottolineare i contributi originali adottati per l'implementazione del progetto
  - Fornire un'analisi della costo computazionale (anche se molto approssimativa)

Oltre al voto

- I progetti consegnati e funzionanti saranno utilizzati per un torneo tutti-contro-tutti per determinare una classifica delle implementazioni più forti di giocatori (M, N, K)
- La classifica verrà aggiornata di volta in volta
- E' solo per nostro divertimento interno, non avrà influenza sul voto anche se la classifica tenderà ad essere consistente con la valutazione
- Il nome che comparirà nella classifica è quello assegnato al giocatore tramite il metodo playerName()