Progetto SO 2019/20

Bini, Radicioni, Schifanella

20 novembre 2019

1 Descrizione

Si intende realizzare un gioco in cui dei giocatori competono tra di loro per la conquista di alcune bandierine poste all'interno di un campo di gioco modellato come una scacchiera. Per conquistare le bandierine i giocatori possono spostare le loro pedine.

Sono presenti 3 tipi di processo:

1. un processo `master` che gestisce il gioco,
2. `SO_NUM_G` processi giocatore,
3. `SO_NUM_P` processi pedina per ogni giocatore.

1.1 Round

Tutta la durata del gioco è suddivisa in *round*. Ogni round:

1. il processo master posiziona un numero di bandierine a sua scelta, ognuna su una cella libera da pedine. Il processo master decide il punteggio assegnato ad ogni bandierina
2. il processo master segnala ai giocatori l'inizio del round, rendendo loro noti posizione e punteggio delle bandierine piazzate
3. i giocatori forniscono indicazioni alle pedine in modo si possano muovere al fine di occupare le celle con le bandierine e ottenere il punteggio relativo
4. ogni volta che una bandierina è conquistata il master viene informato
5. quando tutte le bandierine piazzate nel round sono state conquistate, il processo master:
   (a) ferma il gioco
   (b) stampa a terminale
      - una rappresentazione visuale della scacchiera in ASCII
      - il punteggio attuale dei giocatori e le mosse residue a disposizione
   (c) piazza le nuove bandierine e avvia un nuovo round

Se un round dura più di `SO_MAX_TIME` secondi, il processo master ne è informato e termina il gioco, stampando (come descritto sopra per il termine di un round) posizione finale, punteggio finale e mosse residue totali per ogni giocatore.

1.2 Processo master

Il master genera una scacchiera rettangolare di dimensioni `SO_BASE` e `SO_ALTEZZA`. Poi genera `SO_NUM_G` processi giocatore e attende che i processi giocatore abbiano piazzato le loro pedine.

Dopo che i processi giocatore hanno piazzato tutte le loro pedine, distribuisce le bandierine sulla scacchiera per il prossimo round. Il numero di bandierine è un numero casuale compreso fra `SO_FLAG_MIN` e `SO_FLAG_MAX`. La somma totale dei punti assegnati alle bandierine è uguale a `SO_ROUND_SCORE`.

Quando il round termina, il master stampa a schermo (in ASCII) lo stato della scacchiera: una lettera maiuscola diversa per giocatore su ogni pedina, e il carattere "`.`" su ogni cella vuota.

Il gioco termina quando dall'inizio di un round sono trascorsi `SO_MAX_TIME` secondi senza che il round sia terminato.

1.3 Processo giocatore

All'inizio del gioco, i processi giocatore piazzano **a turno** una pedina alla volta in una cella disponibile, secondo la strategia che ritiene più opportuna. Una pedina è un processo. Dopo aver piazzato tutte le proprie pedine, tutti i processi giocatore rimangono in attesa dell'inizio del primo round.

All'inizio del round, il master piazza le bandierine. I giocatori danno quindi indicazioni alle proprie pedine su come spostarsi. Il tipo di indicazione che il giocatore fornisce alla pedina è a discrezione degli sviluppatori del progetto. I giocatori non devono conoscere le indicazioni date dagli altri. L'obiettivo di ogni giocatore è quello di raggiungere il punteggio più alto ottenuto avendo una propria pedina nella casella della bandierina in modo da guadagnare i punti corrispondenti.

1.4 Processo pedina

Ogni pedina occupa una cella della scacchiera. La posizione delle pedine è nota a tutte le altre pedine (ed ai processi giocatore) in ogni fase del gioco (compresa la fase iniziale di posizionamento delle pedine). Un processo pedina riceve dal proprio giocatore l'indicazione su come spostarsi. All'inizio del round, le pedine di ogni giocatore attendono che **tutti** i giocatori abbiano dato alle prorie pedine le prime indicazioni sugli spostamenti da fare. Invece, nelle fasi successive del gioco tale sincronizzazione non è presente.

La pedina è libera di seguire le indicazioni del giocatore oppure di fare altre scelte se questo è ritenuto vantaggioso. Le pedine si possono muovere nelle quattro direzioni possibili e, prima di spostarsi, provano ad accedere alla cella verso cui intendono dirigersi.

Ogni pedina ha un numero totale `SO_N_MOVES` di mosse a sua disposizione per tutto il gioco. Il numero di mosse residue è uno stato interno del processo pedina e non è visibile al giocatore. Quando il numero di mosse residue raggiunge zero, il processo pedina rimane fermo nella cella occupata.

Se una pedina ritiene di non poter raggiungere il proprio obiettivo (per esempio, perché la cella verso cui si vuole spostare è occupata, perché non ha mosse a sufficienza, etc.), ne può dare comunicazione al proprio giocatore. Il giocatore, se vuole, imposta nuove indicazioni di spostamento. Alternativamente, la pedina può decidere in autonomia come spostarsi se lo ritiene vantaggioso per la propria squarda.

Quando una bandierina viene conquistata, la pedina lo comunica al proprio giocatore, che lo comunicherà al processo master. Il processo master ha tra i suoi compiti anche quello di tenere il punteggio del gioco.

1.5 Scacchiera

La scacchiera è realizzata con una matrice in memoria condivisa. Ogni cella è protetta da un semaforo distinto. Il processo pedina che vuole entrare in una cella chiede l'accesso al semaforo. Attenzione: ci possono essere deadlock con grande facilità!! Per questo motivo, la pedina che non riesce ad accedere ad una cella può anche decidere di desistere (per esempio attraverso `semtimedop(...)` oppure invocando la `semop(...)` con flag il `IPC_NOWAIT`).

1.6 Fine del gioco

Il gioco termina quando un round dura più di `SO_MAX_TIME` secondi. Quando ciò succede, il master stampa:

parametro	“easy”	“hard”
SO_NUM_G	2	4
SO_NUM_P	10	400
SO_MAX_TIME	3	1
SO_BASE	60	120
SO_ALTEZZA	20	40
SO_FLAG_MIN	5	5
SO_FLAG_MAX	5	40
SO_ROUND_SCORE	10	200
SO_N_MOVES	20	200

Tabella 1: Esempio di valori di configurazione.

- stato delle celle
- tempo di gioco trascorso dall’inizio
- punteggio conseguito dai giocatori
- numero totale di mosse residue delle pedine di un ciascun giocatore

2 Configurazione

Il gioco legge i parametri di configurazione dalle seguenti variabili di ambiente, che devono essere state preventivamente impostate. I parametri di configurazione sono i seguenti:

- `SO_NUM_G`: numero di processi giocatore,
- `SO_NUM_P`: processi pedina per ogni giocatore,
- `SO_MAX_TIME`: durata massima di un round
- `SO_BASE`, `SO_ALTEZZA`: dimensioni del campo di gioco
- `SO_FLAG_MIN`, `SO_FLAG_MAX`: numero minimo e massimo di bandierine posizionate per round
- `SO_ROUND_SCORE`: punteggio totale assegnato per round alle varie bandierine
- `SO_N_MOVES`: numero totale di mosse a disposizione delle pedine

La Tabella **??** elenca valori “easy” e “hard” per le configurazione da testare.

3 Requisiti implementativi

Il progetto deve essere realizzato sfruttando le tecniche di divisione in moduli del codice, compreso l’utilizzo dell’utility `make`.