UNIVERSITÀ DEGLI STUDI DI MILANO
Corso di Laurea in Scienze dell'Informazione

Corso di **SISTEMI** per l'Elaborazione dell'Informazione **I**
(Prof. M. Torelli)

Progetto d'esame: **OTHELLO**

1. Introduzione

E' ampiamente noto che l'11 maggio 1997 il campione del mondo di scacchi, Garry Kasparov, ha perso la sfida con l'elaboratore *Deep Blue*, con il punteggio di 2½ a 3½. E' assai meno noto che il 7 agosto 1997 un altro programma, *Logistello*, ha concluso col risultato a favore di 6 a 0 una sfida con il campione del mondo di *Othello*, Takeshi Murakami. Ormai, anche programmi disponibili in commercio ed eseguibili su comuni PC sono in grado di giocare a scacchi al livello dei migliori giocatori italiani.

Lo scopo del presente progetto è quello di avvicinare gli studenti a problematiche connesse con la messa a punto di giocatori artificiali, senza addentrarsi nelle tecniche assai raffinate e specifiche sviluppate nel corso delle ricerche sui programmi per il gioco degli scacchi, ricerche che durano ormai da oltre mezzo secolo. Si propone pertanto di indagare un gioco più semplice, come *Othello*: tale gioco è assai noto in quanto un programma veniva distribuito con alcune versioni di Windows. Lo scopo non è tuttavia quello di produrre un programma analogo a quello distribuito (molto rigido), ma, al contrario, di produrre una struttura flessibile e ricca di informazioni, allo scopo di confrontare diverse versioni delle strategie e di fornire strumenti di valutazione utili ad effettuare i confronti stessi.

2. Othello o Reversi?

Secondo Martin Gardner ("Enigmi e giochi matematici - volume 3°", Sansoni, Firenze, 1985[3]) il *Reversi* è nato in Inghilterra verso il 1870. Il gioco si svolge collocando delle pedine su una scacchiera quadrata con 8 caselle per lato; nel *Reversi* le prime quattro pedine erano collocate a turno dai due giocatori nelle quattro caselle centrali, mentre nell'*Othello* il gioco inizia con quattro pedine già collocate al centro. *Othello* è un marchio registrato nel 1974.

Nota per la copia in rete: purtroppo le figure non sono disponibili, ma se ne trovano di analoghe nei numerosi siti internet, tra cui quelli menzionati.

La numerazione canonica delle caselle va dalla *a1* in alto a sinistra alla *h8* in basso a destra. La posizione iniziale è fissa [si vedano, per esempio, http://www.fngo.it/introduzione.asp e http://www.mclink.it/personal/MC1053/othello/OthNow.pdf] e gioca per primo il *nero* (fare attenzione a queste convenzioni: un programma che non vi si attenga non sarà ammesso a un torneo ed è comunque svalutato).

Ognuno dei due giocatori, a turno, colloca una sua pedina in una casella vuota adiacente a una pedina avversaria lungo una traversa, colonna o diagonale: al termine della fila di pezzi avversari deve trovarsi almeno una pedina propria. Le pedine avversarie intrappolate tra le proprie devono essere "ribaltate" (onde il nome "reversi") così da apparire come pedine proprie, e non più dell'avversario (le pedine fisiche hanno due facce, una bianca e una nera, o colori analoghi). Se non si possono ribaltare pedine avversarie si deve "passare". Al termine delle mosse possibili, il punteggio è dato dal numero di pedine di ciascun colore. Per un paio di esempi si possono vedere le partite al sito http://www.fngo.it/partite.asp .

3. Strategie

Come accade in quasi tutti i giochi interessanti, nessuna strategia singola è in grado di garantire un buon livello di gioco. Il programma più semplice si limita a muovere a caso: è utile come termine di confronto per verificare quanto siano efficaci le diverse strategie. Quasi tutte le strategie sono *avide*, ossia cercano in ogni istante di rendere minimo o massimo qualche parametro: per esempio il numero di pedine. Si noti che una possibile strategia consiste nel *minimizzare* il numero di pedine prima del finale, dal momento che il *massimizzarle*, contrariamente all'intuizione, non porta a grandi risultati (il progetto può servire a verificare questa affermazione!).

Un parametro da massimizzare è la *mobilità*, ossia il numero di mosse a disposizione, mentre una strategia alternativa cerca di accaparrare le *caselle migliori*, individuate secondo *pesi* assegnati staticamente o dinamicamente.

Nel finale alcune pedine divengono *stabili* (non possono più cambiare colore): ovviamente, massimizzare le proprie pedine stabili è una buona strategia, applicabile però solo nel finale, quando può già essere fattibile *analizzare esaustivamente* tutte le possibilità, in modo da conseguire il miglior risultato possibile da quel punto in poi.

I buoni programmi combinano molte di queste strategie e svariate altre (Deep Blue, per gli scacchi, tiene conto di circa 10000 parametri), e la messa a punto e il peso relativo da dare a parametri in conflitto costituiscono aspetti delicati e importanti.

4. Requisiti

Si richiede di allestire un programma che consenta di confrontare almeno alcune delle strategie prima menzionate, più eventuali altre da voi individuate o indicate in letteratura (si confrontino le brevi indicazioni bibliografiche e non si dimentichi che in Internet è disponibile abbondante materiale sull'argomento. Uno dei possibili obiettivi del progetto consiste nell'imparare a reperire e *citare* correttamente informazioni su un dato argomento: se scoprite siti pertinenti e interessanti in rete, non dimenticate di citarli).

Il programma dovrà fornire il maggior numero di informazioni rilevanti (peraltro a richiesta dell'utente: non è il caso di subissarlo di informazioni inopportune) sulla valutazione delle posizioni, sulle possibili mosse, sul confronto delle strategie, eccetera. Si consideri, per esempio, che nella fase finale di ricerca esaustiva può essere interessante conoscere il risultato al quale porterebbe ciascuna mossa possibile, ed è interessante sapere a che punto il programma è in grado di effettuare l'analisi esaustiva in tempi ragionevoli (dipenderà dalle caratteristiche della posizione, oltre che dal programma).

Al momento della valutazione del progetto ci si attende che il programma sia stato usato effettivamente per sperimentare e riportare poi nella relazione le esperienze fatte, anche eventualmente tramite il confronto con altri programmi. Il programma in dotazione con Windows assume che l'"umano" abbia sempre il bianco (in realtà, rosso) e giochi per primo (a meno di un "passo" iniziale): solo per potersi confrontare con questo programma si può lasciare un'opzione per cui il bianco muova per primo (un esempio della flessibilità cui si alludeva in precedenza).

Il programma dovrà ovviamente poter giocare sia col bianco, sia col nero, sia con entrambi i colori (per esempio per confrontare strategie diverse), sia limitarsi ad eseguire mosse introdotte dall'esterno. Ricordarsi di inserire un'opzione di randomizzazione (i due programmi a mia disposizione, a ciascun livello giocano sempre la stessa partita!). Non dimenticare di consentire l'immissione delle mosse sia col mouse, sia usando la notazione, sia leggendo un file; in uscita, sarebbe bello poter stampare i diagrammi numerati, come quelli nelle figure precedenti (le pedine col colore non sono essenziali).

Si riportino i risultati ottenuti, eventuali partite significative, dati statistici e altro: dovrebbe essere evidente l'opportunità che il programma disponga del salvataggio e della stampa dei dati, un aspetto spesso sottovalutato.

Studenti che hanno seguito corsi sulle reti neurali o sull'intelligenza artificiale potrebbero approfondire particolari euristiche o utilizzare algoritmi evolutivi o sperimentare tecniche di apprendimento automatico.

Studenti particolarmente interessati o esperti in giochi diversi da Othello potrebbero proporre il gioco di loro interesse, sottoponendo una breve relazione preliminare all'approvazione del docente (evitando programmi troppo ambiziosi per giochi complessi come gli scacchi). Un ulteriore possibile campo di interesse del progetto riguarda la didattica del gioco: il programma potrebbe avere un taglio prevalentemente didattico, con caratteristiche ad hoc.

Ferme restando le caratteristiche fondamentali del progetto, studenti interessati alla grafica potrebbero curare particolarmente gli aspetti grafici (senza esagerare): a tutti è richiesto di sviluppare un'interfaccia di uso facile e gradevole, che può essere realizzata anche a caratteri. Si ricordino opzioni come la possibilità di ritirare una o più mosse (anche tutte) e di avere suggerimenti, magari con valutazioni e motivazioni. Per quanto riguarda gli scacchi, per esempio, vi sono programmi che forniscono automaticamente un commento in inglese della partita.

Un aspetto importante per la partecipazione a tornei è il controllo del tempo (di solito 30 minuti per giocatore), meno semplice di quel che appaia (anni fa, un programma di scacchi, in un evento a Crema, perse per il tempo!). Se non pensate alle competizioni, è accettabile un programma che semplicemente muova in tempi "ragionevoli".

L'uso effettivo e critico del programma sin dalle prime versioni dovrebbe suggerire ulteriori caratteristiche e funzionalità.

Riferimenti

M. Buro, "The Othello Match of the Year: Takeshi Murakami vs. Logistello", *ICCA Journal* **20** (3) 1997, 189-193**,** e anche http://www.neci.nj.nec.com/homepages/mic/ps/match-report.ps.gz

P. S. Rosenbloom, "A World-Championship-Level Othello Program", *Artificial Intelligence* **19** (1982) 279-320.

Tra i moltissimi siti citiamo ancora solo

http://perso.wanadoo.fr/brunodlb/

Avvertenze

Per la *presentazione* del progetto occorre preparare una breve relazione scritta, poter mostrare almeno a video il listato e naturalmente far funzionare il programma. La *relazione* non dev'essere una piatta descrizione del programma, o peggio l'elenco dei *nomi* di funzioni e variabili. L'accento va messo sulle scelte implementative che sono state fatte: perché così e non in quest'altro modo, quali difficoltà si sono incontrate e come si sono superate (o non superate: se avete lavorato un mese su un'idea che non ha funzionato, spiegate qual era l'idea e perché non ha funzionato, invece di buttarla via semplicemente: qualche altro potrà evitare lo stesso errore), quali algoritmi o strutture dati si sono usati *e perché*, quali le limitazioni e quali le prestazioni conseguite, che cosa si è scoperto: qualche *esempio significativo* o qualche schermata. E nel *manuale utente* non dimenticate di dire come si entra (ed esce) dal programma, specie se la cosa non è ovvia.