UNIVERSITÀ DEGLI STUDI DI SALERNO

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

ED ELETTRICA E MATEMATICA APPLICATA

Relazione progetto

Ms Pacman controller

Autori:

Longobardi Francesco

Montillo Andrea

Indice

Capitolo 1: Introduzione

In questo documento, si descrive la realizzazione di un agente intelligente basato su utilità il cui scopo è quello di giocare al famoso videogioco *Ms. Pacman* nei panni di Ms. Pacman, utilizzando un opportuno algoritmo di ricerca. L'obiettivo del controller è, infatti, quello di far superare alla protagonista il maggior numero di livelli possibile così da massimizzare il suo score.

Per la sua implementazione è stato utilizzato il codice Java fornito per la competizione *Ms. Pacman vs Ghost Team*. Tramite quest'ultimo è, infatti, possibile sviluppare controllori intelligenti sia di Ms. Pacman sia dei Ghost Teams, oltre che far scontrare l'agente che veste i panni della protagonista contro tre differenti squadre di fantasmini: gli *Aggressive Ghosts*, in cui i 4 membri del team si muovono con l'obiettivo di mangiare Ms. Pacman; i *Random Ghosts*, in cui i 4 membri del team si muovono in maniera completamente casuale; i *Legacy Ghosts*, in cui 3 fantasmini hanno un comportamento *aggressive* e 1 ha un comportamento *random*.

Trattandosi di un agente intelligente, è necessario definire la misura della prestazione, l'ambiente, gli attuatori e i sensori, ovvero la *PEAS* (*Performance Measure, Environment, Actuators, Sensors*). In particolare:

- la ***Performance Measure*** è stata definita come lo score raggiunto nel gioco;

- l'***Environment*** è, chiaramente, la scacchiera di gioco. Quest'ultima si presenta come un susseguirsi di 4 tipi di labirinto, i quali si presentano in ordine fino al Game Over, all'avanzare dei livelli. Per ognuno dei labirinti, sono note le posizioni degli attori del gioco, quelle delle pills e quelle delle power pills. Quindi, ciò premesso, l'ambiente si mostra essere:
    - ***completamente osservabile:*** i sensori permettono di conoscere lo stato completo dell'ambiente ad ogni istante di tempo;
    - ***multi-agente competitivo:*** l'agente *Ms. Pacman* si trova a gareggiare contro gli altri 4 agenti;
    - ***deterministico:*** il prossimo stato dell'ambiente è completamente determinato dallo stato corrente e dall'azione eseguita dagli agenti;
    - ***sequenziale:*** la decisione corrente potrebbe influenzare tutte le future decisioni;
    - ***statico:*** l'ambiente non può cambiare mentre un agente sta decidendo l'azione da intraprendere;
    - ***discreto:*** vi è un numero finito di azioni possibili.

- gli ***Actuators*** che permettono all'agente di effettuare le uniche mosse possibili, ovvero il movimento nelle 4 direzioni;

- i ***Sensors*** che permettono all'agente di ottenere lo stato del gioco, ovvero la sua posizione nel labirinto, la posizione dei fantasmini, la posizione delle pillole e la posizione delle power pills.

Nel corso della trattazione, a valle dell'elenco delle regole del gioco che potrebbe risultare utile a comprendere meglio le motivazioni che hanno portato a scegliere la particolare struttura di base dell'agente, verranno illustrati i vari esperimenti effettuati e i risultati ottenuti da questi ultimi.

Si procederà, quindi, in maniera sequenziale, in modo da catturare le peculiarità e i problemi di ognuna delle diverse soluzioni presentate al fine di poter poi effettuare un confronto in termini prestazionali e poter, quindi, discutere dei motivi che hanno portato ad eventuali miglioramenti.

1.1 Regole del gioco

Le regole del gioco Ms. Pacman sono le seguenti:

- ci sono 4 labirinti (A,B,C,D) che si susseguono in ordine. Quando il quarto labirinto è completato, si ricomincia dal primo labirinto così da ripetere la sequenza finché non si raggiunge il game-over. Ogni labirinto presenta un differente layout con pills e power pills posizionate in specifiche locazioni;
- il giocatore (nel caso in esame l'agente artificiale) che controlla Ms. Pacman comincia il gioco nel labirinto A con tre vite. Al raggiungimento dei 10000 punti, guadagna una vita extra;
- l'obiettivo di Ms. Pacman è ottenere il più alto score possibile mangiando tutte le pillole e le power pills presenti nel labirinto così da avanzare di livello. Ogni pillola mangiata fa guadagnare a Ms. Pacman 10 punti, mentre ogni power pills 50;
- il team di 4 fantasmini rappresenta l'ostacolo che Ms. Pacman incontra per raggiungere il suo obiettivo, in quanto ogni volta che un fantasmino la mangia perde una vita;
- una volta che Ms. Pacman mangia una power pill, i fantasmini diventano mangiabili per un certo lasso di tempo. Lo score che ottiene mangiando i fantasmini in successione in questo lasso temporale raddoppia ad ogni fantasma mangiato, andando da 200 punti dopo aver mangiato il primo fantasma fino ad arrivare a 1600 al quarto fantasma;
- i fantasmini restano per un certo tempo, che diminuisce al progredire dei livelli, nella tana situata nel box al centro del labirinto. Ciò accade in differenti situazioni, ovvero all'inizio del gioco, quando sono mangiati, all'inizio del livello successivo e quando Ms. Pacman perde una vita;
- i fantasmini possono invertire la propria mossa in un corridoio o in un angolo con una probabilità di 0.05%, a meno che non si trovino in presenza di un incrocio dove possono invertirla senza limitazioni. Ms. Pacman può, invece, muoversi liberamente nelle 4 direzioni. La velocità dei fantasmini e di Ms. Pacman è la stessa, tranne quando i fantasmini sono mangiabili (in questo caso la loro velocità è dimezzata);
- ci sono due possibilità di game-over:
    - Ms. Pacman perde tutte le vite disponibili, perdendo la partita;
    - Ms. Pacman supera tutti i 16 livelli, vincendo la partita;

1.2 Obiettivi del progetto

Al fine di raggiungere l'obiettivo principale del progetto, ovvero massimizzare lo score di Ms. Pacman, c'è la necessità di scegliere un adeguato algoritmo di ricerca per

permettere all'agente di intraprendere la mossa migliore, nonché definire opportune euristiche in modo da migliorare significativamente le prestazioni.
C'è, altresì, bisogno di tenere in considerazione il tempo che Ms. Pacman ha a disposizione per intraprendere la prossima mossa, fissato a 40 millisecondi.

Un obiettivo definito secondario in quanto intrinseco nel precedente, è ottenere prestazioni migliori rispetto al controller di base fornito per la competizione, chiamato *StarterPacman*. Quest'ultimo segue 3 strategie abbastanza basilari in ordine di priorità, ovvero allontanarsi dal fantasmino se non è mangiabile, avvicinarsi al fantasmino se è mangiabile, avvicinarsi alle pills e alle power pills.
Di seguito, se ne illustrano le prestazioni in 150 diverse partite, 50 per ogni diversa tipologia di Ghosts elencate precedentemente.

StarterPacman vs AggressiveGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
11050	2530	6015.2	1524.197	1	0	0.02

Tabella 1: risultati StarterPacman vs AggressiveGhosts

StarterPacman vs RandomGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
65560	13320	34301.8	13839.852	11	2	5.56

Tabella 2: risultati StarterPacman vs RandomGhosts

StarterPacman vs LegacyGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
9990	1600	5175.2	1919.888645	0	0	0

Tabella 3: risultati StarterPacman vs LegacyGhosts

Si nota come il comportamento di StarterPacman sia mediamente buono nelle partite contro i RandomGhosts, riuscendo a raggiungere il punteggio di 65560 e il livello 11, mentre fatica molto contro i team Aggressive e Legacy, come dimostrato dal massimo punteggio raggiunto nelle 50 sfide contro questi due team, che è addirittura minore del minimo raggiunto contro i Random. Anche i livelli superati in queste sfide sono davvero

molto pochi, non riuscendo mai ad andare oltre il livello 0 contro i Legacy e arrivando al livello 1 un numero irrisorio di volte contro gli Aggressive (come testimonia il livello medio raggiunto pari a 0,02).

Il motivo è da ricercare nel comportamento non molto intelligente dell'agente. Ad esempio, si è notato che tende a sprecare spesso le power pills, non riuscendo mai a sfruttare il vantaggio di poterle utilizzare come un modo per evitare di perdere la vita in situazioni pericolose, oppure come una possibilità di aumentare il proprio punteggio mangiando i fantasmini.
Tuttavia, l'atteggiamento casuale dei fantasmini del team Random consente, comunque, di avere spesso la strada spianata verso le pillole e le power pills e, quindi, Ms. Pacman riesce a limitare il numero di volte in cui resta implicata in situazioni in cui rischia di perdere la vita.

Capitolo 2: Progettazione

In questo capitolo si descrive la fase di progettazione dell'agente intelligente, da cui poi seguiranno le scelte implementative descritte nel capitolo 3.

L'agente che si è deciso di implementare, data la natura del problema, è un agente basato su utilità. Esso si costruisce un modello del mondo attraverso i sensori, tenendo traccia e cercando di capire come evolve il mondo e l'effetto che hanno le proprie azioni sul mondo stesso. Quindi, attraverso una funzione di utilità misura la sua preferenza dei vari stati del mondo e sceglie la prossima azione di conseguenza.

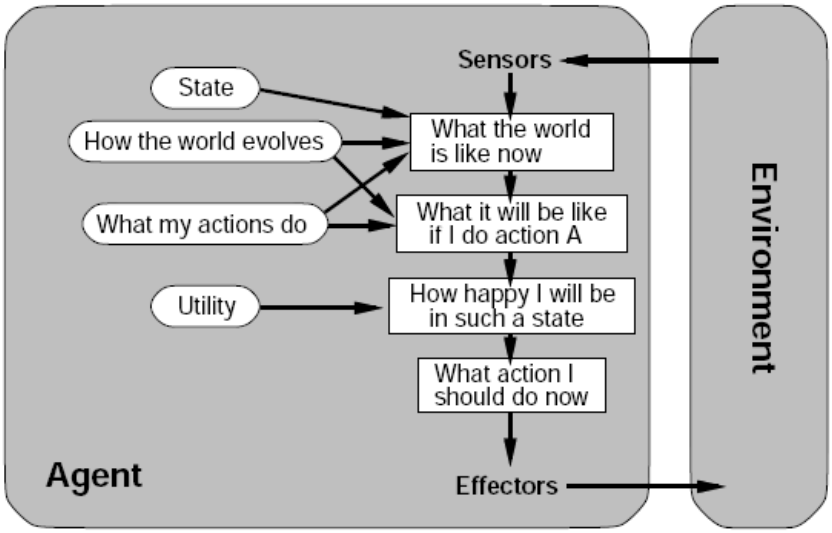


Figura 1: Schema di un agente basato su utilità

2.1 Definizione formale e precisa del problema

Il primo fondamentale step è definire in maniera formale e precisa le cinque componenti che caratterizzano il problema (lo ***stato iniziale*** da cui l'agente può partire per trovare una soluzione, l'insieme ***di azioni*** che l'agente può compiere in un dato stato, il ***modello di transizione*** che descrive lo stato *s2* in cui transita l'agente effettuando un'azione *a* trovandosi in uno stato *s*, il ***goal test*** che determina se un dato stato è uno stato goal, il ***path cost*** rappresentato da una funzione che assegna un costo numerico ad ogni percorso) in modo da poter implementare correttamente un agente in grado di arrivare ad una soluzione.

Nel caso di Ms. PacMan:

- lo *stato iniziale* è lo stato attuale del gioco che i sensori forniscono all'agente;
- l'*insieme di azioni* è rappresentato dai movimenti nelle quattro direzioni (left, right, up, down) che l'attuatore permette all'agente di effettuare;
- il *modello di transizione* è rappresentato, ad esempio, dall'allontanamento o avvicinamento ai fantasmini, alle pillole e alle power pills una volta effettuata una delle azioni possibili;
- il *goal test* potrebbe, ad esempio, essere la verifica che l'agente non si sia portato in uno stato di game-over;
- il *path cost* è rappresentato dallo score ottenuto seguendo quel determinato percorso.

Risulta, inoltre, necessario eseguire un processo di *astrazione*, che consiste nel rimuovere dettagli dalla rappresentazione che non siano rilevanti per raggiungere il goal test, come ad esempio la particolare morfologia dei percorsi che l'agente deve preferire, oltre al fatto che il movimento che viene assegnato potrebbe non essere espresso specificamente in una delle quattro direzioni, ma potrebbe essere fatto in modo che l'agente prediliga un movimento che lo avvicini alle pillole e lo allontani dai fantasmini.

2.2 Una possibile soluzione: Adversarial Search

Una volta formulato il problema, c'è necessità di trovare una soluzione. Quest'ultima può essere vista come una sequenza di azioni, ovvero come un percorso tra gli stati, così che un algoritmo di ricerca possa agire considerando diverse possibili sequenze di azioni che partano dallo stato iniziale, ovvero lo stato attuale della partita, ed arrivino allo stato goal, con l'obiettivo di trovare il percorso ottimale.
Le possibili sequenze di azioni possono essere rappresentate da un albero di ricerca la cui radice è lo stato iniziale, i cui rami sono le possibili azioni e i cui nodi rappresentano gli stati raggiungibili.

Esistono differenti tipi di algoritmi di ricerca che si differenziano nel modo in cui viene effettuata l'analisi dei percorsi possibili da un dato nodo, ovvero nella strategia di espansione dei nodi. Tuttavia, il problema ulteriore da considerare per l'agente in esame è che questi opera in un ambiente multi-agente competitivo. Nell'ambiente abbiamo, infatti, differenti agenti che risultano essere in conflitto dal punto di vista dell'obiettivo da raggiungere. In casi del genere risulta di particolare interesse l'utilizzo degli algoritmi di ricerca con avversari e, quindi, l'utilizzo della teoria dei giochi di cui si discuterà brevemente nel prossimo paragrafo.

2.3 Teoria dei giochi

Negli ambienti multi-agente, siano essi competitivi o cooperativi, ogni agente ha bisogno di considerare le azioni che gli altri agenti intraprendono e quanto queste hanno effetto sulle proprie prestazioni. Nell'intelligenza artificiale, ogni ambiente multi-agente è considerato un gioco. Di conseguenza, è necessario definire differenti componenti che rappresentino correttamente il sistema. In particolare si ha:

- lo ***stato iniziale***, che specifica lo stato del gioco nel momento in cui comincia la ricerca;
- i ***players***, ovvero gli agenti che devono fare la mossa nel determinato stato. Nel caso di Ms. Pacman, in ogni stato a fare la mossa sono Ms. PacMan e i fantasmini;
- le ***azioni***, ovvero le mosse legali in un determinato stato; ad esempio, Ms. Pacman può muoversi in tutte e quattro le direzioni, mentre i fantasmini no, come spiegato nel *paragrafo 1.1*;
- gli ***stati terminali***, ovvero gli stati di game over definiti nel *paragrafo 1.1*;
- una ***funzione di utility***, che definisce il valore numerico finale per un gioco che termina in uno stato terminale *s* per un player *p*. Nel capitolo 3 si illustreranno tutte le diverse funzioni di utility utilizzate.

Nell'IA, inoltre, si possono definire le seguenti categorie di gioco:

- ***a somma zero***: le funzioni di utilità degli agenti hanno segno opposto;
- ***ad informazione completa***: ambiente completamente osservabile e deterministico;
- ***a più giocatori***: ambiente multi-agente, con la necessità di introdurre strategie per vincere a causa della presenza dell'avversario;
- ***a turni alterni***: ogni player ha a disposizione un turno in cui effettuare la propria mossa;
- ***con complessità e vincoli di tempo reale***: ogni player ha a disposizione un tempo massimo entro cui effettuare una mossa affinché questa sia real-time.

Ms. Pacman può essere inquadrato come un gioco ad *informazione completa*, a *più giocatori* e *con complessità e vincoli di tempo reale* in quanto vi è:

- la presenza di sensori che permettono di avere info complete sullo stato del gioco;
- la presenza dei team di fantasmini, avversari dell'agente;
- il vincolo di effettuare una mossa in 40 millisecondi.

2.3.1 Min-Max

In un normale problema di ricerca, la soluzione ottima consisterebbe in una sequenza di azioni legali che condurrebbe ad un nodo terminale che porterebbe alla vittoria. Nel

caso di ricerca con avversario è, invece, necessario considerare le mosse che l'opponente può eseguire, in quanto il suo obiettivo è minimizzare la funzione di utility. Dunque, l'albero di ricerca si sviluppa su 2 giocatori, chiamati MAX e MIN, dove il primo è il player che cerca di massimizzare l'utility (Ms. PacMan), mentre il secondo è l'avversario che cerca di minimizzarla (i fantasmini).

Considerando che lo scopo dell'elaborato è costruire un agente per Ms. Pacman, l'obiettivo dell'algoritmo che si presenterà sarà quello di raggiungere uno stato terminale dell'albero di ricerca con la massimizzazione dell'utilità.

Nei giochi ad informazione completa, la strategia perfetta può essere ottenuta mediante una ricerca esaustiva. In particolare, l'algoritmo che implementa quanto detto è ***minmax***. Il suo funzionamento di base è il seguente:

- si costruisce l'albero delle mosse fino ai nodi terminali;
- si applica la funzione di utilità U(x) ai nodi terminali;
- si usano i valori per calcolare l'utilità dei nodi superiori:
    - U(nodo_sup) = MAX U(nodo_inf) se la mossa è di MAX
    - U(nodo_sup) = MIN U(nodo_inf) se la mossa spetta a MIN.

L'algoritmo *minmax* effettua una completa esplorazione *depth-first* del game tree. Se la massima profondità dell'albero è *m* e ci sono *b* mosse legali ad ogni istante, allora la complessità temporale dell'algoritmo è O(b^m). La complessità spaziale è, invece, O(bm).

Dato che la complessità spaziale e la complessità temporale in molte situazioni risultano essere elevate, si può ricorrere a delle *funzioni di valutazione euristiche* per approssimare la vera utilità di uno stato senza fare una ricerca completa al fine di limitare l'esplorazione dell'albero. Inoltre, esistono delle tecniche di *pruning* le quali permettono di evitare di esplorare porzioni dell'albero di ricerca che risultano ininfluenti dal punto di vista della scelta finale della mossa ottima.
Questo tipo di tecniche può risultare molto utile in giochi con vincoli temporali come nel caso di Ms. Pacman. A tal proposito, nel paragrafo seguente si illustrerà la tecnica di ***alpha-beta pruning*** applicata a *minmax*.

2.3.2 Alpha-Beta pruning

Quando alpha-beta pruning viene applicato ad un albero minmax standard, fornisce la stessa mossa che restituirebbe minmax, ma taglia fuori i rami che non possono influenzare la decisione finale.
L'intuizione importante che consente di ottimizzare l'algoritmo base è la seguente: considerando un nodo n, se nell'albero è presente un altro nodo prima di n che per il giocatore in esame rappresenta la scelta migliore, allora n non sarà mai raggiunto. Quindi, una volta che si verifica questa circostanza, è possibile tagliare quel ramo.

Nella ricerca dell'albero si usano, quindi, due variabili:

- ***alpha***, valore maggiore di MAX nello stato attuale;
- ***beta***, valore minore di MIN nello stato attuale.

Calcolando MAX, si pota il sottoramo di un nodo se un suo figlio ha valore inferiore di *alpha*. Se, invece, tutti i figli hanno valore maggiore, il minimo diventa *alpha.*
Calcolando MIN, si pota il sottoramo di un nodo se un suo figlio ha valore maggiore di beta. Se, invece, tutti i figli hanno valore minore, il massimo diventa *beta.*

In definitiva, *alpha* rappresenterà il valore migliore trovato durante la ricerca per MAX, mentre *beta* sarà il valore migliore lungo MIN.

Di seguito, vengono riportati gli step di un esempio di funzionamento della tecnica in esame:

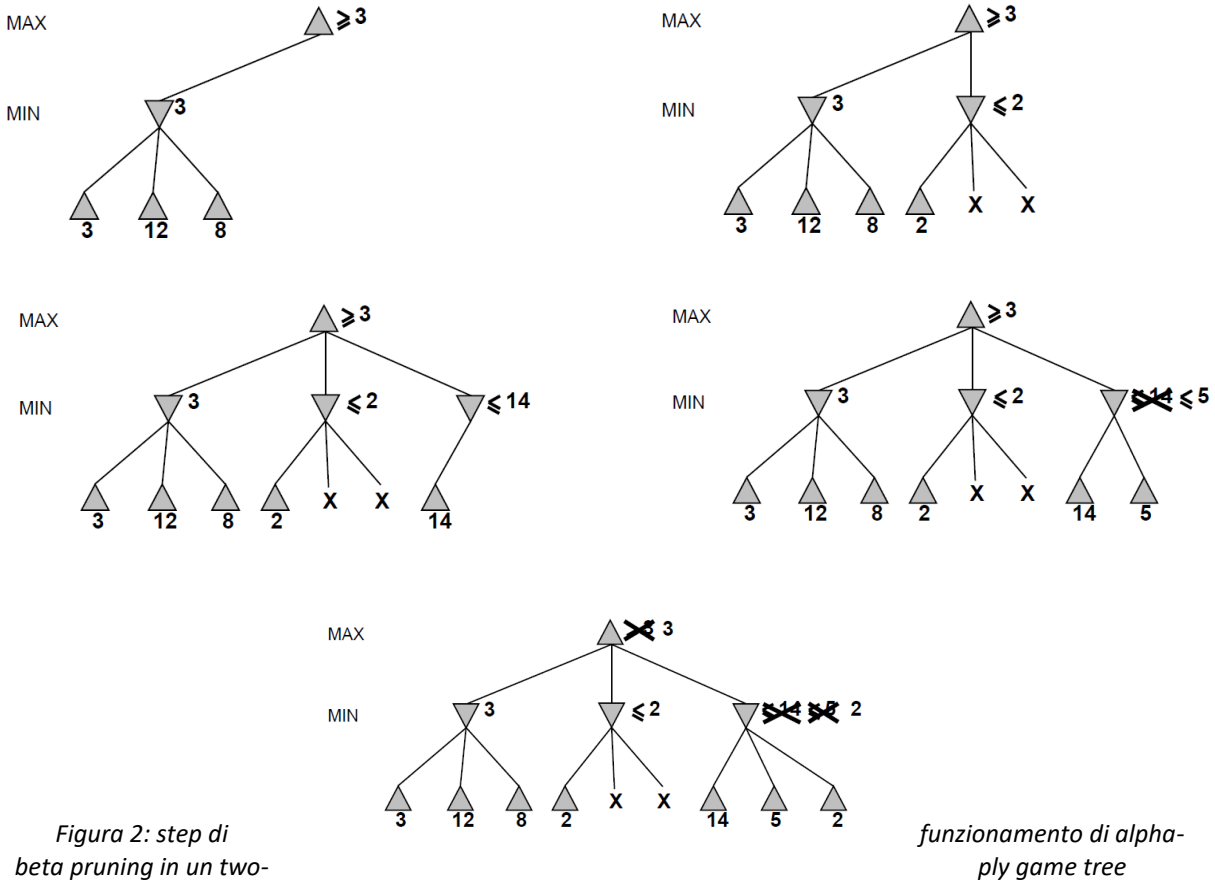


Figura 2: step di funzionamento di alpha-beta pruning in un two-ply game tree

2.4 Necessità di euristiche

Per assicurare il corretto comportamento dell'agente nelle diverse situazioni in cui si trova nel corso del gioco, è necessario definire delle euristiche. Queste consistono in un insieme di strategie che consentono di migliorare le prestazioni nella risoluzione del problema. Uno dei problemi dell'intelligenza artificiale è, infatti, proprio quello di

riconoscere una strada valida e arricchirla di strumenti interni e/o esterni all'algoritmo al fine di ottenere i migliori risultati possibili.

Si possono definire euristiche interne all'algoritmo le quali sono utili per il corretto calcolo del valore di un nodo terminale e per effettuare il *pruning*, tecnica, come già detto, molto utilizzata in quanto permette di ignorare porzioni dell'albero di ricerca che non sono significative per la scelta finale, ed euristiche esterne all'algoritmo, dette ***metaeuristiche***, le quali sono utili per imporre all'agente comportamenti differenti a seconda della situazione in cui si trova.
Nelle varie implementazioni proposte, discusse nel capitolo successivo, ne sono state fornite di diverse con l'obiettivo di migliorare lo score ottenuto dall'agente.

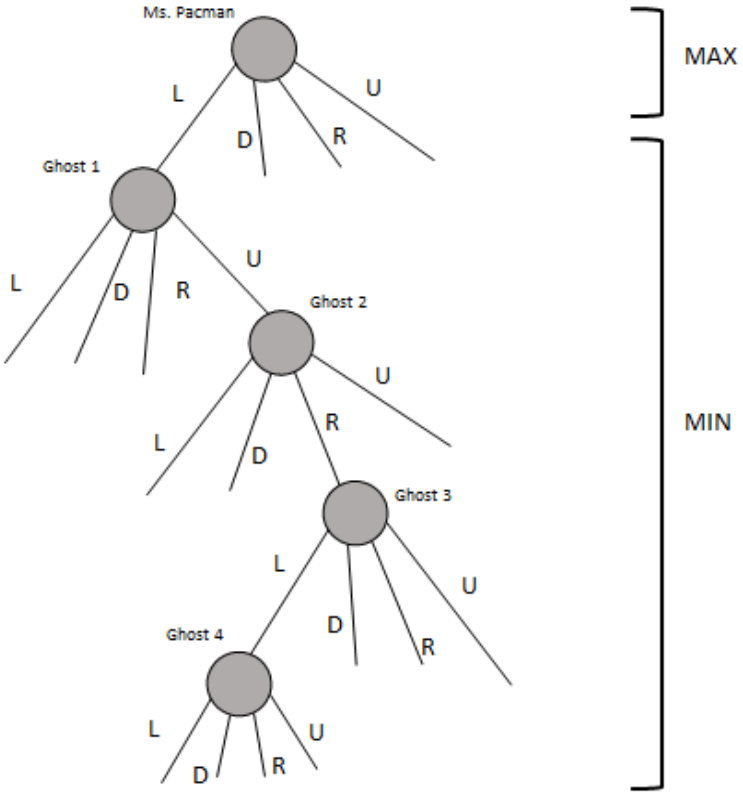
Capitolo 3: Implementazioni e sperimentazioni

In questo capitolo si illustreranno i vari controllori implementati. Si è, infatti, proseguito considerando dapprima una possibile soluzione, se ne sono successivamente analizzati i punti di debolezza e si è, quindi, cercato di agire al fine di migliorare le prestazioni, considerando l'obiettivo principale di massimizzare lo score di Ms. Pacman.

Per ognuno dei controller, si illustrerà il modo in cui è stato implementato l'algoritmo di ricerca, considerando le funzioni di utilità e le euristiche messe in atto. Si procederà, successivamente, ad un'analisi dei risultati del controllore contro i 3 differenti team di fantasmini, confrontandoli con le prestazioni di StarterPacman e con il controllore che si è rivelato essere il migliore fino al momento della descrizione, con l'ausilio di tabelle e grafici. Si effettuerà, inoltre, un'analisi delle criticità e dei problemi di tutte le differenti realizzazioni.
Al fine di rendere il confronto con StarterPacman equo, sono state eseguite 50 partite contro ognuno dei differenti Team, considerando miglior e peggior punteggio, media e deviazione standard dei punteggi delle 50 partite, livello massimo, livello minimo e livello medio raggiunto. Inoltre, per agevolare il confronto, si riporteranno nuovamente le tabelle dei risultati ottenuti da StarterPacman contro i diversi team, oltre che le tabelle del controllore risultato migliore fino al momento della descrizione.

3.1 MinMax versione 1

Nella prima versione, è stato utilizzato l'algoritmo *alpha-beta pruning* nella sua conformazione base. Al livello MAX, l'albero si espande considerando tutte le mosse legali di Ms. Pacman e, dunque, ciascun nodo MAX ha al massimo 4 figli. Al livello MIN, l'albero si espande considerando tutte le mosse legali di tutti e 4 i fantasmini e, dunque, ciascun nodo MIN ha al massimo 4^4figli.



Per quanto riguarda la profondità dell'albero, al fine di rispettare il

vincolo real time di prendere una decisione sulla prossima mossa entro 40 ms, si è deciso di considerare alberi con profondità crescente e fermarsi quando il tempo a disposizione scade. In particolare, per questa versione si è visto che è possibile partire da un albero con profondità pari a 30 e in media si arriva ad una profondità circa pari a 40.

E' stata considerata un'unica *metaeuristica*, che riguarda gli istanti in cui tutti i fantasmini sono nella tana al centro del labirinto. In questa situazione, infatti, non essendoci pericolo che Ms. Pacman possa perdere una vita venendo mangiata, il suo unico obiettivo è quello di cercare di mangiare quante più pillole possibili per aumentare il proprio score e progredire di livello.

La *funzione di utility* implementata è la seguente:

$$utility = score + W_1 d_1 + W_2 d_2 + W_3 d_3 - isLifeLost - gameOver$$

in cui:

- *score* è il punteggio accumulato da Ms. Pacman nel gioco;

- $W_1 d_1$ è il prodotto tra:
  - la *distanza* tra Ms. Pacman e il fantasmino non mangiabile più vicino che si trova nel raggio di 25 metri, ovvero d_1;
  - il *peso* che si intende dare a questo fattore, ovvero W_1, settato a *+4* .

  L'idea è, infatti, quella di incrementare l'utility dell'agente all'aumentare della distanza dal fantasmino, così da farlo scappare da quest'ultimo;

- $W_2 d_2$ è il prodotto tra:
  - la *distanza* tra Ms. Pacman e il fantasmino mangiabile più vicino che si trova nel raggio di 25 metri, ovvero d_2;
  - il *peso* che si intende dare a questo fattore, ovvero W_2, settato a *-5*.

  L'idea è, infatti, quella di decrementare l'utility dell'agente all'aumentare della distanza dal fantasmino, così da farlo avvicinare a quest'ultimo e, quindi, mangiarlo per accumulare punti*;*

- $W_3 d_3$ è il prodotto tra:
  - la *distanza* tra Ms. Pacman e la pillola più vicina, ovvero d_3;
  - Il *peso* che si intende dare a questo fattore, ovvero W_3, settato a *-1.*

  L'idea è, infatti, quella di decrementare l'utility dell'agente all'aumentare della distanza dalla pillola, così da farlo avvicinare a quest'ultima e, quindi, mangiarla per accumulare punti;
- $isLifeLost$ rappresenta il decremento di utility quando Ms. Pacman perde una vita. In particolare, viene settato a *500* così da penalizzare le mosse che portano Ms. Pacman a perdere una vita;

- $gameOver$ rappresenta il decremento di utility quando Ms. Pacman perde tutte le vite a disposizione. In particolare, viene settato a *15000* così da penalizzare le mosse che portano Ms. Pacman a perdere la partita.

Di seguito vengono riportati i risultati ottenuti contro i 3 differenti team di fantasmini.

MinMax1 vs AggressiveGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
13890	910	4620.2	2367.222	3	0	0.3

Tabella 4: MinMax1 vs AggressiveGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
11050	2530	6015.2	1524.197	1	0	0.02

Tabella 5: StarterPacman vs AggressiveGhosts

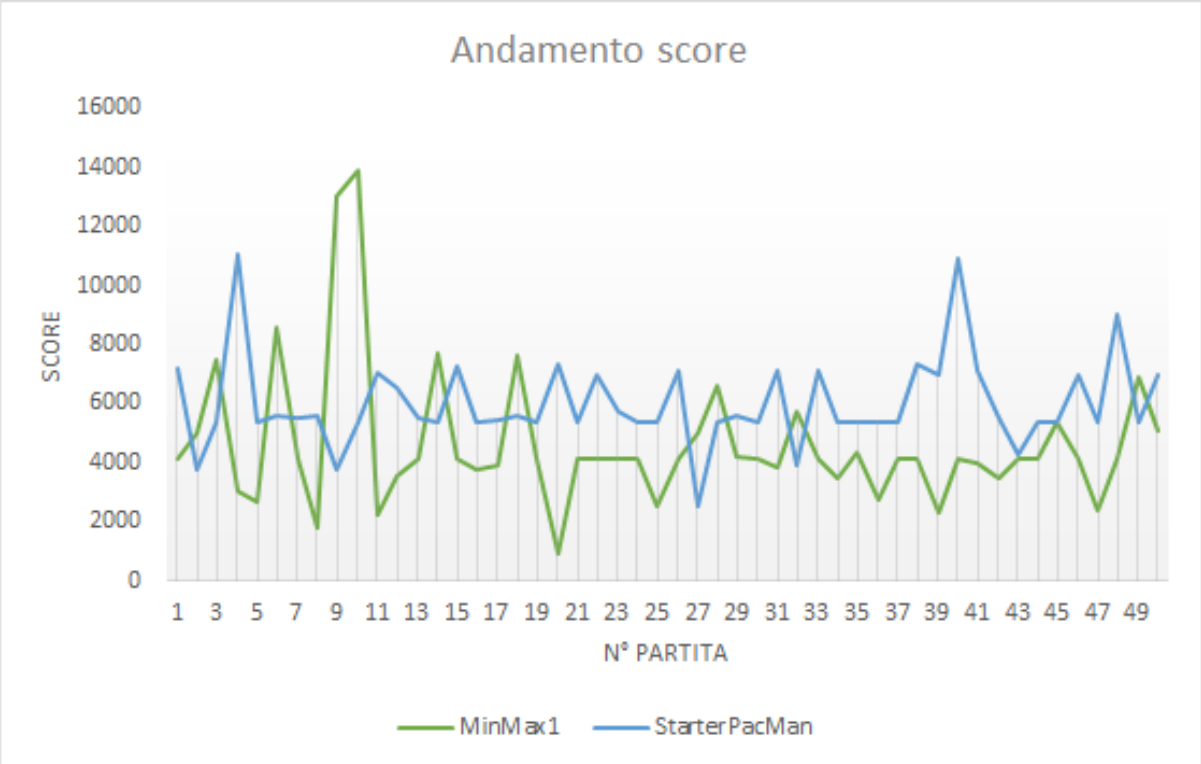


Figura 3: andamento dello score di MinMax1 e Starter PacMan nelle 50 partite contro gli AggressiveGhosts

Le prestazioni di MinMax1 contro gli AggressiveGhosts sono mediamente peggiori di StarterPacman, nonostante il livello raggiunto sia mediamente più alto. Ciò è

conseguenza di un comportamento anomalo di Ms. Pacman che tende ad intraprendere sempre lo stesso percorso che la porta a sopravvivere ma a non accumulare punteggio. In questo caso, infatti, al raggiungimento dei 4000 ticks si avanza di livello, senza però accumulare punteggio bonus rispetto a quello attuale. All'inizio del nuovo livello, dunque, Ms. Pacman raccoglierà alcune pillole per incrementare lo score, ma dopo poco tenderà nuovamente a salvaguardare le proprie vite intraprendendo nuovamente il comportamento appena descritto.

MinMax1 vs RandomGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
12420	780	4913.8	2497.747	5	0	2.2

Tabella 6: risultati MinMax1 vs RandomGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
65560	13320	34301.8	13839.852	11	2	5.56

Tabella 7: risultati StarterPacman vs RandomGhosts

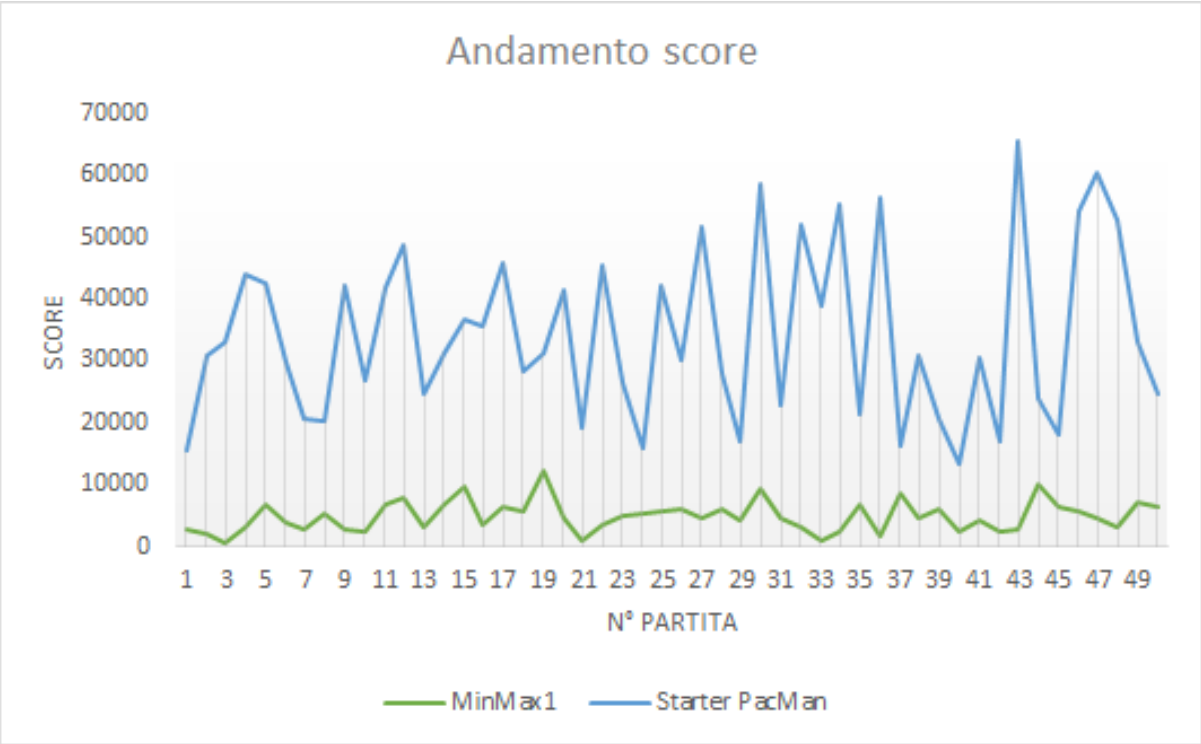


Figura 4 :andamento dello score di MinMax1 e Starter PacMan nelle 50 partite contro i RandomGhosts

Contro i RandomGhosts, il controllore ha prestazioni di gran lunga peggiori rispetto a Starter Pacman. Dal grafico e dalla tabella, infatti, si nota come anche il miglior punteggio di MiniMax1 sia peggiore del punteggio più basso del controllore base. Verificando la colonna del livello medio, si evidenzia ancor di più il problema che si aveva contro gli Aggressive. In particolare, malgrado raggiunga in media circa il livello 2, si nota come il punteggio resta comunque molto basso, soprattutto rispetto a quello che dovrebbe, in realtà, ottenere una volta superati i primi 2 livelli.

MinMax1 vs LegacyGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
5970	970	2285.4	1291.911	1	0	0.02

Tabella 8: risultati MiniMax1 vs LegacyGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
9990	1600	5175.2	1919.888645	0	0	0

Tabella 9: risultati StarterPacman vs LegacyGhosts

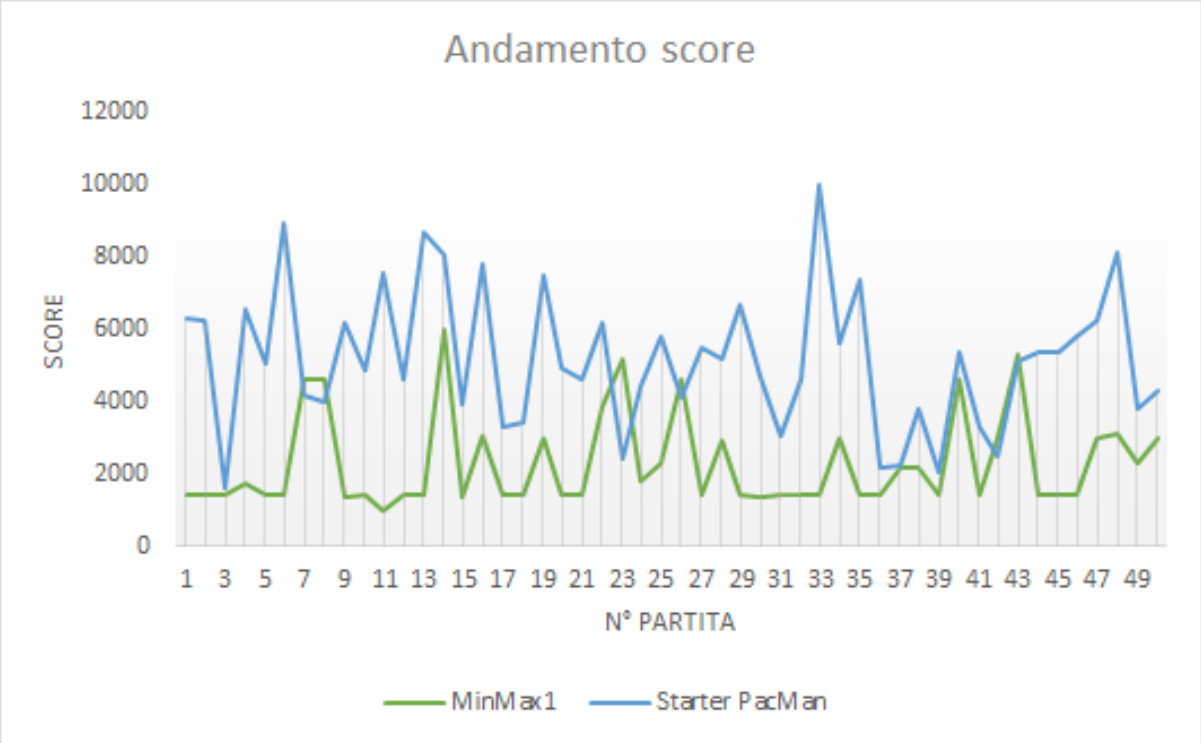


Figura 5: andamento dello score di MinMax1 e Starter PacMan nelle 50 partite contro i LegacyGhosts

Anche in questo caso, non si è riusciti a superare le prestazioni di Starter PacMan. In particolare, si ottengono le peggiori performance proprio contro questo team dato che tre fantasmini hanno un atteggiamento Aggressive e uno ha un atteggiamento Random. Di conseguenza, la strategia di gioco che porta l'agente ad effettuare sempre lo stesso percorso per sopravvivere e superare il livello risulta fallimentare, dal momento che c'è un fantasmino che non lo insegue ma si muove casualmente per la mappa, provocando un eventuale cambio del percorso dell'agente, che, quindi, non riesce più a prendere la decisione migliore, utile a salvarsi la vita.

Considerazioni finali MinMax1

Concludendo il discorso su questo controllore, la coesistenza di più fattori che influenzano la funzione di utility rende difficile equilibrare i loro pesi al fine di raggiungere buoni risultati.

Inoltre, l'atteggiamento che Ms. Pacman ha di cercare prima di tutto di non perdere la vita e poi di raccogliere le pillole, che si traduce nell'intraprendere sempre lo stesso percorso all'interno del labirinto, suggerisce che la funzione di utility considerata risulta essere buona per sfuggire ai fantasmini, ma non sembra essere particolarmente efficiente per raccogliere le pillole presenti nel labirinto.

Infine, dalle sperimentazioni si è visto che viene raggiunto in media una profondità dell'albero di appena 40, cominciando la ricerca da una depth di 30. Ciò influenza significativamente la scelta della mossa dell'agente, in quanto non gli consente di

prevedere correttamente la mossa migliore e, quindi, di ottenere prestazioni che siano elevate.

I successivi controllori implementati mirano a risolvere proprio questi problemi, in quanto considerati come quelli a più alto impatto sulle prestazioni complessive di Ms. Pacman.

3.2 MinMax versione 2

Considerando i risultati ottenuti precedentemente, in questo controllore si è deciso di considerare più *metaeuristiche*, in modo da utilizzare comportamenti standard ed efficaci oppure differenti versioni di alpha-beta pruning che differiscono per la funzione di utility considerata, da sfruttare a seconda della situazione in cui l'agente si trova e per cercare, quindi, di migliorare le prestazioni.

Si elencano le metaeuristiche implementate, in ordine di priorità:

- La prima considera la situazione in cui:
  - Ms. Pacman si trova vicina ad una power pill, ovvero:
    $$distPacmanToPowerPill <= 5$$
  - non ci sono fantasmini vicini a Ms Pacman, ovvero:
    $$distPacmanToNearestGhost >= 4$$
  - Ms. Pacman è più vicina alla power pill di quanto non lo siano i fantasmini, ovvero:
    $$distGhostToPowerPill >= 6$$

  In questo caso, viene imposto alla protagonista di smettere di muoversi e di tendere un'imboscata ai fantasmini vicino alla power pill. In questo modo, si evita che Ms Pacman mangi la power pill quando i fantasmini sono lontani e si massimizza, dunque, il beneficio della power pill stessa.

- La seconda considera la situazione in cui:
  - Ms. Pacman ha un fantasmino nelle vicinanze, ovvero:
    $$distPacmanToNearestGhost <= 8$$
  - esiste una power pill che risulta più vicina a Ms. Pacman che a qualsiasi altro fantasmino, ovvero:
    $$distPacmanToPowerPill <= distGhostToPowerPill$$

  In questo caso, viene applicato alpha-beta prunung con funzione di utility che considera solo il decremento di utilità all'aumentare della distanza dalla power pill e il decremento di utilità quando Ms. Pacman perde una vita. In particolare:
  $$utility = score + W_1 d_1 - isLifeLost - gameOver$$
  dove W_1 è il peso che si vuole dare a questo fattore, settato a *-5* , d_1 è la distanza tra Ms. Pacman e la power pill, mentre gli altri tre fattori hanno il significato descritto nel paragrafo 3.1.

- La terza considera la situazione in cui esistono dei fantasmini mangiabili.
  In questo caso, viene applicato alpha-beta pruning con funzione di utility che considera solo il decremento di utilità all'aumentare della distanza di Ms. Pacman dal fantasmino mangiabile. In particolare:

$$utility = score + W_1 d_1 - isLifeLost - gameOver$$

dove W_1 è il peso che si vuole dare a questo fattore, settato a *-4*, d_1 è la distanza tra Ms. Pacman e il fantasmino mangiabile più vicino, mentre gli altri tre fattori hanno il significato descritto nel paragrafo 3.1.

- La quarta considera il caso in cui si è in una zona in cui Ms. Pacman è abbastanza distante dai fantasmini. In questo caso, viene applicato alpha-beta pruning con funzione di utility che considera solo il decremento di utilità all'aumentare della distanza dalla pillola più vicina. In particolare:

  $$utility = score + W_1 d_1 - isLifeLost - gameOver$$

  dove W_1 è il peso che si vuole dare a questo fattore, settato a *-5*, d_1 è la distanza tra Ms. Pacman e la pillola più vicina, mentre gli altri tre fattori hanno il significato descritto nel paragrafo 3.1.

Come ci si aspettava, essendo invariata la complessità dell'albero, la profondità media raggiunta è circa pari a 40 utilizzando come depth iniziale 30.

I risultati ottenuti verranno illustrati nella pagina seguente.

MinMax2 vs AggressiveGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
13890	910	4620.2	2367.222	3	0	0.3

Tabella 10: risultati MiniMax1 vs AggressiveGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
11050	2530	6015.2	1524.197	1	0	0.02

Tabella 11: risultati StarterPacman vs AggressiveGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
13970	1270	4051.2	2523.641	1	0	0.44

Tabella 12: risultati MiniMax2 vs AggressiveGhosts

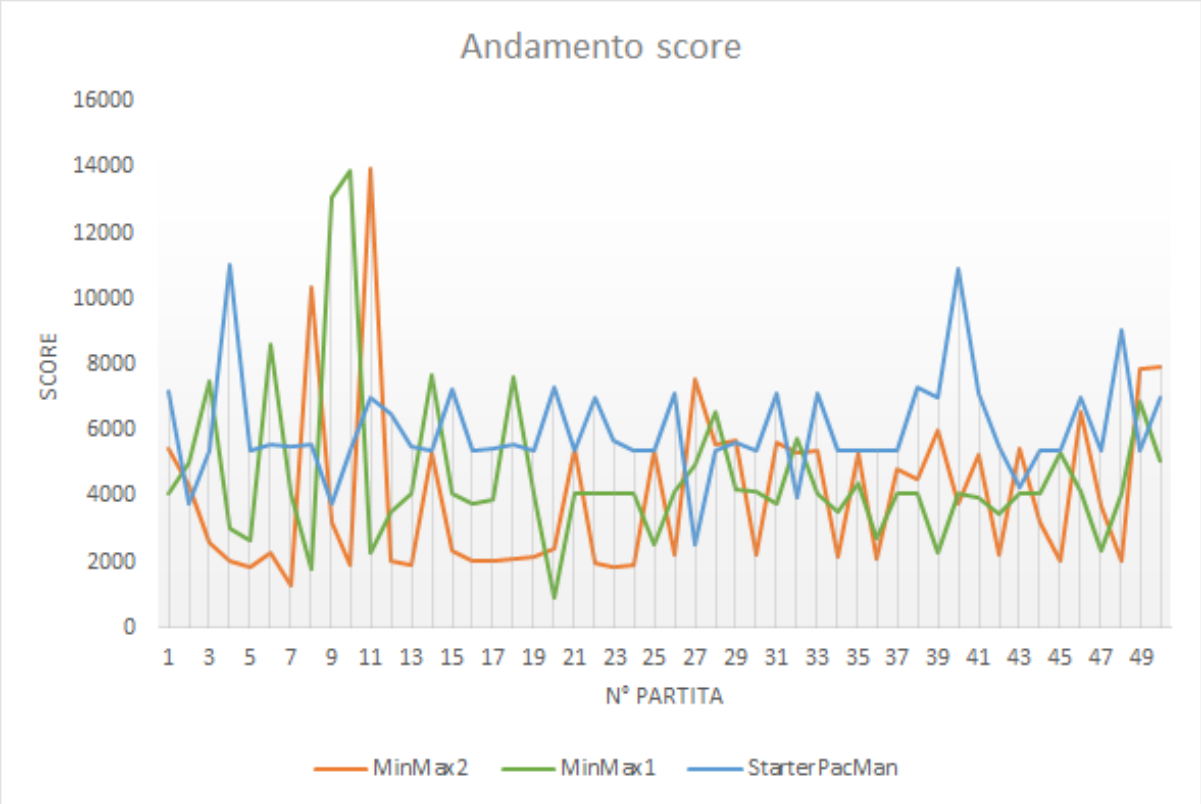


Figura 6: andamento dello score di MinMax1, MinMax2 e Starter PacMan nelle 50 partite contro gli AggressiveGhosts

Il comportamento dell'agente risulta abbastanza simile a quello della versione 1. E', però, possibile notare una sostanziale differenza sul livello medio raggiunto, il quale risulta essere leggermente superiore nonostante il livello massimo raggiunto sia 1, mentre nella precedente versione è pari a 3. Ciò potrebbe portare a dire che:

- l'agente tende ad intraprendere lo stesso percorso dopo un po' di mosse effettuate, come in MinMax1;
- l'agente riesce, in media, a sopravvivere più volte fino al time-out del livello di 4000 ticks nel livello 0;
- l'agente ha più difficoltà di MinMax1 nel livello 1, dato che non riesce mai a superarlo nonostante mediamente lo raggiunga più volte.

In realtà, anche se MinMax1 raggiunge uno score medio migliore, non è possibile asserire alcuna affermazione su quale dei due algoritmi sia il migliore dato che la deviazione standard dei punteggi è elevata e non permette, dunque, una considerazione di questo tipo. Per avvalorare questa ipotesi, è stato effettuato, come suggerito dalla piattaforma online *STAC*, il test di ipotesi *t-test paired* con livello di significatività pari a 0.05:

T-Statistic	p-value	Result
-1.08588	0.28284	H0 is accepted

Tabella 13: t-test paired MinMax2 vs MinMax1 per confronto risultati contro AggressiveGhosts

Si noti come il p-value è risultato ben maggiore rispetto al livello di significatività, dunque l'ipotesi nulla deve essere accettata confermando il fatto che non è possibile distinguere le due distribuzioni di probabilità dal punto di vista statistico.

MinMax2 vs RandomGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello Medio
3730	480	1302.2	626.995	3	0	0.74

Tabella 14: MinMax2 vs RandomGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
65560	13320	34301.8	13839.852	11	2	5.56

Tabella 15: StarterPacman vs RandomGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
12420	780	4913.8	2497.747	5	0	2.2

Tabella 16: MinMax1 vs RandomGhosts

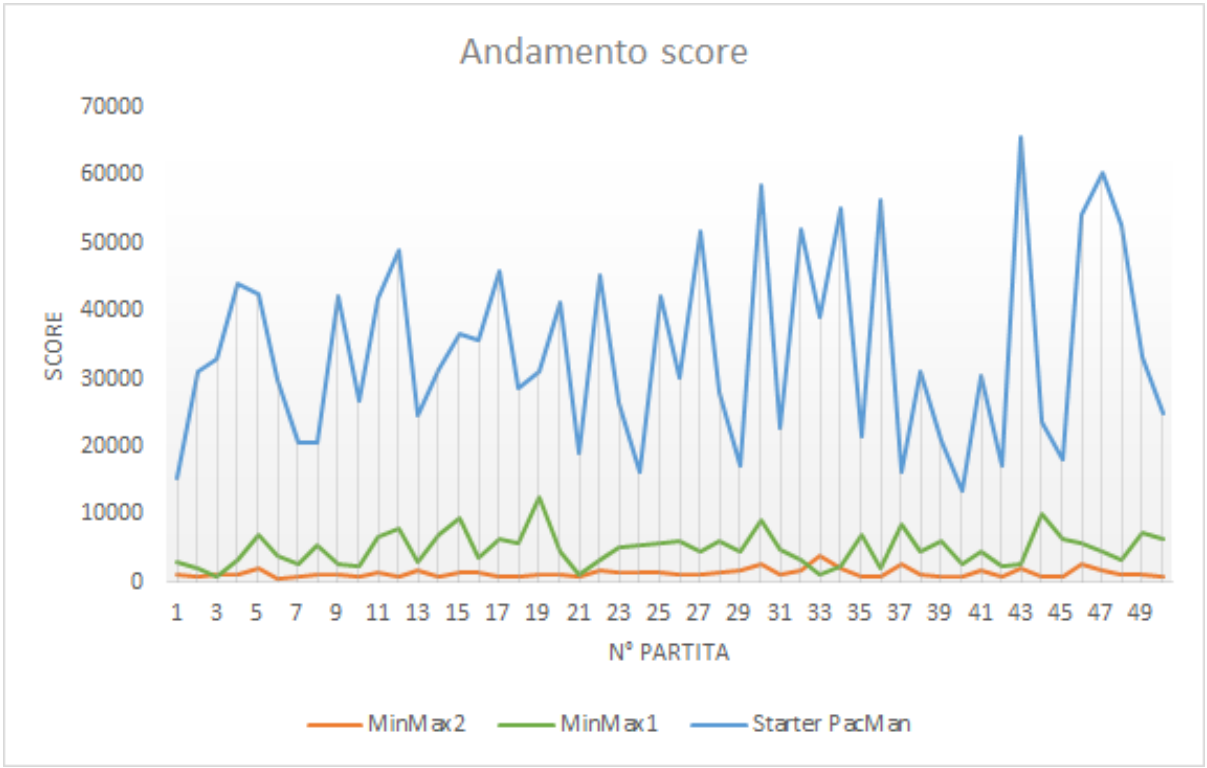


Figura 7: andamento dello score di MinMax1, MinMax2 e Starter PacMan

nelle 50 partite contro i RandomGhosts

In questo caso risulta molto evidente come il controllore implementato sia completamente inefficiente per gareggiare contro i RandomGhosts, come dimostra la media dello score pari a 1302 punti, circa 4000 punti in meno rispetto a MinMax1.

Uno dei fattori che ha, probabilmente, un impatto maggiore sulle prestazioni è la totale randomicità del comportamento dei fantasmini, che rende particolarmente inefficienti le euristiche messe in atto. Ad esempio, al problema che l'agente ha di preferire di non perdere la vita piuttosto che accumulare punti, si aggiunge quello della completa inefficienza delle euristiche pensate per le situazioni in cui Ms Pacman viene attaccata e deve mangiare la power pill in modo da ottenere il massimo beneficio in termini di punteggio.

Nella pagina seguente si prosegue l'analisi considerando il team di fantasmini rimanente, i LegacyGhosts.

MinMax2 vs LegacyGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
5700	1750	3216.8	829.080	0	0	0

Tabella 17: risultati MinMax2 vs LegacyGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
9990	1600	5175.2	1919.889	0	0	0

Tabella 18: risultati StarterPacman vs LegacyGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
5970	970	2285.4	1291.911	1	0	0.02

Tabella 19: risultati MiniMax1 vs LegacyGhosts

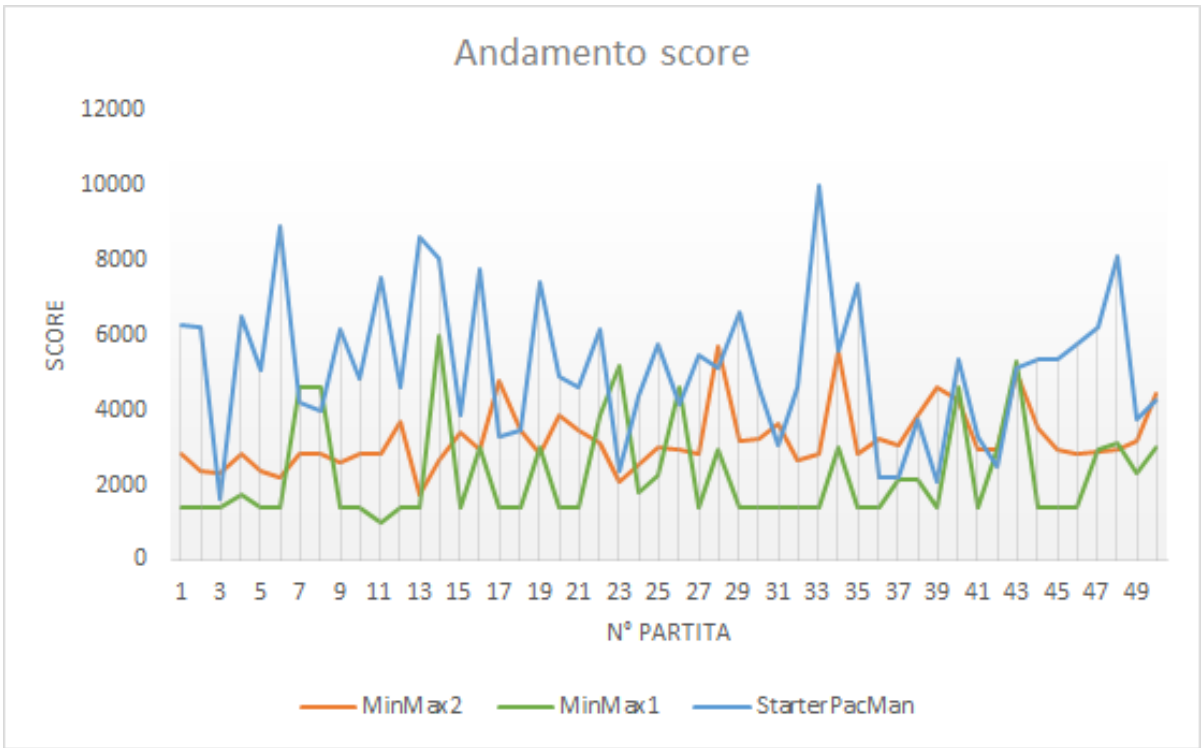


Figura 8: andamento dello score di MinMax1, MinMax2 e Starter PacMan nelle 50 partite contro i LegacyGhosts

Per quanto concerne i risultati contro i LegacyGhosts, è necessario effettuare il test d'ipotesi per comprendere quale dei due algoritmi sia migliore.
In particolare, è stato effettuato il *Wilcoxon test*, come suggerito dalla piattaforma Web *STAC*. I risultati ottenuti sono i seguenti:

Statistic	p-value	Result
229	0.00008029257852296078	H0 is rejected

Tabella 20:Wilcoxon test tra MinMax2 vs MinMax1 per confronto risultati contro LegacyGhosts

Dal test d'ipotesi risulta, dunque, che MinMax2 sia da reputare migliore contro questo team di fantasmini.
Inoltre, dal confronto con i risultati ottenuti contro gli AggressiveGhosts, si nota ulteriormente come la presenza di fantasmini con comportamenti randomici sia un problema rilevante per il controllore.

Considerazioni finali MinMax2

Analizzando i risultati complessivi ottenuti da questa versione dell'agente, si può verificare che le modifiche messe in atto non hanno portato i miglioramenti che si volevano ottenere. Il problema principale risulta, probabilmente, essere la profondità dell'albero raggiunta, comune ai due algoritmi presentati finora. Dunque, prima di procedere con un raffinamento delle euristiche e metaeuristiche utilizzate, si è deciso di provare a ridurre la complessità dell'albero nell'ottica di raggiungere profondità maggiori.

Si è, quindi, ritenuto necessario implementare una nuova versione del controllore, descritta nel paragrafo successivo. Come miglior controllore finora ottenuto di riferimento si utilizzerà ancora MinMax1 che, come si è visto, è risultato peggiore di MinMax2 contro i LegacyGhosts, ha prestazioni equivalenti contro gli AggressiveGhosts, ma risulta essere di gran lunga più efficiente contro i RandomGhosts.

3.3 Max versione 1

L'obiettivo principale che si vuole perseguire con questa nuova versione dell'algoritmo è quello di ridurre la complessità dell'albero in modo da ottenere profondità maggiori, rendendo Ms Pacman capace di considerare un numero di mosse future superiore.

Come prima prova si è deciso di considerare un unico avversario che racchiudesse il comportamento di tutti e 4 i fantasmini, quindi un unico livello Min. In particolare, si è deciso di considerare come unico avversario semplicemente il fantasmino più vicino a Ms Pacman, ignorando dunque gli altri tre.

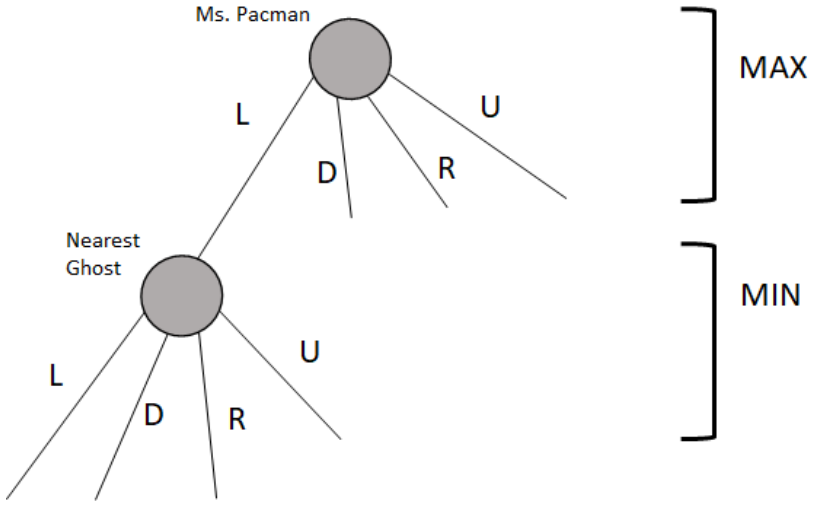


Figura 9: Schema alpha-beta pruning considerato in una fase di prova

In questo modo si è riusciti ad aumentare la profondità iniziale dell'albero e, dunque, si è riusciti a perseguire lo scopo di alleggerire la complessità dell'albero, ottenendo profondità raggiunte maggiori. I risultati, però, non sono stati soddisfacenti in termini di score, difatti Ms Pacman risulta perdere la vita in molte situazioni dato che, pur esplorando un albero minmax più profondo, non prevede le mosse di tutti gli avversari ma solo di quello che le sta più vicino.

Per questo motivo si è deciso di considerare una ulteriore semplificazione dello schema minmax prima di dedicarsi ad uno studio approfondito sui risultati ottenuti.

In particolare, si è deciso di considerare una versione in cui, per ogni fantasmino, viene considerata solo la mossa di avvicinarsi a Ms.Pacman. Si può, dunque, affermare che

non sono più presenti livelli *MIN* dato che la mossa dei fantasmini risulta essere deterministica.

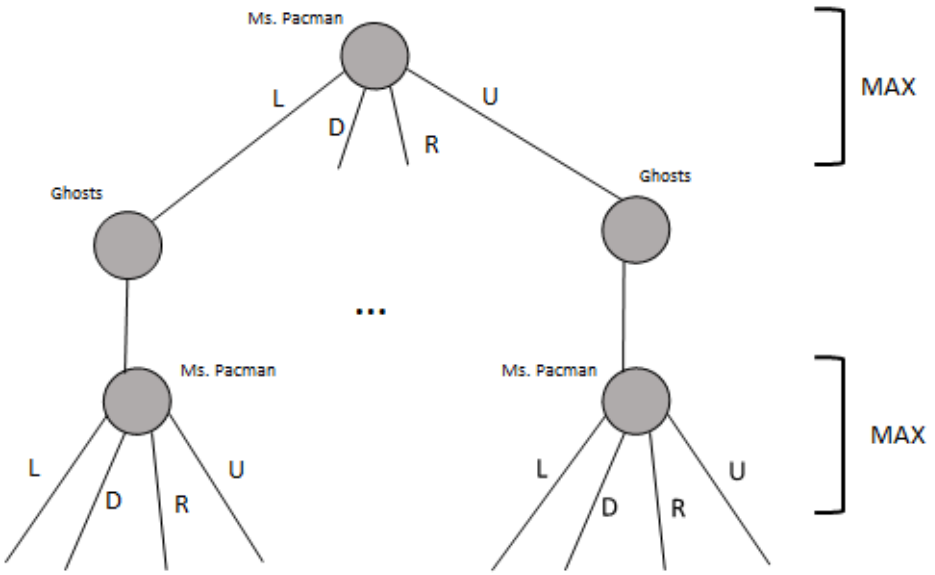


Figura 10: Schema di alpha-beta pruning in MAX versione 1

Le metaeuristiche considerate sono, in ordine di priorità:

1. Come nella prima metaeuristica di MinMax2, si considera la situazione in cui:
   a. Ms. Pacman si trova vicina ad una power pill, ovvero:
   $$distPacmanToPowerPill <= 5$$
   b. non ci sono fantasmini vicini a Ms Pacman, ovvero:
   $$distPacmanToNearestGhost >= 4$$
   c. Ms. Pacman è più vicina alla power pill di quanto non lo siano i fantasmini, ovvero:
   $$distGhostToPowerPill >= 6$$
   In questo caso, viene imposto alla protagonista di smettere di muoversi e di tendere un'imboscata ai fantasmini vicino alla power pill. In questo modo, si evita che Ms Pacman mangi la power pill quando i fantasmini sono lontani e si massimizza, dunque, il beneficio della power pill stessa.

2. se c'è un fantasma non mangiabile troppo vicino, Ms. Pacman deve scappare da lui utilizzando questa particolare versione dell'algoritmo, chiamata *"Max"*, la cui funzione di utility è la stessa presentata per *MinMax1* nel paragrafo 3.1, che, quindi, mira a non perdere vite ma contemporaneamente ad aumentare il punteggio mangiando pills e power pills;
3. se c'è un fantasma mangiabile, si impone che Ms. Pacman vada verso di lui per mangiarlo ed accumulare punti. Si sottolinea come questa euristica venga perseguita solo se non lo sono state le prime due;

4. se ci sono pillole e power pills, Ms. Pacman deve muoversi verso la più vicina. Nuovamente, questa euristica viene perseguita solo se non lo sono state le prime tre.

In questo modo, si è riusciti, semplificando l'albero di ricerca, ad incrementare la profondità iniziale dello stesso ad un valore pari a 70, raggiungendo in media profondità pari a 90. Inoltre, essendo l'albero composto da soli livelli max, avere una tale profondità significa rendere Ms Pacman capace di prevedere effettivamente le prossime 90 mosse possibili, riuscendo dunque a compiere l'azione migliore nello stato attuale in base ai 90^4 possibili futuri.

Max1 vs AggressiveGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
32420	5470	18118.2	6687.846	2	0	0.88

Tabella 21: Max1 vs AggressiveGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
11050	2530	6015.2	1524.197	1	0	0.02

Tabella 22: risultati StarterPacman vs AggressiveGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
13890	910	4620.2	2367.222	3	0	0.3

Tabella 23: MinMax1 vs AggressiveGhosts

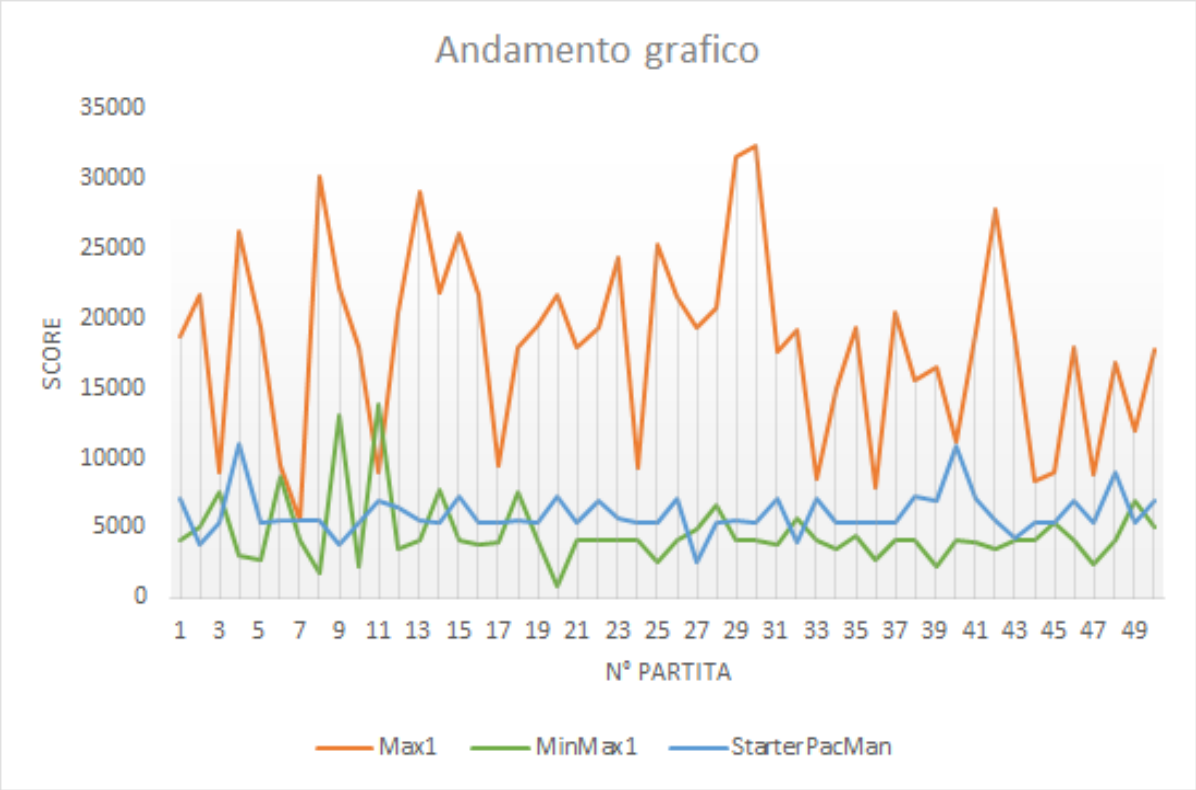


Figura 11: andamento dello score di Max1, MinMax1 e Starter PacMan nelle 50 partite contro gli AggressiveGhosts

Le prestazioni di questo controllore sono di gran lunga migliori di quelle degli altri due. Si riesce, infatti, a raggiungere una media di 18118.2 che risulta essere di circa 12000 migliore di MinMax1. Inoltre, l'agente riesce mediamente a superare il livello 0 raccogliendo quasi tutte le pillole presenti nel labirinto, come testimoniato dai punteggi raggiunti. Uno degli aspetti di maggior impatto su queste prestazioni è senza dubbio la depth raggiunta, che risulta essere mediamente pari a 90, come detto precedentemente.

Inoltre, si è avuto un notevole miglioramento anche perché è stata considerata, per ogni fantasmino, l'unica mossa che realmente intraprende nel caso "Aggressive", ovvero andare verso Ms. Pacman per cercare di mangiarla.

Nella pagina seguente si prosegue l'analisi considerando i team di fantasmini rimanenti, i RandomGhost e in seguito saranno analizzati i LegacyGhosts.

Max1 vs RandomGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
41220	1750	14768.4	3290.730	6	0	1.92

Tabella 24: Max1 vs RandomGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
65560	13320	34301.8	13839.852	11	2	5.56

Tabella 25: StarterPacman vs RandomGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
12420	780	4913.8	2497.747	5	0	2.2

Tabella 26: MinMax1 vs RandomGhosts

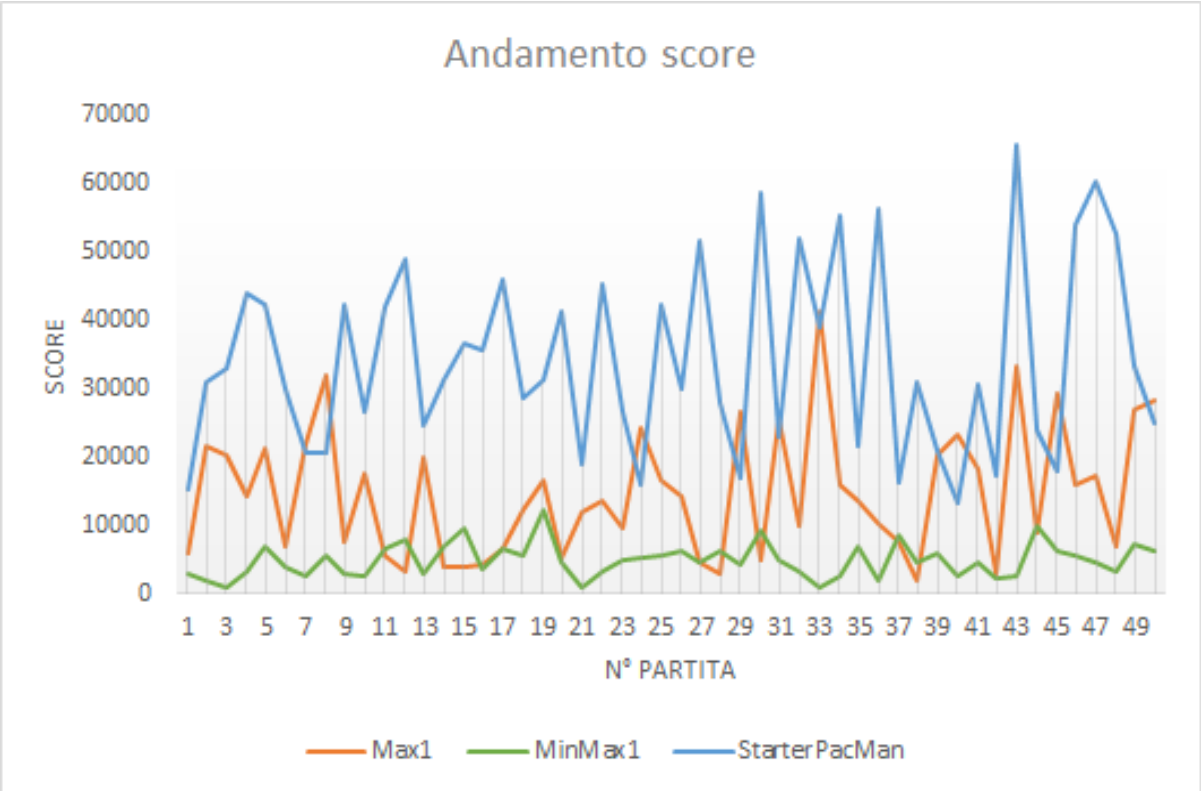


Figura 12: andamento dello score di Max1, MinMax1 e Starter PacMan nelle 50 partite contro i RandomGhosts

Mentre rispetto a MinMax1 le prestazioni sono nettamente superiori, non si può affermare lo stesso nel confronto con Starter PacMan. Il motivo è da ricercare nel fatto che si è considerata come mossa dei fantasmini quella che li avvicina a Ms. Pacman, che non è la mossa che effettuano realmente i fantasmini Random e, dunque, l'agente non riesce a prevedere correttamente le azioni avversarie.
Ciò è confermato dal fatto che l'agente ottiene peggiori risultati medi rispetto a quelli che si hanno contro gli AggressiveGhosts.

Max1 vs LegacyGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
25960	4320	11459	5374.243	2	0	0.52

Tabella 27: risultati Max1 vs LegacyGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
9990	1600	5175.2	1919.889	0	0	0

Tabella 28: risultati StarterPacman vs LegacyGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
5970	970	2285.4	1291.911	1	0	0.02

Tabella 29: risultati MiniMax1 vs LegacyGhosts

Il grafico viene riportato nella pagina seguente.

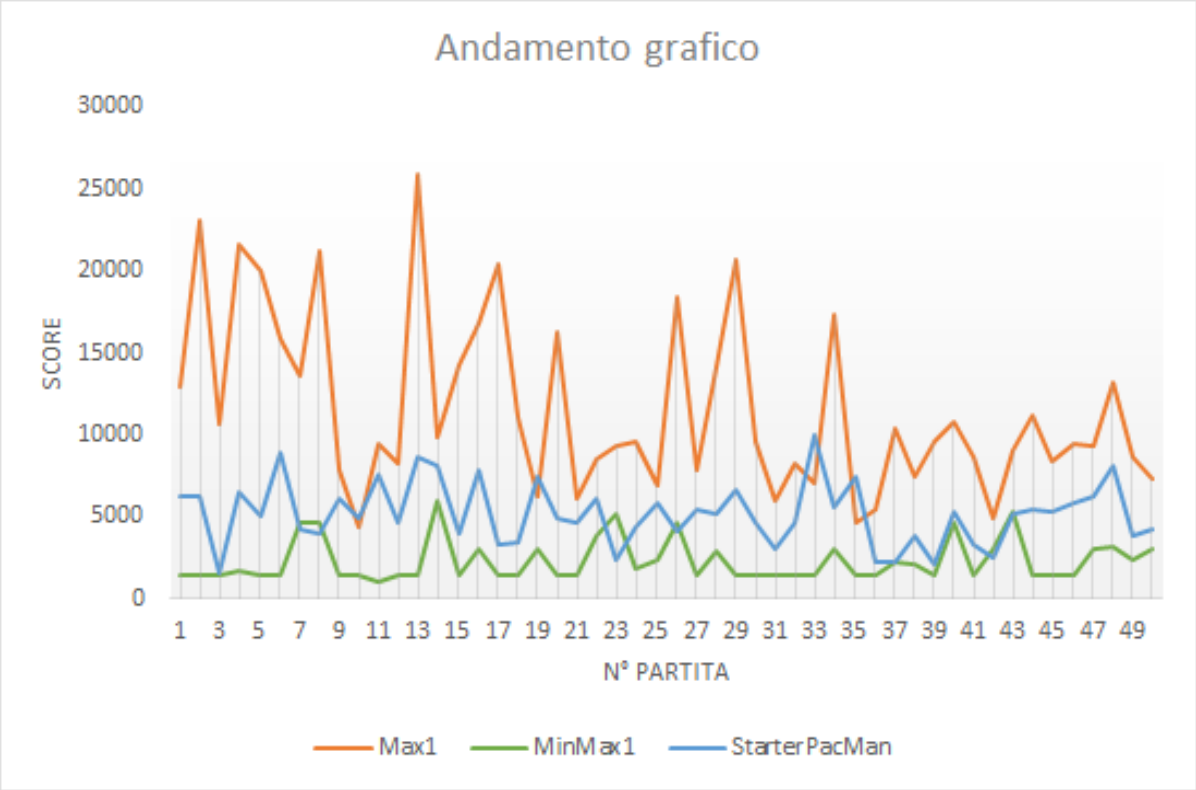


Figura 13: andamento dello score di Max1, MinMax1 e Starter PacMan nelle 50 partite contro i LegacyGhosts

Anche contro i LegacyGhosts è possibile apprezzare un netto miglioramento delle prestazioni rispetto a MinMax1. Il risultato non sorprende considerando soprattutto i risultati ottenuti contro gli *Aggressive* e contro i *Random*, dato che il team in esame è composto da 3 fantasmini con atteggiamento Aggressive e 1 con atteggiamento Random.

Considerazioni finali Max1

I risultati ottenuti evidenziano come l'aver raggiunto una profondità dell'albero che si riesce ad esplorare pari a 90 abbia portato a prestazioni di gran lunga migliori rispetto ai controllori precedenti, dove la depth era infatti uguale a 40.
Un altro fattore sicuramente importante è l'aver definito delle metaeuristiche adeguate che permettono a Ms. Pacman di mettere in atto differenti strategie a seconda della situazione in cui si trova.

Si è provato a ridurre ulteriormente la complessità del problema dopo aver osservato un comportamento che caratterizza gli Aggressive Ghosts. In particolare, questi ultimi tendono a disporsi in fila dietro Ms. Pacman e, dunque, per cercare di migliorare ulteriormente le prestazioni si è considerato che la mossa del fantasma più vicino fosse quella di andare verso Ms. Pacman, mentre per gli altri 3 si è considerata la mossa di muoversi verso il fantasma più vicino a Ms. Pacman. La modifica non ha,

però, portato ad effettivi miglioramenti dal punto di vista prestazionale, per cui si è ritenuto non utile riportare i risultati.

A questo punto, ci si è posti l'obiettivo di migliorare queste performance mediante un raffinamento delle euristiche e delle metaeuristiche. Si considererà, dunque, questa versione del controllore, a cui ci si riferirà col nome Max1, come il migliore ottenuto finora con cui effettuare confronti.

3.4 Max versione 2

Considerando i risultati ottenuti in Max1, si è ritenuto opportuno proseguire utilizzando come base valida da cui partire la versione dell'algoritmo MinMax con solo livello Max. Si ricorda che l'unica mossa considerata dei fantasmini è quella di avvicinarsi a Ms. Pacman, ovvero il loro comportamento è del tutto deterministico.

Al fine di ottenere dei miglioramenti, sono state modificate le metaeuristiche già presenti così da perfezionare i comportamenti dell'agente nelle differenti situazioni.
In particolare, sono stati definiti i seguenti comportamenti in ordine di priorità:

1. Si considera la situazione in cui:
    a. Ms. Pacman si trova vicina ad una power pill, ovvero:
    $$distPacmanToPowerPill <= 4$$
    b. non ci sono fantasmini vicini a Ms Pacman, ovvero:
    $$distPacmanToNearestGhost >= 6$$
    c. Ms. Pacman è più vicina alla power pill di quanto non lo siano i fantasmini, ovvero:
    $$distGhostToPowerPill >= 8$$

    In questo caso, viene imposto alla protagonista di smettere di muoversi e di tendere un'imboscata ai fantasmini vicino alla power pill. In questo modo, si evita che Ms Pacman mangi la power pill quando i fantasmini sono lontani e si massimizza, dunque, il beneficio della power pill stessa. Rispetto a Max1 sono stati scelti valori numerici diversi che hanno portato miglioramenti consistenti.

2. Si considera la situazione in cui c'è un fantasma vicino a Ms. Pacman. In questo caso, si impongono due comportamenti differenti:
    a. se la distanza di Ms. Pacman dalla power pill è abbastanza elevata, ovvero $distPacmanToPowerPill > 5$, e la distanza tra il fantasmino più

vicino a Ms. Pacman e la power pill più vicina è maggiore della distanza tra Ms. Pacman e la power pill stessa, ovvero $distGhostToPowerPill > distPacmanToPowerPill$, e ci sono almeno 3 fantasmini vicini a Ms. Pacman, allora viene utilizzata la versione *"Max"* dell'algoritmo con funzione di utility pari a :

$$utility = score + W_1 d_1 - isLifeLost - gameOver$$

in cui W_1è il peso che si vuole dare questo fattore, settato a *-5*, d_1è la distanza tra Ms. Pacman e la power pill, e gli altri 3 contributi hanno il solito significato.

b. in caso contrario, viene utilizzato l'algoritmo con funzione di utility che considera tutti i fattori, utilizzata già anche per Max1 e descritta nel paragrafo 3.2

3. Si considera il caso in cui esiste almeno un fantasma mangiabile. In questo caso, viene utilizzata la versione dell'algoritmo *"Max"* con funzione di utilità pari a:

$$utility = score + ghostCurrentEdibleScore - isLifeLost - gameOver$$

in cui ghostCurrentEdibleScore è l'incremento dello score che Ms. Pacman ottiene mangiando il fantasmino più vicino. Si ricorda, infatti, che questo valore aumenta a seconda del numero di fantasmini che la protagonista riesce a mangiare dopo aver mangiato una power pill, come descritto nel paragrafo 1.1. Rispetto alle precedenti versioni della funzione di utility, in cui l'utility veniva decrementata in base alla distanza dal fantasmino mangiabile, in questo controllore si aumenta l'utility dell'agente se questo ottiene punti mangiando il fantasmino. La differenza è sottile, ma permette a Ms. Pacman di percorrere la strada che maggiormente preferisce considerando tutte le altre variabili in gioco, come ad esempio il percorso che effettuerà il fantasmino e la posizione di eventuali altri fantasmini non edibili, mentre in precedenza l'unico obiettivo era quello di diminuire la distanza da esso così da tentare di mangiarlo.

4. Se non si trova in nessuna delle precedenti 3 situazioni, Ms. Pacman si trova in uno stato definito safe in quanto non a rischio di perdere la vita e, quindi, può raccogliere le pills e le power pills presenti nel labirinto per accumulare punti e progredire di livello.

La depth media raggiunta risulta essere, anche in questo caso, intorno ai 90. E' stato, però, possibile, dopo una serie di simulazioni eseguite in fase di test delle modifiche, settare la depth iniziale ad 80.

Max2 vs AggressiveGhosts

Miglior	Peggior	Media	Deviazione	Livello	Livello più	Livello

score	score	score	standard	più alto	basso	medio
174110	21680	97898	45065.29	12	1	6.64

Tabella 30: Max2 vs AggressiveGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
11050	2530	6015.2	1524.197	1	0	0.02

Tabella 31: StarterPacman vs AggressiveGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
32420	5470	18118.2	6687.846	2	0	0.88

Tabella 32: Max1 vs AggressiveGhosts

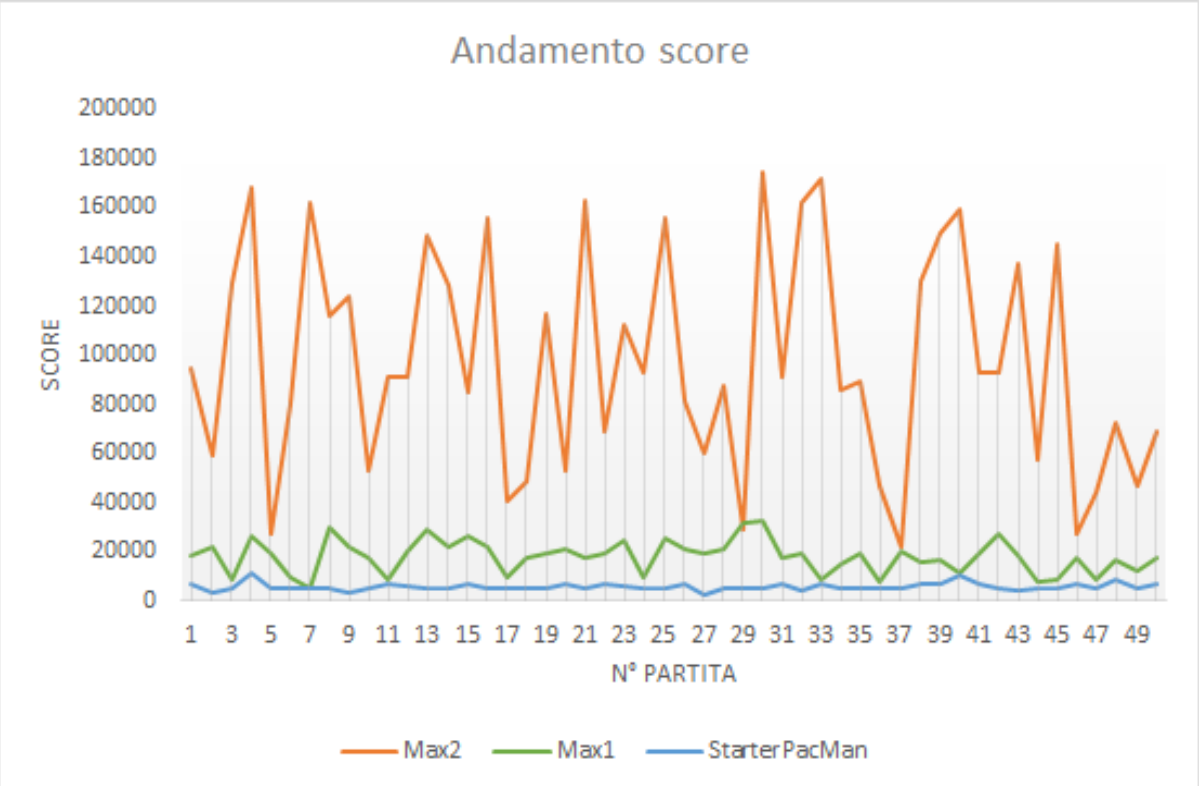


Figura 14: andamento dello score di Max2, Max1 e Starter PacMan nelle 50 partite contro gli AggressiveGhosts

Contro gli AggressiveGhosts il miglioramento è netto. La media degli score è, infatti, aumentata di quasi 5 volte (97898 contro 18118), così come è possibile notare un netto miglioramento nei livelli raggiunti. In particolare, viene sempre superato il livello 0 ed, inoltre, mediamente si arriva al livello 6.64.

In aggiunta, è stato definitivamente risolto anche il problema discusso nei precedenti controllori in cui Ms. Pacman tendeva più a salvaguardare la propria vita piuttosto che a raccogliere pillole per aumentare il proprio score, come testimoniano i punteggi raggiunti che sono indice del fatto che Ms. Pacman termina il livello completando il labirinto.
Un ruolo fondamentale è stato giocato dal considerare algoritmi con funzioni di utility molto più semplici, che tenessero conto solo di ciò che è utile considerare nelle differenti situazioni delle partite, semplificando il settaggio dei pesi di ognuno dei fattori.

Max2 vs RandomGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
54870	32570	41491.6	4550.348	8	6	7.1

Tabella 33: Max2 vs RandomGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
65560	13320	34301.8	13839.852	11	2	5.56

Tabella 34: StarterPacman vs RandomGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
41220	1750	14768.4	3290.730	6	0	1.92

Tabella 35: Max1 vs RandomGhosts

Il grafico di confronto è presente nella pagina seguente.

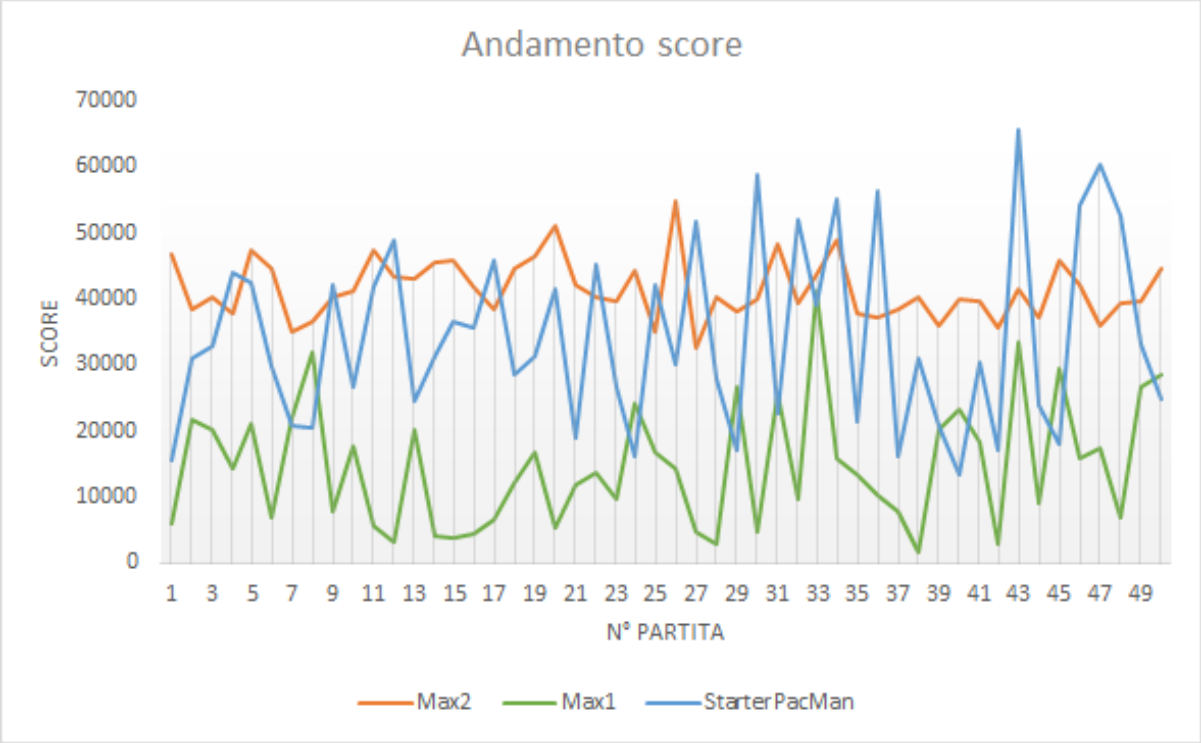


Figura 15: andamento dello score di Max2, Max1 e Starter PacMan nelle 50 partite contro i RandomGhosts

Anche contro i RandomGhosts si ha un netto miglioramento delle prestazioni. In particolare, si nota come venga sempre raggiunto mediamente il livello 7, rispetto a Max1 che raggiunge in media il livello 2. Inoltre, vi è un netto miglioramento anche della media dello score. Restano, comunque, validi i motivi per cui si hanno prestazioni peggiori contro i Random rispetto agli Aggressive. Infatti, anche in questo caso le mosse dei fantasmini considerate sono le mosse che li avvicinano a Ms. Pacman, ovvero le mosse che effettuano proprio gli AggressiveGhosts.

Si migliorano anche le prestazioni del controllore base StarterPacMan, come evidenziato dal test d'ipotesi *Wilcoxon* (ancora una volta suggerito dalla piattaforma Web *STAC*) con livello di significatività pari a 0.05

Statistic	p-value	Result
312	0.0016771655501775634	H0 is rejected

Tabella 36: Wilcoxon test tra Max2 e StarterPacMan per confronto risultati contro RandomGhosts

Max2 vs LegacyGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
80620	7880	36182.2	15373.66	8	0	3.18

Tabella 37: risultati Max2 vs LegacyGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
9990	1600	5175.2	1919.889	0	0	0

Tabella 38: risultati StarterPacman vs LegacyGhosts

Miglior score	Peggior score	Media score	Deviazione standard	Livello più alto	Livello più basso	Livello medio
25960	4320	11459	5374.243	2	0	0.52

Tabella 39: risultati Max1 vs LegacyGhosts

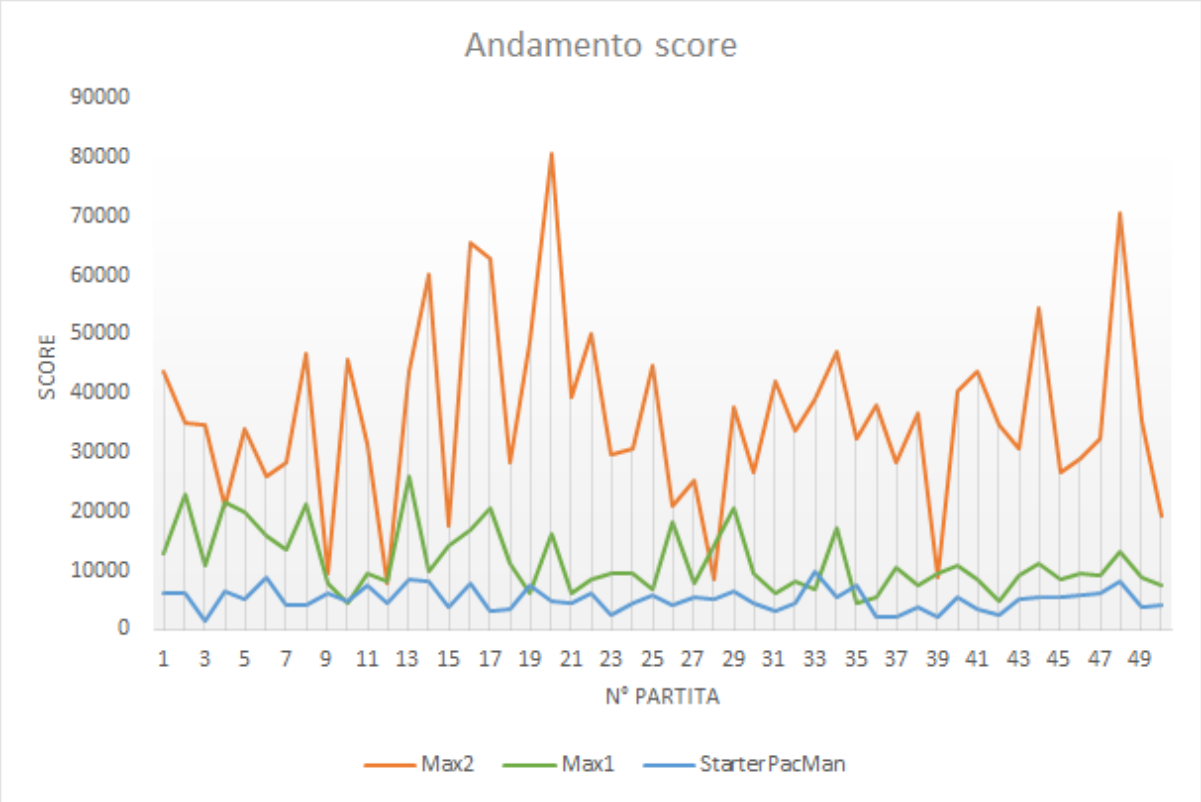


Figura 16: andamento dello score di Max2, Max1 e Starter PacMan

nelle 50 partite contro i LegacyGhosts

Anche contro i Legacy è possibile apprezzare il netto miglioramento in termini di score e livello medio raggiunti. La media score raggiunta è, infatti, 25000 punti superiore al precedente controllore e, mediamente, si raggiunge il livello 3, circa 2 livelli in più di Max1.
Ancora una volta, la presenza del fantasmino con comportamento randomico rende il controllore meno performante rispetto alle partite contro il team composto da fantasmini con solo atteggiamento aggressivo, in quanto non si riesce a prevedere le mosse casuali che il RandomGhost mette in atto.

Considerazioni finali Max2

E' stato ottenuto un netto miglioramento grazie, principalmente, alla depth e alle metaeuristiche considerate.

In particolare, l'essere riusciti ad imporre una profondità iniziale dell'albero pari ad 80 ha come conseguenza il fatto che in ogni situazione l'agente riesce ad esplorare come minimo alberi a profondità 80, mentre precedentemente in alcune situazioni fermava la sua esplorazione ad una profondità di 70.

L'impatto maggiore l'hanno, però, avuto le metaeuristiche. L'aver, infatti, identificato più accuratamente le distanze da considerare per la prima euristica (fermarsi vicino alla power pill per evitare di sprecare i suoi poteri) e l'aver definito due ulteriori funzioni di utility da utilizzare nelle situazioni racchiuse nella seconda (andare verso una power pill se ci sono dei fantasmini vicini) e nella terza (andare verso i fantasmini mangiabili se ce ne sono), ha permesso di migliorare i comportamenti di Ms. Pacman che, quindi, riesce in questo modo a raggiungere livelli avanzati e punteggi molto alti.

Capitolo 4: Conclusioni

Tra tutti i controllori proposti, il migliore in termini prestazionali è, senza dubbio, Max2, in quanto risulta essere l'agente che ha le migliori prestazioni contro tutti e 3 i team di fantasmini.
Questo risultato è frutto, principalmente, di 3 fattori:

- la profondità dell'albero di ricerca che si riesce a raggiungere;
- l'utilizzo di metaeuristiche adeguate e funzionali;
- l'utilizzo di funzioni di utility che abbiano pochi fattori di cui equilibrare i pesi.

In particolare, Max2:

- riesce a raggiungere una profondità pari, in media, a 90 grazie alla semplificazione di considerare come unica mossa possibile dei fantasmini quella di avvicinarsi a Ms. Pacman, che è quindi una mossa deterministica che rende MinMax privo dei rami MIN;
- utilizza 4 differenti metaeuristiche, che consentono di sfruttare differenti versioni dell'algoritmo Max (MinMax senza MIN) e di modificare, quindi, il comportamento dell'agente a seconda della situazione considerata;
- le differenti versioni dell'algoritmo Max differiscono, fondamentalmente, per la funzione di utility considerata. Si ottiene, in questo modo, un vantaggio notevole in termini di complessità di queste funzioni, in quanto risultano essere molto più semplici da interpretare. Di conseguenza, è più semplice calibrare i pesi dei singoli fattori da considerare e, quindi, permettere di imporre determinati comportamenti all'agente.

Ovviamente, l'idea di considerare una mossa deterministica per l'agente Min non dovrebbe portare a risultati migliori in generale. In questo caso, però, si tratta di un gioco con esigenze real time, in cui è perfettamente noto l'atteggiamento degli avversari Aggressive e, dunque, la semplificazione effettuata comporta il miglioramento dei risultati perché permette all'agente Max di prevedere correttamente la mossa degli avversari. Quando invece, questi ultimi hanno un comportamento non noto, la mossa deterministica considerata non corrisponderà alla mossa ottima dell'agente Min né a quella che effettivamente verrà intrapresa e, dunque, l'algoritmo non assicurerà a Max di intraprendere la scelta migliore, comportando un peggioramento dei risultati.
Si evidenziano, infatti, le differenti prestazioni che si hanno a seconda del team avversario contro cui Ms. Pacman si scontra.

Per avere, dunque, prestazioni migliori anche contro gli altri team, occorrerebbe, definire un controllore che generalizzi meglio il comportamento dei fantasmini. Come, però, hanno evidenziato i risultati ottenuti, ciò implicherebbe avere un albero di ricerca più complicato che, a causa dei vincoli real time, comporterebbe raggiungere una profondità più bassa e, quindi, probabilmente prestazioni mediamente peggiori di Max2.

Un’alternativa potrebbe essere implementare controllori ad hoc per ognuno dei differenti team. Il problema fondamentale, in questo caso, continuerebbe ad essere la riduzione della complessità dell’albero di ricerca, in quanto non è possibile prevedere una mossa deterministica per i fantasmini con comportamenti randomici.

In definitiva, il progetto descritto in questo elaborato ha permesso di apprezzare l’importanza di conoscere a fondo il funzionamento dell’algoritmo utilizzato rispetto al semplice utilizzo dello stesso. Una semplice applicazione dell’algoritmo senza le necessarie euristiche e metaeuristiche avrebbe indubbiamente portato a risultati insoddisfacenti.

Anche se, apportando dei miglioramenti al progetto, si sarebbero potuti ottenere risultati sicuramente migliori, l’obiettivo prefissato all’inizio del progetto è stato raggiunto ed i risultati ottenuti sono stati soddisfacenti.