# PROP: Hex

*Descripció dels algorismes d'intel·ligència artificial*

**Dades del projecte**

| | |
|---|---|
| Joc | Hex |
| Clúster | 7 |
| Grup | 3 |
| Versió | 3.1 |
| Data d'entrega | 18 de desembre de 2012 |

**Membres del grup**


| | |
|---|---|
| Javier Ferrer González | javier.ferrer.gonzalez@est.fib.upc.edu |
| Guillermo Girona San Miguel | guillermo.girona@est.fib.upc.edu |
| Marc Junyent Martín | marc.junyent@est.fib.upc.edu |
| Isaac Sánchez Barrera | isaac.sanchez.barrera@est.fib.upc.edu |

# Índex

# Característiques generals

Tots els algorismes d'intel·ligència artificial del joc tenen implementat un moviment d'obertura a la casella inferior esquerra de la més central. Utilitzant la numeració de la nostra implementació, és la casella (4, 2). Aquest és un bon moviment amb estratègia guanyadora, juntament amb la casella més central, com es pot llegir a [Yang 2003].

També implementen uns moviments d'obertura quan fan de jugador B, arran dels arbres de solucions de [Hayward 2003].

# Algorismes d'elecció directa

## Algorismes basats en la funció de cost de *QueenBee*

### IAHexPotencials

Està basat en el càlcul dels potencials de cada casella, fent servir la funció de cost de QueenBee descrita a [Rijswijck 2000]. Si $p$ és una casella que es vol avaluar, es defineix el seu cost

$$c(p) = \begin{cases} -V^* & \text{si } V(p) = 0 \\ V(p) & \text{altrament} \end{cases}$$

on $V(p)$ és el potencial de la casella segons l'algorisme *TwoDistance* de QueenBee i $V^*$ és el segon potencial mínim. Si hi ha dos o més mínims, com que $V(p) \geq 0$, tenim $c(p) = 0$ per a tota $p$ que tingui el potencial mínim.

Donat que aquest càlcul és prou bo, es pot utilitzar directament per escollir un moviment que intenta maximitzar les possibilitats de guanyar contra un jugador amb poca experiència.

# Algorismes amb una cerca basada en minimax

## Algorismes basats en l'aplicació de les regles *AND* i *OR*

### IAHexMiniMax

Aquesta estratègia consisteix en un minimax i utilitza els conceptes de connexions virtuals i semivirtuals i resistència del taulell en la funció d'avaluació.

Primer definim un grup de caselles, un grup és un conjunt de caselles adjacents que pertanyen al mateix jugador. Un grup es tracta com una casella sola.

Una connexió virtual ve donada per dos grups del mateix jugador que es poden connectar independentment del que faci el jugador contrari. Una connexió semivirtual ve donada per dos grups del mateix jugador que es poden connectar si és aquest jugador el que mou primer. Diem que una connexió o semiconnexió és de grau $x$ si els dos conjunts es poden

connectar al cap de *x* torns. La gran majoria de connexions virtuals es poden calcular a partir de les de grau inferior, utilitzant les regles AND i OR tal com explica [Anshelevich 2002].

Així doncs podríem dir que l'objectiu és buscar una connexió virtual entre els dos costats del tauler. Però el càlcul de les connexions virtuals de grau alt és car i complex, així que en el seu lloc proposem utilitzar connexions virtuals de grau 1 com a indicador de la qualitat d'una jugada. Augmentar el nombre de connexions virtuals és bo, sobretot al principi de la partida, però aquestes connexions han d'anar enfocades a unir els dos extrems del tauler i a més, cap al final de la partida, no són útils, en la nostra funció d'avaluació a mida que avancen els torns, les connexions virtuals tenen menys importància.

Considerem el taulell com un conjunt de resistències connectades en una malla. Si la casella pertany al jugador, té resistència 0, si està buida 1, i si és de l'enemic infinit. Amb aquest anàlisis, calculem dos paràmetres, el cost mínim del camí que uneix les dues bandes i la resistència total de banda a banda del tauler. El camí mínim es calcula mitjançant un Dijkstra, la resistència s'hauria de calcular mitjançant la solució d'un sistema lineal d'equacions, com això és molt car en fem una aproximació calculant resistències parcials en funció de les caselles colindants, aquest càlcul dona una aproximació prou bona i compleix la propietat que si no hi ha camí possible torna infinit i si un camí existeix torna 0.

Aquests tres paràmetres: nombre de connexions virtuals, camí mínim i resistència del taulell analitzats per als dos jugadors componen la base de la funció d'avaluació. Si veiem que el jugador contrari té un camí mínim petit es prioritza disminuir el seu camí mínim i resistència a augmentar els nostres (tàctica agressiva), en el cas contrari es té en compte no beneficiar el jugador contrari, però també intentem millorar la nostra posició (tàctica passiva).

## Algorismes basats en la funció d'avaluació de *QueenBee*

### Funció d'avaluació (comuna)

Aquesta funció d'avaluació està basada purament en l'algoritme Two-Distance i el càlcul de potencials. Utilitzant els resultats de [van Rijswijck 2000] el valor de la funció d'avaluació, $A$, es calcula:

$$A = M \cdot (P_a - P_b) + (Q_a - Q_b)$$

On $P_a$ és el potencial mínim del jugador en el tauler, $P_b$ és el potencial mínim del jugador contrari, i $Q_a$, $Q_b$són el nombre de caselles amb el mateix potencial per a cada jugador. $M$ és una quantitat prou gran perquè $Q_a$ i $Q_b$ no siguin rellevants més enllà d'en un cas d'empat, en el nostre cas $M = 100$.

### IAHexQueenBee

Aquesta estratègia és un minimax amb la funció d'avaluació del QueenBee descrita abans a més del cas especial del primer torn on s'utilitza un llibre d'obertures.

### IAHexNegaMonteScout

Aquesta estratègia és similar a la realitzada per *IAHexQueenBee*, però utilitza un algorisme *negaScout* en comptes d'un Minimax amb poda alfa-beta estàndard. En aquests algorismes, la funció que es vol minimitzar i la que es vol maximitzar són la mateixa però canviades de signe (algorisme *Negamax*) i amb una poda alfa-beta més estricta (poda alfa-beta d'un algorisme *SCOUT*). A més, només avalua un subconjunt de mida petita de totes les caselles disponibles, escollint-les amb una distribució uniforme discreta. La mida del subconjunt depèn de la quantitat de fitxes posades, i fa servir la fórmula següent:

$$Total_{a\ visitar} = \min\left\{\max\left\{\frac{Caselles_{restants}}{K \cdot \sqrt{Mida_{tauler}}}, 7\right\}, Caselles_{restants}\right\}$$

on $K$ és una constant ajustada a mà i que depèn de si és a profunditat 0 (abans de fer la crida recursiva) o més gran. En aquest cas, $K_0 = 0.7$ i, per a la resta, $K = 0.85$. L'elecció d'aquestes caselles aleatòries és la que dóna la nomenclatura de Monte Carlo a l'algorisme, utilitzat d'una manera molt similar a com va definir el mètode el matemàtic John von Neumann.

A més, aquesta implementació incrementa la seva profunditat màxima cada $2.3 \cdot Mida_{tauler}$ torns.

Aquests nombres tan poc formalitzats en realitat provoquen que en la majoria de casos l'estratègia jugui prou bé, perquè ajuda a desempatar entre moviments que a cert nivell de profunditat avaluen igual i perquè és poc probable que sempre s'avaluïn els pitjors moviments. I, el que és més important, que es pugui arribar a algun nivell de profunditat major sense trigar un temps excessiu, tot i que depèn en molta mesura de les característiques tècniques de l'ordinador on s'utilitza (incloent-hi la versió de la màquina virtual de Java i el sistema operatiu).

Per altra banda, es pot observar que hi ha vegades que la intel·ligència artificial té una connexió virtual i, en comptes d'aprofitar-la i col·locar-hi una fitxa que provoca el final de la partida, en col·loca una altra que o manté la connexió virtual o en crea una de nova (sempre i quan el contrincant no pugui guanyar afegint només una fitxa). Això té una solució fàcil en el bucle que comprova el nivell 0 d'abans de les crides recursives i que consisteix en retornar la casella si es detecta que fa guanyar la partida. Tanmateix, el fet d'afegir fitxes "inútils" ajuda a fer que el jugador humà no sàpiga com reacciona la intel·ligència artificial i que el joc sigui més divertit.

### IAHexSexSearch

La classe *IAHexSexSearchCtrl* també implementa un algorisme de cerca en l'arbre de moviments de tipus negaScout. La diferència principal amb *IAHexNegaMonteScout* i *IAHexQueenBee* és com s'escullen i ordenen els moviments a provar. Per fer-ho, utilitza la funció de cost que la intel·ligència artificial *IAHexPotencials* com a heurística, ordenant els possibles moviments de manera creixent segons el cost, on els de cost menor són els preferits.

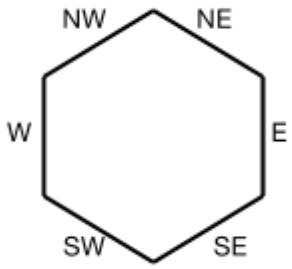
A l'hora de fer les podes i parar les crides recursives es tenen en compte un seguit de factors. Primer de tot, els moviments que tenen una avaluació de cost unes unitats més grans que el moviment amb cost menor no arriben a la profunditat màxima establerta, perquè en general seran moviments pitjors. Després, quan el cost de les successives crides supera el pressupost, també s'avalua estàticament pel mateix motiu. I finalment, dins de la recursió, si els moviments són massa costosos directament no s'avaluen per poder resoldre el problema més ràpidament.

De la mateixa manera, conforme va avançant el joc, es va ampliant la profunditat a què arriba l'algorisme per intentar millorar les jugades. En aquest cas, s'incrementa en un nivell la profunditat cada $1.3 \cdot Mida_{tauler}$ torns.

# Apèndix: Notacions i nomenclatura

Al llarg del document es fan servir un seguit de convenis per facilitar-ne la lectura. Aquests són:

- La mida indica la quantitat total de files i columnes del tauler
- Estem suposant sempre un tauler de mida 7
- Les caselles estan connectades amb d'altres caselles pels costats NW, NE, E, SE, SW, W, seguint el gràfic següent:




- Els índexs comencen sempre per zero (0)
- Les caselles s'escriuen com (*fila*, *columna*)
- El primer jugador que mou és el jugador A i el segon és el B.

# Bibliografia


- [Yang 2003] Yang, J., Liao, S., & Pawlak, M. (2003). New winning and losing positions for 7× 7 Hex. *Computers and games*, 230-248.
- [van Rijswijck 2000] van Rijswijck, J. (2000). Are Bees Better Than Fruitflies?. *Advances in Artificial Intelligence*, 13-25.
- [Hayward 2003] Hayward, R., Björnsson, Y., Johanson, M., Kan, M., Po, N., & van Rijswijck, J. (2003). Solving 7× 7 Hex: Virtual connections and game-state reduction. *Advances in Computer Games*, *263*, 261-278.
- [Anshelevich 2002] Anshelevich, Vadim V. "A hierarchical approach to computer Hex." *Artificial Intelligence* 134.1 (2002): 101-120.


Els moviments d'obertura obtinguts de la recerca de [Hayward 2003] són a http://webdocs.cs.ualberta.ca/~hayward/hex7trees/