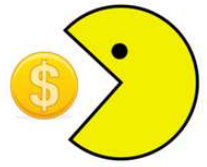


# **PACMAN RETURNS**



**Descripció del projecte**

**Projecte de Programació**  
**Moisés Saus Ten**  
**Oscar Galera i Alfaro**



## **Índex**

<b>Tema</b>	<b>Pag</b>
Introducció.....	3
Aspectes Generals.....	3
<b>Algorítmica</b>	
Camí més curt.....	5
Camí per Maximitzar distància.....	7
Creació de Laberints aleatoris.....	8
Validació de Laberints.....	10
<b>Comportament de diferents elements movibles</b>	
Fantasma 1.....	12
Fantasma 2 .....	12
Fantasma 3.....	13
Ítems.....	14



## Introducció




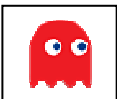
---

Mitja Europa es troba en crisi, crisi que també fa trontollar les butxaques d'en Pacman. Per això en Pacman ha decidit emplenar-se les butxaques de monedes i exiliar-se del país. Però els seus arxienemics també es troben en la mateixa situació i hauran de competir per aconseguir el màxim de diners possible i fugir del país.

## Aspectes Generals

---

- **Personatges:**

-  **PACMAN:** És el personatge que podrà moure el jugador.
-  **FANTASMA1:** És un fantasma que no te el graduat escolar, es va quedar atrapat en les reformes de educació del ministre Wert, i no sap llegir ni escriure ni sumar monedes. És el fantasma menys intel·ligent.
-  **FANTASMA2:** És un fantasma intel·ligent, va deixar la universitat perquè va deixar de rebre beques. Sap buscar-se la vida i pot arribar a agafar un bon grapat de monedes si en Pacman es despista.
-  **FANTASMA3:** És el fantasma més viu dels tres, no te estudis, però te una llarga trajectòria política on ha après a agafar calers d'on sigui. Ha estat imputat en alguns casos de corrupció, però encara ningú a aconseguit demostrar res i seguirà fent de les seves. En Pacman haurà d'estar molt atent si no vol que Fantasma3 li tregui totes les monedes.

- **Descripció:**

El joc consta de 5 nivells. En cada nivell s'han de superar 3 laberints, on en cada laberint hi haurà un fantasma diferent, fantasma1, fantasma2 i fantasma3 respectivament, i a cada nivell s'augmentarà la mida dels laberints.

Nivell 1: Laberint de 10x10

Nivell 2: Laberint de 15X15

Nivell 3: Laberint de 20x20



Nivell 4: Laberint de 25x25

Nivell 5: Laberint de 30x30




Per superar cada laberint s'ha d'obtenir una puntuació superior a la del enemic. La puntuació s'obté agafant monedes.



Existeixen dos tipus de monedes:

-  Monedes normals: Sumen 10 punts a la puntuació.
-  Monedes extra: Sumen 30 punts a la puntuació.

Durant el transcurs de les partides apareixeran en el laberint objectes. Aquests objectes poden ser:

-  **Patins:** Atorga el doble de velocitat al personatge que aconseguixi agafar-los.
-  **X2:** El personatge que l'agafi doblarà el valor de les monedes que vagi recollint.
-  **Mongeta:** Qui tingui la mongeta i aconseguixi atrapar al seu oponent obtindrà tots els punts del seu enemic.

Els objectes apareixeran en el Laberint cada 30% del total de monedes consumides, és a dir, si el numero inicial de monedes del laberint és 100, els objectes sortiran: el primer quan quedin 70, el segon quan quedin 40, i el tercer quan quedin 10, només hi haurà un objecte al mateix temps en el laberint.

Els seus efectes no són per sempre, només duren 15 segons. Aquests objectes aniran passejant per el laberint, es mouen una mica més lents que en Pacman i els fantasmes, però no volen ser atrapats, així que faran tot el possible per fugir si algú els persegueix.

Un cop exhaurides totes les monedes apareixerà una porta de sortida en un punt aleatori del laberint. Qui tingui més puntuació haurà de fugir del laberint, si ho aconseguix serà el guanyador. I qui tingui menys puntuació haurà d'enxampar al seu enemic abans de que aconseguixi sortir, aconseguint d'aquesta manera robar-li tots els punts.

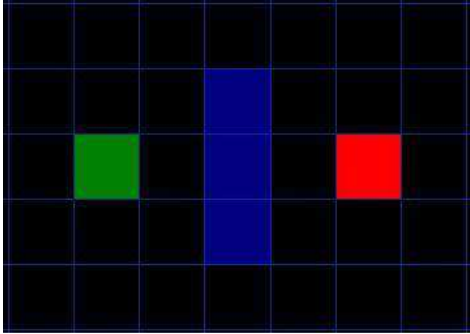


## Algorítmica

- **Algorisme de cerca del camí més curt**

Per a les cerques del camí més curt s'ha utilitzat l'algorisme AStar.

El laberint esta representat per un matriu bidimensional, on cada posició representa una cel·la, així que la nostra area de cerca esta dividida per cel·les.



Ara imaginem que volem anar del punt verd al punt vermell, i per el camí poden haver obstacles que ens impedeixen el pas.

Comencem la cerca:

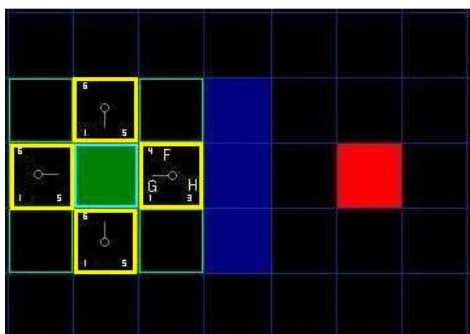
- 1- El primer que fem és afegir la primera casella (la verda) a una llista (una llista que utilitzarem per saber quines posicions haurem de tindre en compte, posicions que formaran part del nostre camí o no)
- 2- Comprovar les caselles adjacents a les quals podem accedir, aquestes caselles les puntuem i les afegim també a la nostra llista, i per cada una d'aquestes caselles que afegim guardem el punt de inici com el seu pare.

La heurística seguida per puntuar-les bé determinada per la funció  $F = G + H$ . On:

- $G$ = El cost del moviment per anar des de el punt inicial (verd) a una certa cel·la, seguint el camí generat per arribar fins a aquesta certa cel·la. El cost que utilitzem per anar de casella a casella es 1, ja que no tindrem en compte els moviments en diagonal.
- $H$ = El cost estimat per anar des de la cel·la que estem avaluant fins al nostre destí. En el nostre cas utilitzem la distancia "Manhattan". És a dir, cada casella esta determinada per una coordenada 'x' i per una coordenada 'y', llavors la distancia "Manhattan" =  $\text{abs}(\text{inici.x} - \text{desti.x}) + \text{abs}(\text{inici.y} - \text{desti.y})$ . És una estimació de la distancia que hi ha fins al nostre objectiu.

- 3- Marquem la casella inicial com a "processada" per no tornar-la a avaluar i la traiem de la llista.

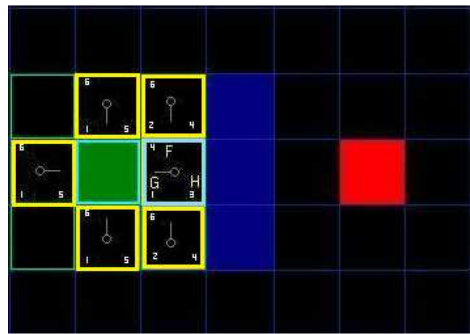
Per tant després d'efectuar aquests 3 primers passos tindrem la següent situació:



On, la casella remarcada amb verd clar seria la casella processada, i les caselles de color groc són les caselles que tenim puntuades (amb F,G i H) i a la llista, i on cada casella apunta al seu pare.



- 4- Per continuar la cerca simplement agafem la casella amb la F més petita de tots els que tenim a la llista i la marquem com a processada.
- 5- Es comprova totes les caselles adjacents, ignorant totes aquelles que ja hagin estat processades o que siguin intransitables.
  - Si no estan a la llista, els afegim i fem com el seu pare la casella actual.
  - Si estan a la llista, comprovem que la G d'aquesta casella sigui més petita que la G que estem utilitzant per arribar fins a ella.
    - Si la G de la casella adjacent que estem comprovant és més petita, no fem res.
    - Altrament Si la G del nou camí es més petita, canviem el pare de la casella adjacent i li assignem com a pare la casella en la que ens trobem actualment.



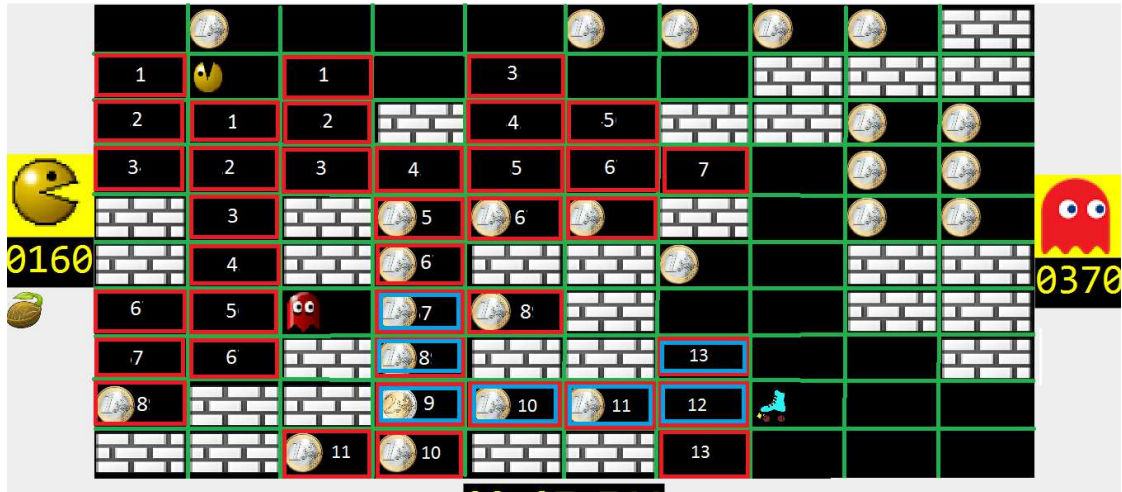
- 6- Així successivament fins que afegim la nostra casella objectiu a la nostra llista, voldrà dir que hem trobat el nostre camí.
- 7- Per aconseguir s'agafa la casella destí i obtenim el seu pare, i així successivament fins que el pare = null (serà la casella de inici).

Cal a dir que els temps de càlculs de l'algorisme són força bons. Els temps de càlcul obtinguts durant una partida es poden observar en el Log del projecte.



- **Algorisme de cerca d'un camí que maximitzi la distància des de un punt respecte de un altre.**

Imaginem el següent cas: En Pacman te mongeta i el fantasma se sent amenaçat, llavors es calcula amb un algorisme de backtracking quin és el camí de  $n$  caselles que es distancien més de la posició d'en Pacman. On  $n$  es la profunditat del backtracking. Durant el joc la profunditat és de 7.



	Area que es te en compte. ( Les caselles processades depenen de la poda de l'arbre de probabilitats)
	Solució al problema
0..n	Distància del camí mínim al punt del qual volem maximitzar la distància. En aquest cas en Pacman.

En el algorisme es busca la millor solució podant l'arbre de possibilitats.

L'algorisme seria:

- 1- Es generen els candidats a la casella actual: Totes aquelles caselles a les que podem accedir i que no hagin estat processades i a cada una d'aquestes caselles candidates anotem la distància mínima ( calculada utilitzant l'algorisme AStar) que hi ha entre la casella candidata i la casella de la qual volem maximitzar la distància.
- 2- Mentre quedin candidats
  - 2.1- Obtenim el la casella candidata **amb la distància més gran**
  - 2.2- Si la distància des de el candidat actual + els passos que hem queden per fer  $>$  que la millor distància aconseguida fins al moment. Llavors es que pot ser una millor solució. Altrament segur que no trobaré una millor solució que la aconseguida fins al moment.
    - 2.2.1 – Es marca la casella com a processada
    - 2.2.2 – Si no he arribat als  $n$  moviments
      - 2.2.2.2 – Crida recursiva amb el candidat actual
    - 2.2.3- Si he arribat als  $n$  moviments i la distància màxima obtinguda és més gran que la ultima millor distància màxima obtinguda, llavors hem guardo la solució actual com a millor solució.
    - 2.2.4 – Es desmarca la casella processada i es deixa com a no processada.

Fi mentre.

- 3- Al sortir de la recursivitat ja tenim el camí que maximitza la distància calculat.



A cada crida recursiva s'utilitza l'algorisme AStar per donar un pes determinat (Distància del camí mínim) a les caselles que processem, això permet fer una poda considerable a l'arbre de probabilitats del backtracking. El recorregut de l'arbre es fa en profunditat, no per nivells.

Els temps de càlcul obtinguts durant una partida es poden observar en el Log del projecte.

- **Creació de laberints aleatoris.**

En primer lloc calculem el nombre de monedes que ha de tenir el laberint i creem un vector de la mateixa mida que el laberint a on anirem afegint els "candidats\*".

Omplim el tauler de parets i afegim en pacman a la posició [0, 0], el fantasma a la posició [costat-1, costat-1] i afegim aquests dos punts al vector de candidats.

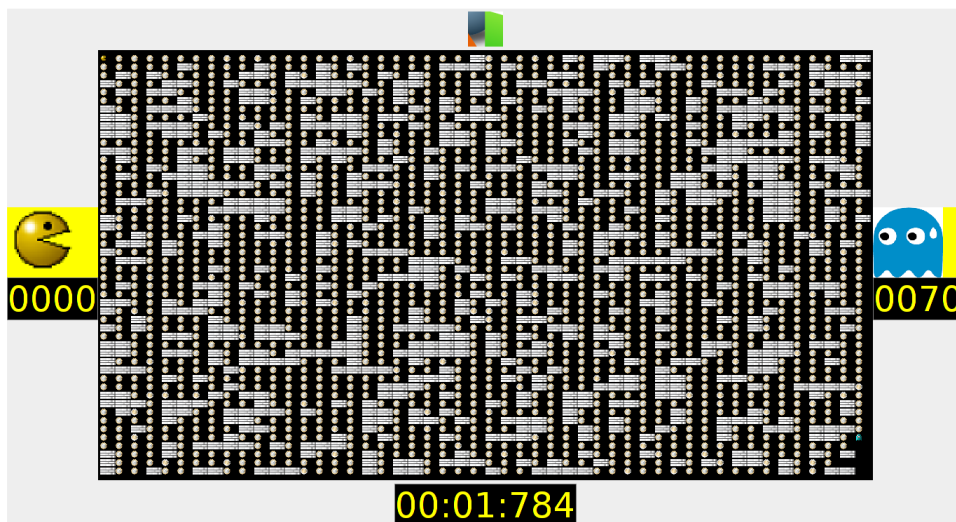
Ara toca anar sorteiant posicions a on afegirem monedes mentre no superem o igualem el nombre de monedes que hem calculat que havia de tenir el laberint. Per cada posició sortejada busquem dins del vector de candidats la posició més pròxima i traça'm un camí entre els dos punts, tots els punts entre origen i destí es van afegint al vector de candidats.

D'aquesta manera podem assegurar que com a molt hi hauran 2 components en el laberint finalment com el nostre laberint per considerar-lo vàlid només pot tenir una component repetim aquest procés mentre el laberint no sigui vàlid amb ajuda del validador de laberints.

Exemples del funcionament de l'algorisme.

Mostra de 5 laberints generats de forma aleatòria

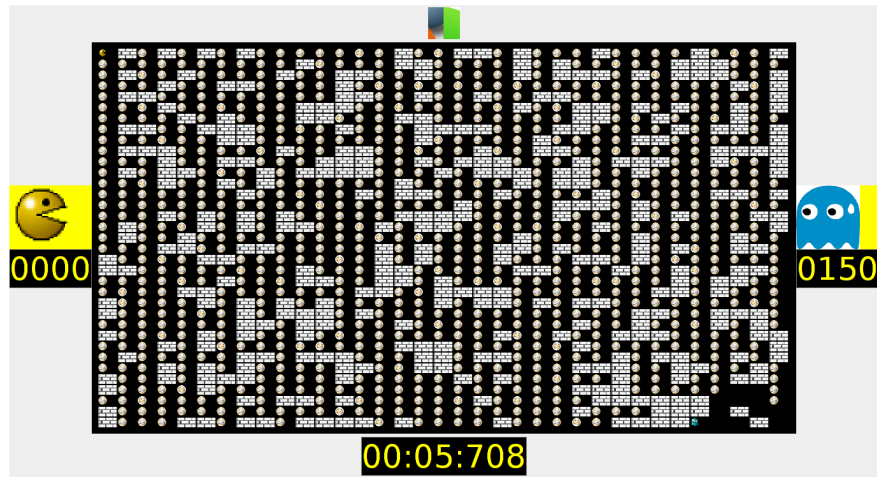
50x50 (Tots 5 laberints vàlids al primer intent)



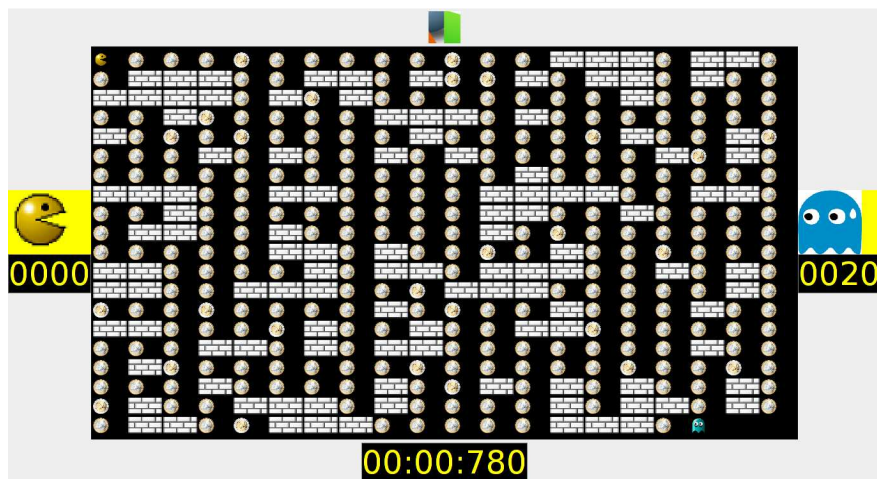




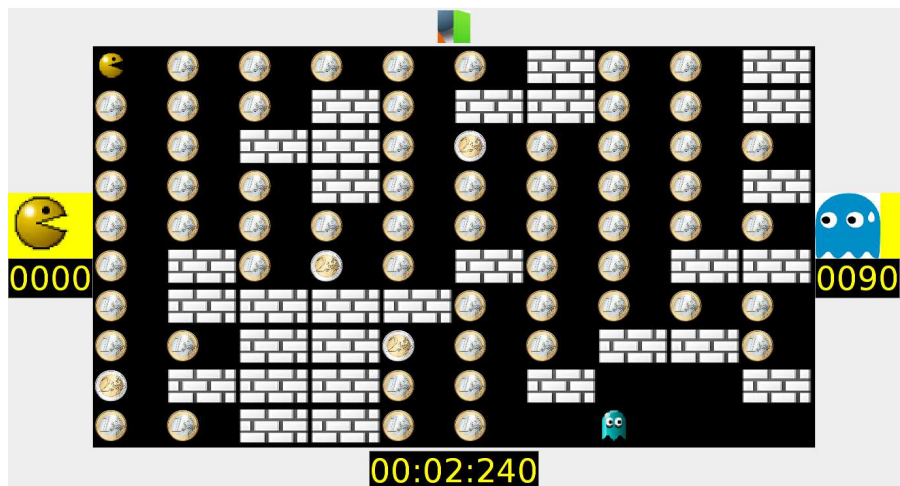
35x35 (Tots 5 laberints vàlids al primer intent)

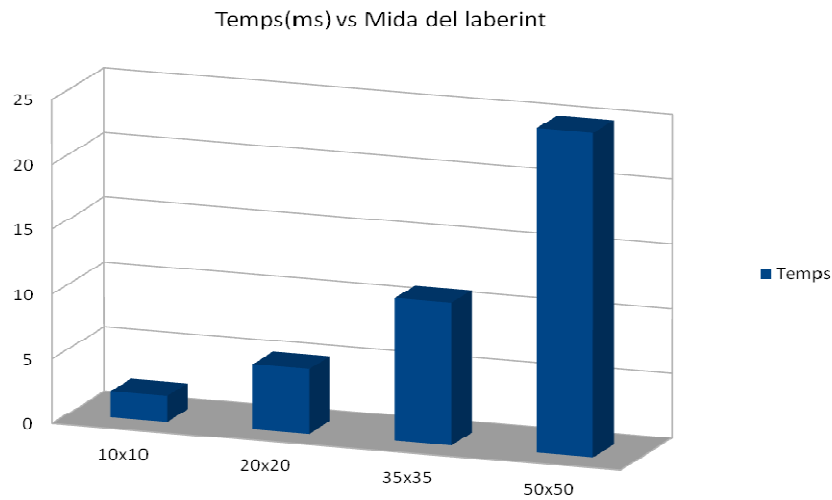


20x20 (Tots 5 laberints vàlids al primer intent)



10x10 (Tots 5 laberints vàlids al primer intent)





Beneficis del algorisme.

Intuïtiu.

Alta probabilitat de obtenir un laberint vàlid al primer intent.

Limitacions del algorisme

Pot haver de repetir-se l'algorisme més d'un cop per obtenir un laberint vàlid.

Un tauler de costat x costat pot no tenir el mateix nombre de monedes.

\*Per candidat entenem tot punt que no és paret.

Característiques rellevants del ordinador

CPU → 3.6 GHz (i7 4770)

RAM → 20 GB (DDR3)

- **Validació de Laberints**

Definició de laberint vàlid → tot laberint connex amb un únic pacman i un únic fantasma.

Descripció del algorisme

Crear una partició per el laberint.

fem una lectura d'esquerra a dreta i de dalt a baix casella a casella on s'ha de mirar que conté cada una.

Si conté una paret o no conté res continuem avaluant.

Altrament si conté un pacman mirem de no haver-lo trobat anteriorment del contrari



sabem que no és vàlid.

Altrament si conté un fantasma mirem de no haver-lo trobat anteriorment del contrari sabem que no és vàlid.

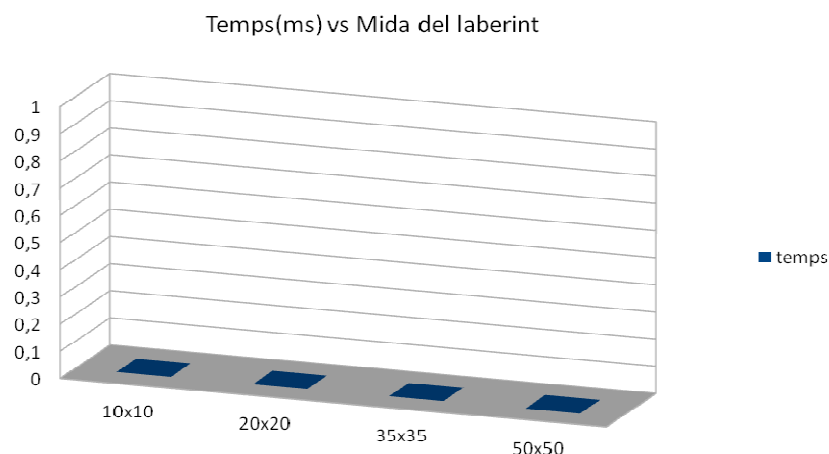
En cas de haver trobat un pacman un fantasma o una moneda (ja sigui normal o extra) mirem que a l'esquerra no hi hagi paret i en aquest cas afegim la casella a la partició de l'esquerra, altrament, en cas que no hi hagi paret a dalt l'afegim a la part de a dalt i si hi ha paret a esquerra i a dalt cal crear una nova subpartició “sempre que es crea una nova subpartició es mira el costat dret i a baix perquè si hi ha paret segur que el laberint no és vàlid”.

Avantatges de l'algorisme  
detecta si el laberint és vàlid lo abans possible.

Inconvenients de l'algorisme  
pot semblar complicat d'entendre a priori.

Cost de l'algorisme  
El cas pitjor es produeix quan el laberint és vàlid fins l'última casella.  
Cost  $O(n^2)$

Característiques rellevants del ordinador  
CPU → 3.6 GHz (i7 4770)  
RAM → 20 GB (DDR3)





### *Comportament dels diferents elements movibles*

---



- **Fantasma 1**

Es mou aleatòriament. És a dir només hi ha quatre possibilitats: amunt, avall, esquerra i dreta. Escull una possibilitat aleatòriament i es desplaça a aquesta posició.



- **Fantasma 2**

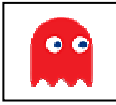
1- Valora les quatre possibilitats amunt, avall, esquerra, dreta.

La valoració ve regida per:

- Recompte de la puntuació que pot aconseguir si escull una direcció determinada fins a la pròxima paret / per el numero de passos.
- Si te mongeta i la direcció que esta avaluant es troba en Pacman i aquest te punts el interès en aquesta direcció determinada és màxim.
- Si la direcció que esta avaluant hi ha un item, el valor d'aquesta direcció també serà màxim.
- Si es troba al final de la partida i la direcció que s'està avaluant es troba la sortida, el interès per aquesta també serà màxim.

2- Decideix la direcció que vol prendre:

- 2.1- Si existeix una direcció amb més interès que les altres, escull dita direcció i empila el seu moviment, per saber per on ha vingut.
- 2.2- Si el interès es 0, és a dir no ha trobat res interessant en cap direcció
  - 2.2.1- Si te moviments en el històric de moviments, llavors torna enrere.
  - 2.2.2- Si no te moviments en el històric, llavors escull una direcció aleatòriament, i es guarda la seva decisió en el històric per saber per on ha vingut.
- 2.3- Si existeix empat en els interessos la prioritat entre les direccions serà: dreta, esquerra, amunt, avall i es guarda la seva decisió en el històric per saber per on ha vingut.



- **Fantasma 3**

Fantasma 3 té tres modes:

- Fugir: En aquest mode es calcula un camí de 6 moviments que maximitza la distància amb el perseguidor, on 6+1 és la profunditat del BackTracking
- Seguiment: Es persegueix un objectiu movable. Es calcula el camí mínim amb AStar a cada moviment i només es porta a terme el primer pas d'aquest camí calculat. Ja que el objectiu és movable i no és eficaç seguir un camí precalculat fins a un punt determinat.
- Navegació: Es persegueix un objectiu estàtic. Es calcula una sola vegada el camí mínim amb AStar i es segueixen els passos fins a aquest objectiu.

El seu raonament és el següent:

1- Valora la situació:

Calcula la distància mínima amb en Pacman.

1.1 - Si es troba en mode navegació:

- 1.1.1 Si no queden monedes i esta perdent, canviarà a mode seguiment per perseguir en Pacman.
- 1.1.2 Si en Pacman te mongeta i es troba a una distància mínima de 15 moviments, canviarà a mode fugir.
- 1.1.3 Si hi ha algun ítem especial a una distància mínima  $< 8$ , llavors canviarà a mode seguiment a perseguir el ítem.
- 1.1.4 Si té mongeta i la puntuació d'en Pacman és més gran que el 30% de la puntuació del fantasma 3, llavors canviarà a mode seguiment per perseguir en Pacman.
- 1.1.5 Altrament segueix en mode navegació i:
  - 1.1.5.1 Si no te cap objectiu o el objectiu al que es dirigia a desaparegut llavors calcula el camí mínim a totes les monedes ( si troba un camí mínim de un sol moviment parará de buscar) i seguirà aquest camí pre-calculat.
  - 1.1.5.2 Altrament segueix el camí que ja tenia calculat.

1.2 – Si es troba en mode Fugir

1.2.1- Si en Pacman te mongeta i es troba a una distància  $< 15$

1.2.1.1 Si queden moviments per fer de la ruta calculada per el backtracking, llavors faig moviment

1.2.1.2 Si no queden moviments calcula una nova ruta i fa moviment.

1.2.2- Altrament canvia a mode navegació.

1.3 – Si es troba en mode Seguiment

1.3.1- Si en Pacman te mongeta i es troba a una distància  $< 15$  llavors canviarà a mode fugir

1.3.2 – Si esta perseguint en Pacman perquè fantasma 3 té mongeta i ja ha atrapat en Pacman o s'han pasat els efectes de la mongeta, llavors canviarà a mode navegació.

1.3.3 – Si esta perseguint un ítem i l'ítem se l'ha menjat en Pacman, llavors



canviarà a mode navegació.

1.3.4 – Altrament segueix en mode seguiment.



Ítems

El Ítems es mouen pseudoaleatoriament.

- 1- Càlcul de les distàncies mínimes respecte en Pacman, i distància mínima respecte el fantasma.
- 2- Si alguna distància  $< 10$  ( en el cas de que les dos distàncies siguin inferiors a 10, fugirà del Personatge que estigui mes a prop).
  - 2.1- Si ja estava fugint
    - 2.1.1 - Si tinc ruta calculada segueix ruta.
    - 2.1.2 – Altrament torno a calcular ruta de fugida.
  - 2.2- Altrament calcula ruta de fugida.
- 3- Altrament
  - 3.1 Si no té moviments per fer, llavors calcula el camí mínim fins a alguna posició aleatòria del laberint i segueix la ruta calculada.
  - 3.2 Si te moviments per fer llavors segueix la ruta calculada.