PACMAN - ARQUITETURA FINAL

Na nossa arquitetura o pacman tem 3 fases distintas: **Verificar se está a ser seguido; Definir o target; Procurar pelo melhor caminho para o mesmo;**

FOLLOWED:

O pacman tem um atributo (booleano) que indica se este está a ser seguido ou não. Este valor é determinado com base na quantidade de ghosts "não zombies" perto do pacman no momento, a quantidade de ghosts existentes no mapa e a quantidade de energias no mapa.

```
def being_followed(self, ghos_pos, mapa):
    ghost_pos = [(g[0][0], g[0][1]) for g in ghos_pos if g[1] == False]
    self.buffer = list(map(lambda x: m_distance(self.pos,x) , ghost_pos))
    following = [b for b in self.buffer if b <=4]
    if len(ghos_pos) < 3:
        self.followed = (len(following) >= 1)
    else:
        if len(mapa._energy) > mapa._initial_energies_count/2:
            self.followed = (len(following) >= 2)
        else:
            self.followed = (len(following) >= 1)
```

PACMAN - ARQUITETURA FINAL

FIND NEXT TARGET:

Como base no atributo followed e outros fatores, o pacman define qual será o seu próximo target:

- Se o pacman não estiver em modo super e se estiver a ser seguido, tenta ir ao boost (se este estiver suficientemente perto e seja seguido por um número de fantasmas que justifique o caminho a percorrer);
- Caso esteja em modo super e existam fantasmas perto que o pacman tenha tempo de comer, o target passa o ghost mais perto;
- Se nenhum dos casos anteriores acontecer, este tenta comer as “melhores” energias, isto é, energias tendo em conta a posição dos ghosts no mapa, caso estejamos num jogo com fantasmas.

Caso existam boosts por comer depois de todas as energias acabarem, o pacman vai comer estes boosts. Isto começa a ser raro de acontecer uma vez que, ao longo do tempo, com a evolução do projeto, temos uma arquitetura que permite fazer decisões inteligentes na altura de comer ou não um boost.

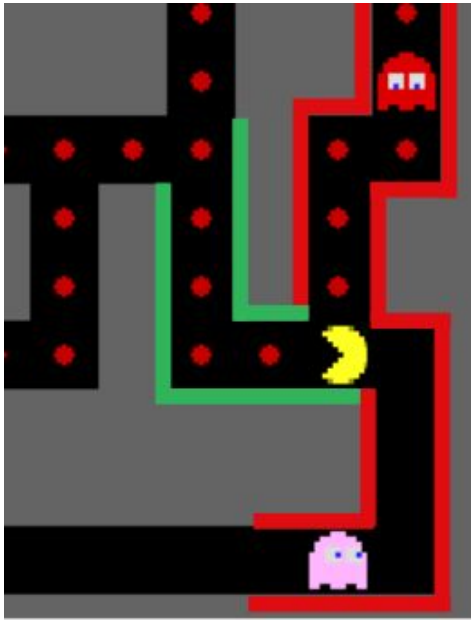
PACMAN - ARQUITETURA FINAL

SEARCH (1 / 2):

Na pesquisa por caminhos em relação ao target, está a ser utilizada uma pesquisa em árvore uniforme.

Na atribuição de custos para os nós existem vários critérios sendo que um deles é o critério de que os nós que fiquem no shortest path de um qualquer ghost até ao pacman têm mais custo. Na figura ao lado, os caminhos a vermelho têm mais custo uma vez que são ambos caminhos que os fantasmas utilizam para chegarem mais rápido ao pacman.

Se um determinado nó estiver num raio muito próximo de um fantasma, este também terá custo mais elevado. A distância ao fantasma é calculada através da distância de manhattan, contudo, se esta for inferior a um limiar (muito próximo do pacman), usamos pesquisa em árvore para validar a distância.



Shortest path dos ghosts ao pacman

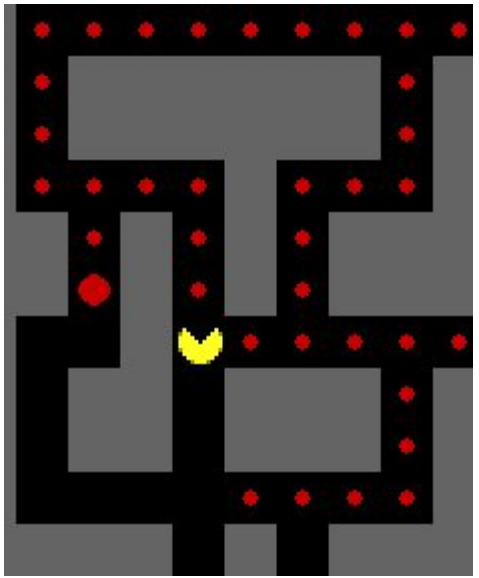
Daniel Nunes (84793) | Rafael Direito (84921)

PACMAN - ARQUITETURA FINAL

SEARCH (2 / 2):

Caso o pacman não encontre qualquer solução para um determinado target ou caso este chegue perto dos 100 ms para enviar a key escolhida, então o pacman tenta escolher ir para a posição que lhe dá maiores garantias de que este pode sobreviver mais tempo e aproximar-se do target.

É ainda de referir que os nós que sejam boosts têm custo mais elevado caso não existam fantasmas muito perto e ainda exista uma grande quantidade de energias no mapa, o que se pode confirmar na figura ao lado.



Gestão inteligente dos boosts

Daniel Nunes (84793) | Rafael Direito (84921)