SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RACUNARSTVA

Dokumentacija projektnog zadatak iz predmeta

INTELIGENTNI MULTIAGENTSKI SUSTAVI

**Pacman**

Autor:

Josip Stuhli, 0036441750

Zagreb, svibanj 2012.

**Sadržaj**

[IMS projekt: Pacman 3](#_Toc327998583)

[Model svijeta 3](#_Toc327998584)

[Model autonomnog agenta Pacman 3](#_Toc327998585)

[Politika agenta Pacman 4](#_Toc327998586)

[Vidljivost svijeta 4](#_Toc327998587)

[Funkcija korisnosti 4](#_Toc327998588)

[Postupak pretraživanja labirinta agenta Pacman 6](#_Toc327998589)

[Strategija agenta Pacman 7](#_Toc327998590)

[Agent duh 8](#_Toc327998591)

[Korištena programska potpora 8](#_Toc327998592)

[Pokretanje simulacije 9](#_Toc327998593)

[Rezultati simulacije 9](#_Toc327998594)

IMS projekt: Pacman

Projektni zadatak bio je izraditi modele autonomnih agenata u popularnoj igri Pacman. Bilo je potrebno definirati strategije svih agenata, te algoritamski opisati njihovo ponašanje. Također bilo je potrebno teorijski i programski ostvariti igru koja može igrati sama protiv sebe (tako da agentom Pacman ne upravlja igrač)

Model svijeta

Svijet se u igri sastoji od 2D matrice polja. Polje može biti zid, hrana ili prazan prostor. Ukoliko polje nije zid, tada se u njemu može nalaziti agent. Isto polje u isto vrijeme može okupirati više agenata (npr. dva ili više duhova). Agenti se mogu kretati iz jednog polja i drugo, ukoliko polje u koje agent prelazi nije zid. Prilikom prolaska preko polja koje sadrži hranu, agent Pacman hranu pojede, pa to polje tada postaje prazno. Agenti duhovi nemaju utjecaja na polje sa hranom. Agenti imaju ograničenu vidljivost svijeta, odnosno mogu vidjeti strukturu samo onih polja koja su od njih udaljena za manje od VIDLJIVOST\_SVIJETA polja po x i y koordinati. Agent vidi polje ukoliko vrijedi , te .

Vidljivi dio okoline je agentima potpuno dostupan, agent ima potpun i točan pristup informaciji o stanju okoline. Kretanje agenata unutar svijeta je deterministično - agent uvijek zna posljedicu svoje akcije (akcija „Idi lijevo“ uvijek rezultira pomakom u polje lijevo). Okolina je statička – ne mijenja se, osim kao posljedica djelovanja agenta Pacman ili agenta duh. Okolina je diskretna, odnosno postoji konačan broj polja i akcija koje se mogu izvršiti.

Model autonomnog agenta Pacman

Autonomni agent Pacman je racionalni agent, odnosno on nastoji optimizirati određenu mjeru performanse (broj ostvarenih bodova u igri). Prema spoznaji o okolini, autonomni agent je reaktivan. Prema vođenju je refleksivan, odnosno vodi se isključivo prema trenutnom stanju svijeta. Prema postojanju memorije pripada u skupinu histereznih, odnosno agent posjeduje memoriju (agent pamti kartu svijeta, te lokaciju hrane na toj mapi). Prema načinu odabira akcija pripada u skupinu refleksivnih agenata.

Agent pri svakom potezu ažurira svoju lokalnu memoriju sa informacijama iz trenutno lokalno vidljivog svijeta (upisuje informacije o novoj vidljivoj hrani). Potom dohvati popis svih mogućih akcija, te za svaku akciju izračuna stanje koje slijedi iz te akcije, te ocijeni to stanje sa funkcijom korisnosti. Agent zatim odabire onu akciju koja ima najveću korisnost. Ukoliko više akcija ima jednako veliku korisnost, tada se stohastički odabire jedna od tih akcija.

Politika agenta Pacman

Politika agenta može se opisati funkcijom

Gdje je opažanje u trenutku t, a akcija koja je odabrana u trenutku t. Iz funkcije se može vidjeti da Pacman reaktivan, ali isto tako da koristi informacije iz prošlih opažanja, ne bi li optimalno planirao svoj put u potrazi za hranom.

Vidljivost svijeta

Vidljivost agenta je lokalna, te on vidi samo polja u svojoj neposrednoj blizini, odnosno za polja za koja vrijedi:

Gdje je VIDLJIVOST\_SVIJETA konstanta koja je definirana unutar igre.

Funkcija korisnosti

Funkcija korisnosti (eng. *Utility function*) govori koliku će isplatu agent primiti ukoliko se nađe u određenom stanju. Funkcija je definirana kao najmanja udaljenost do svakog polja sa hranom. Koristi se koncept umanjenih nagrada, odnosno funkcija sumira vrijednost hrane na svakoj poziciji umanjenu za određeni faktor.

GameScore(s) – vraća broj bodova Pacmana u stanju s

POLJA – skup svih polja

Alfa – faktor umanjenja

Dist(p) – daje najmanju udaljenost agenta do polja p

Special(f) – funkcija koja dodaje (ili oduzima) više bodova posebnim situacijama

Posebne situacije:

* Pacman se nalazi pored polja u kojem se nalazi izolirana hrana (daj više bodova ne bi li Pacman otišao u to polje, pojeo hranu odmah, da se kasnije ne mora vraćati)
* Na polju p se nalazi duh
  + Ako se na na polju p nalazi i Pacman vrati vrijednost minus beskonačno
  + Inače bodove oduzmi od umanjenih nagrada (proporcionalno udaljenosti polja od Pacmana)
* Na polju p nalazi se uplašeni duh
  + Pribroji duha kao hranu sa umanjenom nagradom

Ovako konstruirana funkcija za posljedicu ima to da više bodova nose ona stanja koja su bliža nakupini (eng. *Cluster*) hrane gdje nema duhova u neposrednoj blizini, a ima uplašenih duhova, te ona stanja u kojima Pacman jede izoliranu hranu.

Postupak pretraživanja labirinta agenta Pacman

Pretraživanje labirinta je determinističko i koristi se BFS algoritam koji traži u širinu. Za svako polje provjerava postoji li hrana tamo ili možda duh koji je uplašen. Sukladno tomu pribraja umanjenu nagradu koja ovisi o udaljenost do Pacmana.

|  |
| --- |
| visited = {}  queue = {(pacman current position)}  while queue not empty:  (curPos, cost) = first(queue)  queue = rest(queue)  if curPos in visited then:  continue  visited = visited U curPos  if curPos has food then:  discountedReward += 10.0 \* self.discountFactor ^ cost  if curPos has ghost then:  discountedReward += ghostFood[curPos] \* self.discountFactor ^ cost  for action in possible actions from curPos:  if action not in visited:  append queue (new position after action, cost + 1) |

ghostFood se popunjava na početku, a sadrži kaznu za sva polja do kojih neuplašeni duhovi mogu doći u određenom broju koraka (do izračunavaju se do određene dubine)

|  |
| --- |
| fillGhostFood(ghostFood, gameState, depth)  if depth = 0 return  for each visible ghost in gameState:  for each action ghost can take:  ghostFood += penalty / current depth  fillGhostFood(ghostFood, gameState after action, depth -1) |

Također Pacman prilikom evaluacije pojedinog stanja provjerava, može li duh u jednom potezu doći u to stanje, a ukoliko može, vraća kao povratnu vrijednost minus beskonačno.

|  |
| --- |
| For each visible ghost:  If ghost is not scared then:  For each action ghost can take:  If ghost position after action = my position then:  return -inf |

Strategija agenta Pacman

Agent Pacman prvo dohvati popis svih mogućih akcija, te za svaku akciju izračuna stanje koje slijedi iz te akcije i ocijeni to stanje sa funkcijom korisnosti. Agent zatim odabire onu akciju koja ima najveću korisnost. Ukoliko više akcija ima jednako veliku korisnost, tada se stohastički odabire jedna od tih akcija.

|  |
| --- |
| Foreach possible action:  List = (action, u(next state for action))  Best = max(List) by utility  If |Best| = 1 then:  Take action  Else:  Take one action from Best randomly |

Agent duh

U sklopu projekta ostvareni su i agenti duhovi koji imaju ograničenu vidljivost svijeta, te su sposobni komunicirati sa drugim agentima duhovima koje mogu vidjeti.

Na početku svakog poteza, svaki duh provjeri koje sve duhove može vidjeti, te uspostave „mrežnu vidljivosti“. Odnosno, ako duh A vidi duha B, tada duh A vidi sve što vidi i duh B, i obrnuto. Relacija je tranzitivna, što znači ako postoje 3 duha A, B, C, takvi da A vidi B, a B vidi C, tada sva 3 duha znaju što se nalazi u okolini sva 3 duha.

Nakon što je svaki duh saznao informacije o svijetu, traži vidi li u tom svijetu Pacmana, ukoliko ne vidi Pacmana, tada potraži postoji li u memoriji informacija o zadnjem mjestu na kojem je Pacman viđen. Ukoliko postoji ta informacija ne postoji, tada duh nasumično odabire jednu akciju.

Ukoliko duh vidi Pacmana ili zna njegovu zadnju lokaciju, tada se kreće u smjeru te lokacije tako da manhattan udaljenost bude što manja. Kretanje u tom smjeru nije striktno definirano, već se događa sa određenom vjerojatnosti, što znači da pacman u nekom malom postotku slučajeva neće birati najkraći put, već će odabrati neki drugi put. Ovo se u praksi pokazalo kao dosta dobro rješenje, jer tada duhovi ne love Pacmana tako da zajedno idu istim putem, već se jedan od njih zbog stohastičnosti postupka, može odvojiti, te hvatati Pacmana iz nekog drugog smjera.

Korištena programska potpora

Kao podloga za programsku izvedbu igre koristi se Berklyjeva Python implementacija. U implementaciji dolazi primjer izvedbe duhova i refleksivnog agenta Pacman koji bira onu akciju koja mu u idućem koraku donosi najviše bodova. Agenti duhovi nasumično biraju između jedne od mogućih akcija.

Nadopunjavanje implementacije je izuzetno jednostavno, te se ostvaruje uređivanjem datoteke *multiAgents.py*, kreirajući klasu za agenta Pacman, odnosno uređivanjem datoteke *ghostAgents.py* te dodavanjem klase za agente duhove.

Opisana implementacija Pacmana je u klasi ReflexAgent. Osim te verzije, pokušano je i sa implementacijom MiniMax algoritma, no on se u praksi pokazao kao prespor. Opisana implementacija duhova je u klasi LimitedDirectionalGhost.

Pokretanje simulacije

Simulacija se pokreće pozivanjem naredbe:

python pacman.py -p ReflexAgent –l mediumClassic

Uključivanje pametnih duhova radi se pozivanjem:

python pacman.py -p ReflexAgent –l mediumClassic -g LimitedDirectionalGhost

Rezultati simulacije

Uz stohastičke duhove i razumnu vidljivost (2 polja u svakom smijeru) Pacman pobjeđuje u 95% slučajeva. Ukoliko se vidljivost smanji na 1 polje, te ukoliko je mapa takva da hrana nije povezana, već da na početku igre postoje prazna polja između nakupina hrane, tada se može dogoditi da Pacman ne zna gdje dalje ići, te nasumično luta po mapi (kada dođe do kraja jedne nakupine, zbog vidljivosti ne može vidjeti drugu nakupinu), pa se samim time smanjuje i vjerojatnost pobjede.

Važnost vidljivosti svijeta agentu Pacman eksponencijalno opada. Odnosno Pacman ostvaruje puno bolje rezultate uz vidljivost 2 nego uz vidljivost 1. No kako povećavamo vidljivost, to napredak agenta Pacman više nije toliko očit. Pacman ostvaruje tek nešto malo bolje rezultate, što se može pripisati tomu što agent pamti gdje je vidio hranu, te koristi tu informaciju za izračun svoje slijedeće akcije.

Uz pametne, komunicirajuće, usmjerene duhove, učinkovitost agenta drastično opada, te sada pobjeđuje u tek približno 35% slučajeva.