Univerzita Karlova v Prahe
Matematicko-fyzikálna fakulta

Bakalárska práca

Eva Pešková

Codewars, vojna robotov

Katedra softwarového inžinierstva

Vedúci bakalárskej práce: Mgr. Tomáš Poch

Všeobecná informatika

2009

Prehlasujem, že som svoju bakalársku prácu napísala samostatne a výhradne s použitím citovaných prameňov.
Súhlasím so zapožičiavaním práce.

V Prahe 12. mája 2009

Eva Pešková

Obsah

Kapitola 1

Uvod

Kapitola 2

Analýza

2.1 Codewars a jeho alternativy

2.1.1 Robowars

2.1.2 Pogamut

2.2 Problem optimalnej strategie viacerych hracov

2.3 Prakticke vyuzitie

Kapitola 3

Implementácia prostredia

3.1 Hracia plocha

3.1.1 Generovanie map

3.1.2 Vlastnosti stien

3.2 Jazyk hry a priebeh hry

Po načítaní všetkých robotov, ich programov sa hra klada z trojice (FrontaUdalosti, Mapa, ŽiviBoti). Samotná hra prebieha tak, že pokiaľ existujú živí boti, spracuje sa nsledujúca udalost z FrontyUdalosti. O zaradenie zpať a prípadné vymazanie sa v prípade úmrtia sa objekty starajú sami.
Objekty = steny, boti, strely.
Stav robota vzhladom na svet charakterizuje šestica: (InstruktionStack,Instructionpointer, Hitpoints, X, Y, natocenie)
Jednotlivé inštrukcie sa prejavia takto (zatial nepredpokladame koliziu):

3.2.1 Priebeh penalizacie za instrukciu

3.2.2 Detekcia zacyklenia

3.3 Vlastnosti robotov

Výsledok programu, ktorého inštrukcie robot vykonáva, je závislý na konkrétnom type robota. Typ robota je definovaný niekoľkými parametrami, ktoré si bežný užívateľ samostatne na začiatku hry definuje == UholViditelnost, PolomerViditelnosti, Obrana, Hitpoints, TypStrely, VelkostPamate==

Užívateľ si typ strely definuje tak, že dostane k dispozícií určitź počet bodv a tie musí medzi jednotlive volby rozdeliť
Bot premýšla tak, že sa nechá bežat dovtedy, kým z jeho program nenarazí na inštrukciu, ktorá by ovplyvnila svet, alebo kým neprsiahne timeout. Ak

presiahne timeout, preruší sa, preplánuje vo fronte udalostí ďalej (presnejšie o TIMEOUT tikov ďalej)
Velmi špeciálny je prípad, keď užívateľ nenastaví robotovi žiadnu pamať. Takýto robot nie je schopný si zapatať jedinú premennú a ani vracat návratové hodnoty, teda v kóde programu by nemali byť definované žiadne funkcie, ale iba procedúry.Je nutné povedať, že funkcie a procedúry sa v žiadnom prípade nechovajú ako premenné a teda nezaberajú robotovi pamať. Teda v tomto prípade dojde k masívnemu využitiu rekurzií. Samotne vstavané príkazy sú však vlastne funkcie, ale kedže sa nikam nemajú priradiť, ich návratová hodnota sa zahodí. Nevýhodou tohoto typu bota je však skutočnost, že nemá povolené napriklad strieľanie, pretože to je funkcia, ktorá potrebuje za každých okolností parametre.

3.4 Sietova komunikacia

Tabuľka 3.1: Vplyv robota na svet

Instrukcia	parameter	stav robota po instrukcii	Stav sveta po instrukcii
Viditelne instrukcie vzhladom na svet			
step	Z[4] alebo typy FORWARD, BACKWARD, LEFT, RIGTH	(IS, IP+1, HP, x+(0,1),y+(0,1), natocenie)	(Fronta.pop().insert(bot), Mapa, ZiviBoti)
step	-(default parameter = FORWARD)	rovnako	rovnako
turnLeft	-	(InstruktionStack, Instructionpointer+1, Hitpoints, X, Y, natocenie+1)	(FrontaUdalosti.pop(). insert(bot), Mapa, ŽiviBoti)
turnRight	-	(InstruktionStack, Instructionpointer+1, Hitpoints, X, Y, natocenie-1)	(FrontaUdalosti, Mapa, ŽiviBoti)
turn	Z[4] alebo typy FORWARD, BACKWARD, LEFT, RIGTH	(InstruktionStack, Instructionpointer+1, Hitpoints, X, Y, NoveNatocenie)	(FrontaUdalosti, Mapa, ŽiviBoti)
see	-	(InstruktionStack, Instructionpointer+1, Hitpoints, X, Y, natocenie)	rovnaky efekt
see	[0-9]+	(InstruktionStack, Instructionpointer+1, Hitpoints, X, Y, natocenie)	rovnaky efekt
shoot	[0-9]+, [0-9]+	(InstruktionStack, Instructionpointer+1, Hitpoints, X, Y, natocenie)	(FrontaUdalosti.pop() insert(bot, strela), Mapa, ŽiviBoti)
shoot	[0-9]+	(InstruktionStack, Instructionpointer+1, Hitpoints, X, Y, natocenie)	(FrontaUdalosti.pop() insert(bot, strela), Mapa, ŽiviBoti)
JUMP	(InstruktionStack, Instructionpointer - LABEL, Hitpoints, X, Y, natocenie)	žiadny efekt až na preplánovanie	
CALL	FunctioName	(InstruktionStack + FunctionStack, Instructionpointer = AtFunctionStack, Hitpoints, X, Y, natocenie)	žiadny efekt až na preplánovanie
EndFunction	(InstruktionStack	žiadny efekt až na pre-	

Tabuľka 3.2: Vlastnosti robota

Vlastnost	Vplyv
polomer viditeľnosti	Robot dokáže na určitú vzdialenosť rozpoznať object, táto vlastnosť určuje, koľko políčok dopredu v priamom smere(v smere, akým je bot aktualne otočený) vidí. Tento počet sa pod iným uhlom o niečo zmení,
uhol viditeľnosti	maximálny rozptyl viditeľnosti - robot nevidí celý kruh okolo seba, ale iba určitú výseč
Obrana	Obranne cislo bota. Pouziva sa pri strete s nejakou prekazkou. Viz kolizie.
Hitpoints	životnost bota, akonahle klesne na nulu, z fronty akcií sa jeho nasledujúca akcia odstráni a robot mizne do prepadliša dejin
[1ex] Typ strely	Strela je sa definuje podobne ako samotný bot, ale ma iné parametre

Tabuľka 3.3: Vlastnosti Strely

Vlastnost	Vplyv
Odrážavost	definuje, ako moc sa strela od steny odrazí, v podstate je to to iste ako hmotnosť
Rýchlosť	definuje preplanovanie vo fronte akcií
Hitpoints	dolet strely, ten sa znižuje s poctom tikov a súčasne kolíziami
Utok	Utocne cislo bota, pouziva sa pri utok, viz kolizie

Kapitola 4

Výber a popis testovaných algoritmov

4.1 Algoritmus 1

4.2 Algoritmus 2

4.3 Algoritmus 3

Kapitola 5

Testovanie

Kapitola 6

Vylepšenia

Kapitola 7

Záver

Literatúra