Genetic Algorithm

Artificiell intelligens för digitala spel

Simon Bothén

VT16

**Ordlista**

Unit(s)– En enhet i populationen i simuleringen.

GA – Genetic Algorithm

**Problemet**

Frågeställning kan skrivas som: ”Vilka attribut för en *Unit* är dem mest optimala i en arenamiljö”.

Att sätta alla attribut till dess maxvärde är den mest oslagbara lösningen men inte den rimligaste eller den mest passande, speciellt inte för ett spel. För att göra min *GA* mer intressant introducerade jag därför negativa effekter till mina attribut. Här listas de tre gener varje Unit har.

***Health***

Fördel: Du tål mer.  
Nackdel: Du blir större och därmed lättare att träffa.  
Omvänd effekt: Du blir mindre, men tål inte lika mycket.

***Speed***

Fördel: Du blir snabbare och blir svårare att träffa.  
Nackdel: Du blir svagare och skadar mindre.  
Omvänd effekt: Du skadar mer, men blir långsammare.

***Firerate***

Fördel: Du skjuter snabbare.  
Nackdel: Du har mindre träffsäkerhet.  
Omvänd effekt: Du blir mer träffsäker, men skjuter inte lika snabbt.

För att lösa problemet har jag satt upp en simulering där varje *Unit* utmanar alla andra *Units* var för sig tills han mött alla. Sen är det nästas tur att möta alla andra. Detta möjliggör att alla blir rättvist bemött och på lika villkor. Nackdelen är att det utförs på O(n2) tid.

**Begränsningar**

Positioneringen av alla *Units* är begränsade till en oval. Rörelse kan bara ske moturs eller medurs. Fördelen är att rörelse bara sker i 1D vilket förenklar många problem då projektet har en snar deadline.

Simuleringen kan bara köras i max 1200Hz innan det blir för mycket för min *CPU*. Det är ungefär 200x så snabbt.

**Representationen**

representationen (hur representeras l¨osningen i din GA? Ex.: om du anv¨ander en heltalsvektor f¨or att representera l¨osningen, vad betyder de olika f¨alten och vad betyder olika v¨arden eller olika intervall),

**Fitness-funktionen**

Fitness-funktionen:

Fitness-funktion tar hänsyn till tre olika variabler; *TotalTimeAlive*, *DamageDealt* och *Wins*. Enheten kan tänkas som “värdefulla sekunder”. Varje sekund du är i liv och i strid ökar *TotalTimeAlive* variabel med 1. Varje vunnen match är värd 20 sekunder och var 100:e skada är värd 1 sekund.

* *TotalTimeAlive* belönar en att vara vid liv.
* *DamageDealt* belönar en att ge ut mycket skada.
* *Wins* belönar en att vinna.

Tanken är att fitness-funktionen skall vara balanserad och belöna olika typer av spelstilar. Variablerna återställs till 0 efter varje generation för att börja om profileringen.

**Evolutionen**

**Selektering**

Efter varje generation så sorteras populationen och den sämre hälften plockas bort. Sen delas den bättre hälften in i par som sen får vars två barn. Varje barn kopierar slumpmässigt ut gener från antingen mamman eller pappan till sig själv.

**Mutation**

Varje Unit har 10% chans att muteras efter varje generation. Om det blir en mutering muteras alla tre gener. Då sker en ”Real-Value” mutation på varje gen vilket innebär att genen kan utvecklas både positivt och negativt. Chansen finns också att en gen förändrar sig minimalt.

**Resultat**

100X generationer tog ca xxh. Anledningen till att simuleringen tar så lång tid per generation är på grund av upplägget. Storleken på populationen är 100st vilket innebär att det blir strax under 10,000 matcher per generation om varje Unit skall utmana alla andra plus en returmatch.

• och resultatet f¨or b¨asta, s¨amsta, median, medel, standardavvikelse f¨or “fitness”-funktionen f¨or generationerna 1, 10, 100 och 1000 (l¨ampligen med en tabell).

• samt visa resultaten i en graf