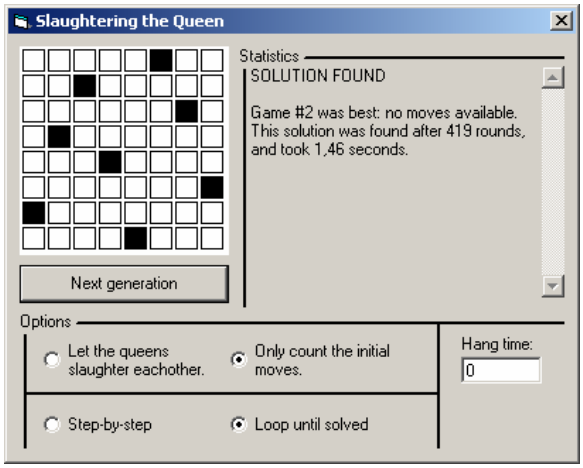
Paul Lammertsma #0305235

Het Acht-Koninginnen Probleem

We gaan uit van een schaakbord met uitsluitend acht koninginnen. Het is de bedoeling dat deze koninginnen elkaar niet mogen slaan; dat wil zeggen: er is maar één koningin per rij, kolom en beide diagonalen. Met behulp van een evolutionair algoritme kunnen we de situatie nabootsen en door middel van een 'toernooi' de beste oplossing vinden.



Hiervoor is het programma geschreven: *Slaughtering the Queen*. Het programma heeft een grafische interface die het ronde en oplossing kan laten zien.

Deze antwoorden op de vragen van de taak beschrijven de werking van het programma.

1) Het programma maakt een reeks bits die het bord voorstelt, die uiteraard 8 x 8 = 64 lang is. Acht willekeurige, verschillende bits worden een 1, die een koningin representeren. De anderen zijn worden 0.

2) Een bord wordt gemuteerd door paren bits te verwisselen. Minstens één, maximaal acht 1-bits worden met 0-bits vervangen. Het is doelloos om bits met dezelfde waarde te verwisselen, dus het algoritme zoekt uitsluitend 1-bits en verwisselt ze met willekeurige 0-bits.

3) Het algoritme berekent het aantal slagen die ieder koningin (1-bit) kan doen. De fitnessfunctie moet een resultaat opleveren dat zo laag mogelijk is: een goeie bord (en dus de oplossing van de algoritme) geeft namelijk nul slagen.

4) Het programma begint met twee willekeurige borden. Nu kan een loop beginnen die het volgende doet:
   - De fitnessfunctie wordt op beide borden toepast;
   - Het bord met de hoogste waarde wordt vervangen met een mutatie van het bord met de laagste waarde.

   Bij het laatste stap wordt voorkeur gegeven voor de mutant. De reden hiervoor is om te voorkomen dat het programma erg lang bezig is een oplossing te vinden voor een onoplosbaar opstelling van het bord. Bijvoorbeeld: zes koninginnen zijn geplaatst, dus moeten nog twee plekjes uitgezocht worden. Het is mogelijk dat er *geen* toegestaande plaats overblijft om deze twee neer te zetten. Dit zou langer duren, omdat een mutant pas wint wanneer zowel een reeds geplaatste koningin en nog minstens één van de overgebleven twee naar een nieuw toegestaande veld geplaatst worden. Het is duidelijk dat dit onwaarschijnlijk is en dus talloze mutanten nodig zijn voordat de correcte oplossing gevonden is.

In het plaatje is zichtbaar dat een oplossing gevonden is. Het heeft 419 rondes (wedstrijden) geduurd, dus van de oorspronkelijk 2 willekeurige borden, werden 418

mutaties gemaakt voor dat een goede bord gevonden werd. Omdat de langzamere taal Visual Basic werd gebruikt, en voor een grafische interface werd gekozen, duurde dit 1,46 seconden.

Een andere aanpak is bijvoorbeeld eerst een ***n*** aantal borden te maken (waar de koninginnen willekeurig geplaatst zijn), en dan wedstrijden van twee laten bepalen welke borden naar de volgende ronde gaan. Bij de volgende ronde (de helft van de eerste) wordt per bord één mutant gemaakt (dus zijn er opnieuw ***n*** borden), enzovoort, totdat een bord ondekt wordt waar geen zetten mogelijk zijn. Hier heb ik pas later aan gedacht, en dus werkt mijn oplossing niet met generaties, maar slechts met enkele wedstrijden.

Op het volgende pagina ziet u een overzicht van de stappen die het programma volgt.

De werking van het programma wordt beschreven in zeven stappen.

1) Het programma maakt een reeks bits die het bord voorstelt. Deze is uiteraard 8x8 = 64 bits lang. Willekeurig worden 8 bits een 1 gemaakt, met als enige regel dat er daadwerkelijk 8 bits de waarde 1 aan moeten nemen.

2) Nu analyseert het programma voor ieder koningin (een 1-bit) of zij een ander koningin kan slaan. Het totaal aantal zetten wordt opgeslagen in een variabel.

3) Een tweede willekeurige bordt wordt nu gemaakt. Dit wordt op dezelfde wijze gedaan als stap 1 en 2.

4) De twee resultaten worden vergeleken. Het kleinste aantal zetten bepaalt de winnaar. Deze gaat dan ook door naar de volgende ronde. (Als het gelijk spel is, wordt altijd voor het mutant gekozen.)

LOOP

5) Volgende ronde. Nu wordt de winnende bord gemuteert: een koningin (1-bits) wordt verplaatst naar een lege veld (0-bit). Het aantal koninginnen dat wordt verplaatst is een willekeurig getal tussen de 1 en 8. Wanneer een koningin verplaatst wordt, wordt eigenlijk alleen de waarde's van de twee bits verwisseld. Er ontstaat dus altijd een mutatie op de oorspronkelijke bord.

6) Opnieuw wordt een toernooi gehouden, zoals stap 4, maar nu tussen de winnaar van de vorige ronde en zijn gemuteerde variant. Opnieuw gaat de winnaar door naar de volgende ronde.

7) De stappen 5 en 6 worden herhaald totdat de winnaar geen zetten kan doen. Hiermee is een oplossing gevonden en eindigt de loop.