Samenvatting evolutionary computation

Sinds de geboorte van computers is evolutie een inspiratiebron voor alghorithme designers. Het resulterende veld, evolutionaire computation heeft al vele wetenschappelijke problemen opgelost. Vandaag de dag staat een nieuwe fase te beginnen, die waarin de virtuele algorithmes zich gaan verplaatsen naar de echte wereld. Een stap naar autonome robots die zich aanpassen aan hun omgeving.

Het voorstel dat evolutie gebruikt kan worden voor het oplossen van problemen kwam met de uitvinding van de computer. Er werden verschillende implementaties bedacht welke uiteindelijk allemaal samenkwamen in het veld dat evolutionary computation heette. Het is succesvol voor een hoop computationele taken zoals optimisatie of design. Daar zijn ook vele succesvolle voorbeelden van. De opkomst van computationele evolutie is een grote stap binnen de evolutie vanuit de biologie naar software, computer gebied. Nieuwe ontdekkingen (zoals 3D printen) introduceren de stap naar nog een overgang, van software naar hardware. In dit artikel wordt beschreven hoe het werkt en wat het verschil is met natuurlijke evolutie.

De essentie van problemen oplossen met evolutie is om voor de individuen binnen een populatie verschillende oplossingen te berekenen en deze een bepaalde fitness toe te kennen. Om zo'n werkende systeem te krijgen moeten er verschillende design stappen gedaan worden. De eerste stap is om een goede representatie structuur te vinden die mogelijke oplossingen voor het probleem kunnen geven. De volgende stap is om een bepaalde kwaliteit aan een individu toe te kennen op basis van de oplossing van het probleem. De laatste stap is het speciferen van de selectie en mutatie operators. Een evolutionair algoritme werkt op twee niveaus. Op hoger niveau zijn er phenotype waarvan de fitness gemeten wordt. Selectie mechanismen kiezen dan een pool van ouders op basis van hun fitness en bepalen welke ouders en hun 'offspring' verder gaan naar de volgende generatie. Op lager niveau zijn er genotypen die phenotypen reproduceren op een manier dat deze alsnog aangepast kunnen worden. Zo worden met behulp van mutaties telkens nieuwe generaties met oplossingen geintroduceerd die telkens een hogere fitness hebben. De grote van de populatie wordt meestal constant gehouden. Evolutionaire algorithmes kunnen gemakkelijk ook in een andere applicatie worden toegepast omdat er maar twee componenten probleem afhankelijk zijn, de manier waarop genotypen worden omgezet in phenotypen en de fitness functie. Een simpele evolutionair computationeel systeem kan al een enorme verscheidenheid aan problemen oplossen. In andere woorden, een relatief klein aantal genotypen kan een hoop phenotypen ondersteunen. Net als in de natuur. Het is dus ook heel simpel om een evolutionair algorithme te designen zolang het maar met dezelfde genotypen werkt.

Mensen hebben twee rollen binnen de evolutie, ze ondergaan evolutie, maar ze hebben ook de evolutie van andere soorten beïnvloed. Dit hebben we gedaan door te kiezen welk dier moet paren of overleven. Zo hebben we evolutie gebruikt om nieuwe voedselbronnen voor onszelf te creeëren of meer nuttige dieren. Toch begrepen we toen we hiervan gebruik begonnen te maken nog niet hoe dit werkte. De evolutie op een andere manier (recombination mechanisms or mutations) beïnvloeden was nog niet binnen ons bereik. Dit veranderde met de uitvindingen van de computer, waarin digitale werelden gemaakt kunnen worden welke zeer flexibel zijn. Vanuit een wetenschappelijk oogpunt zijn evolutionaire algorithmen randomized zoekmachines gebaseerd op het genereer en test principe. Hetgeen wat zich onderscheidt van andere methoden is dat de populatie gedeeltelijk bewaard blijft tijdens het zoeken. Vaak worden evolutionaire algorithmes geordend op basis van compleetheid, optimaliteit en efficientie. De compleetheid kan behaald worden door een goeie representatie en variatie operator. De optimaliteit is meer complex aangezien niet elk probleem hetzelfde is. De problemen die het best opgelost kunnen worden door EC (evolutionairy computation) zijn optimalisatie problemen of iets dat kan worden omgezet in zo'n

probleem. Design is een van de gebieden waar EC veel gebruikt zou kunnen worden. Toch zijn EC's niet perfect. Ze benaderen perfectie maar je bereiken het niet.

EC kunnen lastige, nog niet goed begrepen problemen oplossen. Evolutionary problem solvers hebben zich al in meerdere velden van de wetenschap en techniek kunnen bewijzen. Er zijn bijvoorbeeld antenne's voor NASA gemaakt met behulp van EC welke veel efficiënter waren dan degenen die ze voorheen gebruikten. Het grote voordeel van EC is dat er heel veel verschillende designs getest worden. Hierdoor is het uiteindelijke resultaat origineel. Vooral als er afwegingen gedaan moeten worden zijn EC's erg efficient om de beste oplossingen te vinden. Ook op het gebied van machine learning hebben EC's hun kracht getoond en bij het maken van de controllers van physieke machines in bijvoorbeeld fabrieken. EC's zijn volgens het artikel eigenlijk gewoon overal goed voor.

Alhoewel er eerst kritiek was op EC is het nu uitgegroeid tot een heel belangrijk wetenschappelijk veld binnen de computational intelligence. Er is weinig werk nodig voor een designer om unieke en enorm slimme oplossingen uit een EC te krijgen en dit is tot nu toe nog ongeevenaard. Evolutie leert langzaam maar dankzij steeds snellere processors kunnen er meer generaties gesimuleerd worden binnen dezelfde tijdspanne. Er zijn zelfs vergelijkingen gedaan tussen mens en EC en deze schenen zeer gelijkwaardig te zijn. Het succes van de EC's kan gekoppeld worden aan verschillende algorithmische functies. Ten eerste zijn ze assumptie vrij, ze kunnen gemakkelijk overal op toegepast worden. Ten tweede zijn ze flexibel en werken ze samen met al bestaande methoden. Ten derde zijn ze ook nog robuust, de kans dat een EC vastloopt op een niet optimale oplossing is kleiner dankzij mutaties. Ten vierde zoeken ze niet naar 1 oplossing maar naar meerdere die goed werken. En als laatste ze denken out of the box doordat ze helemaal niks over het gehele onderwerp weten verder.

Tegenwoordig wordt er naar de volgende onderwerpen onderzoek gedaan voor EC's:
-Er wordt automatische design hulp voor EC's gemaakt zodat men hierbij minder snel fouten kan maken.
-Om efficienter te kunnen werken qua computationele berekeningen wordt bij een nieuwe methode niet de fitness van elke individu berekend maar alleen van een paar geselcteerden.
-Er worden meerdere doelen gesteld voor de EC's maar tegelijk worden de doelen ook efficienter gemaakt doordat onderzoekers zich gaan afvragen wat ze nou werkelijk willen weten.
-Door meer complexere codes in het genotype-phenotype mapping kunnen steeds complexere dingen gedaan worden met EC, bijvoorbeeld Evolutionary Robotics.