 ISSN 0254-3486 = SA Tydskrif vir Natuurwetenskap en Tegnologie 17, no. 3 1998

Inleiding tot die toepassing van genetiese algoritmes in ingenieurswese

I.S. Shaw
Navorsingsgroep vir Industriële Elektroniese Tegnologie, Fakulteit Ingenieurswese, Randse Afrikaanse Universiteit, Posbus 524, Aucklandpark, 2006

Ontvang 10 Mei 1997; aanvaar 27 Februarie 1998

Uittreksel

Genetiese algoritmes is 'n nuwe navorsingsveld en behoort tot die gebied van kunsmatige intelligensie. Hierdie artikel is gemik op die toepassing daarvan in spesifieke vakgebiede van die ingenieurswese waar die tegniek reeds goeie resultate gelewer het. Die doel van hierdie werk is om 'n basiese inleiding aan studente asook ervare ingenieurs, wat hul kennis wil opgradeer, te verskaf. 'n Besondere kenmerk van kunsmatige intelligensie is dat óf direkte menslike ervaring, óf sekere funksies van die menslike brein, in stede van wiskundige modelle, vir die modellering van fisiese verskynsels gebruik word. Op hierdie manier kan sekere eienskappe van menslike intelligensie, soos byvoorbeeld die ignorering van minder belowende benaderings tot 'n probleem, of die waarneming van verborge patrone, nageboots word. 'n Poging is aangewend om 'n Afrikaanse vakterminologie vir hierdie nuwe veld te vestig, wat in hierdie artikel aan die lesers vir kommentaar voorgelê word.

Abstract

Introduction to the application of genetic algorithms in engineering

Genetic algorithms constitute a new research area in the field of artificial intelligence. This work is aimed at their application in specific areas of engineering where good results have already been achieved. The purpose of this work is to provide a basic introduction for students as well as experienced engineers who wish to upgrade their knowledge. A distinctive feature of artificial intelligence is that instead of mathematical models, either direct human experience or certain functions of the human brain for the modelling of physical phenomena are used. In this manner certain properties of human intelligence such as, for example, the ignoring of less promising approaches to a problem or the recognition of hidden patterns can be emulated. An effort was made to establish an Afrikaans terminology in this new field which is now submitted to the readers for their evaluation.

1. INTELLIGENTE STELSELS

'n Sogenaamde intelligente stelsel verskaf treffende *probleemoplossende* response wat deur probleeminsette gegenereer is. 'n Probleemoplossende respons is een wat

- nie deur die ontwerper vooruitgeprogrammeer is nie;
- as gevolg van 'n nuwe en onverwagte inset gegenereer mag word;
- vanuit voorbeelde geleer kan word.

Die werking van intelligente stelsels word gewoonlik na analogie van biologiese stelsels beskryf, soos byvoorbeeld hoe prosesbeheer, patroonherkenning en besluitneming deur mense uitgevoer word. Bostaande is tipiese eienskappe van menslike intelligensie, dus het die begrip *kunsmatige intelligensie* ontstaan, wat 'n nuwe dissipline is wat bestudeer

- hoe probleme deur mense op 'n intelligente wyse opgelos word;
- hoe intelligente, probleemoplossende, menslike gedrag en denkwyse deur masjiene nageboots kan word.

Die uiteindelike doel van intelligente stelselontwerp is die skepping van *outonomiese* stelsels wat komplekse beheertake onder die operasionele toestande van 'n aanleg of proses kan uitvoer, selfs in die teenwoordigheid van falings, en sonder enige menslike ingryping of toesig. In so 'n geval sal dit voldoende wees om die stelsel te laat weet *wat* gedoen moet word, maar nie *hoe* die opdrag uitgevoer moet word nie.

2 ROBUUSTHEID

Een van die fundamentele doelwitte van ingenieurswese is om *robuuste* stelsels te ontwerp wat 'n doeltreffende diens kan lewer ten spyte van omgewings- en komponentverouderings. Ingenieurs soek robuuste apparatuur en programmatuur wat sulke ongunstige veranderings kan weerstaan. Byvoorbeeld, beheerstelsels gebruik negatiewe terugvoer om die stelsel se sensitiwiteit vir aanlegparameter- asook temperatuurveranderinge te verlaag. In hierdie verband moet 'n mens na biologiese stelsels kyk, wat besonder goeie eienskappe vir oorlewing onder minder gunstige omstandighede ontwikkel het. Biologiese stelsels is inherent robuust, buigsaam en doeltreffend om hul voortbestaan te verseker. Hulle kan by omgewingsveranderings aanpas en komponentdegradasie oorleef, te danke aan selfgeneesvermoë, selfleiding en reproduksie. Genetiese algoritmes (GA) pas die meganisme van biologiese genetika op kunsmatige stelsels soos ingenieurswese, besigheidsbestuur, ekonomie, ens. toe. Sommige van hul voordele is dat hulle teoreties asook empiries bewys is, dat hulle eenvoudig, tog kragtig is, en dat hulle nie deur beperkende aannames oor die oplossingsruimte, soos byvoorbeeld kontinuïteit, differensieerbaarheid, ens. beïnvloed word nie. 'n Nadeel wat genoem moet word, is die feit dat die regte keuse van spesifieke berekeningstegnieke partykeer slegs deur 'n sekere interaktiewe eksperimentele benadering bereik kan word.

3. DIE BASIS VAN GENETIESE ALGORITMES

GA's verteenwoordig 'n baie doeltreffende metode toepaslik vir enige probleem wat 'n optimale oplossing verg. GA's is soektog-algoritmes gebaseer op die meganisme van natuurlike seleksie en biologiese genetika. Natuurlike seleksie beteken die oorlewing binne 'n gegewe omgewing van dié met die hoogste fiksheid. Volgens hierdie metode word individue vanuit 'n gegewe populasie met die beste oorlewingsvermoë gesoek. Vervolgens sal die stelling "die individu met die hoogste fiksheid wat in 'n gegewe omgewing kan oorleef" omgesit word na terme wat in ingenieurswese gebruik word. In hierdie verband beteken optimisasie die deursoek van 'n *oplossingsruimte*, dit wil sê al die moontlike oplossings van 'n probleem, om die optimale, met ander woorde, die mees robuuste oplossing te vind. GA's het die vermoë om lineêre asook nie-lineêre, probleme op te los. 'n Geometriese afbeelding wat later in meer detail bespreek word, gee 'n goeie insig in die werking van die genetiese operators, *voortplanting* en *kruising*, wat as 'n instrument dien om sekere areas in die oplossingsruimte te delimiteer om belowende oplossings te identifiseer.

Daar bestaan egter ook sekwensiële optimisasiemetodes wat onder sekere beperkende omstandighede goed kan werk. Vervolgens sal die gewildste van hierdie benaderings bespreek en met die GA vergelyk word. In 'n praktiese geval is daar dikwels nie genoeg tyd om 'n groot oplossingsruimte volkome te ondersoek nie. Wanneer 'n mens 'n optimiseringsmetode kies om tyd te bespaar, moet jy vra of die optimale oplossing vanuit 'n betreklik klein willekeurig gekose monster van die oplossingsruimte verkry kan word al dan nie. Ons sal later sien dat in hierdie verband genetiese algoritmes 'n besonders belowende benadering is.

4. DIFFERENSIAALREKENINGGEBASEERDE METODES

In die gewilde differensiaalrekeninggebaseerde metode word die oplossingsruimte van 'n gegewe probleem gedefineer in terme van 'n *n*-dimensionele *objektiewe funksie* of *kostefunksie*, en die maksimum of minimum van hierdie funksie sal gesoek word. Die gesogte ekstreme waarde moet deur die opklim of afklim op die funksie in die stylste rigting (gradiënt) gesoek word. Die optimale oplossing lê waar die gradiënt van die kostefunksie in alle rigtings nul is. Ingeval van minima verteenwoordig 'n globale minimum die optimale oplossing, terwyl plaaslike minima waarvan die diepte minder as dié van die globale een is, suboptimale oplossings lewer. Soos in die geval van 'n beheerstelsel, byvoorbeeld, kan 'n kostefunksie 'n tydintegraal wees:

$$\int_0^\infty e^2 \, dt = \int_0^\infty [y_{stelpunt}(t) - y(t)]^2 \, dt \qquad (1)$$

Een van die tekortkomings van differensiaalrekeningmetodes is dat hulle 'n enkele punt as wegspringpunt gebruik, dus is hulle geneig om plaaslike minima in stede van die globale een te vind, veral as die wegspringpunt naby 'n plaaslike minimum lê. Indien die soektog binne 'n plaaslike minimum beland, kan dit nog steeds voortgesit word deur die gebruik van spesiale metodes, soos byvoorbeeld, deur die superposisie van ruis wat die funksie sal "rondskud" om uit die suboptimale vallei uit te kom. 'n Ander praktiese metode is om die begintoestande, dit wil sê, die wegspringpunt van die soektog te verander en die algoritme weer te laat loop. In ander gevalle, soos in neurale trupropagasienetwerke waar differensiaalrekeningmetodes gedurende die opleidingsfase gebruik word om 'n optimale netwerkstruktuur te bereik, word beweer dat plaaslike minima baie selde sal voorkom, dus is die probleem nie baie ernstig nie. 'n Aansienlik groter probleem is egter dat differensiaalrekening-gebaseerde metodes nie gebruik kan word nie indien die kostefunksie nie glad en differensieerbaar is nie. Dus is dié metode tot 'n betreklik klein domein beperk en kan die mees robuuste oplossing in die algemene geval nie gelewer word nie.

Alhoewel dit in die algemeen nie nodig is nie, neem slegs vir hierdie bespreking aan dat die stelsel wat geoptimiseer moet word oor 'n wiskundige model (soos byvoorbeeld 'n lineêre oordragsfunksie) beskik, waarvan elke stel parameters aan 'n bestemde oplossing behoort, dus kan *N* sulke parameterstelle gekies en in die model ingesit word om *N* oplossings te verkry. Afhangende van sekere optimisasiekriteria sal sommige van hierdie stelle "beter" vaar as andere. Die doelwit van optimisasie is dat in 'n sekwensieel gegenereerde reeks van genetiese operasies elke nuut gegenereerde oplossing 'n verbetering moet toon in vergelyking met die vorige een, dus sal 'n konvergensie tot 'n optimum getoon word. Die kwessie is egter die kriteria om 'n "beter" oplossing (dit wil sê, parameterstel) te verkry, hoe die tydrowende verwerking van te veel parameterstelle vermy kan word, en of daar 'n metode bestaan soortgelyk aan menslike intelligensie wat vinnig en doeltreffend irrelevante, nuttelose of suboptimale oplossings kan uitskakel en die optimale oplossing spoedig kan identifiseer.

5. BASIESE TERME VAN GA

GA word gunstig toegepas op probleme met 'n baie groot aantal potensiële oplossings. Elke oplossingskandidaat (met ander woorde, 'n stel van ooreenkomstige kostefunksieparameters) word as 'n **individu** beskou, wie se vermoë om in 'n gegewe omgewing (dit wil sê, die **kostefunksie**) te oorleef, getoets sal word. Elke parameterstel van die kostefunksie sal deur 'n **gekodeerde string** verteenwoordig wees. Somtyds word die gekodeerde string ook 'n **chromosoom**, in ooreenstemming met biologiese genetika, genoem. Die eerste stap is om 'n monster van die gegewe **populasie** van gekodeerde stringe (individue), die sogenoemde **broeivoorraad**, willekeurig vanuit alle moontlikes te kies. Die volgende stap is om die ooreenkomstige parameterwaardes van elke string te dekodeer en in die kostefunksie (dit wil sê, die omgewing) te substitueer, waardeur die **fiksheidsindeks** van elke gedekodeerde individu bepaal sal word.

Die substituering van 'n individu in die kostefunksie is gelykstaande aan die plasing van 'n individu in 'n omgewing waar hy moet oorleef, en sy potensiële vermoë daarvoor word deur sy fiksheid bepaal. Die fiksheidsindeks is eintlik die numeriese resultaat wat deur die kostefunksie gelewer word nadat 'n spesifieke parameterstel daarin gesit is. Dus word 'n fiksheidsindeks vir elke lid van die gekose broeivoorraad verkry. Individue met die hoogste fiksheidsindekse het 'n hoër potensiaal om in die volgende **generasie** 'n groter nageslag te genereer. Die volgende stappe van die algoritme bestaan uit *genetiese operasies* soos **voortplanting, kruising** en **mutasie**, wat se doel is om die fiksheidsindekse van individue oor 'n aantal generasies moontlik te verhoog.

6. GENETIESE OPERASIES

In *voortplanting* sal individue met hoë fiksheidsindekse potensieel 'n groter nageslag (kopieë) genereer. In hierdie stap word die verwagte aantal kopieë vir elke individu bereken. Die grootte van die broeivoorraad moet egter konstant gehou word, dit wil sê, dieselfde as dié van die vorige generasie.

In *kruising* is die doel om die genetiese materiaal van twee *willekeurig* gekose individue te vermeng, waardeur die goeie eienskappe van elke individu (dit wil sê, wat die oorlewingsvermoë aansienlik verbeter) na die nageslag oorgedra word. Eers word 'n broeipaar (die "ouers") willekeurig vanuit die broeivoorraad gekies om 'n nageslag te genereer. Gestel x en y is binêre vektore, $x = \{x_i\}$, $y = \{y_i\}$. 'n Willekeurige getal r word gegenereer vanuit 'n uniforme verdeling van 1 tot m en twee nuwe individue (die "kinders") x_i' en y_i' ontstaan soos volg:

$$\begin{aligned} x_i' &= x_i \text{ indien } i < r \\ &= y_i \text{ in alle andere gevalle} \end{aligned}$$

$$\begin{aligned} y' &= y_i \text{ indien } i < r \\ &= x_i \text{ in alle andere gevalle} \end{aligned}$$

Voorbeeld:

$$x_1 x_2 x_3 x_4 x_5 \; x_6 x_7 x_8$$

$$y_1 y_2 y_3 y_4 y_5 y_6 \; y_7 y_8$$

'n Kruisingsplek, soos byvoorbeeld na bis *3* vanaf links, word *willekeurig* gekies en 'n kruisingsoperasie sal, as volg, uitgevoer word.

Voor kruising: $y_1 y_2 y_3 \wedge y_4 y_5 y_6 \; y_7 y_8$

Na kruising: $x_1 x_2 x_3 \wedge y_4 y_5 y_6 \; y_7 y_8$

Voor kruising: $x_1 x_2 x_3 \wedge x_4 x_5 \; x_6 x_7 x_8$

Na kruising; $y_1 y_2 y_3 \wedge x_4 x_5 \; x_6 x_7 x_8$

Let op dat kruising gewoonlik slegs vir 'n sekere persentasie van die broeivoorraad uitgevoer word.

Laat ons kyk na 'n binêre voorbeeld. Gestel x = *100:01010* en y = *010:10100* waar die kruisingspunt deur ":"gekenmerk is. Die uitslagstrukture is dan: x' = *100:10100* en y' = *010:01010*.

Soos voorheen vermeld, is kruising 'n instrument wat 'n sekere area in die soektogruimte delimiteer en vir beter oplossings toets. Die n-dimensionele soektogruimte word deur hipervlakke verdeel waarvan die digtheid gelyk is aan die individu se koderesolusie. Kruising voer twee komplementêre soektogfunksies uit. Eerstens word nuwe toetspunte op die populasie se reeds teenwoordige hipervlakke verskaf. In bostaande numeriese voorbeeld, x en x' verteenwoordig die hipervlak *100#####*, waar "*#*" 'n "enigeen" geval beteken, dus word addisionele kennis oor hierdie hipervlak versamel. Hierdie kennis bestaan uit die uitkoms van berekeninge waarvolgens die fiksheidsindeks van die "kinders" groter of nie groter as vantevore sal uitkom nie. Tweedens, verteenwoordigers van 'n nuwe hipervlak in die bestaande populasie word ingevoer. In die vorige voorbeeld verteenwoordig x' hipervlak *#1001###* wat geeneen van die ouervektore x of y behels het nie. Indien dié hipervlak as 'n hoë werkverrigtingsarea bewys word, sal die evaluasie van x' tot verdere ondersoeke van dié hipervlak lei. Elke evaluasie van 'n struktuur van lengte L dra kennis by tot die werkverrigting van dié struktuur se 2^L hipervlakke. Die krag van GA's bestaan hoofsaaklik uit hul vermoë om hierdie geakkumuleerde kennis deur middel van 'n betreklik eenvoudige seleksiemeganisme te versamel. Die meganisme van voortplanting en kruising is eenvoudig, omdat hulle bloot uit die afskrywing van stringe en die uitruiling van substringe bestaan. Die soektogdoeltreffendheid van GA's is egter in 'n groot mate die gevolg van hierdie eienskappe.

Die doel van *mutasie* is om genetiese stagnasie te vermy, met ander woorde, om "nuwe bloed" in die genetiese proses in te bring, byvoorbeeld, wanneer daar sekere bisposisies bestaan wat nooit 'n kans gehad het om gedurende voortplanting en kruising te verander nie. Mutasie is die toevallige verandering van 'n waarde by 'n sekere bisposisie. Die doel is om te verhinder dat sekere inligting verlore raak. 'n Generasie kan ontstaan met 'n gebrek aan 'n sekere waarde by 'n sekere bisposisie. Byvoorbeeld, 'n generasie kort 'n "*1*" in bisposisie *3*, alhoewel dit mag 'n kritiese invloed op die uitkoms van 'n hoëkwaliteitresultaat hê. Nóg voorplanting nóg kruising alleen kan in opeenvolgende generasies 'n "*1*" in bisposisie *3* produseer. Mutasie laat egter toe om die bestaande "*0*" in bisposisie *3* na 'n "*1*" te verander, dus kan 'n kritiese inligtingselement in die populasie heringestel word. In die geval van binêre mutasie word 'n bis in 'n willekeurig gekose bisposisie gekomplementeer met 'n waarskynlikheid p_m'. Gestel 'n eendimensionele vektor $x = \{x_i\}$, dan

$$\begin{aligned} x_i' &= 1 - x_i, \text{ indien } p_m < p_m' \\ &= x_i \qquad \text{in alle ander gevalle} \end{aligned}$$

In biologiese genetika asook in genetiese algoritmes het mutasie 'n lae waarskynlikheid: 'n tipiese mutasietempo is een per elke *1000* verwerkte bisposisies.

$$M = L \,.\, N \,.\, p_m \tag{2}$$

waar L = stringlengte, N = aantal stringe in die broeivoorraad, p_m = mutasietempo, en die resultaat moet afgerond word.

Die voorafgaande is slegs 'n opsomming van die grondslae van GA's. Die prosedure wat verskaf word en hieronder in 'n eenvoudige voorbeeld getoon is, kan as 'n algemene uitgangspunt gebruik word. Daar bestaan egter etlike verfynings wat in die vakliteratuur nageslaan kan word. Byvoorbeeld, in sommige gevalle mag die fiksheidsindeks van 'n nageslag minder wees as dié van die "ouers". Om dit te vermy, kan die sogenoemde *keurgroeptegniek* gebruik word, waardeur die nageslag se fiksheidsindeks ten minste dieselfde as die ouer s'n gehou sal word. Buitendien bied GA's baie keuses vir die ontwerper en sulke keuses kan slegs empiries deur simulasie getoets word, soos byvoorbeeld populasiegrootte, kruisingstempo, mutasietempo, voortplantingstempo, en hoe hierdie keuses die konvergensie tot 'n optimale fiksheid sal beïnvloed.

7. KOSTEFUNKSIE EN KODERING

Die basiese skema van alle GA's is dieselfde, behalwe die parameterkodering en die kostefunksie wat uniek is aan elke probleem.

Die *kostefunksie* moet óf gegewe óf geskep word en dit is gewoonlik nie 'n triviale saak nie. Dit mag byvoorbeeld 'n oordragsfunksie wees, wat in die vorm van 'n analitiese funksie, 'n opsoektabel of selfs 'n wasigelogikafunksie uitgedruk is. Die buigsame keusemoontlikhede van die kostefunksie wat nie lineariteit vereis nie, is 'n wesenlike voordeel: die kostefunksie mag algebraïes, opsoektabelgebaseerd, stukgewys lineêr, diskreet, ens. wees. In industriële beheertegniek, soos vroeër vermeld, is die integraal van die kwadraat van die respons se afwyking vanaf die gewenste stelpunt een van die moontlike kostefunksies.

Die kostefunksie het 'n aantal parameters waarvan elkeen deur 'n string verteenwoordig word en elke aaneengeskakelde reeks van parameterstringe 'n sekere enkele resultaat gee, dus verteenwoordig elke stringreeks een moontlike oplossing van 'n probleem. Die som van hierdie moontlike resultate genereer 'n populasie waarvan 'n willekeurige aantal parameterstring-reeks (dit wil sê, K stringe van lengte N elk) as 'n broeivoorraad willekeurig gekies moet word. *Kodering* is 'n manier om die vorm van 'n probleemveranderlike na 'n GA-gebaseerde vorm te vertaal. As 'n parameterkode kan 'n binêre, Gray, skuifpunt, en nog baie ander vanuit die vakliteratuur gekies word, elkeen met sy besondere voordele en nadele. Binêre kode is dikwels 'n goeie uitgangspunt. Die aantal bisse is afhanklik van die bestek van waardes wat die resolusie bepaal. Die ontwerper moet eers die parameterbestek, $\Delta\theta = \theta_{maks} - \theta_{min}$, en die gewenste resolusie, A, bepaal.

Die kodes van die gekose *broeivoorraad* moet *gedekodeer* word, dit wil sê, hul ooreenkomstige ongekodeerde parameterwaardes moet bepaal word. 'n Mens mag vra hoekom die kodering van parameters nodig is, aangesien sommige van hulle weer gedekodeer moet word. Kodering van die populasie se parameterstringe is nodig omdat hierna die volgende genetiese operasies op die gekodeerde vorm van die parameters uitgevoer sal moet word. Dekodering is nodig slegs vir die broeivoorraad se parameterstringe wat in die kostefunksie ingesit moet word om hul individuele fiksheidsindekse te bereken. Natuurlik sal die gekodeerde vorm van die broeivoorraad se parameterstringe ook in die genetiese operasies gebruik word. Die lengte van 'n binêre string om die parameter te verteenwoordig, word soos volg, bereken:

$$N = log\ 2^{\Delta\theta/A} = [log_{10}\ (\Delta\theta/A)] / [log_{10}\ 2] \qquad (3)$$

Die volgende taak is om 'n eerste-orde lineêre *inverse afbeeldingsfunksie* te skep om binêr-na-desimale-omsetting vir dekodering te kan uitvoer. (In sommige gevalle mag hierdie inverse afbeeldingsfunksie heelwat ingewikkelder wees.)

$$M = m\,x + \theta_{min} \qquad (4)$$

waar m 'n afbeeldingsfaktor is, x die heelgetal omgesit vanaf 'n binêre string na desimaal is, en θ_{min} die laer limiet van die parameter is. Dus die afbeeldingsfaktor is

$$m = (\theta_{maks} - \theta_{min}) / (2^N - 1) \qquad (5)$$

Indien daar meer as een parameter in die parameterstring bestaan, sal die binêre string (dit wil sê, die *chromosoom*) die aaneenskakeling van alle individuele binêre stringe wees met die totale lengte van

$$\sum_{i=1}^{K} N(i) \qquad (6)$$

Byvoorbeeld, indien $\theta_{min} = 0$, $\theta_{maks} = 5000$, $A = 0.1$, dan $N = 3.6989/0.31010 = 12.28 \text{ -> } 13$ bisse; indien $x = 1011100001001 = 5896$, dan $m = (5000 - 0) / (8192 - 1) = 5896\ x\ 0.6610427 = 359.9$

Die individuele fiksheid, $F(x_i)$ word bereken as die individu se werkverrigting, $f(x_i)$ met betrekking tot dié van die broeivoorraad:

$$F(x_i) = f(x_i) / \sum_{i=1}^{N} f(x_i) \qquad (7)$$

waar N = broeivoorraad se grootte, en x_i die fiksheidswaarde van individu i verteenwoordig.

Hierdie metode ontstaan soos volg: gestel die waarskynlikheid dat 'n individu i vir die volgende generasie op grond van sy fiksheidsindeks geselekteer sal word is P_i. 'n Reeks van willekeurige getalle word gegenereer en vergelyk met die kumulatiewe waarskynlikheid C_i van die populasie:

$$C_i = \sum_{i=1}^{I} P_i \qquad (8)$$

waar I die populasiegrootte is. 'n Individu i sal geselekteer word en in die volgende populasie ingesluit word indien $C_{i-1} \leq C_i$. Daar is verskillende metodes om 'n sekere waarskynlikheid aan 'n individu toe te wy. Een van die gewildste is die sogenoemde *roulettewielmetode* waar die waarskynlikheid P_i vir 'n individu soos volg, gedefinieer word:

$$P_i = F_i \ / \ C_i = \sum_{i=1}^{I} F_i \qquad (9)$$

waar F_i die fiksheidsindeks van individu I beteken. In voortplanting word die individuele stringe gekopieer op grond van hulle fiksheidsindekse. Daar word verneem dat in die volgende generasie die populasie (broeivoorraad) se grootte dieselfde as dié van die vorige een gehou sal word.

Stringe (dit wil sê, individue) met hoër fiksheidsindekse sal 'n groter aantal kopieë verkry, terwyl stringe met lae waardes geëlimineer sal word. Die aantal kopieë word soos volg bereken:

$$N_i = N_{totaal}\ x\, f_i / f_i \qquad (10)$$

8. VOORBEELD

Die volgende voorbeeld van 'n eenvoudige funksiebenaderingsprobleem illustreer die algemene ontwerpstappe van 'n GA. Gestel die maksimum van die kostefunksie $f(x) = x^2$ word in die interval $0 < x < 31$ met die resolusie van 1 gesoek.

Indien die parameter x deur 'n binêre string verteenwoordig word, word die aantal bisse soos volg bereken:

$$L = log_2\,[(\ 31 - 0) / 1] = 5 \qquad (11)$$

'n Beginpopulasie $N = 4$ word willekeurig gekies deur 'n muntstuk $4\ .\ 5 = 20$ keer te werp.

01101; 11000; 01000; 10011

Elkeen van hierdie kodes het 'n ooreenkomstige desimale waarde: *13;24;8;19;*

Dekodeer die stringe deur substitusie in die kostefunksie om die fiksheidsindekse te verkry:

$$f(x_1) = 13^2 = 169; f(x_2) = 24^2 = 576; f(x_3) = 8^2 = 64;\ f(x_4) = 19^2 = 361;$$

Dus $\sum f = 1170$

Bereken die relatiewe fiksheid $f_i = f(x_i)/\sum f$:

$$f_1 = 169/1170 = 0.14; f_2 = 576/1170 = 0.492; f_3 = 64/1170 = 0.06; f_4 = 361/1170 = 0.31;$$

$\sum f_i = 1.00$; Gemiddeld: $= 0.25$

Voortplanting:

Die broeivoorraad van die volgende generasie word $\sum f_{ei} = 4$ gehou, en die individue met die hoogste fiksheid sal meer kopieë verkry.

$f_{e1} = 0.24 \cdot 4 = 0.56; \Rightarrow 1$ kopie; $f_{e2} = 0.49 \cdot 4 = 1.96; \Rightarrow 2$ kopieë; $f_{e3} = 0.06 \cdot 4 = 0.24 \Rightarrow 0$ kopieë; $f_{e4} = 0.31 \cdot 4 = 1.24;$ $\Rightarrow 1$ kopie. Na voortplanting is die nuwe generasie: *01101; 11000; 11000; 10011;*

Kruising
Broeipare asook 'n kruisingsbis word willekeurig gekies. Tabel 1 wys die resultate.

Mutasie
Gestel 'n mutasietempo $p_m = 0.001$; dus die verwagte aantal mutasies is $4 \cdot 5 \cdot 0.001 = 0.02$ Dus word geen mutasies binne hierdie generasie verwag nie.

Gevolgtrekkings
Binne een generasie het die gemiddelde fiksheid vanaf *293* na *439* en die maksimale fiksheid vanaf *576* na *729* vermeerder. Na slegs een generasie is die beste individu *27* wat reeds naby aan die oplossing van *31* is. Die oplossing tot die probleem word gevind deur op dieselfde wyse voort te gaan met die *GA*.

9. TOEPASSING IN INDUSTRIËLE BEHEERTEGNIEK

Meervoudigedoel- en meervoudige-eienskap-optimisasie-probleme kan suksesvol met behulp van GA opgelos word, soos byvoorbeeld die optimale instemming van 'n *PID*-beheerder waar die gelyktydige optimale waardes van die wins, asook die integraal- en differensiaaltydkonstante gesoek word. Buitendien word GA's ook in wasige beheerstelsels vir die outomatiese generering van reëls en lidmaatskapfunksies gebruik.

10. TOEPASSING IN BEDRYFSINGENIEURWESE

Die volgende praktiese voorbeeld is 'n vereenvoudigde weergawe van 'n skeduleringsprobleem wat in die plaaslike nywerheid voorgekom het.

'n Staalfabriek moes *10* kliëntebestellings van gietblokke *(A1 - A10)* per dag in 'n sekere volgorde verwerk. 'n Bestemde volgorde van gietblokke word as 'n *skedule* gekenmerk. Die keuse van dié volgorde is kritiek weens sy wesenlike invloed op die verwerkingskoste. Die rede hiervoor is die dikwels onvermydelike wanaanpassing tussen gietblokeienskappe wat 'n tydvertraging in die verwerking kan veroorsaak. In hierdie vereenvoudigde voorbeeld het elke gietblok slegs twee eienskappe: *wydte* en *afleweringstyd*. Gestel twee willekeurig gekose gietblokke word in die skedule as opeenvolgende prosesse ingesit. Ingeval die wydteverskil tussen die gietblokke 'n sekere drempelwaarde oorskry, moet die produksieproses gestaak en die toerusting verstel word om die grootteverskil te kan akkommodeer. Dus sal 'n sekere tydvertraging in die skedule ontstaan wat die produksiekoste aansienlik kan verhoog, veral as dit dikwels gebeur. Die doel van optimisasie is om die beste volgorde van gietblokke te bepaal waar die aantal van sulke wanaanpassings 'n minimum sal wees.

Die volgende eienskaptabel met relatiewe waardes is vir elke gietblok opgestel:

TABEL 2 Gietblokeienskappe

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
Wydte	1	2	1	4	8	5	3	9	7	4
Afl.tyd	7	1	6	2	7	3	2	8	1	9

'n Populasie van *50* is aanvaar. Die optimale skedule moet tipies binne 'n paar minute bereken kan word. Let wel dat die totale aantal permutasies van *10* gietblokke *A1...A10* gee *10! = 3,628,800* moontlike skedules, dus is daar nie tyd om alle moontlikes te ondersoek nie. GA is as optimisasiemetode gekies en binêre kodering met *22* bisse per individu was gebruik, waar *8* bisse vir die wydte, *8* vir die afleweringstyd, en *6* vir die uitset gebruik is. Die algebraïese kostefunksie is empiries soos volg, opgestel:

$$koste\,(gietblok_{k+1} - gietblok_k) = (wydte_{k+1} - wydte_k)^2 + (posisie_k - afleweringstyd_k)^n$$

waar *posisie*$_k$ = posisie in die skedule met verwysing na die gewenste afleweringstyd
$n = 2$ *indien* $posisie_k > afleweringstyd_k$ *(met ander woorde, die gietblok is alreeds laat)*
$n = 1$ *indien* $posisie_k$ £ $afleweringstyd_k$
$k = 1,\ldots,10$

Na die uitvoer van voortplanting, kruising en mutasie, het die beste fiksheidsindeks van *160* generasies tot *83* gekonvergeer en die ooreenkomstige optimale skedule (wat vir ons doeleindes nie belangrik is nie) is vanuit hierdie getal bepaal. Van meer belang is hoe doeltreffend hierdie genetiese algoritme was. Onder normale omstandighede is dit nie nodig om 'n kontrole-berekening uit te voer nie, maar as 'n bewys word die resultate hiervan voorgestel. Die koste van 'n oplossingsruimte van *10! = 3,628,800* skedules is geëvalueer. Daar was *530* skedules met 'n koste minder as *90*, met ander woorde, baie naby die optimum van *83*. Die belangrikste resultaat is egter dat hierdie aantal skedules slegs 0,015% van die oplossingsruimte verteenwoordig, dus was die optimum een van die top 0,015% van alle moontlike oplossings. Tweedens, die konvergensie is na slegs 160 generasies bereik, wat 'n baie spoedige optimisasieproses bewys.

TABEL 1 Kruising

Broei-voorraad	Maat	Kruisingsbis	Nuwe populasie	Reële waarde	Fiksheid $f(x)=x^2$
01101	2	4	01100	12	144
11000	1	4	11001	25	625
11000	4	2	11011	27	729
10011	3	2	10000	16	256
Som	1754				
Gemiddeld	439				
Maksimum	729				

11. OPSOMMING

Hierdie inleidende werk het bewys dat GA 'n belowende optimisasiewerktuig is wat in praktiese gebiede van ingenieurswese gunstig gebruik kan word. Daar is egter kommersieel beskikbare ontwikkelingsprogrammatuur, soos byvoorbeeld MATLAB™ wat vir meer ingewikkelde gevalle gebruik kan word. Meer inligting vir verdere gespesialiseerde navorsingsdoeleindes kan in die onderstaande literatuuropgawe nageslaan word, alhoewel dit slegs 'n beperkte seleksie aanbied. 'n Poging is aangewend om 'n Afrikaanse vakterminologie vir hierdie nuwe te vestig, wat in hierdie artikel vir die leser se kommentaar voorgestel is.

LYS VAN TERME

Afrikaans	**Engels**
genetiese algoritme	genetic algorithm
chromosoom	chromosome
voortplanting	reproduction
voortplantingstempo	reproduction rate
kruising	crossover
kruisingstempo	crossover rate
mutasie	mutation
mutasietempo	mutation rate
fiksheid	fitness
fiksheidsindeks	fitness factor
broeivoorraad	mating pool
robuust	robust
oorlewing	survival
generasie	generation
nageslag	offspring
populasie	population
meervoudige doel	multiobjective
meervoudige eienskap	multiattribute
aaneenskakeling	concatenation
enigeentoestand	don't care condition

SUMMARY

Genetic algorithms (GA), a relatively new approach to optimization, show a great promise. GAs are making their way into the mainstream of commercially applied technology. The idea is to imitate nature's way of doing research, i.e. via the natural selection process, or survival of the fittest.

A GA represents potential solutions on coded data structures known as *chromosomes*. An initial population of such structures that may also be considered as "individuals", is randomly generated. The GA evaluates the *fitness* of each individual (i.e. the "goodness of the solution"). The fitness function is equivalent to a *cost function*, totalling the costs (or profits) of a particular solution, with penalties for constraint violations. When the GA has evaluated all of the individuals in a population, *selection* and *breeding* occur: the algorithm creates a new generation of individuals (i.e. chromosomes) out of the genetic material of the previous population. GA researchers have derived numerous techniques for selection and breeding. Such approaches are, for example: transfer directly the best *10%* of chromosomes from the parent population to the next generation; or automatically exclude the worst *10%*; or compute the remaining *80%* by cross-breeding the parent chromosomes, where the probability of breeding is proportional to the fitness of each chromosome. The important thing is to apply selection pressures, as in nature, to favour the individuals with the highest fitness. Once the new generation is created, it is evaluated and the entire process is repeated. As the process continues, the fitness of the population increases. After many generations the best chromosomes evolve to near-optimal fitness.

The GA approach has several advantages over traditional optimization techniques. Firstly, it is very flexible. One can measure fitness as needed: a fitness criterion can include highly customized and unconventional definitions of objectives and constraints. As has been said before, the cost function does not necessarily have to be analytic: it can be a look-up table, piecewise linear, etc. Also, a GA can be fast, especially when it is necessary to sample a large solution space with many combinations of alternatives to consider. Finally, the GA can find a feasible solution quickly, and continue to improve its results the longer you let it run.

On the downside, we cannot say that the GA strictly guarantees optimality or, for that matter, feasibility for it is a statistical approach. In practice, this is not an issue as long as the problem is well-formulated and the solution correctly implemented.

The speed of the GA is particularly important, for example, in industrial scheduling operations. New orders, equipment problems, and other challenges may necessitate so-called "reactive scheduling" to incorporate new information into the production plan without excessive disruptions in production. The term "dynamic scheduling" refers to scheduling techniques that apply both predictively and reactively. This is a distinct advantage when, for example, compared with traditional PERT techniques.

GAs combined with object-oriented knowledge-based technology are proving to be a valuable tool. The technology has a bright future in a variety of industries that depend on batch processes.

LITERATUURVERWYSINGS

1. Greffenstette, J.J. (1986). Optimization of genetic algorithms, *IEEE Trans.Syst.Man.Cybern.*, SMC-16(1), 122-128.
2. Goldberg, D.E. (1989). Genetic algorithms in search, optimization and machine learning (Addison Wesley).
3. Karr, C. et al. (1995). Least median squares curve fitting using a genetic algorithm, *Engng.Appl.Artif.Intell.*, 8, 177-189.
4. Man, K.F., Tang, K.S., Kwong, S. (1996). Genetic algorithms: concepts and applications, *IEEE Trans.Industr.Appl.* 43(5), 519-533.
5. Shaw, I.S. (1996). Production scheduling by means of a fuzzy multiattribute decisionmaking model, *Elektron*, 13(8), 12-14.
6. Stachowitz, M., Yao, D., Chen, T. (1996). Tuning the PID controller based on a genetic algorithm. Lab for Intell. Sys., Dept Electr. and Comp. Eng., Univ. of Minnesota.
7. Lipton, M.J. et al. (1996). Genetic algorithms at work: two case studies, *PCAI Magazine*, 10,5, Phoenix, Arizona, USA.