8. kapitola

Evolučné stratégie

8.1. Historický úvod

Evolučná stratégia patrí historicky medzi prvé úspešné stochastické algoritmy. Bola navrhnutá už počiatkom 60-tych rokov Rechenbergom a Schwefelom, aj keď knižne publikovaná bola až neskoršie [1-3]. Podrobný popis je možné nájsť v poslednej Schwefelovej knihe [4], ale dobrý prehľad poskytnú aj časopisecké články, konferencie a správy [5-10]. Evolučná stratégia vychádza zo všeobecných predstáv prirodzeného výberu, no o mnoho vágnejších ako napríklad u genetického algoritmu. Naviac, na rozdiel od väčšiny ostatných stochastických metód, evolučná stratégia nie je založená na binárnej reprezentácii premenných, manipuluje priamo s "reálnou" reprezentáciou premenných (ako všade, aj tu existujú výnimky, diskrétna optimalizácia bola použitá napr. Herdym [11], ktorý smerodajnú odchýlku nahradzoval rôznymi parametrami určujúcimi "veľkosť" mutácie - u problému obchodného cestujúceho je to dĺžka podreťazca permutácie určujúcej poradie miest, ktorý bude invertovaný).

Ide teda o minimalizáciu funkcie $f(\boldsymbol{x})$, kde $\boldsymbol{x} \in \mathbf{R}^n$, teda funkcia má ako nezávislé premenné n reálnych čísiel zoradených do vektora $\boldsymbol{x}$. Prípustné hodnoty, ktorých môžu jednotlivé prvky (premenné) vektora $\boldsymbol{x}$ nadobúdať, môžu byť pritom obmedzené, napr. dolnou a hornou hranicou, $x_i \in \langle a,b \rangle$, teda $\boldsymbol{x} \in \langle a,b \rangle^n$.

Ingo Rechenberg, Hans-Paul Schwefel (teraz už obidva profesori) a Bienert sa začali v r. 1963 ako študenti na Technickej univerzite v Berlíne zaoberať experimentmi vo veternom tuneli, kde optimalizovali tvar telies, aby tieto kládli čo najmenší odpor v prúdení. Pretože ani intuitívny prístup, ani jednoduché gradientové metódy neboli veľmi úspešné, začali sa zaoberať myšlienkou náhodných zmien parametrov definujúcich tvar, podobne ako je tomu u mutácií.

Evolučné stratégie sa od tej doby rozvinuli, no stále ostávajú spojené predovšetkým s technickými aplikáciami v inžinierskych oblastiach, a používajú sa najviac v Nemecku. Prvé evolučné stratégie, ktoré si teraz opíšeme, používali ako jediný rekombinačný operátor mutáciu, a iba jedno riešenie v populácii, s maximálne dvoma "jedincami" uchovávanými súčasne - rodič a navrhovaný potomok. Taký prístup sa volá dvojčlenná evolučná stratégia, alebo (1+1)-ES.

Základom evolučnej stratégie je nasledujúci predpis, ktorý "mutuje" aktuálne riešenie $\boldsymbol{x}$ na nové riešenie $\boldsymbol{x}'$ (viď obr. 8.1 a 8.2),

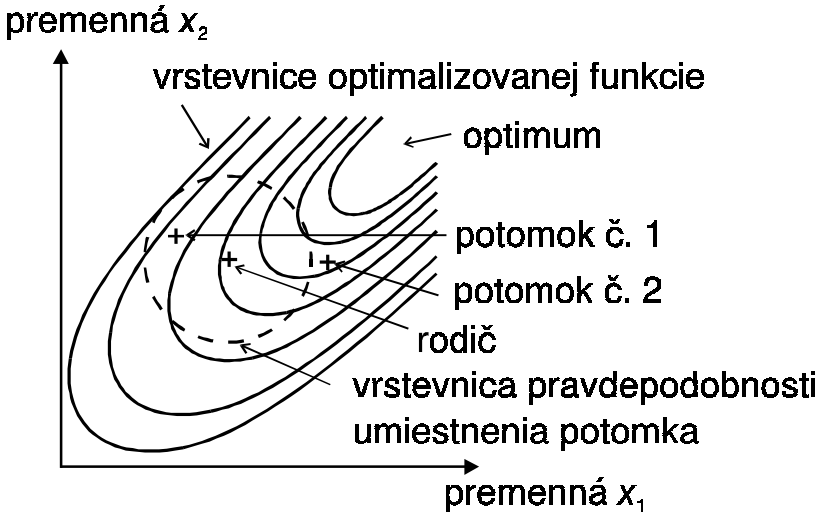
$$\boldsymbol{x}' = \boldsymbol{x} + \mathbf{N}(0,\sigma) , \tag{8.1}$$

kde $\mathbf{N}(0,\sigma)$ je vektor nezávislých náhodných čísel s nulovou strednou hodnotou a štandardnou odchýlkou σ. Tieto jednotlivé čísla sú náhodne distribuované podľa Gaussovej distribúcie, teda menšie čísla sa objavia viac pravdepodobne ako väčšie čísla. Štandardná odchýlka je merítkom rozptylu generovaných hodnôt, merítkom variácie náhodnej premennej. Určuje šírku dvojrozmerného "kopčeka pravdepodobnosti", ktorého základňu tvorí os možných hodnôt vygenerovaných čísel a ktorého výška v tom ktorom bode odpovedá pravdepodobnosti, s akou bude také číslo vygenerované. Tento "kopček pravdepodobnosti" sa volá hustotná funkcia a pre normálnu gaussovskú distribúciu je opísaný ako

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-0.5\frac{(x-\mu)^2}{\sigma^2}} \tag{8.2}$$

kde μ je stredná hodnota premennej x a σ je štandardná odchýlka. Čím väčšia je štandardná odchýlka, tým väčšia bude najskôr vzdialenosť medzi rodičom a vygenerovaným potomkom.

Problém akceptácie nového riešenia $\boldsymbol{x}'$ je striktne deterministický, riešenie $\boldsymbol{x}'$ je akceptované (úspešné), keď $f(\boldsymbol{x}') < f(\boldsymbol{x})$. Teda generujeme z rodiča po jednom potomkovi, ktorý sa líši skôr málo



Obrázok 8.1. Je riešená optimalizácia funkcie $f(x_1, x_2)$ dvoch premenných x_1 a x_2 znázornenej na obrázku neprerušovanými vrstevnicami rovnakých hodnôt funkcie, kedy v pravom hornom rohu je optimum. Čiarkovanou čiarou je označená vrstevnica rovnakej pravdepodobnosti vygenerovania potomka. To ale neznamená, že potomok bude niekde na tejto kružnici. Čím bližšie k rodičovi je nová pozícia, tým pravdepodobnejšie podľa Gaussovej krivky normálneho rozdelenia tam bude potomok umiestnený. Vygenerujeme napr. potomka č. 1. Pretože odpovedá horšiemu riešeniu ako rodič (čo je vidno z vrstevníc funkcie), zabudneme na tohto potomka a vygenerujeme ďalšieho potomka č. 2. Ten odpovedá lepšiemu riešeniu ako rodič, rodiča teda nahradíme potomkom č. 2 a ďalších potomkov už generujeme z potomka č. 2.

ako veľa od rodiča. Keď potomok odpovedá horšiemu riešeniu ako rodič, vymažeme potomka a generujeme iného. To robíme tak dlho, dokiaľ potomok neodpovedá lepšiemu riešeniu ako rodič a nezaujme jeho miesto. To prakticky odpovedá gradientovému prístupu alebo hillclimbingu.

8.2. Stratégia (1+1), teda jeden rodič a jeden potomok

Stratégia, kedy sa do ďalšej generácie vyberú lepší jedinci (tu iba jeden jedinec) tak z populácie rodičov, ako aj potomkov, sa volá PLUS stratégia. Pre jedného rodiča a jedného potomka je to teda (1+1) stratégia. Rechenberg robil základné pokusy na dvoch základných funkciách, teda na "sférickej funkcii", kde vrstevnice rovnakých hodnôt funkcie tvorí pre osi tvorené akýmikoľvek dvoma premennými sústredné kružnice $f(\boldsymbol{x}) = \sum_{i=1}^{n} x_i^2$, a pre koridorový model $f(\boldsymbol{x}) = -\sum_{i=1}^{n} x_i$, kde tvoria tieto vrstevnice rovnobežné čiary (pridané ohraničenia tvoria obdĺžnik, ku ktorého dlhšej strane sú vrstevnice kolmé). Pretože ide o zjednodušenú stratégiu [7], mohol byť Rechenbergom odvodený predpis na zmenu veľkosti "mutácie" - teda vzdialenosti potomka od rodiča, a na "zmenu tejto zmeny". Štandardná odchýlka σ sa v priebehu evolučnej stratégie mení podľa pravidla 1/5 úspešnosti. Nech $\varphi(k)$ je koeficient úspešnosti definovaný ako pomer počtu úspešných mutácií v priebehu posledných k iterácií k počtu k iterácií, z ktorých bola úspešnosť meraná, potom

$$\sigma' = \begin{cases} c_d . \sigma & (\varphi(k) < 1/5) \\ c_i . \sigma & (\varphi(k) > 1/5) \\ \sigma & (\varphi(k) = 1/5) \end{cases} \tag{8.3}$$

kde $c_i > 1$ a $c_d < 1$ riadia zväčšovanie resp. zmenšovanie štandardnej odchýlky, v literatúre [1,3,4] sú tieto koeficienty špecifikované c_d=0.82 a $c_i = 1/c_d$=1.22. Intuitívne zdôvodnenie pravidla 1/5 úspechu je zvýšenie efektívnosti prehľadávania: pri úspešnosti by malo prehľadávanie pokračovať vo väčších

krokoch, pri neúspešnosti by mali byť kroky kratšie. Toto pravidlo odvodené pre uvedené dve funkcie sa potom všeobecne používa aj pre ďalšie neznáme funkcie, pretože pre mnohé problémy jeho používanie pomáhalo udržať najrýchlejší postup k optimu. Algoritmus evolučnej stratégie v pseudopascale má tento tvar:

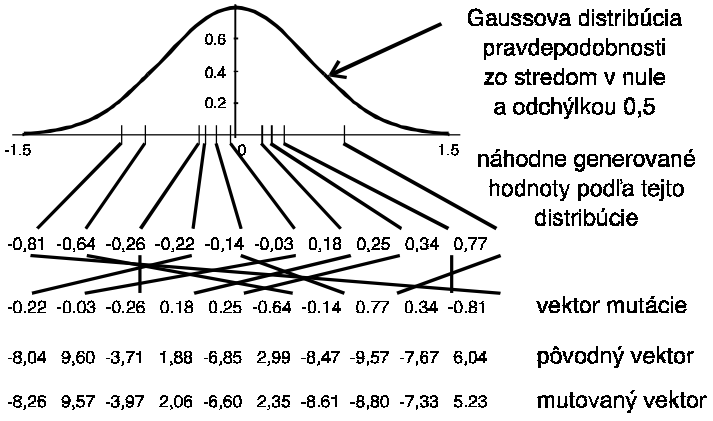
Evolučná stratégia:

```
1  x:=náhodne generovaný vektor reálnych premenných;
2  t:=0; σ:=σini; x*:=x;
3  WHILE t<tmax DO
4  BEGIN i:=0; k:=0;
5        WHILE i<imax DO
6        BEGIN i:=i+1; x':=x+r(0,σ);
7              IF f(x') < f(x) THEN
8              BEGIN k:=k+1; x:=x';
9                    IF f(x) < f(x*) THEN x*:=x;
10             END;
11       END;
12       IF k/imax<0.2 THEN σ:=cd*σ ELSE IF k/imax>0.2 THEN σ:=ci*σ;
13 END;
```

Algoritmus 8.1. Evolučná stratégia (1+1)

Premenná t je počítadlo epoch evolučnej stratégie. Algoritmus obsahuje dva WHILE-cykly, vonkajší a vnútorný. Vo vnútornom cykle sa pre dané σ opakuje elementárny krok evolučnej stratégie i_{max}-krát, pričom premenná k zaznamenáva úspešnosť mutácií v tomto vnútornom cykle. Vonkajší cyklus, s počítadlom t, aplikuje pre rôzne hodnoty σ evolučnú stratégiu t_{max} -krát. Štandardná odchýlka σ je v 2. riadku inicializovaná hodnotou σ_{ini}. V 6. riadku sa vykonáva modifikácia riešenia $\boldsymbol{x}$ pomocou generátoru náhodných čísel s nulovou strednou hodnotou a zo štandardnou odchýlkou σ. Voľba základných parametrov evolučnej stratégie (t_{max} , σ_{ini} , i_{max} a k_{max}) si vyžaduje určité experimentovanie, pomocou ktorého tieto konštanty nastavíme. Zvyčajne je σ_{ini} blízke jednotke, a konštanta i_{max} sa rovná rádovo tisícom.

Ako pre každý algoritmus, aj pre tento existujú drobné modifikácie. Napríklad zmena smerodajnej odchýlky sa môže robiť po každých n generáciách, začínajúc 10 n generáciami. Spočítame vždy, koľko úspešných potomkov, ktorí nahradili rodiča, bolo vygenerovaných za posledných 10 n generácií. Pokiaľ je to menej ako $2n$, vynásobíme veľkosť smerodajnej odchýlky tak,



Obrázok 8.2. Príklad mutácie vektora s reálnymi premennými za použitia Gaussovy distribúcie pravdepodobnosti pre určenie malej náhodnej zmeny (ne vždy je nutné mutovať všetky hodnoty vektora, stačí si náhodne vybrať, ktorý prvok sa bude mutovať).

aby sa zmenšila (napr. číslom 0.82), keď bolo viac ako 2*n* pokusov úspešných, potom násobíme smerodajnú odchýlku tak, aby sa zväčšila (napr. číslom 1.22).

Je ale vždy potrebné ohraničiť zmeny smerodajnej odchýlky tak, aby nedosiahla nulovú hodnotu pre danú presnosť.

Keď niektorá hodnota x_i nezapadá do svojho vymedzeného intervalu po pripočítaní vektoru mutácie k pôvodnému vektoru, t.j. $x_i \notin \langle a,b \rangle$, je potrebné použiť opravný proces, aby sa zaistilo, že opravená hodnota spadá do vymedzeného intervalu. Najjednoduchší taký opravný proces je tzv. zrkadlový odraz, kedy umiestnime hodnotu x_i na opačnú stranu hranice intervalu, v rovnakej vzdialenosti od hranice. Teda keď je napr. b=3.5 a x_i=4, potom nová hodnota x_i=3, teda je vzdialená 0.5 od hodnoty b, iba na opačnú stranu. Pokiaľ sa hodnota x_i dostane týmto presunom za hranice intervalu a na opačnej strane, postupujeme rovnako. Algoritmicky tento prístup vyjadrený procedúrou Perturbation nahradí výraz $\boldsymbol{x}'$:=$\boldsymbol{x}$+**N**(0,σ) v riadku 6 algoritmu 8.1. Symbol N(0, σ) tu vyjadruje jedno náhodné číslo vygenerované podľa Gaussovej distribúcie s nulovou strednou hodnotou a štandardnou odchýlkou σ.

```
PROCEDURE Perturbation(x,x',σ)
BEGIN FOR I:=1 TO N DO
        BEGIN x'i:= xi + N(0, σ);
                WHILE (x'i >b) OR (x'i <a) DO
                IF x'i > b THEN x'i := 2 b - x'i ELSE
                IF x'i < a THEN x'i := 2 a - x'i;
        END;
END;
```

Algoritmus 8.2. "Zrkadlenie hranice " - úprava hodnôt navrhovaného riešenia, keď algoritmus prekročí definičný obor optimalizovanej funkcie.

Príklad 8.1. Naprogramujte evolučnú stratégiu (1+1) a aplikujte ju pre optimalizáciu funkcie $y=x^2$, pričom σ_{ini}=1 a štartovný bod bude 100. Použite pravidlo zmeny σ 1/5 úspešnosti a nakreslite graf, kde na vodorovnej osi bude počet generácií a na zvislej momentálna hodnota rodiča. Porovnajte priebeh tohto grafu s obdobným grafom, kde sa v algoritme smerodajná odchýlka σ nebude meniť a ostane konštantná, teda rovná jednej.

8.3. Distribúcia pravdepodobnosti a generovanie náhodných čísel

Prečo je Gaussova distribúcia lepšia ako prehadzovania bitov v binárnom kóde? Majme reálne číslo z intervalu [-10,10] povedzme rovné nule, zakódované do 12 binárnych číslic.

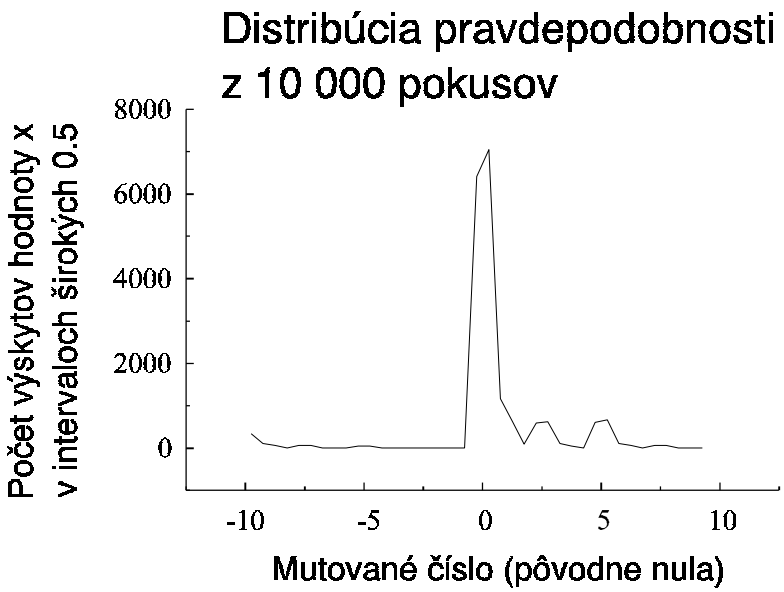
Bude transformované na $\frac{0-(-10)}{10-(-10)}\left(2^{12}-1\right)=2047.5$, čo bude binárne zakódované 100000000000

Mutáciou každého bitu by sme dostali čísla

000000000000 → -10.0
110000000000 → +5.00366
101000000000 → +2.50305
100100000000 → +1.25275
100010000000 → +0.27595
100001000000 → +0.15018
100000100000 → +0.5873
100000010000 → +0.805861
100000001000 → +0.41514
100000000100 → +0.21978
100000000010 → +0.1221
100000000001 → +0.0732601

Ako je možné vidieť, mutované čísla nemajú dobrú distribúciu okolo nuly. Nasledujúci príklad potvrdzuje, že to nie je náhoda, čiže tento problém existuje vo všeobecnosti.

Binárne číslo 100000000000 bolo mutované 10000 krát, každý bit v každej mutácii s



Obrázok 8.3. Distribúcia pravdepodobnosti výsledku mutácie binárneho čísla 100000000000 je veľmi nerovnomerná, a dokonca nie je symetrická! Prečo? Dôvodom je tzv. **Hammingova bariéra**: binárne čísla 100000000000 a 011111111111 sú susedmi v reálnej reprezentácii, no bolo by potrebné zmeniť hodnoty všetkých 12 bitov, aby sme z prvého čísla dostali to druhé.

```
help:=0;

FUNCTION returngaussdev: real;
BEGIN
    IF help = 0 THEN BEGIN
    REPEAT v1:=2.0*random() – 1.0;
           v2:=2.0*random() – 1.0;
           r:= v1^2 + v2^2;
    UNTIL (r<1.0);
    fac:= sqrt(–2.0*ln(r) / r)
    savegaussdev:=v1*fac;
    returngaussdev:=v2*fac;
    help:=1;
END
ELSE BEGIN
      help:=0;
      returngaussdev:= savegaussdev;
      END
END
```

Algoritmus 8.3. Generátor náhodnej premennej s normálnym (Gaussovským) rozdelením s nulovou strednou hodnotou a jednotkovou smerodajnou odchýlkou. Verzia podľa Numerical Recipes [12].

```
help:=0;

FUNCTION returngaussdev: real;
BEGIN
    IF help = 0 THEN BEGIN
    a:= sqrt(–2.0*ln(random()))
    b:=2 π random();
    savegaussdev:= a*cos(b);
    returngaussdev:= a*sin(b);
    help:=1;
END
ELSE BEGIN
      help:=0;
      returngaussdev:= savegaussdev;
      END
END
```

Algoritmus 8.4. Generátor náhodnej premennej s normálnym (Gaussovským) rozdelením s nulovou strednou hodnotou a jednotkovou smerodajnou odchýlkou. Jednoduchšia verzia (no vzhľadom k použitiu trigonometrických funkcií výpočtovo o niečo náročnejšia) používaná Schwefelom [4].

pravdepodobnosťou 1/12 (viac ako jeden bit mohol zmeniť svoju hodnotu). Po transformácii výsledných binárnych čísel na reálne čísla sme rozdelili interval [-10,10] do 40 oblastí a spočítali, koľko čísel sa nachádzalo v každej oblasti (viď obr. 8.3).

Ako dostať náhodné čísla s normálnym rozložením podľa Gaussovej distribúcie? Algoritmy 8.3 a 8.4 sa zakladajú na použití Box-Mullerovy metódy, implementáciu môžete nájsť v knihách Numerical Recipes (existujú pre Pascal, C, aj Fortran [12]). Tieto pseudopascalovské algoritmy vracia náhodnú premennú s nulovou strednou hodnotou a jednotkovou smerodajnou odchýlkou. Používajú pritom generátora náhodných premenných z intervalu (0,1] s uniformným rozložením - teda každé číslo z tohto intervalu dostanete s rovnakou pravdepodobnosťou. Pokiaľ je k dispozícii iba generátor náhodných prirodzených čísel, vygenerujte také číslo, preveďte na reálne číslo a vydeľte ho maximálnou možnou hodnotou dosiahnuteľnou generátorom, to by malo odpovedať výsledku funkcie random(). Funkcia vlastne vyrába vždy dve náhodné premenné s normálnym rozložením returngaussdev a savegaussdev. Prvú dáva ako výsledok hneď, druhú pri ďalšom volaní funkcie.

Zložitejší, no v mnohých prípadoch rýchlejší je tzv. trapézový algoritmus, opísaný v [13], ktorého kód vo fortrane je uvedený v [4].

Pretože ale generátor náhodných čísel s normálnou distribúciou je používaný často a podstatne ovplyvňuje časovú náročnosť výpočtu, obrátime pozornosť na logistickú distribúciu (viď obr. 8.4). Táto distribúcia je veľmi podobná normálnej distribúcii, ale je jednoduchšia na výpočet, a pomerne často nám vyhovujúco môže slúžiť ako jednoduchšia alternatíva. Formálne sa tvar logistickej distribučnej funkcie $f_\sigma(x)$, dá vyjadriť pomocou nasledovne, kde funkcia $\omega_\sigma(x)$ predstavuje odpovedajúcu hustotnú funkciu.

$$f_\sigma(x) = \frac{1}{1+e^{-x/\sigma}} \tag{8.4}$$

$$\omega_\sigma(x) = f'(x) = \frac{1}{\sigma}\frac{e^{-x/\sigma}}{\left(1+e^{-x/\sigma}\right)^2} \tag{8.5}$$

kde parameter σ s kladnou hodnotou (približne odpovedajúci štandardnej odchýlke u normálnej distribúcie) určuje "šírku" hustotnej funkcie $\omega_\sigma(x)$. Distribúcia $f_\sigma(x)$ mapuje celú reálnu os na otvorený interval (0,1), f_σ: R $\rightarrow$ (0,1). Môže byť použitá na jednoduchú konštrukciu generátora náhodných reálnych čísel zachovávajúcich logistickú distribúciu. Keď vyriešime rovnicu

$$r = \frac{1}{1+e^{-x/\sigma}} \tag{8.6}$$

pre $r \in$ (0,1) dostávame

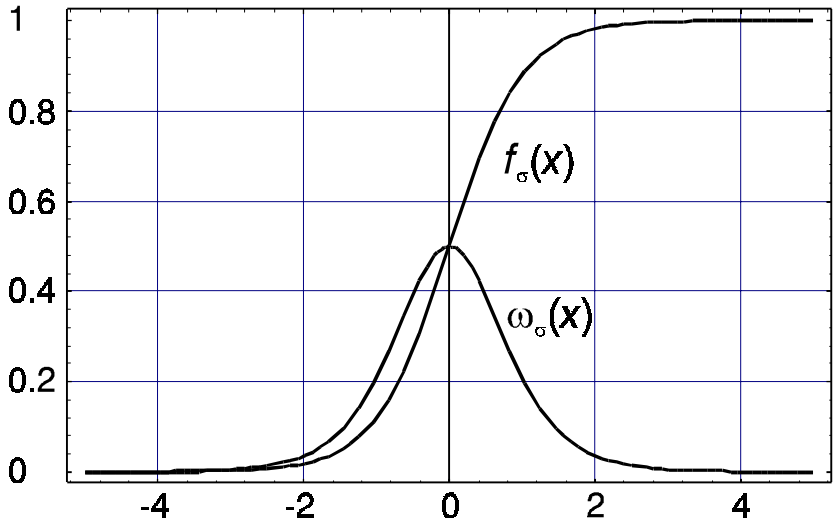
$$x = \sigma \ln\frac{r}{1-r} \tag{8.7}$$

Teda, keď r je náhodné číslo s uniformnou distribúciou, potom x určené pomocou (8.7) je náhodné číslo s nulovou strednou hodnotou, spĺňajúce logistickú distribúciu. Ako interpretovať a určiť parameter σ? Požadujeme, aby 100 p % náhodných čísel (pre 0<p<1) je generované z intervalu (–σ,σ), potom

$$\int_{-\rho}^{\rho} \omega_\sigma(x)\,dx = p \tag{8.8}$$

Pri využití vzorcov (8.4, 8.5) pre $f_\sigma(x)$ a $\omega_\sigma(x)$ po jednoduchom spracovaní dostaneme

$$\sigma = \rho\left(\ln\frac{1+p}{1-p}\right)^{-1} \tag{8.9}$$



Obrázok 8.4. Zobrazenie logistickej distribučnej funkcie $f_\sigma(x)$ a jej hustotnej funkcie $\omega_\sigma(x)$ pre $\sigma = 0$.

Napríklad, v prípade že p=0.99 (teda 99% generovaných náhodných čísel patrí do intervalu $(-\sigma,\sigma)$, potom

$$\ln\frac{1+p}{1-p} = \ln 199 \approx 5.3 \tag{8.10}$$

To zhruba znamená, že σ by malo byť približne päťkrát menšie ako ρ, za predpokladu, že 99% náhodných čísel patrí do intervalu $(-\sigma,\sigma)$.

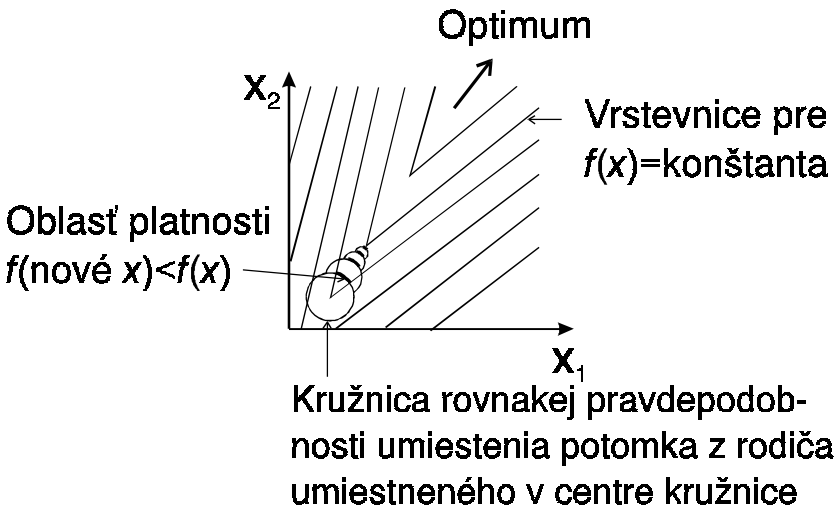
Podobne, ako pre simulované žíhanie, bolo dokázané aj pre evolučnú stratégiu [1,14], že potenciálne poskytuje globálny extrém optimalizovanej funkcie $f(\boldsymbol{x})$. Za predpokladu regulárnosti optimalizovaného problému (teda že optimalizovaná funkcia je spojitá a definičný obor funkcie tvorí uzatvorenú množinu a niekoľkých ďalších podmienok väčšinou splnených u inžinierskych problémov) je možné dokázať konvergenčný teorém (prehľad v [7], obecne [15, 16]). Tento konvergenčný teorém tvrdí, že globálne optimum bude dosiahnuté s jednotkovou pravdepodobnosťou; nehovorí ale, za ak dlho generácií.

8.4. Viacčlenné stratégie a kovariancie

Schwefelom so spolupracovníkmi [3-4,6] boli navrhnuté ďalšie sofistikovanejšie verzie evolučnej stratégie, takže v súčasnosti je možné už hovoriť o celej triede evolučných stratégií. Odpadá tu nutnosť zavádzania pravidiel pre zmeny štandardnej odchýlky, čo môže viesť k predčasnému ukončeniu optimalizácie, ako je tomu na obrázku 8.5.

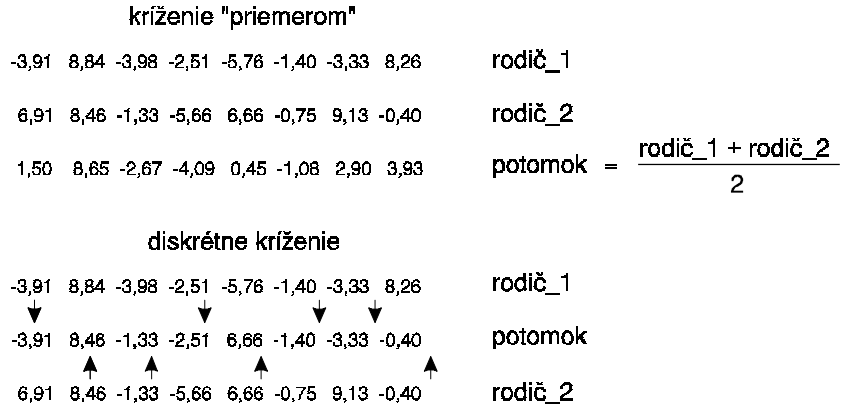
Schwefel novšie zaviedol viacčlenné evolučné stratégie, označované $(m+l)$ alebo (m,l), ktoré imitujú ďalšie základné princípy organickej evolúcie. Pracuje sa tu potom s celým súborom rodičovských vektorov $\boldsymbol{x}$, ktorých počet je m. V prípade PLUS stratégie $(m+l)$ vygenerujeme z týchto rodičov / potomkov. Potom takto navrhnutých vektorov - potomkov a pôvodných "rodičovských" vektorov dovedna zoradíme podľa optimálnosti riešení, a prvých m najlepších jedincov - riešení zoberieme ako rodičov do ďalšej populácie. V prípade "ČIARKA" stratégie (m,l) vyberáme budúcich m rodičov iba z momentálne vygenerovaných / potomkov, a rodičia týchto potomkov vymierajú. Život každého jedinca je obmedzený iba na jednu generáciu. Okrem mutácie je u obidvoch stratégií používané kríženie, teda čiastočná výmena informácií medzi vektormi reálnych čísel (viď obr. 8.6), na rozdiel od kríženia bitových reťazcov používaného u ďalej preberaných genetických algoritmov.

Kríženie dvoch alebo viacerých rodičov vždy produkuje jedného potomka, na rozdiel od genetických algoritmov, kedy dvaja rodičia generujú vždy dvoch potomkov. Rovnako ako u genetických algoritmov môže byť chromozóm vybraný ako rodič aj niekoľkokrát.



Obrázok 8.5. Ukážka zlyhania prístupu zmenšovania štandardnej odchýlky riadenej pravidlom 1/5 úspešných pokusov. Toto pravidlo bolo pôvodne vytvorené, aby algoritmus rýchlo našiel minimum, keď sa dostane do jeho blízkosti. Vtedy nájde menej ako 1/5 lepších riešení, a teda so zmenšením štandardnej odchýlky začne prehľadávať bližšie okolie. Keď má ale funkcia "úzke údolie", alebo sú aktívne iné ohraničenia, dĺžka kroku bude stále klesať bez toho, že by sa zvyšovala úspešnosť algoritmu, a ten potom "zamrzne" ďaleko od minima.

Najväčším "objavom" evolučnej stratégie je ale možnosť nebrať jednu hodnotu štandardnej smerodajnej odchýlky pre všetky hodnoty optimalizovaného vektora reálnych čísel $\boldsymbol{x}$, no uchovávať pre každú premennú z vektora $\boldsymbol{x}$ vlastnú hodnotu smerodajnej odchýlky (rozptyl). Každý jedinec je teda tvorený dvoma vektormi: vektorom $\boldsymbol{x}$ a vektorom smerodajných odchyliek $\boldsymbol{\sigma}$. Nesie si tak zo

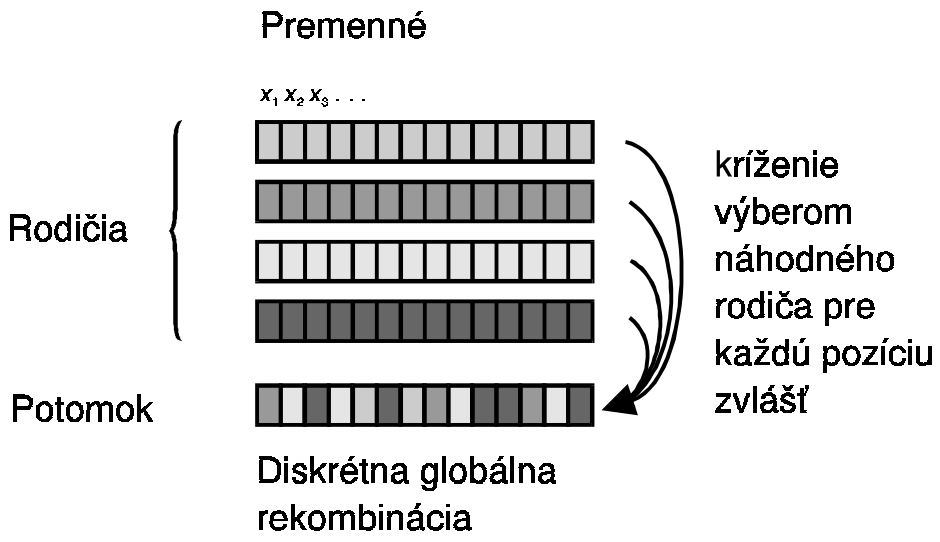


Obrázok 8.6. Dva z mnohých druhov kríženia používaných u evolučnej stratégie. Kríženie "priemerom" zoberie dva vektory, zosumuje hodnoty ich prvkov na odpovedajúcich miestach a vydelí celý vektor dvoma. Kríženie "diskrétne" preberá hodnoty na odpovedajúcich miestach náhodným výberom z jedného alebo druhého vektora "rodičovských jedincov".

sebou aj informáciu o distribúcii pre novú mutáciu. Mutácie sú obecne robené rovnako ako u stratégie (1+1) s pomocou štatistického rozdelenia tak, že drobné zmeny sú viac pravdepodobné a závažné zmeny nepravdepodobné. Táto tendencia sa s časom zvyšuje spolu s približovaním populácie k optimu, teda blízko optima sú mutácie už len malé. Ako ale poznať, že sme už blízko optima? Viacčlenné evolučné stratégie túto otázku obchádzajú meta-evolučnou samoadaptačnou technikou, kedy aj veľkosť mutácií sa v priebehu algoritmu mení. Evolúcia teda nebude prebiehať iba u premenných určujúcich riešenie, ale súčasne sa bude vyvíjať informácia o tom, ako každé z riešení bude generovať nového potomka, tzv. strategické premenné. Keď pri súčasnej hodnote jednej premennej jej zmena neprináša veľké efekty na hodnotu optimalizovanej funkcie, potom aj hodnota smerodajnej odchýlky priradenej tejto premennej sa vyvinie ako veľká. Teda, zmeny majú iné merítko u premenných, ktoré majú väčší vplyv na hodnotu funkcie ako iné premenné. Každý nový jedinec je ohodnotený účelovou funkciou. Učenie prebieha na dvoch úrovniach, jedna úroveň hodnotí dosiahnuté výsledky funkcie, druhá úroveň predstavuje schopnosť jedinca učiť sa, zakódovanú v smerodajnej odchýlke určujúcej veľkosť mutácie (aj keď táto schopnosť sa zhodnocuje len nepriamo pomocou ohodnotení výsledkov funkcie).

Pri tejto použití smerodajnej odchýlky pre každú premennú poskytuje väčšinou lepšie výsledky "čiarková" stratégia, kde populácia rodičov celkom vymiera. Aj keď momentálne môže dojsť k zabudnutiu najlepšieho doteraz nájdeného riešenia, v dlhodobom merítku táto stratégia funguje lepšie, lebo sa dostane cez lokálne minimá, kedy nejaké riešenie má síce momentálne vysokú hodnotu optimalizovanej funkcie, ale nastavenie smerodajných odchyliek je zlé. Zabúdanie rodičovských chromozómov preferuje vybudovanie vnútornej štruktúry - teda smerodajných odchyliek - optimálne nastavených na ďalší vývoj. Takýto prístup potom funguje lepšie pre optimalizáciu zašumených alebo časovo premenných funkcií. Môžeme si predstaviť hodnoty vektora $\boldsymbol{x}$ ako analógiu fenotypu a hodnoty smerodajných odchyliek ako genotyp. Výberom určitého pomeru počtu rodičov a potomkov *m/l* je možné nastavovať rýchlosť konvergencie evolučnej stratégie. Keď chceme rýchlu, no lokálnu konvergenciu, zvolíme si silne reštriktívnu selekciu, napr. (5,100). Keď hľadáme globálne optimum, musíme selekciu zmierniť, napr. (15,100). Podľa sférického modelu je ideálny pomer rodičia/potomkovia asi 1/7, no rodičov musí byť pritom rádovo aspoň desať.

Rodičia pre kríženie sa vyberajú z celej populácie s rovnakou pravdepodobnosťou, nezáleží teda na optimálnosti riešenia (konvergencia je dostatočne zabezpečená selekciou potomkov pre



Obrázok 8.7. Globálne uniformné kríženia (alebo ináč globálna rekombinácia), kedy potomok má viac ako dvoch rodičov (neexistovalo by rozlíšenie na "mužské" a "ženské" pohlavie, uvedený prístup by odpovedal viacero rovnocenným pohlavím). Každý prvok vektora riešenia (premenná) je braný z náhodne zvoleného rodičovského vektora.

ďalšiu generáciu rodičov). Pre kríženie sa väčšinou používa "uniformného kríženia", známeho z analógie z genetických algoritmov, kde potomok preberá hodnoty postupne z jedného alebo druhého rodiča s rovnakou pravdepodobnosťou (viď. obr. 8.6). Kríženie teraz už zahŕňa nielen výmenu hodnôt vektora $\boldsymbol{x}$, no tiež hodnôt vektora odpovedajúcich smerodajných odchyliek σ.

Ďalším z možných spôsobov krížení je kríženie priemerom, ktoré ale často vedie k príliš rýchlej lokálnej konvergencii.

Najnovšie sa objavili ďalšie druhy kríženia, ktoré boli použité ak v genetických algoritmoch, tak v evolučnej stratégii súčasne [17]. Dva vektory $\boldsymbol{x}_1$ a $\boldsymbol{x}_2$ produkujú dvoch potomkov $\boldsymbol{y}_1$ a $\boldsymbol{y}_2$, ktoré sú lineárnou kombináciou svojich rodičov

$$\begin{aligned} \boldsymbol{y}_1 &= \mathrm{a}\,\boldsymbol{x}_1 + (1\text{-a})\,\boldsymbol{x}_2 \\ \boldsymbol{y}_2 &= (1\text{-a})\,\boldsymbol{x}_1 + \mathrm{a}\,\boldsymbol{x}_2 \end{aligned} \tag{8.11}$$

Okrem tohto kríženia bolo ešte navrhnuté diskrétne globálne uniformné kríženie, ktoré si hodnoty vektorov $\boldsymbol{x}$ a $\boldsymbol{\sigma}$ nevyberá postupne náhodne z dvoch vektorov, no môže si vyberať zo všetkých rodičovských vektorov (viď. obr. 8.7). Obdobou "kríženia priemerom" pre globálne uniformné kríženie je použitie hodnôt jedného rodiča pre všetky premenné, zatiaľ čo druhý rodič sa vyberá pre každú premennú zvlášť. Podľa Schwefelových skúseností [2,3] sa osvedčilo diskrétne kríženie s dvoma rodičmi pre premenné x_i spolu s krížením priemerom (prípadne aj jeho globálnym variantom) pre smerodajné odchýlky a uhly α opísané ďalej.

Pre zmenu smerodajných odchyliek sa pri viacčlenných evolučných stratégiách už nepoužíva deterministického algoritmu, ako bolo pravidlo 1/5 úspešnosti. Namiesto toho sú smerodajné odchýlky podrobené evolučnému procesu. Každá smerodajná odchýlka je teraz mutovaná

$$\sigma_i' = \sigma_i \exp(N(0, \Delta\sigma_0)) \tag{8.12}$$

kde $N(0, \Delta\sigma_0)$ je náhodná premenná vygenerovaná s normálnym rozdelením so stredom v 0 a smerodajnou odchýlkou $\Delta\sigma_0$. Táto smerodajná odchýlka $\Delta\sigma_0$ je parametrom metódy, konštantou, ktorá je rovnaká pre všetky smerodajné odchýlky všetkých premenných v celom časovom priebehu algoritmu. Novo vytvorená smerodajná odchýlka σ_i' potomka, príslušná nejakej i-tej hodnote x_i vo vektore premenných $\boldsymbol{x}$ sa hneď použije na vygenerovanie novej hodnoty

$$x_i' = x_i + N(0, \sigma_i'). \tag{8.13}$$

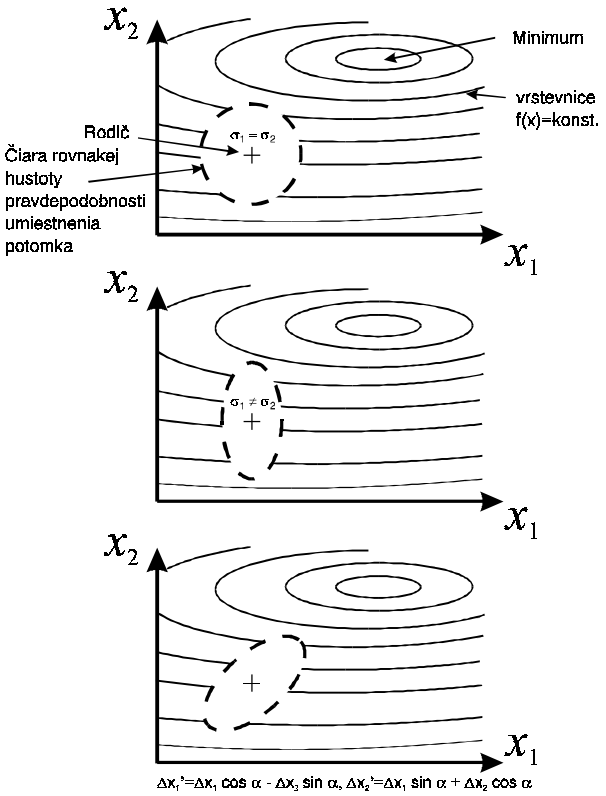
Okrem vlastnej hodnoty premennej vo vektore a jej smerodajnej odchýlky môže byť súbor premenných charakterizovaný aj ďalším vektorom "strategických" premenných, ktorý riadi koreláciu mutácií. Pre dve premenné s rôznymi hodnotami smerodajnej odchýlky vrstevnice pravdepodobnosti umiestenia mutovaného vektoru nepredstavujú kružnicu, ale elipsu. Táto elipsa však je orientovaná v smere súradnicových os. Môžeme si ale predstaviť, že optimum nie je umiestnené v smeru dlhšej osy elipsy, ale niekde naprieč. Ideálne by potom bolo, keby sme mohli túto elipsu vrstevníc pravdepodobností umiestnenia mutovaného vektoru natočiť tak, aby hlavná os elipsy smerovala k optimu. Tak by mutovaný vektor mal najväčšiu šancu čo najviac sa priblížiť k optimu (viď obr. 8.8). Toto natočenie ide zaistiť kovarianciami. Vektor "strategických" premenných teda môže pre n-rozmerný vektor $\boldsymbol{x}$ zahŕňať n rozptylov $c_{ii} = \sigma_i^2$ rovnako ako $n(n\text{-}1)/2$ kovariancií c_{ij} zobecnenej n-rozmernej normálnej distribúcie s hustotou pravdepodobnosti vektoru mutácií $\boldsymbol{z}$

$$p(\boldsymbol{z}) = \frac{\exp\left(-\frac{1}{2}\boldsymbol{z}^T \boldsymbol{C}\,\boldsymbol{z}\right)}{\sqrt{(2\pi)^n \cdot \det \boldsymbol{C}}} \tag{8.14}$$

Pre zaistenie kladnosti a konečnosti kovariančnej matice $\boldsymbol{C}^{-1} = c_{ij}$ algoritmus používa odpovedajúce rotačné uhly α_j ($-\pi \le \alpha_j \le \pi$) namiesto koeficientov c_{ij}. Tieto uhly sú v nasledovnom vzťahu k varianciám a kovarianciám:

$$\tan(2\alpha_{ij}) = \frac{2c_{ij}}{\sigma_i^2 - \sigma_j^2} \tag{8.15}$$

Osi mutačných elipsoidov (povrchu rovnakej hustoty pravdepodobnosti umiestnenia potomka vytvoreného mutáciou z rodiča umiestneného v strede elipsoidu) sú potom rovnobežné s koordinačnými osami iba ak kovariančná matica je diagonálna. V obecnejšom prípade nenulových kovariancií môže byť elipsoid ľubovolne natočený v priestore riešení a mutácie premenných sú



Obrázok 8.8. Prvý graf ukazuje čiary rovnakej hustoty umiestnenia potomka z rodičovského chromozómu umiestneného v strede čiarkovaného oválu. Keď sa štandardné odchýlky pre obidve premenné rovnajú, čiara je tvorená kružnicou. No priali by sme si, aby potomok mal väčšiu šancu byť umiestnený v smere minima označeného sústrednými vrstevnicami. Keď zmeníme štandardné odchýlky, dostaneme z kružnice určujúcej pravdepodobné umiestnenie potomka elipsu. No osy tejto elipsy sú stále rovnobežné zo súradnicovými osami. Natočiť elipsu tak, aby svojou hlavnou osou smerovala k minimu dokážeme iba pootočením o uhol α, ktoré je realizáciou kovariancie.

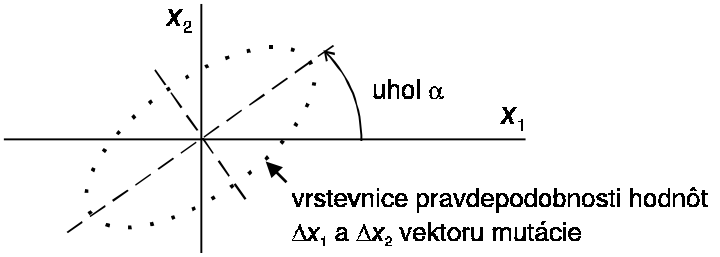
lineárne korelované. Takto vytvorení potomkovia sa môžu adaptovať na ľubovoľný výhodný smer prehľadávania, čo je výhodné v prípade existencie úzkych údolí v optimalizovanej funkcii.

Mutácie, ktoré berú do úvahy aj kovarianciu sú robené nasledovne

$$\begin{aligned} \sigma_i' &= \sigma_i \cdot \exp(\tau_0 \cdot N(0,1) + \tau N_i(0,1)) \\ \alpha_j' &= \alpha_j + \beta \cdot N_j(0,1) \\ x_i' &= x_i + z_i(\sigma', \alpha') \end{aligned} \qquad (8.16)$$

Notácia N(0,1) označuje vygenerovanú normálne distribuovanú jednorozmernú náhodnú premennú pri nulovej strednej hodnote a jednotkovej smerodajnej odchýlke, zatiaľ čo $N_i(0,1)$ označuje, že náhodná premenná je vygenerovaná vždy znova pre každú hodnotu čítača *l*. Táto sada vzorcov je použitá pre $\forall i \in \{1,...,n\}$, $\forall j \in \{1,...,n\cdot(n-1)/2\}$. Globálny faktor $\exp(\tau_0 \cdot N(0,1))$ dovoľuje celkovú zmenu veľkosti všetkých mutácií, zatiaľ čo $\exp(\tau \cdot N_i(0,1))$ povoľuje individuálnu zmenu strednej veľkosti kroku σ_l. Výraz $\exp(\tau \cdot N_i(0,1))$ je teda individuálnym parametrom generovaným zvlášť pre každú premennú z vektora $\boldsymbol{x}$, dovoľujúc tak individuálne zmeny "priemerných" zmien σ_i každej premennej x_i vektora $\boldsymbol{x}$.

Týmto spôsobom sú mutácie premenných korelované pomocou hodnôt vektora α (viď obr. 8.9). Konštanty τ_0, τ a β sú parametre, ktoré Schwefel [2,3] navrhuje nastaviť $\tau_0 \propto 1/\sqrt{2\sqrt{n}}$ a $\tau \propto 1/\sqrt{2n}$ a $\beta \approx 0.873$ ($=5^\circ$ v radiánoch). Pri použití kovariancií, teda uhlov, nemusíme brať do úvahy vzťahy medzi všetkými premennými, teda počet uhlov môže byť aj menší ako n $(n-1)/2$. To by ale



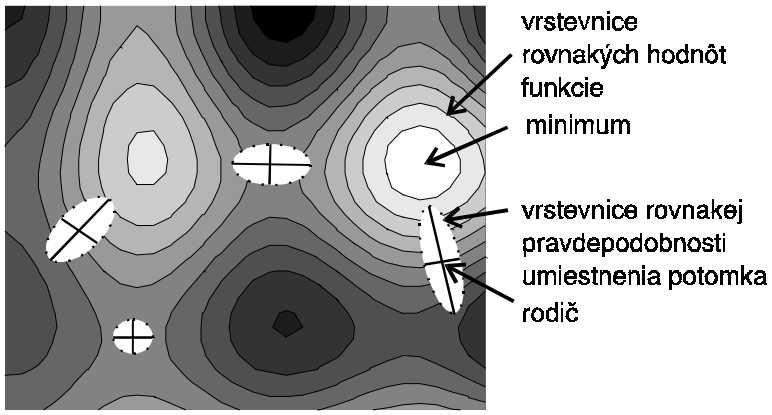
Obrázok 8.9. Pootočenie elipsy rovnakých hodnôt pravdepodobnosti vygenerovania zmien Δx_1 a Δx_2.

vyžadovalo predbežnú analýzu alebo optimalizáciu výberu uhlov, ktoré je možné zanedbať. Konštanty τ_0, τ môžu byť interpretované ako "rýchlosti učenia", podobne ako je tomu u neurónových sietí. Najlepšie nastavenie týchto parametrov môže závisieť od druhu optimalizovanej funkcie. Rovnako ako u (1+1) stratégie, smerodajné odchýlky musia ostať väčšie ako nejaká minimálna nenulová hodnota. Schwefel požaduje splnenie podmienky $\sigma_i \geq \varepsilon |x_i|$, kde $\varepsilon>0$ je malé kladné číslo. Pokiaľ uhly α_j' prekročia hranice $(-\pi,\pi)$, používame zrkadlového odrazu podobne ako v procedúre PERTURBATION opísanej pri (1+1) stratégii.

Žiaľ, generovať náhodný vektor mutácií s členmi $z_i(\sigma',\alpha')$ uvedenými v (8.16) s distribúciou pravdepodobnosti $p(\mathbf{z})$ uvedenou v (8.14) nemusí byť triviálnym problémom (je obtiažne garantovať ortogonálnosť výsledného koordinačného systému, čo odpovedá pozitívne definitnej koordinačnej matici). V praxi sa tento problém nahradzuje postupnou rotáciou. Povedzme, že máme dve premenné x_1 a x_2 zo smerodajnými odchýlkami σ_1 a σ_2, takže vrstevnice rovnakých hodnôt pravdepodobnosti by vytvárali elipsu s osami rovnobežnými s hlavnými osami. Korelačný koeficient c_{12} odpovedá uhlu α, o ktorý sa táto elipsa pravdepodobnosti pootočí (viď obr. 8.9, 8.10 a 8.11). Takže keď sú pôvodné odchýlky premenných x_1 a x_2 so smerodajnými odchýlkami σ_1 a σ_2 rovné Δx_1 a Δx_2, potom odchýlky upravené pomocou korelačného koeficientu majú tvar $\Delta x_1' = \Delta x_1 \cos\alpha - \Delta x_2 \sin\alpha$, $\Delta x_2' = \Delta x_1 \sin\alpha + \Delta x_2 \cos\alpha$.

Pre tri premenné by museli byť urobené tri následné rotácie, v rovine $(\Delta x_1, \Delta x_2)$ o uhol α_1 s výsledkom $\Delta x_1'$ a $\Delta x_2'$, v rovine $(\Delta x_1', \Delta x_3)$ o uhol α_2 s výsledkom $\Delta x_1''$ a $\Delta x_3'$, a v rovine $(\Delta x_2', \Delta x_3')$ o uhol α_1 s výsledkom $\Delta x_2''$, $\Delta x_3''$. Výsledné zmeny premenných by teda boli $\Delta x_1''$, $\Delta x_2''$ a $\Delta x_3''$.

Pre viac premenných sa daný prístup jednoduchšie vyjadrí pomocou rotačných matíc. Každá matica odpovedá jednému uhlu, teda jednej kovariancii dvojice premenných vektora $\boldsymbol{x}$. Budeme mať



Obrázok 8.10. Zobrazenie vrstevníc hodnôt funkcie dvoch premenných s optimami označenými svetlou farbou. Biele bodkované ovály označujú miesta rovnakej pravdepodobnosti umiestenia potomka vzniklého mutáciou z rodiča umiestneného v stredu oválu označeného krížením os. Dlhšia os oválov je nasmerovaná smerom k optimu, tak ako sa to "jedinec" naučil v priebehu optimalizácie, kedy každá z premenných si uschováva vlastnú hodnotu smerodajnej odchýlky, a ešte sa uschováva výhodný smer korelovaných odchýlok.

teda n (n–1)/2 rotačných matíc $R(\alpha_{ij})=(r_{kl})$. Tieto matice sú jednotkové, teda na hlavnej diagonále majú samé jedničky a inde samé nuly, s výnimkou štyroch prvkov: $r_{ii} = r_{jj} = \cos \alpha_{ij}$, a $r_{ij} = - r_{ji} = - \sin \alpha_{ij}$. Trigonometrické výrazy sú teda umiestnené vždy v priesečníkoch riadkov a stĺpcov i a j. Násobenie týmito maticami odpovedá vlastne transformácii súradníc vzhľadom k osám i a j pri otočení o uhol α_{ij}. Môžeme násobiť vždy prvú maticu $R(\alpha_{1,2})$ s vektorom odchýlok premenných (Δx_1, Δx_2, ..., Δx_n) premenných x_1, x_2, ... , x_n so smerodajnými odchýlkami σ_1, σ_2, ..., σ_n , a potom násobíme ďalšiu maticu $R(\alpha_{2,3})$ s výsledným vektorom predchádzajúceho násobenia (Δx_1', Δx_2', Δx_3, ..., Δx_n), atď. Jednoduchšie je ale najskôr vynásobiť všetky matice medzi sebou a až výsledok násobiť vektorom (Δx_1, Δx_2, ..., Δx_n). Formálne sa to dá vyjadriť vzorcom

$$\text{rotácia}(\Delta x_1, \Delta x_2, \ldots, \Delta x_n) = \left(\prod_{i=1}^{n-1} \prod_{j=i+1}^{n} \boldsymbol{R}(\alpha_{ij}) \right) \cdot (\Delta x_1, \Delta x_2, \ldots, \Delta x_n) \qquad (8.17)$$

Výskum Rudolpha [18] potvrdzuje správnosť tohto prístupu, pretože ukázal, že tento prístup garantuje ortogonálnosť transformácií a súčasne umožňuje vytvorenie akýchkoľvek ortogonálnych transformácií. Použitie rotačných uhlov teda nijako neobmedzuje možnosť vygenerovať akúkoľvek mutáciu.

Evolučná stratégia: pokročilá viacčlenná verzia

1 t=0; σ:=σ_{ini};
2 P_0:={náhodne generovaná populácia chromozómov (=vektorov reálnych premenných spolu so štandardnými odchýlkami a prípadne variancemi)}
3 Ohodnotenie každého chromozómu z populácie P_0 odpovedajúcou funkčnou hodnotou;
4 **WHILE** $t<t_{max}$ **OR** dostatočne dobré riešenie nie je nájdené **DO**
5 **BEGIN** t:=t+1;
6 Q={nové chromozómy vytvorené krížením a vždy použitou mutáciou z náhodne vybraných chromozómov z P_{t-1} };
7 Ohodnotenie každého chromozómu z Q odpovedajúcou funkčnou hodnotou;
8 P_t:={najlepšie chromozómy z Q; tiež stratégia výberu najlepších z $P_{t-1} \cup Q$ je prípustná;}
9 **END**;

Algoritmus 8.5. Evolučná stratégia typu (m,n) alebo (m+n) v závislosti na tvorbe P_t. Počet chromozómov podpopulácie Q je ohraničený podmienkou $|Q|>=|P_t|$

Už zmienená viacčlenná evolučná stratégia dovoľuje každému členu vytvoriť niekoľko potomkov. Z tých ("ČIARKA" stratégia), a prípadne ešte z pôvodných členov - rodičov (PLUS stratégia), sa vyberie ďalšia skupina rodičov pre novú generáciu na základe ich zoradenia pomocou im odpovedajúcich hodnôt minimalizovanej funkcie. Pseudokód je uvedený v algoritmu 8.5.

Pri krížení nekombinujeme hodnoty premenných x_1, x_2, ... , x_n, no aj strategické premenné, teda smerodajné odchýlky σ_1, σ_2, ..., σ_n a uhly α_1, α_2, ..., $\alpha_{n\cdot(n-1)/2}$. Môžu byť pritom použité rôzne typy rekombinačných operátorov pre hodnoty premenných, smerodajné odchýlky a uhly. Štandardné nastavenie parametrov evolučnej stratégie je σ_{ini} = 3 pre funkcie, o ktorých nič nepoznáme, počet rodičov $|P_t|$ = 15, počet potomkov $|Q|$ = 100, stratégia výberu je "čiarka", teda P_t:={najlepšie chromozómy z Q}. Pokiaľ sa týka smerodajných odchyliek a uhlov, najjednoduchšie je použiť iba jednu smerodajnú odchýlku pre celý chromozóm s diskrétnym krížením. O trocha zložitejšie funkcie vyžadujú použitie celého vektora smerodajných odchyliek pre každý chromozóm, zasa s diskrétnym krížením, ale ešte bez použitie uhlov α. Konečne najkomplexnejší algoritmus používa aj $n\cdot(n-1)/2$ uhlov α a kríženia priemerom medzi týmito uhlami.

8.5. Ukážka aplikácie algoritmu na Rosenbrockovu funkciu

Ako príklad riešenia optimalizačného problému pomocou evolučnej stratégie si tu uvedieme riešenie Rosenbrockovej funkcie [19]. Táto funkcia dvoch premenných má hlboké "údolie", kde iba v jednej časti (v bode x_1=1, x_2=1) sa nachádza minimum, viď obr. 8.11. Takáto funkcia by sa ľahko riešila gradientovou metódou, no napr. pre genetický algoritmus predstavuje táto funkcia "nočnú moru", pretože genetický algoritmus nemá nastavený smer mutácií = smer pohybu v priestore riešení, a nie je schopný "zahýbať". Genetický algoritmus by pomerne rýchlo našiel údolie vyznačené na obrázku 8.11 čiarkovanou čiarou, no nebol by už schopný dostať sa v rozumnom čase do globálneho optima. Funkcia je definovaná nasledovne:

$$f(\boldsymbol{x}) = 100(x_2 - x_1^2)^2 + (x_1 - 1)^2 \tag{8.18}$$

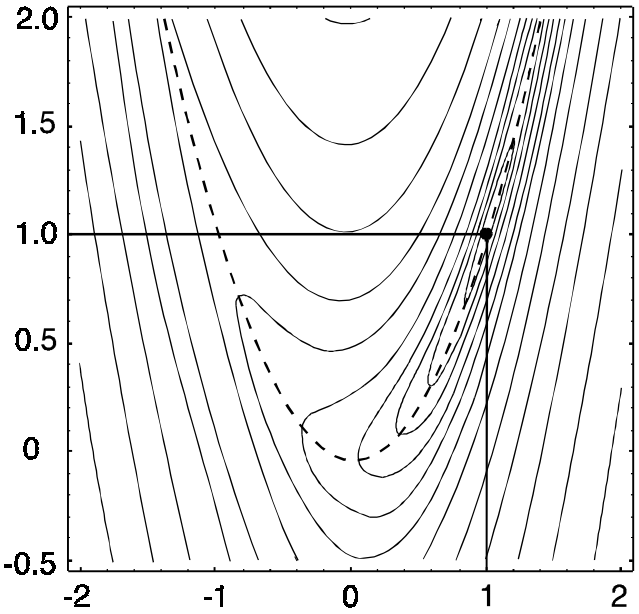
Minimum $f(1,1)=0$ je na obrázku 8.11 vyznačené veľkým bodom.

Táto funkcia bola riešená viacčlennou verziou evolučnej stratégie s použitím smerodajných odchyliek pre každú premennú zvlášť a kovariancie medzi premennými, teda uhlu α. Každý chromozóm sa teda skladal z piatich členov:

$$\text{chromozóm}=\{ x_1, x_2, \sigma_1, \sigma_2, \alpha\} \tag{8.19}$$

Počiatočné hodnoty premenných boli pre $\boldsymbol{x}$ zvolené náhodne z intervalu (–100,100), smerodajné odchýlky boli rovné 3.0 a uhol α bol nulový. Použili sme stratégiu (15,100), to znamená, že z 15 chromozómov vždy vyrobíme krížením a následnou mutáciou 100 chromozómov. Pri problémoch väčšej dimenzie by sa samozrejme hodilo viac chromozómov. Rodičovských chromozómov po vygenerovaní potomkov zmažeme. Vzhľadom k tomu, že z potomkov berieme vždy striktne iba najlepšie chromozómy, nie je treba prepočítavať funkčné hodnoty na fitness ako u genetických algoritmov, stačí iba zoradiť 100 novo vytvorených chromozómov podľa veľkosti funkčných hodnôt od najmenšej k najväčšej (alebo naopak, pokiaľ hľadáme maximum) a zobrať iba prvých 15 chromozómov do ďalšej generácie.

Bolo použité uniformné kríženie s náhodným výberom vždy dvoch rodičov pre premenné x_1,



Obrázok 8.11. Vrstevnicový graf Rosenbrockovej funkcie, kde horizontálna os je x_1 a vertikálna os x_2. Veľký bod predstavuje umiestnenie minima v "údoliu" vyznačenom čiarkovanou parabolou.

x_2, a kríženie priemerom pre strategické premenné σ_1, σ_2, α. Pretože počet premenných n=2, boli použité konštanty s nasledujúcimi hodnotami: $\tau_0 = 1/\sqrt{2\sqrt{n}} = 0{,}5946$; $\tau = 1/\sqrt{2n} = 0{,}5$; β=0.0873. Jednotlivé premenné sú teda po krížení chromozómov mutované pomocou vzorcov

$$\sigma_i' = \sigma_i \exp(\tau_0 N(0,1)) \exp(\tau N_i(0,1)), \text{ teda}$$

$$\text{konštanta} = N(0,1),\ \sigma_1' = \sigma_1 \exp(0.5946 \cdot \text{konštanta}) \exp(0.5\ N(0,1))$$

$$\text{a}\ \ \sigma_2' = \sigma_2 \exp(0.5946 \cdot \text{konštanta}) \exp(0.5\ N(0,1)) \qquad (8.20)$$

$$\alpha' = \alpha + \beta\ \Delta\alpha\ \text{, teda}\ \alpha' = \alpha + 0.0873\ N(0,1) \qquad (8.21)$$

kde N(0,1) je výsledok generátora náhodných čísel s normálnym rozdelením, s nulovou strednou hodnotou a jednotkovou smerodajnou odchýlkou. Pretože pre dve premenné je iba jedna kovariancia, používame iba jeden uhol α bez indexov. Pri výpočte zmien premenných x_1 a x_2 používame rovnaký generátor, iba s novo zmutovanou smerodajnou odchýlkou:

$$\Delta x_1 = N(0, \sigma_1') \text{ a } \Delta x_2 = N(0, \sigma_2') \qquad (8.22)$$

Tieto hodnoty ešte "pootočíme" pomocou uhla α:

$$\Delta x_1' = \Delta x_1 \cos\alpha' - \Delta x_2 \sin\alpha',\ \Delta x_2' = \Delta x_1 \sin\alpha' + \Delta x_2 \cos\alpha' \qquad (8.23)$$

Teraz konečne môžeme dostať nové hodnoty vektora $\boldsymbol{x}$:

$x_1' = x_1 + \Delta x_1'$ a $x_2' = x_2 + \Delta x_2'$ a z týchto hodnôt vypočítať funkčnú hodnotu $f(x_1', x_2')$, ktorou bude chromozóm ohodnotený. Nasleduje príklad tvorby nového chromozómu (ktorý bude zaradený do medzi 100 nových) z dvoch náhodne zvolených rodičovských chromozómov (z 15 možných):

premenná	x_1	x_2	σ_1	σ_2	α	$f(x_1,x_2)$
rodič_1 = {	0.415204	0.129087	0.039035	0.023917	0.202127	0.529536 }
		↓	↓	↓	↓	
potomok = {	0.316526	0.129087	0.0286605	0.0278155	0.399014	XXXXXX }
			$= \frac{0.039035 + 0.018286}{2}$			
	↑		↑	↑	↑	
rodič_2 = {	0.316526	0.129394	0.018286	0.031714	0.595900	0.552434 }

V prvej fáze sa teda za premennú x_1 zobrala premenná z druhého rodiča, a za premennú x_2 hodnota tejto premennej z prvého rodiča (u každej z premenných sme si "hodili korunou", z ktorého rodiča ju zobrať), zatiaľ čo namiesto hodnôt σ_1, σ_2 a α sa dosadili priemery obidvoch rodičov. Samozrejme, hodnota $f(x_1,x_2)$ potomka sa spočíta až po skončení mutácií. Ako prvá boli zmutované hodnoty σ_1, σ_2 a α:

konštanta=N(0,1)= 0.528959,
$\sigma_1' = \sigma_1 \exp(0.5946 \cdot \text{konštanta}) \exp(0.5 \cdot N(0,1))$
$= 0.0286605 \exp(0.5946 \cdot 0.528959) \exp(0.5 \cdot 0.380766\ ) = 0.047485$
$\sigma_2' = \sigma_2 \exp(0.5946 \cdot \text{konštanta}) \exp(0.5 \cdot N(0,1))$
$= 0.027815 \exp(0.5946 \cdot 0.528959) \exp(0{,}5 \cdot (-0.419033)) = 0.030895$
$\alpha' = \alpha + 0.0873\ N(0,1)$
$= 0.399014 + 0.0873 \cdot 0.548100 = 0.446863$
$\Delta x_1 = N(0, \sigma_1') = N(0, 0.047485) = 0.023767$
$\Delta x_2 = N(0, \sigma_2') = N(0, 0.030895) = -0.042886$
$\Delta x_1' = \Delta x_1 \cos\alpha - \Delta x_2 \sin\alpha$
$= 0.023767 \cos(0.446863) - (-0.042886) \sin(0.446863) = 0.038562$
$\Delta x_2' = \Delta x_1 \sin\alpha + \Delta x_2 \cos\alpha$
$= 0.023767 \sin(0.446863) + (-0.042886) \cos(0.446863) = -0.024535$
$x_1' = x_1 + \Delta x_1' = 0.316526 + 0.038562 = 0.355087$
$x_2' = x_2 + \Delta x_2' = 0.129087 + (-0.024535) = 0.104552$
$f(x_1,x_2) = 0.462288$

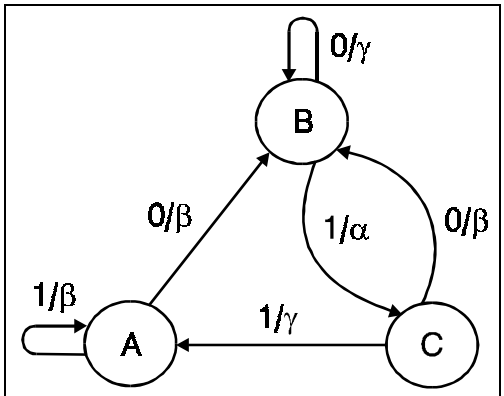
Výsledný potomok upravený mutáciou smerodajných odchyliek, uhlu α, a mutáciou premenných x_1, x_2 s použitím nových mutovaných odchyliek, s nasledovným pootočením týchto súradníc pomocou zmutovaného uhlu α vyzerá nasledovne:

premenná	x_1	x_2	σ_1	σ_2	α	$f(x_1,x_2)$
potomok = {	0.355087	0.104552	0.047485	0.030895	0.446863	0.462288 }

Príklad 8.2. Použite viacčlenné stratégie pre nájdenie optima Rosenbrockovej funkcie. Program zbehnite 100× a vypočítajte priemerný počet generácií, ktorý bol potrebný, aby najlepšia nájdená funkčná hodnota bola nižšia ako 10^{-6}. Porovnajte to s priemerným počtom generácií, nutným k nájdeniu rovnakej funkčnej hodnoty, ale bez použitia kovariancie, teda bez použitia uhlu α. Podobne skúste odstrániť kríženie pri zachovaní použitia kovariancií, a diskutujte vplyv tvaru alebo typu funkcie na efektívnosť použitia kríženia alebo kovariancie pri nájdení minima. (Symetria a multimodálnosť indikujú pravdepodobnú výhodnosť aplikácie kríženia, minimá uložené v "údoliach" umiestnených naprieč k osám súradnicového systému budú nájdené rýchlejšie pri použití kovariancií, čo platí aj pre "viacrozmerné údolia".)

8.6. Evolučné programovanie

Evolučné programovanie je prístup využívajúci simuláciu evolúcie na populácii súťažiacich algoritmov. V počiatkoch tohto prístupu navrhnutého okolo r. 1962 Lawrencom Fogelom [20-22] bolo cieľom predpovedať nové stavy prostredia a udalosti a na základe predpovedi prispôsobiť vlastnú reakciu podľa daného cieľa, čo je základnou podmienkou inteligencie organizmu. Evolučné programovanie



Obrázok 8.12. Trojstavový konečný automat. Vstupné symboly sú uvedené naľavo, výstupné napravo od lomítka. Štartovná pozícia je v A (podľa Fogela [21]).

Momentálny stav	A	A	B	B	C	A
Vstupný symbol	1	0	0	1	1	0
Budúci stav	A	B	B	C	A	B
Výstupný symbol	β	β	γ	α	γ	β

Tabuľka 8.1. Odpovede konečného automatu z obrázku 8.12 v závislosti na vstupných symboloch. Ohodnotenie automatu je dané funkciou, ohodnocujúcou výstupné symboly v závislosti na vstupných (stavu prostredia).

rovnako ako evolučná stratégia kladie skôr dôraz na podobnosť správania sa rodičov a ich potomkov ako na napodobňovanie genetických operátorov tak, ako to môžeme pozorovať v biologických procesoch na molekulárnej úrovni. Fogel začal u evolučného programovania s predpoveďami symbolických reťazcov generovaných konečnými automatmi z tzv. Markovovských procesov a nestacionárnych časových sérií.
Spracovanie vstupu automatom sa v prípade tabuľky 8.1. začína v stave A, na ktorý prichádza vstupná hodnota 1. Pretože z "krúžku" A vedie šípka 1/β presunieme sa pozdĺž tejto šípky. Hodnota za lomítkom, teda β, tvorí odpoveď automatu. Šípka je ale sľučkou a dostávame sa zasa do stavu A. Nový vstup je 0, a pozdľž šípky 0/β sa dostávame do stavu B, pročom β je odpoveď automatu. Takto pokračujeme, až pokial na sadu signálov 100110 nedostaneme odpoveď β β γ α γ β. Túto odpoveď ohodnotíme hodnotiacou funkciou, nech už vyzerá akokoľvek.

U takýchto konečných automatov existuje v zásade päť prirodzených typov mutácií: zmena výstupného symbolu, zmena budúceho stavu pre nastavený konečný stav a výstupný symbol (premiestnenie šípky), pridanie stavu, vymazanie stavu, zmena počiatočného stavu.

Na rozdiel od Johna Kozy [23], ktorý v svojom genetickom programovaní pracuje s programami vo forme ohodnotených stromov a používa okrem mutácie aj kríženie, Fogel sa v počiatku obmedzil vo svojich úvahách na konečné automaty bez použitia kríženia. Evolučné programovanie napodobňuje totiž vývoj druhov, ktoré medzi sebou súťažia a medzidruhové kríženie väčšinou nenastáva. Princíp súťaženia je v tomto prístupu podobný ako pri evolučnej stratégii, zo skupiny rodičov a potomkov sa vyberú do ďalšej populácie iba striktne najlepší jedinci. Z každého rodiča sa pritom vyrába mutáciou iba jeden potomok, teda zo skupiny rodičov a potomkov sa vyberie lepšia polovina ako rodičia do ďalšej populácie.

V nových variantoch evolučného programovania majú aj horší potomci ešte šancu, pokiaľ sú vybraní do slabšej skupiny. Skupinky sú náhodne vybrané z nových (t.j. mutovaných) jedincov aj z rodičovských jedincov. V skupinkách potom dochádza k "turnaji", potomci sú zoradení podľa svojich výsledkov v turnaji, a tí s najlepšími výsledkami v turnaji sú vybraní do ďalšej populácie. Počet jedincov v populácii nemusí byť konštantný a jedinec môže produkovať aj viac potomkov.

Evolučné programovanie sa postupne rozvíjalo rovnakým smerom ako evolučné stratégie. Výsledné algoritmy sú v súčasnej dobe natoľko podobné, že ich rozlišovanie nemá význam. Pretože ale tieto algoritmy majú rozdielny historický pôvod, v USA sa evolučné programovanie stále udržiava ako samostatný algoritmus [24-31].

Spoločnou črtou evolučného programovania a evolučnej stratégie, ktorá je rozdielna od genetických algoritmov, je reprezentácia jedinca (riešenia). Tá sa podriaďuje problému. Povedzme, že máme problém, ktorého riešenie zahŕňa nie len reálne premenné, ale aj ďalšie prvky. Tak je tomu napr. u zadania neurónovej siete [24], kde okrem váhových parametrov (čísel charakterizujúcich spojenie medzi uzlami siete) určujeme aj vlastnú topológiu, teda počet uzlov a existenciu spojení medzi nimi. Nemusíme sa predsa zaoberať tým, ako celú sieť zakódovať do bitového reťazca, ako to robia genetické algoritmy. Stačí si predstaviť jedinca ako sieť, a mutácie budú rôzneho druhu, jedny pre váhové faktory, teda reálne čísla, a iné pre topológu siete, teda pre pridávanie alebo uberanie uzlov a spojení medzi nimi.

8.7. Záver

Zoznam viac ako 300 aplikácií evolučných stratégií sa dá nájsť napr. v SyS-2/92 report [10]. Evolučné stratégie sú schopné riešiť multidimenzionálne nelineárne problémy s veľa lokálnymi optimami a s lineárnymi alebo nelineárnymi obmedzeniami. Optimalizovaná funkcia môže tiež byť výsledkom simulácie, nemusí byť v presne definovanej forme. To sa týka tiež obmedzení, ktoré môžu predstavovať výsledok napr. metódy konečných prvkov (FEM). Rovnako ako ostatné evolučné algoritmy, evolučné stratégie boli použité aj na nastavovanie parametrov (učenie) neurónových sietí [32]. Evolučné stratégie boli vytvorené na optimalizáciu vektorov reálnych čísel, no môžu slúžiť tiež

ako heuristika pre NP-úplné kombinatoriálne problémy ako problém obchodného cestujúceho, alebo problémy s časovo premennými alebo zašumenými hodnotami optimalizovanej funkcie.

Distribučná funkcia mutácie neurčuje smer mutácie, iba veľkosť, takže k veľkej mutácii môže dojsť s rovnakou pravdepodobnosťou aj v opačnom smere, preč od minima. Zaujímavé ale je, že pri návrhu pravdepodobnosti výskytu nového potomka sa neoplatí posunúť stred kružnice alebo elipsy tak, aby sa nezhodoval s pozíciou rodiča, ale aby bol podstatne posunutý smerom, v ktorom nastal predchádzajúci posun od rodiča k potomkovi - terajšiemu rodičovi. Síce by to zvýšilo momentálnu úspešnosť generovaných potomkov, no v dlhodobom merítku pre zložité multimodálne funkcie všeobecne nie je tento prístup výhodný. Pre jednoduchšie funkcie ale môže byť výhodný.

Literatúra

[1] I. Rechenberg: *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. (Evolution Strategy: Optimization of technical systems by means of biological evolution.) Fromman-Holzboog, Stuttgart, 1973, 1993 2nd edn.

[2] H.-P. Schwefel: *Numerische Optimierung von Computermodellen mittels der Evolutionsstrategie*, Birkhäuser, Basel, 1977.

[3] H.-P. Schwefel. *Numerical Optimization for Computer Models*. Wiley, Chichester, UK, 1981.

[4] H.-P. Schwefel: *Evolution and Optimum Seeking*. J. Wiley, New York, 1995.

[5] H.-P. Schwefel and R. Manner, (eds). *Proceedings of the First International Conference on Parallel Problem Solving from Nature.* Dortmund, Germany, 1990.

[6] F. Hoffmeister and T. Bäck: *Genetic Algorithms and Evolution Strategies: Similarities and Differences*, University of Dortmund, Dept. of CS, SyS-1/92. (1990, 1992). Available by ftp from ftp://lumpi.informatik.uni-dortmund.de

[7] T. Bäck, F. Hoffmeister, and H.-P. Schwefel: A Survey of Evolution Strategies. In R.K. Belew and L.B. Booker, (eds). *Proceedings of the Fourth International Conference on Genetic Algorithms,* pp. 2-9. Morgan Kaufmann, San Mateo, CA 1991.

[8] T. Bäck and H.-P. Schwefel: An Overview of Evolutionary Algorithms for Parameter Optimization. *Evolutionary Computation*, 1(1) (1993) 1-23.

[9] T. Bäck, G. Rudolph, and H.-P. Schwefel: Evolutionary Programming and Evolution Strategies: Similarities and Differences. In *Proc. of the 2nd Annual Conf. on Evolutionary Programming*, D.B. Fogel and W. Atmar (eds), published by the Evolutionary Programming Society, (Attn: Bill Porto, Treasurer), 9363 Towne Centre Dr., San Diego, CA 92121, 1993, pp. 11-22.

[10] T. Bäck, F. Hoffmeister, and H.-P. Schwefel: *Applications of evolutionary algorithms*. Technical report SYS-2/92, Systems Analysis Research Group, University of Dortmund, Department of Computer Science, July 1993, available at ftp://lumpi.informatik.uni-dortmund.de/pub/EA/papers/ea-app.ps.gz

[11] M. Herdy: Application of the Evolution Strategy to Discrete Optimization Problems. In H.-P. Schwefel and R. Männer (eds), *Proceedings of the First Interational Conference on Parallel Problem Solving from Nature* (PPSN), Dortmund, Germany, 1990, pp. 188-192.

[12] W.H. Press, B.P. Flannery, S. A. Teukolsky, and W.T. Vetterling: *Numerical Recipes in Pascal*. Cambridge University Press, 1989. http://cfatab.harvard.edu/nr/bookc.html (v postscripte dostupná iba verzia v Pascale a Fortrane)

[13] J.H. Ahrens, U. Dieter: Computer Methods for Sampling from the Exponential and Normal Distributions. *Communications of ACM*, 15 (1972), pp. 873-882, 1047

[14] J. Born. *Evolutionstrategien zur numerischen Lösung von Adaptationsaufgaben*. Dissertation A, Humboldt-Universität, Berlin, 1978.

[15] F.J. Solis and R.J.-B. Wets: Minimization by Random Search Techniques. *Math. Operations Research*, 6 (1981) 19-30.

[16] G. Rudolph: Parallel Approaches to Stochastic Global Optimization. In W. Joosen and E. Milgrom, (eds.), *Parallel Computing: From theory to Sound Practice, Proceedings of the European Workshop on Parallel Computing*, pp. 256-267. IOS Press, Amsterdam, 1992.

[17] Z. Michalewicz and C. Janikow: Handling Constraints in Genetic Algorithms. In R.K. Belew and L.B. Booker, (eds), *Proceedings of the Fourth International Conference on Genetic Algorithms*, Morgan Kaufmann, San Mateo, CA, 1991, pp. 151-157.

[18] G. Rudolph: On correlated mutations in evolution strategies. In R. Männer and B. Manderick (eds.), *Parallel Problem Solving from Nature 2* (PPSN II), North-Holland, Amsterdam, 1992, pp. 105-114.

[19] H.H. Rosenbrock: An automatic method for finding the greatest or least value of a function, *Comp. J.* 3 (1960) 175-184.

[20] L.J. Fogel: Autonomous Automata. *Industrial Research*, 4 (1962) 14-19.

[21] L.J. Fogel: On the Design of Conscious Automata. Final Report under Contract No. AF49(638)-1651, AFOSR, Arlington, VA.

[22] L.J. Fogel: A.J. Owens, and M.J. Walsh. *Artificial Intelligence through Simulated Evolution*. John Wiley, New York, 1966.

[23] J. R. Koza: *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, 1992.

[24] D.B. Fogel, L.J. Fogel, and V.W. Porto: Evolving Neural Networks. *Biological Cybernetics*, Vol. 63:6 (1990) 487-493.

[25] D.B. Fogel and W. Atmar, eds.: *Proceedings of the First Ann. Conf. on Evolutionary Programming. Evolutionary Programming Society*, La Jolla, CA, 1993.

[26] D.B. Fogel and W. Atmar, eds.: *Proceedings of the Second Ann. Conf. on Evolutionary Programming. Evolutionary Programming Society*, La Jolla, CA, 1993.

[27] A.V. Sebald and L.J. Fogel, eds.: *Proceedings of the Third Annual Conference on Evolutionary Programming*. World Scientific Publishers, River Edge, NJ, 1994.

[28] J.R. McDonnell, R.G. Reynolds, and D.B. Fogel DB (eds.): *Evolutionary Programming IV: The Proceedings of the Fourth Annual Conference on Evolutionary Programming*. MIT Press, Cambridge, MA, 1995.

[29] L.J. Fogel, P.J. Angeline, T. Bäck (eds.): *Evolutionary Programming V*. MIT Press, Cambridge, MA, 1996.

[30] P.J. Angeline, R.G. Reynolds, J.R. McDonnell, and R. Eberhart (eds.): *Evolutionary Programming VI*. Springer Verlag, Berlin, 1997.

[31] D.B. Fogel: *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. IEEE Press, New York, 1995.

[32] R. Lohmann: Structure Evolution and Incomplete Induction. In R. Männer and B. Manderick (eds.), *Parallel Problem Solving from Nature 2* (PPSN II), North-Holland, Amsterdam, 1992, pp. 175-186.