Princip optimalizačních metod inspirovaných přírodou

Tomáš Kroupa

20. května 2014

Tento studijní materiál je spolufinancován Evropským sociálním fondem a státním rozpočtem České republiky.

Obsah

Úkol a základní princip

Úkol

Nalézt

$$\min_{\mathbf{x}} \{f(\mathbf{x})\} \tag{1}$$

kde f je funkcí mnoha proměnných (parametrů)

$$f = f(x_1, x_2, ..., x_N), \tag{2}$$

na oblasti Ω, která je definovaná tak, že $x_1 \in \left\langle x_1^{lb}, x_1^{ub} \right\rangle$ atd.
Často se značí

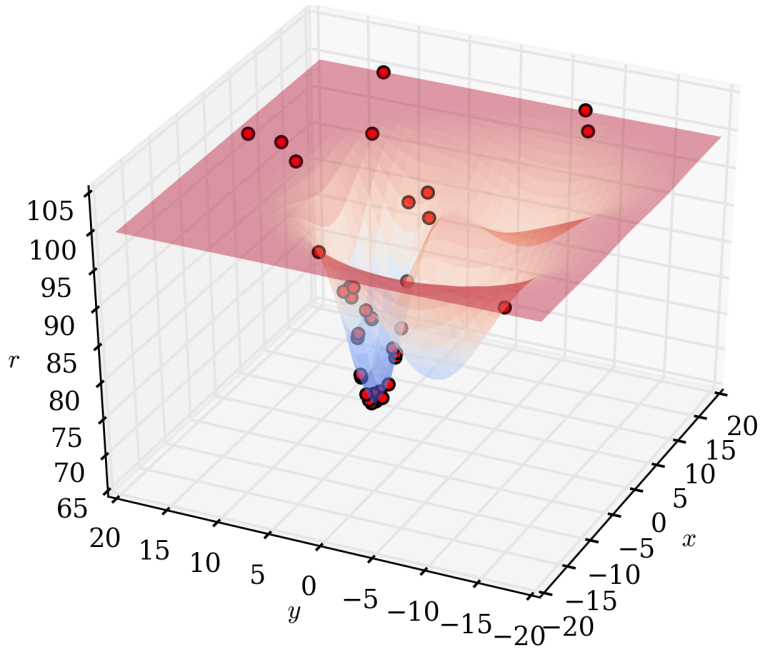
$$\mathbf{x} = [x_1, x_2, ..., x_N]^{\mathrm{T}}. \tag{3}$$

Často je nutné uvažovat ještě podmínky rovnosti a nebo nerovnosti

$$\mathbf{0} \geq \mathbf{h}_{\mathrm{in}}(\mathbf{x}) \quad \text{a nebo} \quad \mathbf{0} = \mathbf{h}_{\mathrm{eq}}(\mathbf{x}). \tag{4}$$

Úkol

Funkce f nemusí být popsaná jen analyticky, ale i tak, že pomocí parametrů $\mathbf{x}$ se postaví MKP model a výstup bude například maximální průhyb modelu, nebo první vlastní frekvence. A funkce f může být rozdíl mezi vlastní frekvencí, na kterou chceme model „naladit“ nebo ten samotný průhyb, který chceme minimalizovat apod.



Praktické příklady

1. Identifikace materiálových parametrů
   - f – Rozdíl mezi MKP a exp.
   - $\mathbf{x}$ – E_1, E_2, σ^y, ...
   - $0 \geq h$ – Relaxační časy u viskoelastického modelu větší než 0
2. Minimalizace hmotnosti lávky
   - f – Hmotnost lávky
   - $\mathbf{x}$ – Všechny možné rozměry, skladba vrstev laminátu,...
   - $0 \geq h$ – Průhyb, První vlastní frekvence $>$ 6 Hz
3. Identifikace místa dopadu impaktoru
   - f – Rozdíl mezi signály z analýzy a exp.
   - $\mathbf{x}$ – Pozice razníku v analýze
   - $0 \geq h$ – Hodnoty rázové funkce musí být vždý vetší nebo rovny 0

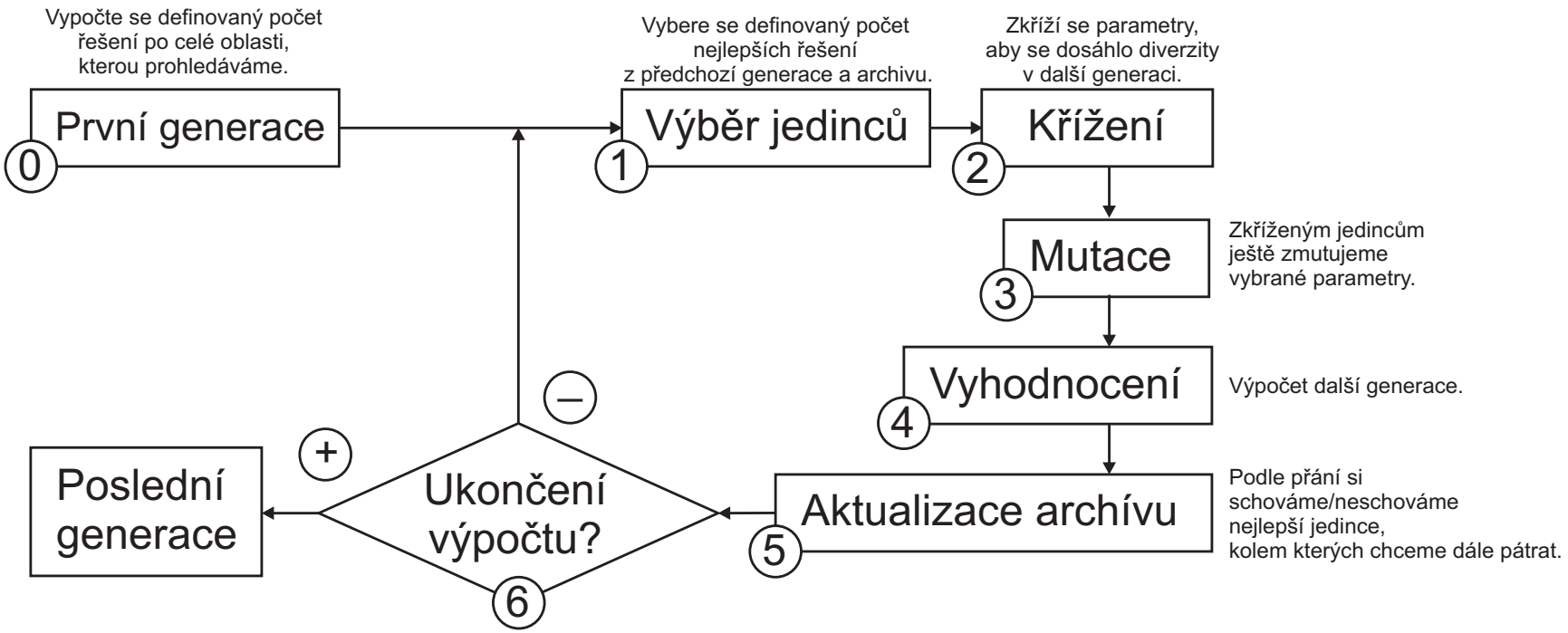
Princip

Z generace na generaci hledáme lepší řešení pomocí matematicky popsaných jevů jako jsou „křížení genů“ nebo „mutace genů“ a definujeme výběr nejlepších jedinců z dané generace. Viz příroda ...

Rodiče, kteří se našli mezi těmi nejlepšími lidmi;-), přenáší na děti geny. Geny se zkříží, zmutují a je z toho (občas ;-)) lepší jedinec.

Včelky hledají květiny na opylovaní a chtějí najít tu nejsladší. Tak vysílají do různých míst dělnice. Když se dělnice vrátí, předají si informace, jak sladké byly květy a kde je nalezli. Další generace výsadku poupraví cíle letu podle svého nejvetšího úspěchu s přihlédnutím k nejlepším výsledkům celého úlu.

Vývojový diagram



Nutné znalosti

Fittness

Tzv. *fitness* $f_i^f(\mathbf{x})$ (i číslo designu) odpovídá tomu jak dobrý je daný jedinec (jakou hodnotu má $f_i(\mathbf{x})$).

Nejlepší jedinec z party určené k reprodukci dostane nejvyšší hodnotu.

Příklad

Řekněme, že minimalizujeme průhyb nosníku (f_i je tedy hodnota průhybu nosníku v designu i).

i	f_i	f_i^f
1	0.1	3
2	10.5	1
3	1.2	2

Fittness – poznámka

Stanovovat $fitness$ není úplně šikovné v každém kroku tím, že škálujeme funkci podle nejvyšší hodnoty dosažené během celé optimalizace, třeba takto divně:

$$f_i^f(\mathbf{x}) = \frac{1}{1 + \frac{f_i}{\max\limits_j(f_j)}}, \text{ kde } j = 1, ..., \text{počet designů}, \qquad (5)$$

kde f_i je minimalizovaná funkce.

Fittness – uvažování omezení

Pokud uvažujeme omezení, lze stanovit $fitness$ takto:

$$F_i^f(\mathbf{x}) = f_i^f(\mathbf{x}) - \left(cliff + \sum_{k=1}^{\mathrm{n}_{eq}} |\mathbf{h}_{eq}(\mathbf{x})| + \sum_{k=1}^{\mathrm{n}_{in}} |\min\left[0, \mathbf{h}_{in}(\mathbf{x})\right]| \right) \tag{6}$$

kde parametr $cliff$ je volen vzhledem k očekávané velikosti minimalizované funkce.

Manuál optiSLangu se chlubí, že nejlepší je to dělat jako oni a využít následující metodu...

Fittness – uvažování omezení

1. Rozdělit populaci (všechny dosud vypočtené designy) na ty které vyhovují všem podmínkám (řešitelné) a ty, které ne (neřešitelné).
2. Stanovit *fitness* f_i^f pro všechny řešitelné designy na základě hodnoty f_i. Je jich μ a tak ten s nejnižší hodnotou f_i dostane například hodnotu μ a nejhorší „třeba“ 1.
3. Stanovit *rank* všech neřešitelných designů na základě kritéria dominance[1].
4. Stanovit *fitness* všech neřešitelných designů přidáním nejhorší hodnoty fitness řešitelných designů[2].

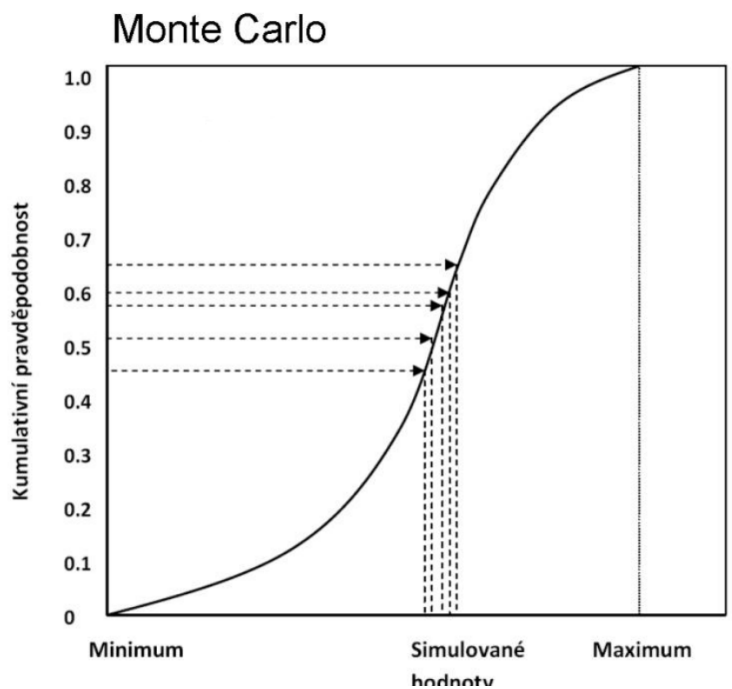
[1] Autor si není jistý, co má toto kritérium znamenat u problému s jednou minimalizovanou funkcí!

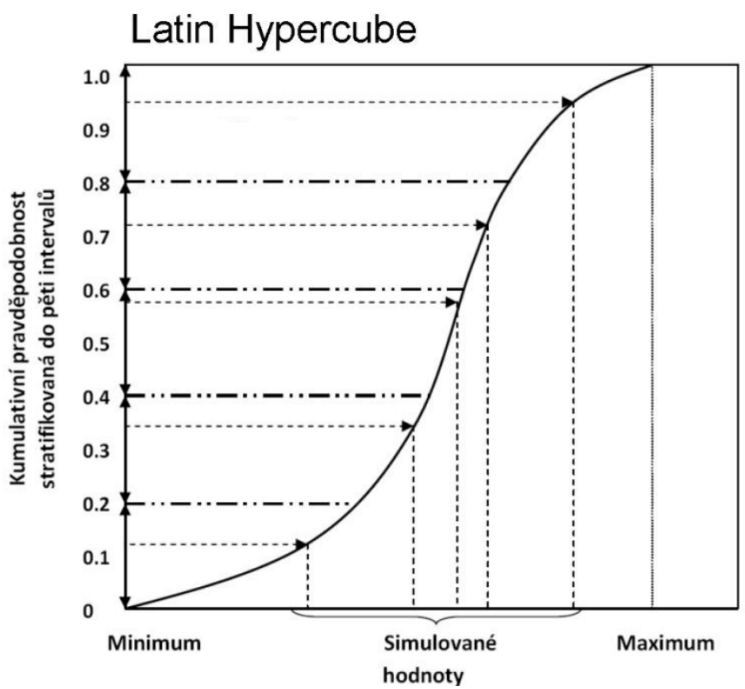
[2] Takže *fitness* neřešitelných je záporná? - tímto se asi vyhneme znalosti parametru $cliff$

Jednotlivé kroky

0 – Počáteční populace

- Ručně
- Z předchozích analýz.
- Pravidelné rozmístění bodů – pokud využiji všechny „rohy“ a „prostředky“ může být problém časově neřešitelný díky nespočitatelnému množství výpočtů (designů).
- Monte Carlo – nerovnoměrné rozmístění výpočtů (designů).
- Latin Hypercube – nerovnoměrnější rozmístění výpočtů (designů).





1 – Výběr jedinců – ranking

Proces, při kterém se vyberou rodiče pro další generaci potomků.

Jedinci se vyberou z archívu a z nejlepších v předchozí generaci.

- Lineární ranking – Z μ jedinců dostane ten nejlepší fitness $f_i^f = \mu$ a nejhorší fitness $f_i^f = 1$.

$$rank(i) \approx f_i^f \tag{7}$$

- Exponenciální ranking – Tolik neupřednostňuje nejlepší jedince.

$$rank(i) = \sqrt{f_i^f - \min_{i=1..\mu}\left(f_i^f\right)} + 1 \tag{8}$$

1 – Výběr jedinců – výběr

- Ruleta
  – Virtuální losování jako při ruletě, kdy velikost chlívečku je úměrná *rank*u.

$$slot(i) = \frac{rank(i)}{\sum_{i=1}^{\mu} rank(i)} \in \langle 0, 1 \rangle \tag{9}$$

$$\sum_{i=1}^{\mu} rank(i) = 1 \tag{10}$$

  – Vždy se losuje patřičný počet jedinců pro reprodukci zvlášť.

$$p_k \in \langle 0, 1 \rangle , \qquad k = 1, ..., g \tag{11}$$

1 – Výběr jedinců – výběr

- Stochastic universal sampling
  – Virtuální losování jako při ruletě, kdy velikost chlívečku je úměrná *rank*u.
  – Vždy se losuje patřičný počet jedinců pro reprodukci tak, že se pravidelně rozmístí kolem osudí rulety a „zatočí se jen jednou“.

$$p_0 = \frac{1}{\mu} R\left\langle 0, 1\right\rangle, \qquad k = 1, ..., g \tag{12}$$

$$p_k = p_0 + k\frac{1}{\mu}, \qquad k = 1, ..., g \tag{13}$$

1 – Výběr jedinců – výběr

- Výběr pomocí turnaje
  1. Náhodně se vybere předem určený počet jedinců z archivu a současné generace.
  2. Ten co má nejlepší *rank* (v tomto případě stačí $fittness$) je vybrán k reprodukci.

  – Velikost turnaje může být od 2 do velikosti generace. Čím větší turnaj tím větší upřednostňování nejlepšího jedinec z celé generace.

2 – Křížení/rekombinace – myšlenka

Proces, kdy dva rodiče, zkříží parametry (geny, chromozomy) a vyprodukují dva potomky.

Dominantní vyhledávací operátor pro genetické algoritmy

Příklad:
Mějme dva rodiče s parametry

$$R_1 = [1, 2] \qquad \text{a} \qquad R_2 = [3, 4]$$

a skřížíme jejich parametry a dostaneme dva potomky, například

$$P_1 = [3, 2] \qquad \text{a} \qquad P_2 = [1, 4]$$

Způsobů křížení existuje celá řada.

2 – Křížení/rekombinace – typy

- Singlepoint/multipoint crossover
  1. Náhodně se vybere předem určený počet bodů pro křížení. Na ukázku provedeme dvou bodový crossover, vyberou se náhodně dvě čísla i a j, kdy $i \leq j$.
  2. Parametry rodičů (celkem jich je n) se rozdělí do sekcí.

$$R_1 = \left[r_1^1, ..., r_i^1, r_{i+1}^1, ..., r_j^1, r_{j+1}^1, ..., r_n^1\right] \quad (14)$$

$$R_2 = \left[r_1^2, ..., r_i^2, r_{i+1}^2, ..., r_j^2, r_{j+1}^2, ..., r_n^2\right] \quad (15)$$

  3. Každá druhá sekce se prohodí.

$$P_1 = \left[r_1^1, ..., r_i^1, r_{i+1}^2, ..., r_j^2, r_{j+1}^1, ..., r_n^1\right] \quad (16)$$

$$P_2 = \left[r_1^2, ..., r_i^2, r_{i+1}^1, ..., r_j^1, r_{j+1}^2, ..., r_n^2\right] \quad (17)$$

2 – Křížení/rekombinace – typy

- Aritmetický crossover
  1. Předem se definuje číslo $\lambda \in \langle 0, 1 \rangle$.
  2. Provede se plně deterministický výpočet.

$$p_i^1 = (1 - \lambda)\, r_i^1 + \lambda r_i^2 \tag{18}$$

$$p_i^2 = (1 - \lambda)\, r_i^2 + \lambda r_i^1 \tag{19}$$

2 – Křížení/rekombinace – typy

- Simulovaý binární crossover[3] lze provést pomocí vztahů

$$p_i^1 = 0.5\left[\left(r_i^1 + r_i^2\right) - \beta_i \left|R^2 - R^1\right|\right] \quad (20)$$

$$p_i^2 = 0.5\left[\left(r_i^1 + r_i^2\right) + \beta_i \left|R^2 - R^1\right|\right] \quad (21)$$

kde $i = 1, ..., n$ a

$$\beta_i = (\alpha_i u)^{\frac{1}{n_c+1}}, \quad \text{pokud} \quad u \leq \frac{1}{\alpha_i} \quad (22)$$

$$\beta_i = \left(\frac{1}{2 - \alpha_i u}\right)^{\frac{1}{n_c+1}}, \quad \text{pokud} \quad u > \frac{1}{\alpha_i} \quad (23)$$

kde $n_c \in \langle 0, \infty)$ je parametr náhodného rozdělení, $u \in \langle 0, 1\rangle$ je nahodné číslo a $\alpha_i = 2 - \delta_i^{-n_c+1}$ a

$$\delta_i = 1 + \frac{2}{r_i^2 - r_i^1} \min\left[(r_i^1 - r_i^{lb}), (r_i^{ub} - r_i^2)\right] \quad (24)$$

[3] To, že se β počítá pro každé i zvlášť není jisté, ale autor si nedovede představit, jak by to mělo fungovat jinak.

2 – Křížení/rekombinace – poznámky

- Každý typ křížení se hodí pro jiný typ analýzy.
- Volba křížení závísí na zkušenostech velectěného pána/paní kteří provádí danou analýzu.

3 – Mutace – základní myšlenka

Proces, kdy se parametry potomků náhodně pozmění pomocí náhodného rozdělení, tak aby nebyly jen pouhým zkřížením rodičů.

Dominantní vyhledávací operátor pro evoluční algoritmy

$$p_i^{t+1} = p_i^t + N\left(0, \sigma_i^t\right) \tag{25}$$

- Ještě lze definovat jak často dochází k mutaci, často se doporučuje zadat tzv.

$$\text{mutation rate} = \frac{1}{n} \tag{26}$$

kde n je počet parametrů (vstupních $= x_i$)

3 – Mutace – nastavení a typy

- Ruční zadání
  - Rozptyl normálního rozdělení lze zadat ručně.
- Sebe-adaptivní mutace
  - Ty parametry které mají zmutovat se zmutují viz. (25)
  - Ostatní parametry se zmutují pomocí logaritmického rozdělení pravděpodobnosti s jiným rozptylem[4].
- Adaptivní mutace určená omezením
  - Rozptyl normálního rozdělení pravděpodobnosti pro mutaci je závislý na vzdálenosti daného parametru od meze, kdy nejsou splněny podmínky nerovnosti.
  - Detaily – Manuál optiSLangu – je aktivní jen, když jsou podmínky nerovnosti v modelu.

[4] Viz. google, wikipedie a manuál optiSLangu.

3 – Mutace – nastavení a typy

- Modulovaná adaptivní mutace
  - Rozptyl normálního rozložení pravěpodobnosti je upravován během chodu optimalizace za pomocí informací z úspěšných mutací (potomek má lepší $fitness$ než rodič)[5]

[5] Zatím autor nenašel inteligentní a stručný vztah, který tohle popisuje

4 – Vyhodnocení

Provedení výpočtů minimalizované funkce f_i a hodnot omezeních $\mathbf{h}_{in}$ a $\mathbf{h}_{eq}$.

U nás nepochybně pomocí Abaqusu ;-).

5 – Aktualizace archívu

(μ, λ) Schováme si μ nejlepších jedinců z generace vypočtené v kroku 4. (Neuchováme si žádného nejlepšího jedince z celé populace)

$(\alpha(\mu) + \lambda)$ Schováme si μ nejlepších jedinců z nejlepších α jedinců z populace bez jedinců z generace vypočtené v kroku 4 a přidáme λ nových jedinců z generace vypočtené v kroku 4. (Uchováme si vždy „pár“ dobrých jedinců z celé populace.) Toto je dobrý kompromis.

$(\mu + \lambda)$ Schováme si μ nejlepších jedinců z populace bez jedinců z kroku 4 a přidáme λ nových jedinců z generace vypočtené v kroku 4. (uchováme si μ nejlepších jedinců z celé populace po celou dobu výpočtu) Toto je nebezpečné pokud jsme náhodou uvízli v lokálním minimu.

6 – Ukončení výpočtu

Výpočet se ukončí:

- Pokud je počet generací větší než uživatelem povolený počet.
- Pokud se po určitém počtu generací nenajde lepší jedinec.

Zamyšlení

Zamyšlení

Uvažte lidskou společnost (planetu Zemi)...

- *Pokud je naše společnost jen algoritmus, proč už dávno nebyl ukončen?*
- *Jak pracovník BVC[6] nebo MVC[7] nastavil náš výpočet?*
- *Snad běžíme někde na serveru a nikdo o nás neví, vzhledem k tomu, že odpověď na základní otázku Života, Vesmíru a vůbec je známá už nějaký čas[8].*

[6] Božského Výpočtového Centra

[7] Mimozemského Výpočtového Centra

[8] Odpověď je 42, pro ty, kteří by náhodou nevěděli ;-).

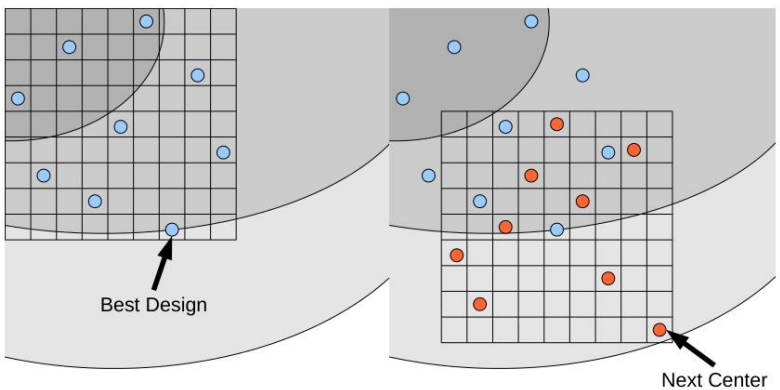
Obsah

Typy

Genetický algoritmus Zejména křížení

Evoluční algoritmus Zejména mutace

Single design improvement algoritmus Čisté prohledávání kolem nalezeného nejlepšího řešení. Nejlepší řešení je to, které má nejmenší hodnotou minimalizované funkce a nejméně porušených omezujících podmínek. Takový jedinec je zvolen jako střed další generace. Kolem tohoto jedince se provede vzorkování pomoci latin hypercube na předem definovaném okolí.



Typy

Particle swarm algoritmus Začne se generací a každý jedinec si pamatuje svoje nejlepší řešení a zároveň se „nestydí“ inspirovat se globálním nejlepším řešením.

$$\mathbf{v}_{t+1}^{i} = w_t \mathbf{v}_t^i + c_{p,t} R\langle 0,1\rangle \left(\mathbf{x}_{p,t}^i - \mathbf{x}_t^i\right) + c_{g,t} R\langle 0,1\rangle \left(\mathbf{x}_{g,t}^i - \mathbf{x}_t^i\right) \quad (27)$$

$$\mathbf{x}_{t+1}^i = \mathbf{x}_t^i + \mathbf{v}_{t+1}^i \quad (28)$$

kde i je číslo designu, t je číslo generace, $\mathbf{x}_{p,t}^i$ je nejlepší řešení jedince, $\mathbf{x}_{g,t}^i$ je nejlepší řešení ze všech jedinců, $\mathbf{x}_t^i$ je kombinace vstupních parametrů v daném designu, w_t je váhový koeficient (?vektor?), $c_{p,t}$ kognitivní koeficient (koeficient akcelerace jedince), $c_{g,t}$ sociální koeficient (koeficient akcelerace hejna), $\mathbf{v}_{t+1}^i$ kam se pohnou jedinci v další generaci (dalším výletu včelek pro pyl na poli) a $R\langle 0,1\rangle$ náhodné čílo mezi 0 a 1.

Tato prezentace je spolufinancována Evropským sociálním fondem a státním rozpočtem České republiky v rámci projektu č. **CZ.1.07/2.2.00/28.0206**
„Inovace výuky podpořená praxí“.

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Tento studijní materiál je spolufinancován Evropským sociálním fondem a státním rozpočtem České republiky.