Prednáška 1

Klasifikácia algoritmov podľa zdroja inšpirácie

- Prírodou inšpirované algoritmy
  - Biologicky inšpirované algoritmy
    - *Biologické algoritmy inšpirované inteligenciou roja (SI)*
    - *Biologické algoritmy neinšpirované inteligenciou roja (nie SI)*
      - Genetické algoritmy
      - Neurónové siete
      - Iné
  - Fyzikálne a chemicky inšpirované algoritmy
  - Iné

Heuristické algoritmy

- Heuristika (heuristická metoda) je praktický prístup na riešenie problémov, pre ktoré nepoznáme algoritmus alebo presnejší spôsob riešenia.
- Heuristika nemusí byť nutne optimálne riešenie, ale malo by byť dostatočné vzhľadom k stanovenému cieľu.

Toto riešenie je vetšinou iba približné, založené na skúsenosti, odhadu, intuitivnom úsudku a zdravom rozume.

Metaheuristické algoritmy

Metaheuristika je množina algoritmických konceptov používaných na definovanie heuristických metód aplikovateľných na širokú triedu aplikácií. Rovnaký postup je použitý s malými modikáciámi na riešenie rôznych problémov.

Stochastika je matematický odbor, ktorý sa zaoberá skúmaním a modelovaním náhodných javov.

Stochastický - náhodný, nahodilý. Protikladom je deterministický.

Optimalizácia

- Numerická optimalizácia
- Kombinatorická optimalizácia

Alan Turing v roku 1948 načrtol inovatívne myšlienky strojového učenia, neurónových sietí a evolučných algoritmov.

Prednáška 3

Účel optimalizácie je minimalizovať/maximalizovať účelovú funkciu.

Rozhodovacie premenné:

$$\boldsymbol{x} = (x_1, x_2, \ \ldots, x_D) \in R^D$$

Numerická optimalizácia - vyhľadávací priestor R^D
Kombinatorická optimalizácia - vyhľadávací priestor N^D

Obmedzenia:

$h_i(\boldsymbol{x}) = 0, \quad (i = 1,2, \ \ldots, M)$

$g_j(\boldsymbol{x}) \leq 0, \quad (j = 1,2, \ \ldots, N)$

Prírodou inšpirované počítanie

- oblasť, ktorá skúma: možnosti použitia výpočtovej techniky na modelovanie prírodných javov a prírodné javy za účelom zlepšenia používania výpočtovej techniky,
- spája vedné disciplíny: matematika, informatika a biológia,
- často úzko spätá s oblasťou umelej inteligencie a strojového učenia,
- ponúka nové možnosti riešenia zložitých úloh.

V energetike vieme optimalizáciu získať najlepšie parametre pre predikčné metódy.

Sekvenovanie – proces – textová reprezentácia DNA

Cytometria - meranie fyzikálnych a chemických vlastností buniek

Inteligencia roja

- jedinci nemajú žiadne vedomosti o riešení daného problému

- inteligenciu vidno až pri správaní sa roja týchto agentov

- agenti môžu meniť aj samotné prostredie

- Medzi hlavné vlastnosti inteligencie roja patrí:
  - **pružnosť**
    roj má schopnosť rýchlej adaptácie sa na zmenu prostredia,
  - **robustnosť**
    kolónia je úspešná aj keď niektorí jedinci zahynú,
  - **samoorganizácia**

kolónia nemá žiadne centrálne riadenie,

- **decentralizácia**
  riadiaca moc sa presúva na nižšie orgány – jedincov,
- **paralelizmus**
  súbežné vykonávanie rovnakých akcií viacerými jedincami.

Stimergia

- Stigmergia je mechanizmus spontánnej nepriamej koordinácie medzi jednotlivcami spoločenstva.

- Je to **forma samoorganizácie**, kde spoločenstvo vytvára zložité, zjavne inteligentné štruktúry **bez** nutnosti akéhokoľvek **ústredného plánovania a riadenia**. Podporuje efektívnu spoluprácu medzi veľmi jednoduchými živočíchmi, ktoré nemusia mať akúkoľvek pamäť a inteligenciu.

Neorientovaný graf sa nazýva **súvislý**, ak medzi ľubovoľnými dvoma jeho vrcholmi existuje cesta.

PREDIKČNÉ METÓDY

- Moving Average

- Random Forest

- ARIMA

- SARIMA

- Support Vector Machine

- Logistická regresia

Harmonizačný algoritmus vyhľadávania

- na riešenie problémov ako je:
  - distribúcia vody
  - modelovanie dopravy
  - plánovanie

ALGORITMY

BIOLOGICKY INŠPIROVANÉ ALGORITMY	
ALGORITMY INŠPIROVANÉ INTELIGENCIOU ROJA	**NEROJOVÉ ALGORITMY**
Optimalizácia rojom častíc, Zrýchlené PSO, Mravčia kolónia, Včelia kolónia, Hierarchický včelí úľ, Hľadanie potravy baktériami, Netopieří algoritmus, Algoritmus čmeliakov, Optimalizácia húfom mačiek, Siví vlci, Kukučie vyhľadávanie, Orlia stratégia, Algoritmus svätojánskych mušiek, Húf rýb, Hierarchický model roja, Roj pancieroviek antarktických, Algoritmus zamorenia švábmi, Algoritmus pohybu lososov	Umelé neurónové siete, Evolučné algoritmy, Umelý imunitný systém, Algoritmus opeľovania kvetín, Optimalizácia mozgových vzruchov, Diferenciálna evolúcia, Echolokácia delfínov, Algoritmus volania rosničiek japonských, Ekologicky inšpirovaný evolučný algoritmus, Algoritmus egyptského supa, Rybie vyhľadávanie, Genetická expresia, Algoritmus inšpirovaný inváziou buriny, Algoritmus evolúcie včelej kráľovnej, Optimalizačný algoritmus skákajúcej žaby

FYZIKÁLNE A CHEMICKY INŠPIROVANÉ ALGORITMY
Simulované žíhanie, Dynamika formovania riek, Teória veľkého tresku, Čierna diera, Samohybné častice, Inteligentná kvapka vody, Gravitačné vyhľadávanie, Optimalizácia založená na biogeografii

INÉ ALGORITMY
Algoritmus inšpirovaný správaním sa ľudí, Optimalizácia anarchickej spoločnosti, Evolúcia gramatiky, Sociálna emocionálna optimalizácia

ABC Umelá kolónia včiel
BBO Biogeografická optimalizácia
DE Diferenciálna evolúcia
GA Genetický algoritmus
GWO Optimalizácia pomocou sivých vlkov
PSO Optimalizácia rojom častíc

Model včelieho úľa

- využitie v prostredí webu
- zaradený medzi najlepšie modely
- parametre modelu:
  - počet včiel
  - distribúcia včiel
  - max. čas tancovania
  - čas strávený v pozorovatelni
  - informačný šum
  - chyba vyhodnotenia kvality zdroja
  - vhodnosť zdroja

Sledovanie vyvíjajúceho sa príbehu

- Pavol Návrat, Anna Bou Ezzedine

- Vhodný pre Web

Zisťovanie kvality výrobkov na základe odporúčania používateľov internetového obchodu

- Pavol Návrat, Anna Bou Ezzedine

- Vhodný pre Web

Hierarchický model

- Pavol Návrat, Anna Bou Ezzedine

- Vhodný pre Web

- pozrieť z poslednej prednášky !!!!!!!

Umelá kolónia včiel

- vhodný pre DNA sekvencie

- pozrieť z poslednej prednášky !!!

Mravčia kolónia

- vhodný pre DNA sekvencie
- metaheuristický, stochastický prístup pre KOMBINATORICKÉ úlohy, ktoré sú reprezentované matematickým grafom
- Základná črta ACO algoritmov je kombinácia informácie o atraktívnosti daného riešenia s informáciou o jeho úspešnosti
- Pred použitím ACO, je nutné optimalizovaný problém transformovať na problém hľadania najlepšej cesty váženého grafu.
- Agenti pohybom po grafe vytvárajú riešenia problému a aplikovaním špecifickým pravidiel počas daného počtu iterácií dosiahnu nami požadované globálne riešenie.
- pozostáva z populácie jednoduchých agentov, ktorí vzájomne **pôsobia** sami **na seba** a **na svoje okolie**
- Mravce do prostredia vylučujú pachovú stopu nazývanú **feromón**.
- Množstvo feromónu predstavuje parameter priradený hranám grafu, ktorého hodnoty menia mravce počas riešenia.
- Proces riešenia je stochastický.
- **umelý mravec**, ktorý si pamätá cestu a dĺžku prejdených hrán.
- Umelý mravec - dva režimy pri hľadaní zdroja potravy.
    - **Dopredný režim:**
        - mravec **vytvára riešenie** pomocou náhodnej voľby ďalšieho vrcholu.
    - Pred spätným (deterministickým) pohybom sa z cesty eliminujú slučky.

- **Spätný režim**
    - mravec na spiatočnej ceste **vylúči feromón**. Jeho množstvo závisí od dĺžky nájdenej cesty – mravce sa nasmerujú k lepším trasám a urýchli sa konvergencia riešenia.

- Vrcholy, medzi ktorými hľadáme minimálnu cestu nazývame **zdrojový** a **cieľový**.
- Ku každej hrane a_{ij} priradíme premennú τ_{ij}, ktorú nazveme *umelá feromónová stopa*. Na začiatku má každá hrana rovnaké nenulové množstvo feromónu τ_0

S hranami grafu sú spojené aj heuristické hodnoty η_{ij}, ktoré predstavujú **užívateľsky definovaný parameter**.

Heuristické hodnoty slúžia na nasmerovanie algoritmu do perspektívnych oblastí na začiatku vyhľadávacieho procesu. Ich veľkosť zväčša zodpovedá prevrátenej hodnote dĺžky hrán a_{ij}

$$\eta_{ij} = \frac{1}{d_{ij}}$$

- V i-tom uzle mravec číta feromónovú stopu $\tau_{ij}(t)$ a heuristickú informáciu η_{ij}
- Z týchto hodnôt určí mieru pravdepodobnosti voľby ďalšieho vrchola
- Týmto sa mravce vyhnú návratu do predchádzajúceho vrchola okrem prípadu, keď je množina N_i^k prázdna.
- Parametre, v rovnici na určenie pravdepodobnosti, $\alpha > 0$ a $\beta > 0$ určujú váhu množstva feromónu a heuristickej hodnoty. $\alpha \approx \beta \approx 2$
- Mravec prechádza z vrcholu do vrcholu opakovane, až pokiaľ nedosiahne cieľ (potravu).
- Potom sú eliminované z cesty všetky slučky.
- Pri spätnom pohybe *k*-ty mravec prechodom cez hranu a_{ij} zmení množstvo stopovacieho feromónu τ_{ij} medzi vrcholmi *i* a *j*
- Potom ako každý mravec dokončí svoju cestu sa časť feromónu vyparí
- Vyparovaním feromónu redukujeme vplyv stôp z počiatočných fáz, kedy mravce nájdu menej kvalitné riešenia a tým sa umožní prieskum nových ciest.

PSEUDOKÓD ACO

definuj účelovú funkciu $f(x), x = (x_1, \dots, x_d)^T$

definuj parametre vyparovania feromónu;

pokiaľ (nie je splnená ukončovacia podmienka):
    pre všechny mravence $i \in \{1, \dots, n\}$:
        spočítaj nové riešenia;
        priraď novým riešeniam mieru feromónu;
        aktualizuj mieru feromónu všetkých riešení;
    koniec
    nájdi najlepšie riešenie;
    $t = t + 1$;
koniec

AKÉ ÚLOHY JE MOŽNÉ RIEŠIŤ

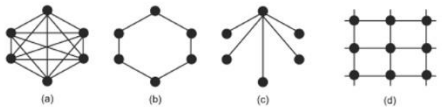
ACO algoritmus vo všeobecnosti úlohy, ktoré sa dajú transformovať do podoby grafu.

- problém obchodného cestujúceho (TSP),
- problém rozvozu, delenej dodávky (VRP)
- problém poradia sekvencií – skladanie DNA sekvencií),
- problém farbenia grafu (GCP),
- trasovanie v komunikačných sieťach.

Optimalizácia rojom častíc

- vhodný pre DNA sekvencie, pre problémy, ktoré nie sú diferencovateľné
- jednoduchý na implementáciu
- nastavuje iba pár parametrov simulácie
- inšpirovaný vtákmi
- každý jednotlivec si pamätá svoje najlepšie riešenie a globálne najlepšie riešenie
- smer pohybu sa skladá z troch zložiek
    - zotrvačnosť pokračovať v danom pohybe
    - pohyb smerom k jeho vlastnému najlepšiemu riešeniu
    - pohyb ku globálnemu najlepšiemu riešeniu
- účinne udržuje rozmanitosť
- Informácia dostupná každej častici je založená na jej **vlastnej skúsenosti** a **znalosti chovania sa ostatných častíc v jej okolí**.
- priebeh algoritmu:
    1. Inicializácia roja.
        - Každej častici je priradená náhodná pozícia v prehľadávanom priestore.
    2. Pre každú časticu je vypočítaná hodnota fitness funkcie.
    3. Porovnanie súčasnej hodnoty fitness funkcie častice s jej p_{best}.
        - Ak je súčasná hodnota lepšia, je označená za p_{best} a do p_i je uložená súčasná poloha častice.
    4. Nájdenie častice s najlepšou fitness funkciou v populácii. Jej hodnota je g_{best} a jej poloha p_g .
    5. Aktualizácia pozícii a rýchlosti častíc podľa rovníc

TOPOLÓGIA PSO



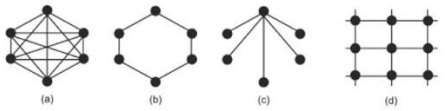
V PSO môžu byť dva základné druhy susedstva.

Je to globálne susedstvo, pri ktorom sú častice priťahované k najlepšiemu nájdenému riešeniu z celého roja.

Môžeme si to predstaviť ako plne prepojenú sieť, kde každá častica má prístup k všetkým informáciám. (obrázok a)

Druhou možnosťou je lokálne susedstvo. Častice sú priťahované k najlepšiemu riešeniu vybranému z bezprostredného susedstva. V tomto prípade existujú dve najčastejšie varianty: kruhová topológia (b) – každá častica je prepojená s dvomi susedmi, alebo centralizovaná topológia (c) – jednotlivé častice sú od seba oddelené a všetky informácie sú zhromažďované v hlavnom jedincovi.

TOPOLÓGIA PSO



Predpokladá sa, že plne prepojená sieť konverguje k riešeniu rýchlejšie, ale môže uviaznúť v lokálnom optime.

Prístup s obmedzeným susedstvom má väčšiu šancu nájsť optimálne riešenie, ale pomalšie.

Predpokladá sa, že najlepšie výsledky by mal dosiahnúť roj s von Neumannovou topológiou (obr. d).

- nastavenie rýchlosti častíc je dobré spraviť dynamicky, aby sa postupne znižovala

Genetický algoritmus

- vhodný pre DNA sekvencie

Diferenciálna evolúcia

- vhodná pre DNA sekvencie, pre multidimenzionálné nediferencovateľné funkcie, pre nespojité alebo zašumené problémy, problémy, ktoré sa menia v čase
- inicializácia je náhodná
- zavádza mutáciu, ktorá k danému jedincovi pripočíta rozdiel viacerých náhodne vybraných jedincov. Tento rozdiel sa prenásobí parametrom F, ktorého rozsah je (0, 2), najčastejšie 0.8
- vďaka mutácii je DE nemenná vzhľadom k rotáciám priestoru a ku škálovaniu osí
- pri krížení sa vyberie náhodný rodič a s určitou pravdepodobnosťou (zvyčajne 10%) sa do zmutovaného jedinca prenesie číslo z rovnakej pozície od rodiča. Vždy sa prenesie aspoň 1 hodnota.
- selekcia funguje tak, že sa porovnáva zmutovaný, skrížený jedinec s pôvodným jedincom a lepší z nich ide do ďalšej generácie
- základné kroky evolučných algoritmov:

- inicializácia
- mutácia
- kríženie
- selekcia

Simulované žíhanie

- stochastický optimalizačný algoritmus

- základ vo fyzike

- vhodný na kombinatorické úlohy (travelling salesman problem)

Znázornenie fyzikálnej realizácie žíhania. Teleso sa vloží do pece, ktorá je vyhriata na vysokú teplotu T_{max}. Teplota sa programovacím zariadením postupne znižuje na teplotu T_{min}. Častice telesa sa môžu postupne dostať do rovnovážnej polohy, čiže energia telesa sa znižuje. Týmto spôsobom sa odstránia štrukturálne defekty vyskytujúce sa v telese.

SIMULÁCIA FYZIKÁLNEJ EVOLÚCIE

Ak je rozdiel $\Delta E = E_{perturbed} - E_{current}$ medzi porušeným a aktuálnym stavom negatívny ($E_{perturbed} < E_{current}$) potom proces pokračuje s novým porušeným stavom.

V opačnom prípade, ak $\Delta E \geq 0$, pravdepodobnosť akceptovania porušeného stavu

$$\Pr(perturbed \leftarrow current) = \min(1, \exp(-\Delta E / kT))$$

Toto pravidlo akceptovania porušeného stavu sa nazýva Metropolisovo kritérium.

Tento tvar metódy Monte Carlo sa vo fyzike nazýva Metropolisov algoritmus.

IMPLEMENTÁCIA METROPOLISOVHO ALGORITMU

```
Procedure Metropolis_algorithm
Begin k : = 0; x : = x_ini;
        while k < k_max do
        begin k : = k+1;
                x' = : O_pertur(x);
                Pr : = min ( 1, exp ( - ( f ( x') - ( x ) ) / T ));
                if random < Pr then x : = x';
        end;
        x_out : = x;
end;
```

- vstupné parametre sú T_{min}, T_{max}, k_{max}, α
- výstupný parameter je x_{opt} .
- Algoritmus je inicializovaný náhodným generovaním počiatočného stavu x_{ini} a maximálnou teplotou T_{max} .
- While – cyklus sa opakuje tak dlho, pokiaľ platí $T > T_{min}$
- teplota T sa znižuje pomocou parametra α vzťahom $T := \alpha * T$.
- Po ukončení while-cyklu výsledný stav x_{out} je považovaný za výsledné riešenie označené x_{opt} .
- počiatočná teplota by mala byť zvolená tak, aby približne 50% porušených stavov bolo akceptovaných Metropolisovým algoritmom
- algoritmus akceptuje aj horšie riešenia (narozdiel od gradient descent)

paralelné simulované žíhanie

- na riešenie komplikovaných kombinatoriálnych grafovo teoretických problémov, kde štandardná verzia simulovaného žíhania nie je schopná poskytnúť globálne riešenie
- medzi stavmi sa používa kríženie

Algoritmus svätojánskych mušiek

- inpiraciou boli tropické svetlusky
- svetlusky vytvaraju svetlo pomocou bioluminiscencie na prilakanie partnerov a potencialnu korist a odohnat predatorov

- niektore druhy zosynchronizovavaju blikanie, cim vznika samoorganizacia spravania sa
- intenzita svetla v určenej vzdialenosti od zdroja svetla je nepriamo úmerná druhej mocnine vzdialenosti, tým pádom sa svetlušky vidia iba do určitej vzdialenosti od seba
- atraktivita svetlušky je priamo úmerná jej žiarivosti (jasu) a nepriamo úmerná jej vzdialenosti
- každá svetluška je priťahovaná k najatraktívnejšej svetluške vo svojom okolí. Ak žiadna atraktívna svetluška v jej okolí nie je, pohybuje sa náhodne
- žiarivosť svetlušky je daná jej polohou a účelovou funkciou
- pri inicializácii algoritmu je dôležité, aby boli svetlušky po ploche rozmiestnené pokiaľ je možné čo najrovnomernejšie. Tým sa zvyšuje pravdepodobnosť vyhľadania globálneho optima účelovej funkcie
- koeficient absorbcie (gamma) ovplyvňuje správanie algoritmu a rýchlosť konvergencie, vyjadruje do akej vzdialenosti sú mušky schopné vidieť, ak sa mušky nevidia (gamma je velka hodnota), hýbu sa náhodne, ak je gamma mala hodnota, algoritmus sa stava podobnym PSO
- zakladny algoritmus pozostava z 3 krokov po inicializácii polôh mušiek
    - Určenie žiarivosti mušiek
    - Určenie atraktívnosti každej mušky ku ostatným muškám, aby sme zistili ku ktorej bude priťahovaná
    - Pohyb mušky

- mutacia je pouzita pre globalne aj lokalne prehladavanie

- specialnou vlastnostou tohto algoritmu je pouzitie **pritazlivosti** – lokalna pritazlivost je silnejsia ako priazlivost na vacsie vzdialenosti a tym padom sa musky rozdelia na podskupiny, okolo lokalnych extremov, najlepsie najdene riesenie je optimalnym riesenim skumaneho problemu

- ak gamma = 0 a alpha (mutacia) = 0, tak sa z tohto algoritmu stave diferencna evolucia

- ak gamma = 0, alpha = 0, beta = 0 (atraktivnost musky), tak sa z algoritmu stava simulovane zihanie

- ak musky nie su porovnavane s inymi muskami x_j ale s najlepsou muskou, tak sa z algoritmu stava zrychlena verzia PSO

- z toho vyplyva, ze DE, SimZihanie a PSO su specialnymi pripadmy algoritmu svatojanskych musiek, preto ma algoritmus svatojanskych musiek vyhody vsetkych troch algoritmov

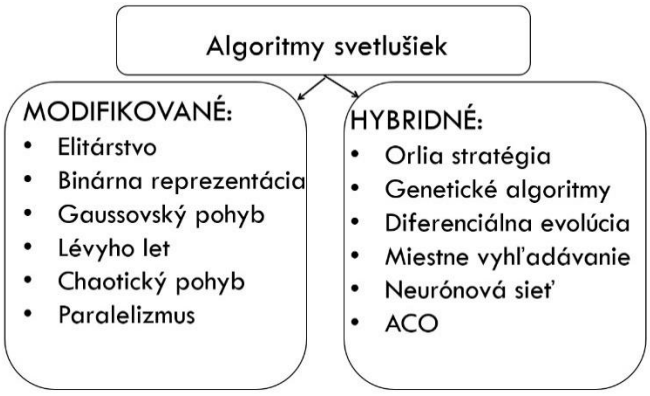
- algoritmus svatojanskych musiek sa povazuje za globalny optimalizator, jeho vykonnost patri medzi najlepsie (hlavne lepsi ako PSO), avsak nie je samozrejme najlepsi pre kazdy problem (lebo no free lunch)

- pouziva sa najme na optimalizaciu spojitych problemov a na viacucelovu optimalizaciu (kvoli rozdeleniu svetlusiek do podskupin)

Algoritmus FA je používaný najmä pri:

- **Optimalizácii**
    - Kombinatorická optimalizácia
    - Spojitá optimalizácia
    - Obmedzujúca optimalizácia
    - Multiobjektívna optimalizácia
    - Dynamická optimalizácia
- **Klasifikácii**
- **Inžinierkých aplikáciách** – spracovanie obrazu, robotika, sémantické weby, obchod, sieťové problémy, priemyselná optimalizácia, chémia, stavebníctvo

- hybridizacia moze prebiehat v takmer kazdej zlozke algoritmu (inicializacia, hodnotenie fitness, hodnotenie pohybu)



Generalized Evolutionary Walkthrough Algorithm

GEWA

Na základe spoločných charakteristik meta-heuristických algoritmov kukučie vyhľadávanie, genetický algoritmus, simulované žíhánie a PSO.

X. S. Yang navrhol nový obecný meta-heuristický algoritmus pre optimalizáciu a nazval ho Generalized Evolutionary Walk Algorithm (GEWA - generalizovaný evolučný chodiaci algoritmus.

GEWA má nasledujúce tri hlavné časti:

- globálny prieskum randomizáciou

- intenzívne lokálne vyhľadávanie pomocou náhodnej prechádzky,

- výber najlepšieho pomocou elitárstva.

Zaujímavé je, že tento algoritmus sa snaží vyvážiť prieskum a využiť ho pre nový kontrolný parameter randomizácie.

YANG, X. S. Review of meta-heuristics and generalised evolutionary walk algorithm, *International Journal of Bio-Inspired Computation*, 2011, 3 (2), 77-84.

Kukučí algoritmus

- inspirovany parazitizmom kukuciek pri hniezdeni, kukucky obvykle znasaju svoje vajcia do hniezd inych druhov vtakov, ktorych vajcia sa vzhladom aj velkostou podobaju kukucim a aj nacasovanie znasania vajec

- existuju zakladne 3 druhy parazitizmu

    - vnutrodruhovy

    - kooperacny

- preberanie hniezd

- ak hostitelsky vtak zisti, ze ma v hniezde cudzie vajicko, moze sa ho zbavit alebo hniezdo opustit

- kukucie mladata sa liahnu skorej a ich instiktom je po vyliahnuti zbavit sa ostatnych vajec, aby si zaistili vyhodu pri krmeni

- okrem parazitizmu je tento algoritmus doplneny Levyho letom, co zarucuje lepsi pohyb ako pri nahodnom kroku

- pravidla algoritmu

  - kazda kukucka znesie jedno vajce do nahodne vybraneho hniezda v kazdom casovom kroku

  - do dalsich generacii bude zachovane hniezdo s najkvalitnejsimi vajciami

  - pocet hostitelskych hniezd je konstantny

  - je dana nejaka pravdepodobnost, ze hostitelsky vtak objavi kukucie vajce

  - ak je vajce objavene, hostitel sa ho zbavi, alebo hniezdo opusti a vybuduje si nove (nove nahodne riesenia)

- v najjednoduchsej implementacii moze mat kazde hniezdo iba jedno vajicko

- cielom algoritmu je v kazdom kroku nahradit riesenie v hniezde lepsim (kukucim) riesenim (vajcom)

- Lévyho let je inspirovany pohybom zvierat schopnych letiet na dlhu vzdialenost, je to vlastne nahodna prechadzka, pri ktorej je dlzka kroku urcena levyho distribuciou

definuj účelovú funkciu $f(\boldsymbol{x})$ $\quad \boldsymbol{x} = (x_1, \dots, x_d)^T$
pre všetky hniezda $i \in \{1, \dots, n\}$:
 inicializuj počiatočné polohy $\boldsymbol{x_i^0}$;
pokiaľ (nie je splnená ukončovacia podmienka):
  vyber kukučku i;
  generuj nové riešenie (pomocou Lévyho letu)
  vyber hniezdo j;
 pokiaľ $f(x_i) < f(x_j)$:
    nahraď riešenie v hniezde riešením kukučky;
  koniec
  znič p_a časť najhorších hniezd;
  náhodne generuj nové hniezda;
  ohodnoť riešenia v hniezdach a nájdi najlepšie;
  t = t + 1;
koniec

Netopieri algoritmus

- velmi efektivny algoritmus

- simuluje spravanie netopierov pri hladani potravy

- netopiere su jedine cicavce s kridlami

- netopiere vyuzivaju echolokaciu (meranie vzdialenosti medzi netopierom a prekazkou/potravou)

- pri netopieroch sledujeme rychlost pohybu, frekvenciu a hlasitost

- ak netopier narazi na potencialnu potravu, znizi hlasitost a zvysi frekvenciu

- netopier (agent) ma premenne: pozicia, rychlost, hlasitost, frekvencia a/alebo vlnova dlzka

- lokalne vyhladavanie: akonahle je riesenie vybrane zo sucasnych najlepsich rieseni, nove riesenie pre kazdeho netopiera je generovane lokalne pomocou nahodnej prechadzky (ku rieseniu sa pripocita nejaky nahodna hlasitost netopiera

- netopieri algoritmus byva oproti PSO lepsi, pretoze pouziva frekvencne ladenie a riadenie parametrov vyhladavania

- pri jednotlivych iteraciach sa aktualizovava hlasitost a rychlost vydanych zvukovych impulzov

    - hlasitost klesa, zatial co rychlost impulzov sa zvysuje

- pravidla algoritmu

    - netopiere pouzivaju echolokaciu a poznaju rozdiel medzi prekazkou a koristou

    - netopiere lietaju nahodne s rychlostou v a polohou x, mozu sa automaticky nastavit

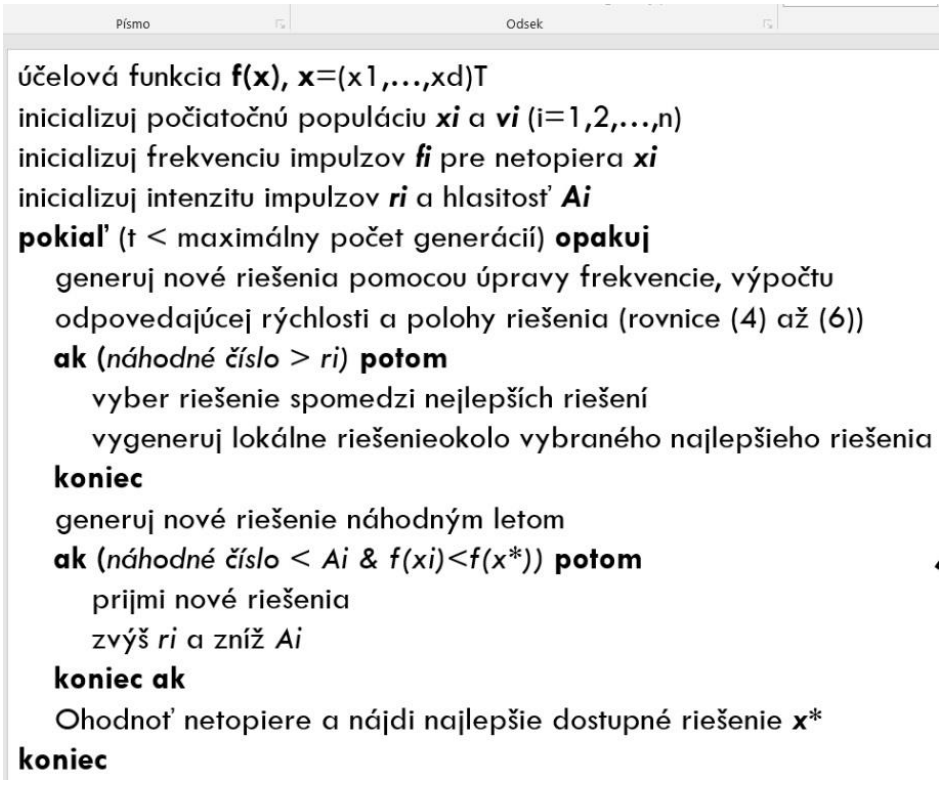
    - frekvencia a rychlost vydavaneho impulzu zavisi od vzdialenosti koristi

    - hlasitost sa meni od A_{max} po A_{min} , kde A je hlasitost

    - vyssie frekvencie dojdu na kratsiu vzdialenost

- kvoli vypoctovemu zjednoduseniu mozeme pri multidimenzionalnej optimalizacii zjednodusit alg na:

    - frekvencia koresponduje vlnovej dlzke, popripade vlnove dlzky nemusime vobec pouzivat



Algoritmus opelovania kvetov

- inspirovany opelovanim kvetov

- globalne opelovanie: opelovanie na velke vzdialenosti pomocou levyho letu

- lokalne opelovanie: samoopelovanie

- pravdepodobnost reprodukcie zavisi od podobnosti kvetov, ktore sa opeluju navzajom

- na kontrolovanie lokalneho a globalneho opelovania sa pouziva switch probability (nahodne cislo medzi 0 a 1)

- kazda rastlina ma iba jeden kvet, ktory ma v sebe iba jedno zrniecko pelu, tym padom mozeme riesenie povazovat ako jeden kvet alebo ako jednu rastlinu

```
Vytvor populáciu n kvetín s náhodne vygenerovanými riešeniami;
Ohodnoť riešenia;
Nájdi najlepšie riešenie z populácie;
Nastav switch probability p ϵ[0, 1] ;
t = 0;
Definuj maximálny počet iterácií (maxIter);
while t < maxIter do
    for každá kvetina v populácií do
        if rand < p then
            Vykonaj globálne opelenie
        end
        else
            Vykonaj lokálne opelenie
        end
        Ohodnoť riešenia;
        Updatni riešenie ak je nové lepšie ako aktuálne;
    end
    Nájdi aktuálne najlepšie riešenie g* a updatni g*;
    t = t + 1
end
Vypíš najlepšie riešenie;
```

Algoritmus bakterialnej optimalizacie

- sklada sa zo 4 zakladnych krokov

    - chemotactic

    - reprodukcia

    - eliminacia

    - rozptylenie

- chemotactic: bakteria sa pohybuje smerom k oblasti s vela zivinami, pokym nenarazi na nepriaznivu oblast

    - ulohou algoritmu je minimalizovat pohyb bakterie v urodnej oblasti

    - na konci chemotactic kroku sa usporiadaju bakterie zostupne podla fitness

- reprodukcia: kazda bakteria z lepsej polovica sa zduplikuje

- eliminacia: kazda bakteria z horsej polovice umrie

- rozptylenie: nahodne rozmiestnenie bakterii (vdaka tomu sa neuviazne v lokalnom extreme)

Sociálny algoritmus optimalizácie pavúkov

- tento algoritmus obsahuje dva zakladne atributy: pavuky a pavucinu

- pavuky sa delia na samicky a samcov

- kolonia pavukov sa sklada zo 70% samcov a 30% samiciek

- samicky su bud pritahovane alebo odpudzovane

- samicky su bud pritahovane alebo odpudzovane od ostatnych pavukov na zaklade ich vibracii, ktore suvisia s ich vahou (fitness) a vzdialenostou

- samci sa delia do dvoch skupin:

    - dominantni

    - submisivni

- dominantny pavuk ma lepsiu fitness ako nedominantny

- v tomto algoritme existuje „mating operator“, ktory sa pouziva pri vymene informacii medzi samickami a dominantnymi pavukmi

- dominantni pavuci sa stretavaju so samickami a vytvaraju potomstvo (nove riesenia)

Algoritmus skakajucej zaby

- Tento algoritmus kombinuje Memetic algorithm a PSO

- je inspirovany pohybom ziab, ktore si tymto pohybom predavaju informacie pri hladani potravy

- kazda zaba predstavuje potencialne riesenie daneho problemu

- po vypocitani fitness vsetkych ziab, su tieto zaby zoredene podla fitness zostupne

- potom sa zaby rozdelia do skupiniek nazyvanych memplex

- v kazdej skupinke je rovnaky pocet ziab

- pre kazdy memeplex sa vypocita najlepsie riesenie

- v kazdom memeplexe prebiehaju memetics evolucie

- potom su vsetky zaby spojene a vykona sa globalna evolucia

Umely imunitny system

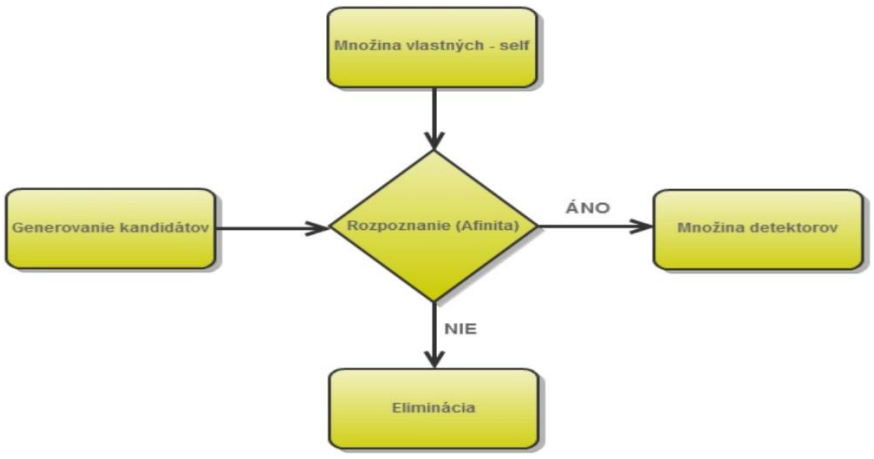
- decentralizovany system

- vrodena a speficka imunita (dva typy)

- zakladne vlastnosti imunitneho systemu:

- jedinecnost
- distribuovanost (decantralizovanost)
- paralelita
- schopnost ucenia (pamat)
- robustnost
- odolnost voci sumu
- detekcia anomalii

- algoritmy umeleho imunitneho systemu
  - pozitivna selekcia
    - inspirovany dozrievanim T-lymfocitov v tymuse
    - aplikuje sa na eliminaciu T-lymfocitov bez receptorov alebo zbytocnych T-lymfocitov
    - vytvaranie mnoziny detektorov pre vlastne bunky
    - algoritmus (self – mnozina vlastnych prvkov, ktore treba chranit, n – detektori)
      - inicializovanie (nahodne vygenerovanie mnoziny kandidatov na detektory)
      - cenzura (pokial nebola vyprodukovana mnozina detektorov o velkosti n)
        - vyhodnotenie afinity medzi kazdym vlastnym prvkov a kandidatom
        - selekcia (ak kandidat rozpozna niektory element mnoziny self, je tento kandidat pridany do mnoziny detektorov. V opacnom pripade je eliminovany



  - negativna selekcia
    - inspirovany dozrievanim lymfocitov v detskej zlaze
    - lymfocity sa testuju, ci dokazu rozpoznat vlastne bunky
    - ak lymfocit rozpozna vlastnu bunku, je eliminovany, zabezpecuje sa tak to, ze bude utocit iba na cudzie bunky

- vytvaranie mnoziny detektorov pre cudzie bunky

Algoritmus

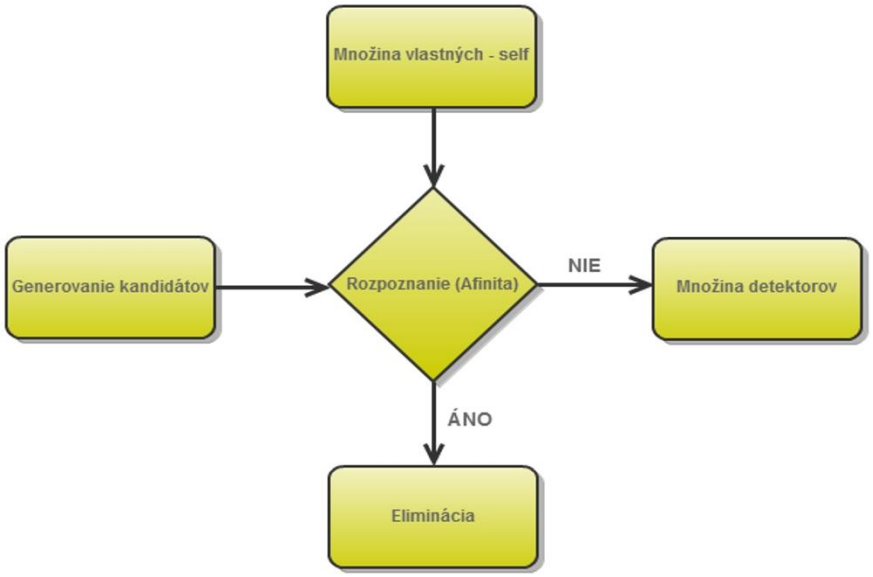
Majme množinu vlastných prvkov self, ktoré chceme chrániť a definovanú veľkosť množiny detektorov n. Potom samotný algoritmus vyzerá nasledovne:

1. Inicializovanie - náhodne vygenerujem množinu kandidátov na detektory.

2. Cenzúra - pokiaľ nebola vyprodukovaná množina detektorov o veľkosti n

(a) Vyhodnotenie afinity medzi každým vlastným prvkom a kandidátom.

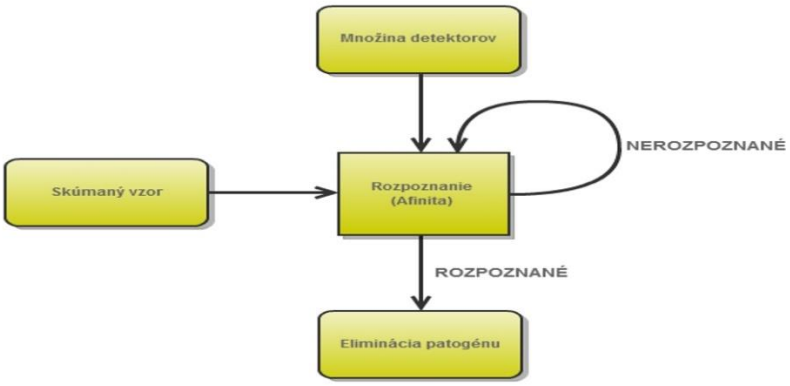
(b) Ak kandidát rozpozná niektorý element množiny self, je tento kandidát eliminovaný. V opačnom prípade je umiestnený do množiny detektorov.



ALGORITMUS DETEKCIE

V prípade tohto algoritmu ide v podstate o beh umelého imunitného systému. **Nerozpoznaný prvok** je **vlastný** (self – imunitný systém nezasahuje).

Rozpoznaný prvok je **nevlastný** (non-self - nastáva reakcia systému a **eliminácia patogénu**). V tomto prípade je množina detektorov porovnávaná s kontrolovanou množinou. Ak je prvok z kontrolovanej množiny rozpoznaný, môžeme o ňom prehlásiť, že je nevlastný **non-self.**



ALGORITMUS KLONÁLNEJ SELEKCIE

Tento algoritmus v svojich rôznych úpravách patrí v oblasti umelých imunitných systémov k najčastejšie používaným technikám.

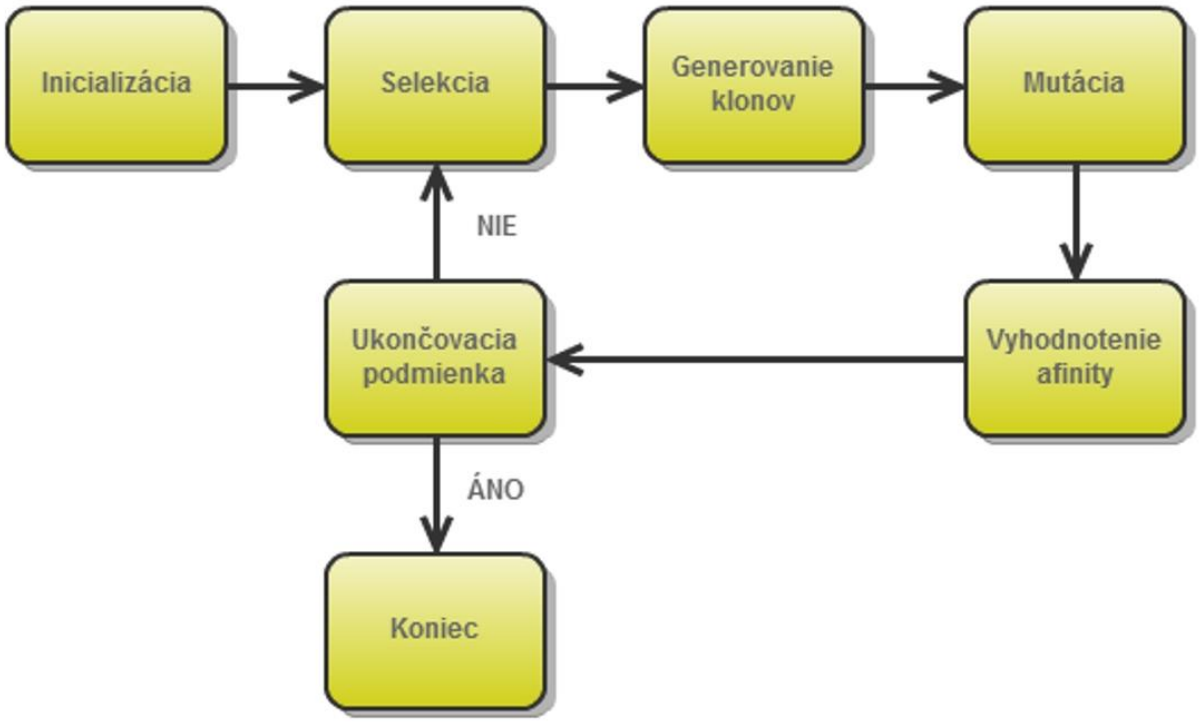
Algoritmus

Majme množinu antigénov, ktoré chceme rozpoznávať a veľkosť množiny protilátok n, ktoré chceme vyprodukovať.

1. Inicializovanie - náhodne vygenerujem populáciu imunitných buniek.

2. Generovanie populácie - pre každý antigén
   - Vyberieme len tie bunky, ktoré majú najvyššiu afinitu k antigénu.
   - Generovanie klonov - čím lepšie daná bunka antigén rozpoznáva, tým viac kópií bunky vyprodukujeme.
   - Mutácia - každú novú bunku zmutujeme podľa pravidla - čím je afinita väčšia, tým budú mutácie danej bunky menšie.
   - Vyhodnotenie afinity - pre každú zmutovanú bunku vyhodnotíme afinitu k antigénu

3. Krok 2 opakujeme až do splnenia ukončovacieho kritéria (miera afinity, počet cyklov. . .)



Algoritmus klonálnej selekcie

Memeticke algoritmy

- Memetické algoritmy sú metódou heuristického prehľadávania založenou na paradigme **kultúrnej evolúcie**.
- V princípe **kombinujú heuristiky lokálneho prehľadávania s operátormi kríženia** (krizenie najcastejsie greedy algoritmami)

MEMETICKÝ ALGORITMUS

V každej generácii sa najprv vykoná selekcia jedincov na kríženie, počiatočný stav pre potomka sa následne vytvorí prostredníctvom operátora kríženia.

Potomok sa následne stáva začiatočným stavom pre lokálne hľadanie.

Do nasledujúcej generácie sa potom štandardne vyberá stav z lokálneho hľadania s najvyššou hodnotou fitness funkcie.

Memetický algoritmus

Vstup: počet jedincov |P|, maximálny počet generácií gmax, maximálny počet iterácií lokálneho hľadania tmax

Výstup: výstupný stav S

```
1 P = generuj_začiatočnú_populáciu()
2 pre g = 1..gmax
3 ak P obsahuje optimálne riešenie S
4 vráť S
5 Q = Ø
6 pre i = 1..|P|
7 [P1, P2, ..., Pk] = selekcia(P)
8 O = kríženie(P1, P2, ..., Pk)
9 O = lokálne_hľadanie(O,tmax)
10 Q = Q ∪ O
11 P = Q
12 vráť S = najlepší_nájdený_stav()
```

MÉMY

Mémy sú mentálne replikátory; entity, ktoré parazitujú v našich mysliach a ovplyvňujú naše správanie.

Ich jediným cieľom je - prežiť a šíriť sa ďalej - bez ohľadu na spôsob a cenu týchto aktivít.

Memetika nachádza uplatnenie v psychológii, politike, reklame a marketingu, sociológii a evolučných teóriách.