**Dokumentácia**

**Cat Swarm Optimization**

**Lukáš Gereg**

**Martin Matta**

**Barbora Kubalcová**

**Martina Marasová**

**Technická univerzita v Košiciach Katedra kybernetiky a umelej inteligencie**

**Fakulta elektrotechniky a informatiky Inteligentné systémy, 2. ročník**

Obsah

[Mačací algoritmus (CSO) – popis a inšpirácia 3](#_Toc121574403)

[Matematický model 4](#_Toc121574404)

[Hlavná časť algoritmu 4](#_Toc121574405)

[Pozorovací mód (seeking mode) 5](#_Toc121574406)

[Prenasledovací mód (tracing mode) 6](#_Toc121574407)

[Terminačné kritérium 6](#_Toc121574408)

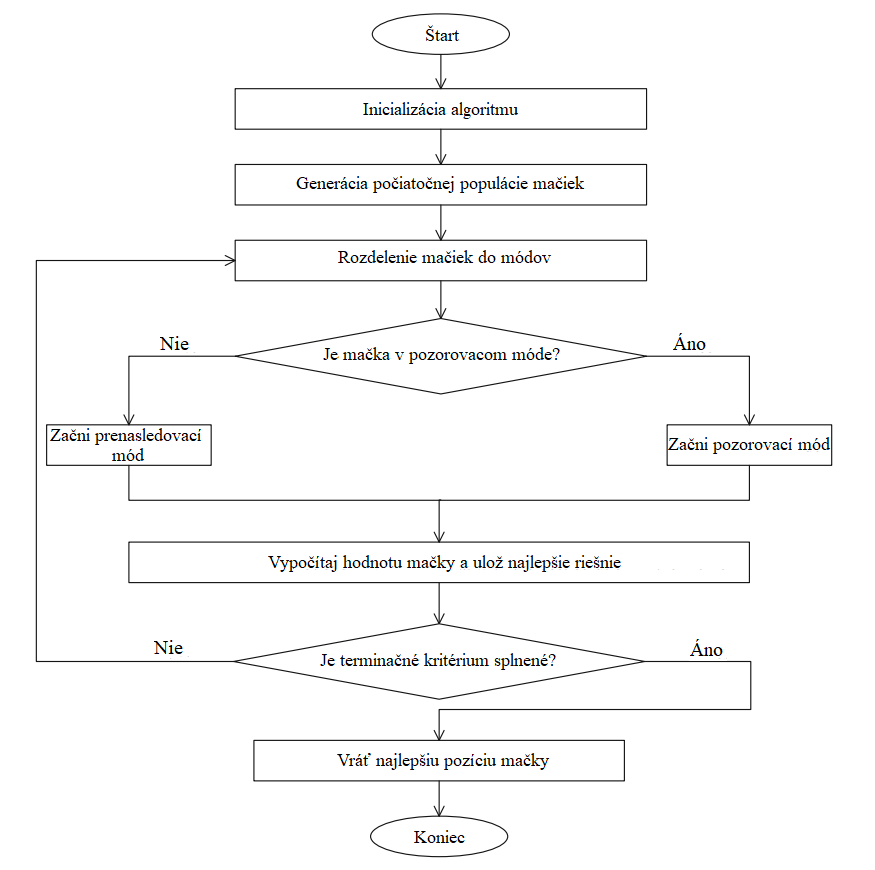
[Ohodnocovacia funkcia 6](#_Toc121574409)

# Mačací algoritmus (CSO) – popis a inšpirácia

CSO je algoritmus inšpirovaný správaním mačiek v reálnom svete. **Prírodné mačky** strávia väčšinu času v pokoji, sústredia sa na pozorovanie svojho okolia a striehnu na korisť. Aj keď sú v tejto fáze nečinné, sú veľmi ostražité a ostávajú v pozore. Ak zaznamenajú nejaký pohyb, pustia sa do akcie a začnú korisť prenasledovať. **Umelá mačka** má preto dva módy – pozorovací a sledovací.

* **Pozorovací mód** (seeking mode) - mačka stojí na mieste a pozoruje okolie
* **Prenasledovací mód** (tracking mode) - mačka prenasleduje korisť a mení svoju pozíciu

Na začiatku problému rozhodneme, koľko mačiek chceme v algoritme použiť – čím viac, tým lepšie. Vytvoríme populáciu *n* mačiek a náhodne ich rozmiestnime po *m*-dimenzionálnom priestore. Každá mačka v tomto priestore predstavuje jedno riešenie. Mačky následne rozdelíme do dvoch skupín – tie, čo pozorujú a tie, čo prenasledujú. Každú mačku následne ohodnotíme pomocou ohodnocovacej funkcie a uložíme najlepšiu mačku. Mačka predstavuje jedno potencionálne riešenie problému. Ako sa už spomínalo vyššie, väčšinu svojho času strávia pozorovaním, preto bude skupina pozorujúcich mačiek väčšia. Pomer mačiek v skupinách nám udáva **MR** (mixture ratio), ktorý si taktiež stanovíme na začiatku.



Obr. 1 – Vývojový diagram CSO algoritmu

## Matematický model

Každá mačka je umiestnená v *m*-dimenzionálnom priestore kandidátov, a teda má *m* súradníc, ktoré určujú jej pozíciu *Xn*. Ďalej má označenie, v ktorom z módov sa aktuálne nachádza, určitú rýchlosť *vn* pre každú dimenziu a hodnotu z ohodnocovacej funkcie. Posledné riešenie po ukončení algoritmu bude súčasne aj najlepším nájdeným riešením, keďže uchovávame doposiaľ najlepšie riešenie až do splnenia ukončovacej podmienky. Tento algoritmus nám síce nezaručuje, že nájdeme optimálne riešenie, ale výsledok je dostatočný.

### Hlavná časť algoritmu

Postup algoritmu je znázornený na vývojom diagrame (Obr. 1). Slovný opis:

1. Vytvor inicializačnú populáciu mačiek, umiestni ich náhodne do *m*-dimenzionálneho priestoru a každej mačke priraď náhodnú rýchlosť.
2. Každej mačke priraď mód vo zvolenom pomere (MR).
3. Pre každú mačku vypočítaj hodnotu a do pamäte ulož najlepšiu mačku. Najlepšie riešenie predstavuje pozícia mačky.
4. Na základe prideleného módu, vykonaj hľadanie alebo pozorovanie.
5. Ak je splnené terminačné kritérium, ukonči proces, inak sa vráť ku kroku 2.

##### Pseudokód

Input: π, n n – počet mačiek, π - problém

Output: Xbest Є S Xbest – pozícia najlepšej mačky

Initialize n cats

**Repeat**

**For** i = 1 : n

Set mode for cat[i]

Evaluate cat[i]

**If** g(Xbest) < g(cat[i].Xi) **then**

Xbest = cat[i].Xi

**Endif**

**Endfor**

**For** i = 1 : n

**If** cat[i] is in seeking mode **then**

Start seeking mode

**Else**

Start tracing mode Mačka sa pohne k najlepšiemu riešeniu

**Endif**

**Endfor**

**Until** term()

**Return** Xbest

### Pozorovací mód (seeking mode)

V tomto móde mačka ostáva na mieste a pozoruje svoje okolie. V prípade detekcie koristi (poprípade nebezpečenstva) sa rozhodne zmeniť svoju polohu na základe štyroch faktorov:

* **SMP** (seeking memory pool) – počet vytvorených kópií (klonov) mačky
* **SRD** (seeking range of the selected dimension) – „ako ďaleko“ sa vie mačka pozrieť (posun v dimenzii). Ak sa niektorá z dimenzií bude meniť, tak rozdiel medzi novou a starou hodnotou bude SRD.
* **CDC** (counts of dimension to change) – koľko dimenzií bude zmenených
* **SPC** (self-position consideration) – boolovská hodnota. Rozhodnutie mačky ostať na aktuálnej pozícii alebo odísť.

#### Postup

1. Urob *i* počet kópií každej mačkyi, pričom *i* = SMP. Ak je SPC pravdivé, vytvor *i* = (SMP - 1) kópií mačky, pričom jedna ostane na pôvodnom mieste.
2. Na základe CDC vypočítaj novú pozíciu pre každú kópiu mačky pomocou rovnice

kde

*Xcn* je nová pozícia mačky,

*Xc* je aktuálna pozícia,

*R* je náhodné číslo z intervalu <0, 1>

1. Vypočítaj hodnoty mačiek (*FS*) pre ich nové pozície. Ak sú všetky hodnoty z ohodnocovacej funkcie rovnaké, nastav pravdepodobnosť výberu na 1 pre všetkých kandidátov, inak vypočítaj pravdepodobnosť výberu podľa vzorca

kde

*Pi* je pravdepodobnosť výberu aktuálneho kandidáta

*FSi* hodnota mačky z ohodnocovacej funkcie

*FSmax* maximálna hodnota ohodnocovacej funkcie

*FSmin*  minimálna hodnota ohodnocovacej funkcie

Riešime

*FSb = FSmax* pre minimalizačný problém

*FSb = FSmin* pre maximalizačný problém

1. Pomocou vybranej stratégie (roul-wheel, tournament, best fitness-self, random choice) náhodne vyber cieľový bod a zmeň pozíciu mačkyi.

V našej implementácii používame „best fitness-self“ a „random choice“ stratégiu“

* Random choide – úplne náhodný výber
* Best fitness-self – zoradíme mačky podľa výsledku ohodnocovacej funkcie od najhoršej po najlepšiu a vrátime najlepšiu mačku

### Prenasledovací mód (tracing mode)

Po tom ako mačka zaznamenala korisť v pozorovacom móde, zmenila svoj mód na prenasledovací a začína loviť. Mačka sa teda pohne nejakou rýchlosťou smerom ku koristi.

#### Postup

* + 1. Aktualizuj rýchlosť mačky v každej dimenzii (*vk,d*) na základe rovnice

kde

*vk,d* je rýchlosť mačky v dimenzii *d*

*Xbest,d* je pozícia najlepšej mačky

*Xk,d* pozícia mačkyk

*c1* konštanta

*r1* náhodná hodnota z intervalu <0, 1>

* + 1. Skontroluj, či výsledná rýchlosť je v intervale <0, *vmax*>. Ak nie, nastav rýchlosť na *vmax*.
    2. Aktualizuj pozíciu mačky pomocou rovnice

kde

*Xk,d,new* je nová pozícia mačkyk v dimenzii *d*

*Xk,d,old* je aktuálna pozícia mačkyk v dimenzii *d*

### Terminačné kritérium

Terminačné kritérium nám určuje, kedy sa ma prehľadávanie priestoru kandidátov ukončiť. Jeho správne nastavenie je veľmi dôležité, pretože keď skončíme prehľadávanie skôr, nemusíme dostať najlepší výsledok. Ako terminačné kritérium sa môže brať počet iterácií, veľkosť zlepšenia, čas a podobne.

V našej implementácii riešime problém MAX SAT, preto je naším terminačným kritériom počet splnených klauzúl – buď všetky alebo čo najviac.

### Ohodnocovacia funkcia

Funkcia dostane na vstupe jednu mačku a hľadá, koľko klauzúl je splnených pri danom nastavení pozície mačky. Vráti počet splnených klauzúl, pričom maximálna hodnota, ktorú môžeme dostať, je dĺžka listu s problémom.