Rješavanje optimizacijskih problema algoritmima evolucijskog računanja u Javi
Algoritam simuliranog kaljenja.

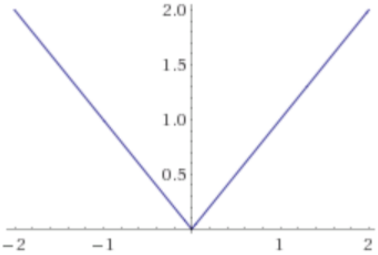
dr.sc. Marko Čupić

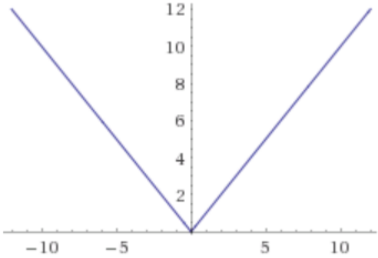
Fakultet elektrotehnike i računarstva
Sveučilište u Zagrebu
Akademska godina 2013./2014.

17. listopada 2013.

Primjer

Tražimo minimum funkcije: $f(x) = |x|$ u intervalu $[-10, 10]$. Funkcija je prikazana na slici.





Minimum se postiže u $x^* = 0$ i iznosi $f(x^*) = 0$.

Pohlepni algoritam

Pretpostavimo da radimo sa sljedećim optimizacijskim algoritmom.

Generiraj početno rješenje $\omega \in \Omega$
Izračunaj vrijednost funkcije $f(\omega)$ te dobrotu rješenja $fit(\omega)$
Ponavljaj dok nije zadovoljen uvjet zaustavljanja
    Generiraj susjedno rješenje $\omega^{'} \in N(\omega)$
    Izračunaj vrijednost funkcije $f(\omega^{'})$ te dobrotu rješenja $fit(\omega^{'})$
    Ako je $fit(\omega^{'}) > fit(\omega)$ prihvati $\omega^{'}$, tj. postavi $\omega \leftarrow \omega^{'}$
Kraj ponavljanja
Vrati ω kao rješenje

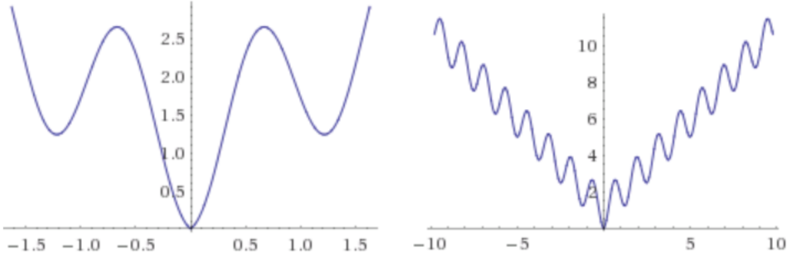
Pohlepni algoritam

Pretpostavimo sada da je u našem primjeru funkcija koja izvlači jedno slučajno rješenje iz susjedstva rješenja x definirana kao $x' = x + unifRand(-0.1, 0.1)$.

- Kakvo ponašanje možemo očekivati od pohlepnog algoritma uz takvo susjedstvo? Zašto? (isprobati)
- Što bi se promijenilo u ponašanju ako bismo susjedstvo definirali kao $x' = x + unifRand(-10, 10)$? Poopćite zaključak na funkcije velikog broja varijabli.

Primjer 2

Tražimo minimum funkcije: $f(x) = |x| + 1 - \cos(5 \cdot x)$ u intervalu $[-10, 10]$. Funkcija je prikazana na slici.



Minimum se postiže u $x^* = 0$ i iznosi $f(x^*) = 0$.

Primjer 2: pitanja

Pretpostavimo sada da je u našem primjeru funkcija koja izvlači jedno slučajno rješenje iz susjedstva rješenja x definirana kao $x' = x + unifRand(-0.1, 0.1)$.

- Kakvo ponašanje možemo očekivati od pohlepnog algoritma uz takvo susjedstvo? Zašto? (isprobati)
- Što bi se promijenilo u ponašanju ako bismo susjedstvo definirali kao $x' = x + unifRand(-10, 10)$? Poopćite zaključak na funkcije velikog broja varijabli.

Zaključak

Pohlepni algoritam već i na jednostavnim problemima može zapeti u lokalnom optimumu.

- Determinističko prihvaćanje isključivo boljih rješenja nije dobra strategija!
- Treba osigurati mogućnost izlaska iz lokalnih optimuma.

Analogija iz stvarnog života

Kako bi se dobili metali s povoljnim karakteristikama, u metalurgiji se koristi proces *kaljenja* metala.

- Metal se zagrijava do vrlo visokih temperatura.
- Potom se postupno hladi.
- Posljedica:
  - konfiguracija atoma metala postupno se preslaguje u strukturu iz koje malo po malo nestaju sve nepravilnosti;
  - čitav sustav nizom takvih promjenom prelazi iz stanja visoke energije u stanje s niske energije
- Ako se proces hlađenja napravi prebrzo, neće biti dovoljno vremena da se rekonfiguriraju sve nepravilnosti $\Rightarrow$ sustav će ostati u stanju nešto više energije.

Fizikalni opis postupka

Svaka rekonfiguracija položaja atoma sustav prevodi iz stanja energije E_1 u stanje energije E_2. Time svakom rekonfiguracijom dolazi do promjene energije $\Delta E = E_2 - E_1$.

- Rekonfiguracije uz koje je $\Delta E < 0$ su uvijek i fizikalno moguće.
- Međutim, unatoč procesu hlađenja koji iz sustava uklanja energiju, moguće su i rekonfiguracije uz koje je $\Delta E > 0$, tj. uz koje dolazi do prelaska u više energetsko stanje. Takve se rekonfiguracije događaju uz vjerojatnost određenu izrazom:

$$P(\Delta E) = \exp(\frac{-\Delta E}{k \cdot t}) \tag{1}$$

gdje je k Boltzmannova konstanta $(1.3806503 \cdot 10^{-23} \ \frac{m^2 kg}{s^2 K})$. Što je ΔE veći, takva promjena je manje vjerojatna.

Rubno ponašanje

Što se događa s vjerojatnostima pri rubnim temperaturama? Za ograničeni iznos $\Delta E > 0$ te kada $t \to \infty$ vrijedi:

$$\begin{aligned}\lim_{t\to\infty} \frac{1}{\exp(\frac{\Delta E}{k \cdot t})} &= \frac{1}{\exp(\frac{\Delta E}{\infty})} \\ &= \frac{1}{\exp(0)} \\ &= \frac{1}{1} \\ &= 1\end{aligned}$$

U tom slučaju sve su promjene moguće, neovisno o tome koliko se time pravilnost konfiguracije atoma narušava (i energija raste).

Rubno ponašanje

Što se događa s vjerojatnostima pri rubnim temperaturama? Za ograničeni iznos $\Delta E > 0$ te kada $t \to 0$ vrijedi:

$$\begin{aligned}
\lim_{t \to 0} \frac{1}{\exp(\frac{\Delta E}{k \cdot t})} &= \frac{1}{\exp(\frac{\Delta E}{0})} \\
&= \frac{1}{\exp(\infty)} \\
&= \frac{1}{\infty} \\
&= 0.
\end{aligned}$$

U tom slučaju bilo koja promjena koja bi narušila pravilnost konfiguracije atoma i dovela do porasta energije je nemoguća.

Kaljenje je optimizacijski proces

Postupak kaljenja direktna je fizikalna realizacija optimizacijskog postupka kojim se sustav (metal) pokušava prevesti u stanje minimalne energije rekonfiguriranjem položaja atoma.

Kaljenje metala	Kombinatorička optimizacija
Moguća stanja sustava	Prihvatljiva rješenja
Energija sustava	Funkcija kazne
Promjena stanja sustava	Prelazak u susjedno rješenje
Temperatura	Parametar koji simulira temperaturu
Zamrznuto stanje	Optimum (lokalni ili globalni)

Minimizacija vs maksimizacija

Algoritmi direktno temeljeni na postupku kaljenja oponašaju proces smanjivanja energije sustava $\rightarrow$ rade minimizaciju.

Pri tome pozitivni ΔE označava pogoršanje rješenja i definirana je vjerojatnost prihvaćanja takve promjene.

Negativni ΔE označava poboljšanje rješenja i takva se promjena prihvaća.

Stoga možemo postupati na sljedeći način:

- Ako radimo minimizaciju, E poistovjećujemo s iznosom funkcije čiji tražimo minimum, računamo $\Delta E = E_2 - E_1 \approx f_2 - f_1$.
- Ako radimo maksimizaciju, E poistovjećujemo s iznosom lošoće rješenja (npr. minus iznos funkcije, računamo $\Delta E = E_2 - E_1 \approx (-f_2) - (-f_1) = f_1 - f_2$ odnosno ako poistovjetimo $E_i \approx f_i$ računamo $\Delta E = E_1 - E_2$. Dalje sve ostaje isto.

Algoritam simuliranog kaljenja

Generiraj početno rješenje $\omega \in \Omega$
Postavi brojač za promjenu temperature na $k = 0$
Odaberi plan hlađenja t_k i početnu temperaturu $t_0 \geq 0$
Odredi plan za M_k – broj ponavljanja petlje pri temperaturi t_k
Ponavljaj dok nije zadovoljen uvjet zaustavljanja
    Ponavljaj za m je 1 do M_k
        Generiraj susjedno rješenje $\omega' \in N(\omega)$
        Izračunaj $\Delta_{\omega,\omega'} = f(\omega') - f(\omega)$
        Ako je $\Delta_{\omega,\omega'} \leq 0$, prihvati ω', tj. postavi $\omega \leftarrow \omega'$
        Inače ako je $\Delta_{\omega,\omega'} > 0$,
            postavi $\omega \leftarrow \omega'$ s vjerojatnošću $\exp(\frac{-\Delta_{\omega,\omega'}}{t_k})$
    Kraj ponavljanja
Kraj ponavljanja
Vrati ω kao rješenje

Vjerojatnost prihvaćanja lošijeg rješenja: vjerojatnost je skupa!

Klasično se koristi:

$$P(\Delta_{\omega,\omega'}) = \exp(\frac{-\Delta_{\omega,\omega'}}{t_k})$$

Izračun eksponencijalne funkcije računski je vrlo skup. Alternativa:

$$P(\Delta_{\omega,\omega'}) = \begin{cases} a_1 x^{k-1} & \text{ako je } \Delta_{\omega,\omega'} > 0, \\ 1 & \text{inače.} \end{cases}$$

Uočiti: vjerojatnost ne ovisi o iznosu pogoršanja – računski efikasnije.

Neki od planova hlađenja

Plan hlađenja definira način na koji se u sustavu mijenja temperatura. U praksi istraživan čitav niz izvedbi.

- Linearni

$$T_k = T_0 - k \cdot \beta$$

- Geometrijski

$$T_k = \alpha^k \cdot T_0$$

- vidi knjigu za druge...

Algoritam ograničenog demona

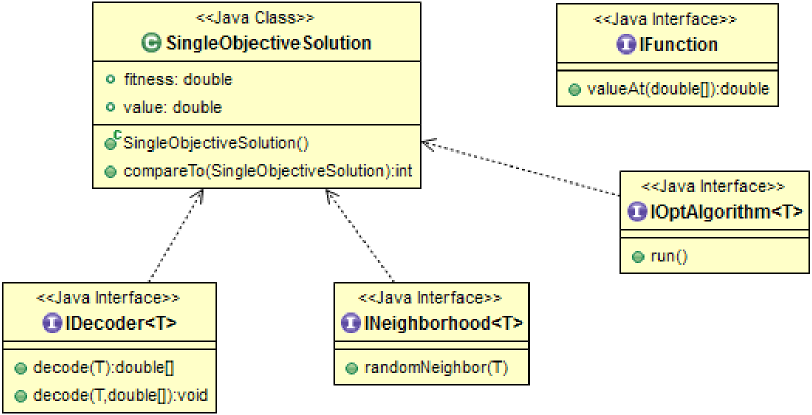
U sustavu postoji demon koji raspolaže određenom količinom energije. Promjena sustava u više energetsko stanje moguća je samo ako demon može isporučiti potrebnu energiju (bez koje time ostaje). Prelaskom u niže energetsko stanje energija se vraća demonu. Međutim, demon ne može imati više od unaprijed zadanog maksimuma.

1. Odaberi početno rješenje ω.
2. Odaberi početnu energiju demona $D = D_0 > 0$.
3. Ponavljaj dok nije zadovoljen uvjet zaustavljanja:
    1. Generiraj susjedno rješenje $\omega^{'} \in N(\omega)$.
    2. Izračunaj promjenu energije $\Delta E = f(\omega^{'}) - f(\omega)$.
    3. Ako je $\Delta E \leq D$ prihvati novo rješenje: $\omega \leftarrow \omega^{'}$ i korigiraj energiju demona $D \leftarrow D - \Delta E$.
    4. U suprotnom odbaci rješenje $\omega^{'}$.
    5. Ako je $D > D_0$, korigiraj $D \leftarrow D_0$.

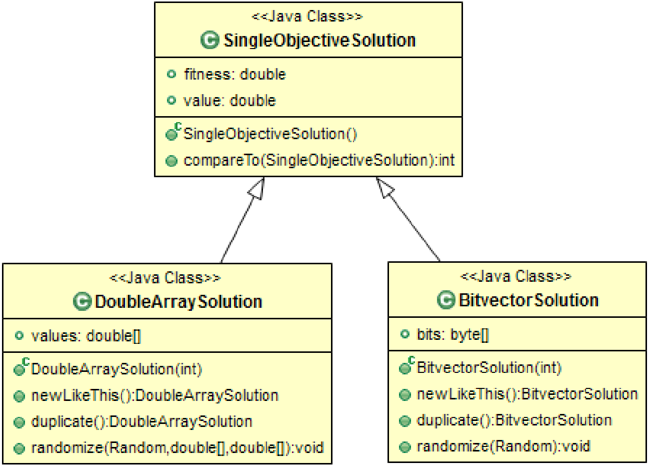
Algoritam kaljenog demona

1. Odaberi početno rješenje ω.
2. Odaberi početnu energiju demona $D = D_0 > 0$.
3. Ponavljaj dok nije zadovoljen uvjet zaustavljanja:
    1. Generiraj susjedno rješenje $\omega^{'} \in N(\omega)$.
    2. Izračunaj promjenu energije $\Delta E = f(\omega^{'}) - f(\omega)$.
    3. Ako je $\Delta E \leq D$ prihvati novo rješenje: $\omega \leftarrow \omega^{'}$ i korigiraj energiju demona $D \leftarrow D - \Delta E$.
    4. U suprotnom odbaci rješenje $\omega^{'}$.
    5. Ako je postignut ekvilibrij, umanji energiju demona prema planu; npr. $D \leftarrow \alpha \cdot D$.

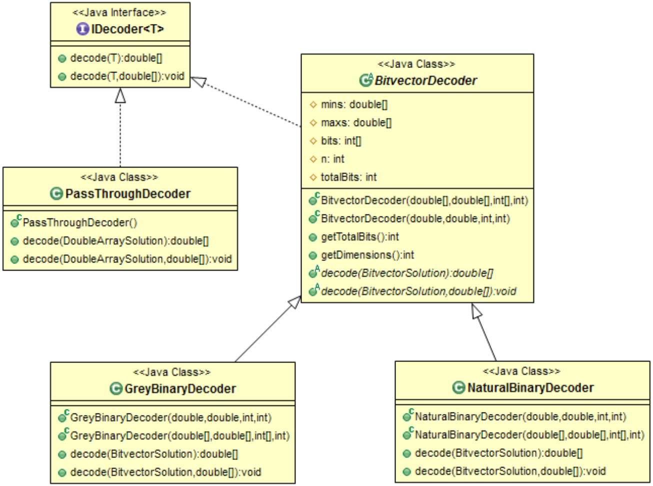
Primjer organizacije koda (1)



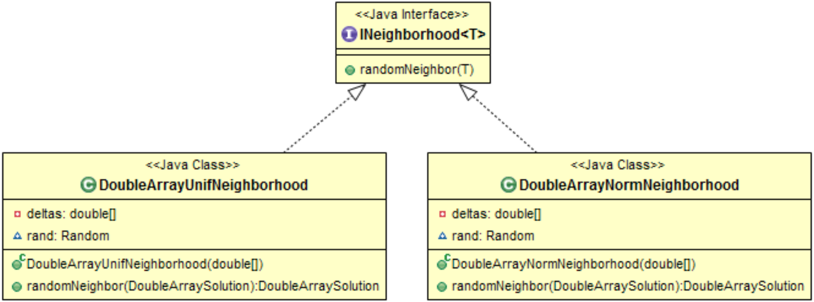
Primjer organizacije koda (2)



Primjer organizacije koda (3)



Primjer organizacije koda (4)



Primjer organizacije koda (5)