Calcul neuronal și evolutiv.

Lab 4: Optimizare combinatorială – problema comis voiajorului. Simulated annealing.

---

1. Problema comis-voiajorului (TSP - Travelling Salesman Problem)

Este una dintre cele mai cunoscute probleme de optimizare combinatorială putând fi enunțată după cum urmează: "se consideră o mulțime de n orașe și un comis-voiajor care trebuie să viziteze toate orașele, trecând o singură dată prin fiecare și să se întoarcă în orașul de pornire astfel încât costul total al circuitului să fie cât mai mic". Din punct de vedere formal, în varianta clasică problema este echivalentă cu cea a găsirii unui circuit Hamiltonian de cost minim într-un graf complet.

Problema este importantă atât din punct de vedere teoretic cât si din punct de vedere practic întrucât o serie de probleme concrete pot fi formulate ca TSP:

- Identificarea rutei optime pentru mijloace de transport (persoane, mărfuri)
- Generarea traseelor urmate de dispozitivele de producere a circuitelor integrate

Diferite aplicații pot fi găsite la [http://www.tsp.gatech.edu/apps/index.html]

Exista diferite variante ale problemei:

- *Varianta asimetrică*: costul trecerii de la un nod la altul depinde de sensul circuitului (Assymetric TSP)
- *Variante cu restricții de precedență*; restricțiile specifică faptul că un anumit nod trebuie vizitat înaintea altuia (Sequential Ordering Problem - SOP)
- *Variante cu mai multe mijloace de transport cu anumite capacități care asigură aprovizionarea* (Capacitated vehicle routing problem - CVRP)
- *Variante generalizate* în care există mai multe trasee directe între două noduri sau nodurile sunt înlocuite cu clustere de noduri (Generalized TSP)

Problema poate fi rezolvată exact (prin analiza tuturor circuitelor posibile) pentru valori mici ale numărului de orașe, însă cum numărul de circuite este (n-1)!/2, pentru valori mari ale lui n nu există metode exacte eficiente. Dealtfel, este cunoscut că TSP face parte din clasa problemelor NP-complete.

Pentru dimensiuni mari problema poate fi rezolvată utilizând metode euristice. Majoritatea metodelor euristice se bazează pe transformarea iterativă a unuia sau mai multor trasee inițiale.
Una dintre euristicile cel mai frecvent folosite este euristica Lin-Kernighan care se bazează pe ideea de a înlocui unele arce ale traseului cu altele în scopul reducerii costului traseului. Cazul cel mai simplu este cel în care se înlocuiesc două arce ceea ce echivalează cu schimbarea ordinii de parcurgere a unei porțiuni din traseu (transformarea 2-opt).

Exemplu: In cazul a 6 orașe: A,B,C,D,E,F dacă traseul inițial este (A,C,B,E,F,D) prin inlocuirea arcului (A,C) cu arcul (A,F), a arcului (F,D) cu arcul (C,D) și inversarea ordinii de parcurgere a orașelor B și E se obține traseul (A,F,E,B,C,D). Se observă că acest traseu se poate obține din cel inițial prin inversarea ordinii în secvența (C,B,E,F).

2. Rezolvarea problemei comis-voiajorului folosind "Simulated Annealing" (SA)

In rezolvarea unei probleme folosind metoda SA se parcurg următoarele etape:

- *Alegerea modului de codificare a soluțiilor.* In cazul problemei comis-voiajorului codificarea naturală este cea de tip permutare: un traseu pentru n orașe se specifică ca o permutare de ordin n. Această reprezentare asigură satisfacerea restricțiilor problemei (un oraș este vizitat exact o dată).

  Exemplu: Dacă orașele sunt numerotate astfel: 1-A, 2-B, 3-C, 4-D, 5-E, 6-F atunci traseul (A,C,B,E,F,D) corespunde permutării (1,3,2,5,6,4)

- *Alegerea mecanismului de căutare locală (construirea unei noi configurații pornind de la cea curentă).* Cea mai simplă variantă este cea bazată pe transformarea de tip 2-opt:
  - se aleg aleator doi indici i și j astfel încât 1<=i<j<=n
  - se inversează ordinea elementelor din permutare ai căror indici sunt cuprinși între i și j

  Exemplu: Dacă traseul curent corespunde permutării (1,3,2,5,6,4) iar i=2 și j=5 atunci permutarea obținută în urma transformării este: (1,6,5,2,3,4)

- *Alegerea probabilității de acceptare a unei noi configurații.* Probabilitatea de acceptare a configurației C' obținută din transformarea C se poate calcula folosind distribuția Boltzmann:

  $P(C'|C)=\min\{1,\exp(-(cost(C')-cost(C))/T(k)\}$

  unde cost(C) este costul configurației C iar T(k) este parametrul de control al procesului (temperatura).

- *Alegerea schemei de răcire.* Daca se notează cu T(k) temperatura corespunzătoare iterației k atunci valoarea corespunzătoare iterației următoare se poate stabili în una dintre variantele:
  - $T(k+1)=T(k)/log(k+1)$
  - $T(k+1)=T(k)/k$
  - $T(k+1)=a\ T(k)$, cu a o valoarea subunitara dar apropiată de 1 (de exemplu, a=0.99)

Observație. Valoarea inițială a temperaturii (T(0)) trebuie să fie suficient de mare pentru a permite trecerea între oricare două configurații.

- *Stabilirea criteriului de oprire.* Criteriul de oprire se poate referi la valoarea temperaturii (se stabilește o valoare minimă a temperaturii), la numărul de iterații parcurse (acest criteriu este corelat cu cel referitor la temperatură) sau la valoarea atinsă de funcția de optimizat (în cazul problemei comis-voiajorului este vorba de costul traseului).

Aplicație 2. Să se implementeze algoritmul Simulated Annealing pentru rezolvarea problemei comis-voiajorului. Pentru testare se pot folosi exemple de test de la http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/

Indicatie. O varianta de implementare este descrisă în SA_TSP.m în care coordonatele punctelor sunt selectate manual.

Un exemplu de apel este: SA_TSP(10000,0.001)

Exerciții:

1. Să se implementeze o funcție pentru preluarea din fișier a coordonatelor punctelor (în acest caz se transmite numele fișierului ca parametru în funcția SA_TSP)
2. Să se analizeze comportarea algoritmului pentru problemele specificate in fișierele: eil51.tsp, eil76.tsp, eil101.tsp
3. Să se testeze comportarea algoritmului pentru fiecare dintre schemele de răcire specificate mai sus
4. Să se modifice funcția de calcul a costului astfel încât distanțele dintre puncte să fie calculate o singură dată (indicație: se vor reține distanțele dintre oricare două puncte într-o matrice)

Temă. Implementați un algoritm de tip Simulated Annealing pentru a rezolva problema rucsacului (varianta discretă): fiind dat un rucsac de capacitate cunoscută și un set de obiecte pentru care se cunosc dimensiunile și valorile (suma dimensiunilor tuturor obiectelor depășește capacitatea rucsacului) să se selecteze un subset de obiecte a căror dimensiune totală nu depășește capacitatea rucsacului și pentru care valoarea cumulată este maximă.

Indicație. Se poate utiliza o reprezentare binară (pentru un set de n obiecte se utilizează un vector cu indicatori de prezență – pentru obiectul i valoarea indicatorului este 0 dacă obiectul nu e selectat respective valoarea 1 dacă obiectul este selectat). Generarea de noi configurații se poate baza pe complementarea unui număr mic (chiar una singură) de poziții ale vectorului binar. Pozițiile care se modifică se selectează uniform aleator.