Algoritmi metaeuristici.

Lab 5: Differential Evolution. Algoritmi coevolutivi.

---

1. Differential Evolution.

Este o tehnică de optimizare foarte populară care se bazează pe o regulă simplă de construire a unor noi soluții candidat ce implică utilizarea unor diferențe între elementele populației. Ideea fundamentală este că, la fiecare iterație, pentru fiecare element x(i) al populației se efectuează următoarele prelucrări:

- Se construiește un vector "mutant", y, prin combinarea unor elemente ale populației. Două dintre cele mai folosite variante de construire a unui mutant sunt:
  - DE/rand/1/bin: y=x(r1)+F*(x(r2)-x(r3)) unde r1,r2,r3 sunt indicii unor elemente selectate aleator din populație (cu restricția să fie diferite între ele)
  - DE/best/1/bin: y=x(ibest)+F*(x(r2)-x(r3)) unde x(ibest) este cel mai bun element din populație iar r1,r2 sunt indicii unor elemente selectate aleator din populație (cu restricția să fie diferite între ele)
  - Obs: in ambele cazuri F este un factor de scalare cu valori în (0,2).
- Se construiește o nouă soluție candidat, z, prin încrucișarea componentelor lui y cu cele ale elementului current x(i) după regula (cunoscută sub numele de încrucișare binomial):
  - z(j)=y(j) cu probabilitatea CR
  - z(j)=x(i,j) cu probabilitatea 1-CR
  - Obs: CR (cu valori în (0,1)) reprezintă cel de al doilea parametru al algoritmului

Aplicatie 1. Să se implementeze algoritmul DE (ambele variante) și să se compare performanța lui în raport cu cel al unei strategii evolutive simple (vezi lab 3).

Indicație: un exemplu de implementare este în fișierul DE.sci; comparația se poate efectua folosind functiile test utilizate la lab 3.

2. Algoritmi coevolutivi.

Eficiența metaeuristicilor bazate pe populații scade în cazul problemelor de dimensiuni mare (cu mai mult de 100 de variabile). O posibilitate de a asigura scalabilitatea acestor metode este de a utiliza tehnica coevoluției cooperative care se bazează pe ideea descompunerii problemei inițiale în mai multe subprobleme de dimensiune mai mică:

- Cele n variabile ale problemei se grupează în k<n componente, C1,C2,…,Ck; ideal ar fi ca variabilele care sunt corelate între ele să fie plasate în aceeași componentă; în cazul în care nu se cunosc interacțiunile dintre variabile alocarea variabilelor la componente se poate face aleator
- Pentru fiecare componentă se aplică un algoritm metaeuristic (se consideră că fiecare componentă definește o problemă de dimensiune mai mică); pentru a evalua fiecare componentă trebuie construită o soluție candidat completă (prin adăugarea variabilelor corespunzătoare celorlalte componente, care formează contextual de evaluare a componentei curente). Există mai multe moduri de a construi contextul:
  - Se consideră variabilele corespunzătoare celui mai bun element din populația corespunzătoare generației anterioare

- Se consideră variabilele corespunzătoare unui element aleator din populația corespunzătoare generației anterioare

Obs.

- Algoritmul corespunzător fiecărei componente poate fi executat pentru o singură iterație sau pentru mai multe iterații înainte de sincronizarea acestora.
- Algoritmii aplicați componentelor pot fi executați și în manieră asincronă, adică modificările corespunzătoare componentei sunt operate asupra la nivelul populației complete imediat dupa finalizarea iterațiilor aferente componentei (fără a se aștepta etapa de sincronizare).

Aplicatie 2. Modificați algoritmul pentru DE (de la aplicația 2) pentru a implementa varianta coevolutivă și testați-l pentru probleme de dimensiune 100, 500 și 1000.