Evolutív algoritmusok

1. **Hátizsák-feladat** **3 p.**
Van egy C össztérfogatú hátizsákunk, valamint N darab árucikk, egyenként p_j *értékkel* és w_j *térfogattal*. Feladatunk az, hogy töltsük meg a hátizsákot (térfogat-túllépés nélkül) úgy, hogy a *lehető legnagyobb* legyen a szállított árucikkek teljes értéke.

Útmutatás: tároljuk az árucikkeket egy bináris sztring-ben, ahol a „cipelt" árucikkhez kapcsolt indikátor értéke 1, máshol nulla. A genetikus operátorok ezen sztring-ek fölött lesznek definiálva. Ha az össz-térfogat nagyobb a megengedettnél, akkor az illető kromoszóma elhal.

Feladat:

(a) Vázoljunk egy genetikus algoritmust (dok) a következőkkel:
- a feladat reprezentációja;
- keresztezés;
- mutáció;
- szelekció.

(b) Implementáljuk a feladatot. A program egy file-ból olvassa be a bemeneteket, ahol az első sorban a hátizsák térfogata, a többi sorokban az árucikkek jellemzői vannak tárolva, az árucikk értéke, valamint a térfogata.

(c) Elemezzük az algoritmust (dok): különböző méretű feladatokra számítsuk ki az egzakt optimumot, hasonlítsuk össze a két megoldás (1) futási idejét (2) optimumait.

2. **Utazóügynök feladata** **3 p.**
Egy ügynök N várost kell, hogy bejárjon, mindegyiket egyszer érintve, úgy, hogy a *teljes útvonal* hossza a legkisebb legyen. A feladatunk, hogy az N város koordinátáinak az ismeretében határozzuk meg a városok bejárási *sorrendjét*.

Útmutatás: az útvonalat egy *permutációban* tároljuk, melyhez hozzárendeljük az illető sorrendhez tartozó *teljes* hosszt.

Feladat:

(a) Vázoljunk egy genetikus algoritmust (dokumentáció) a következőkkel:
- a feladat reprezentációja;
- keresztezés;
- mutáció;
- szelekció.

(b) Implementáljuk a feladatot. A program egy file-ból olvassa be a bemeneteket, ahol sorok szerint a városok síkbeli koordinátáit adjuk meg.

(c) Elemezzük az algoritmust (dokumentáció): különböző véletlen konfigurációra számítsuk ki az egzakt optimumot, majd hasonlítsuk össze a két megoldás (1) futási idejét (2) optimumait.

3. *Illesztés* http://en.wikipedia.org/wiki/Multiple_sequence_alignment **8 pt.**
   Egy karaktersorozat N klónja véletlenszerűen fel van darabolva és egy zsákba helyezve. Feladatunk, hogy a fentiek ismeretében állítsuk helyre az eredeti karaktersort.
   Például: legyen az ismeretlen karaktersorozat `s='mikormerremegy'` és legyen *két példányunk* a karaktersorból. A zsákban a következő sorozatok vannak:
   S_{ossz}=`{'kormer','mi','mikor','megy','merre','remegy'}`
   Célunk a sztring-ek egy sorrendjének a meghatározása úgy, hogy az összefűzés eredménye a legkisebb sztring-et eredményezze, ahol a sztring-eket egymásba lehet fűzni, például a `'mikor' + 'mi' = 'mikor'`.

   (a) Vázoljunk egy genetikus algoritmust (dokumentáció) a következőkkel:
   - a feladat reprezentációja;
   - keresztezés;
   - mutáció;
   - szelekció.

   (b) Implementáljuk a feladatot. A program olvassa be a `seq_sm.txt` file-ból a karaktersorozatokat és állítsa fel egy sorrendjét az 589 sztring-nek úgy, hogy az össz-hossz minimális legyen. Tudjuk, hogy *három* példányunk van a sztring-ből.