**Descrierea problemei**

Problema abordată este **Knapsack Problem**, unde avem:

* **Număr de obiecte (nnn)**: 20.
* Fiecare obiect are asociată o **greutate** și o **valoare**, iar scopul este să maximizăm valoarea obiectelor selectate fără a depăși capacitatea rucsacului (CCC).
* Capacitatea rucsacului (CCC) și valorile obiectelor au fost variate pentru a analiza performanța algoritmului în diferite scenarii.

**Exemple de setări pentru problema rucsacului**:

1. C=30C = 30C=30, un rucsac mic cu constrângeri stricte.
2. C=50C = 50C=50, un rucsac mediu.
3. C=70C = 70C=70, un rucsac mai mare, care permite mai multe soluții fezabile.

**Rezolvarea problemei**

* **Limbaj de programare**: Python.
* **Implementare**:
  + Algoritmul genetic a fost scris **de la zero**, incluzând funcții pentru:
    - **Inițializarea populației**.
    - **Calcularea funcției de fitness**.
    - **Crossover** și **mutație** pentru generarea de soluții noi.
    - **Selecție cu elitism**, păstrând cele mai bune soluții din fiecare generație.

### ****Metodologia experimentelor****

1. **Setarea parametrilor**:
   * Numărul de obiecte, capacitatea rucsacului, populația, generațiile, rata de elitism și rata de mutație.
2. **Executarea algoritmului**:
   * Algoritmul a fost rulat cu diferite combinații de parametri pentru a observa impactul acestora asupra soluției optime.
3. **Înregistrarea rezultatelor**:
   * S-au notat fitness-ul maxim, soluția (chromosome-ul), generația unde a fost găsită soluția optimă sau cea mai bună și timpul de execuție.

### ****Rezultatele experimentale****

* **Fitness maxim**:
  + Depinde de **capacitatea rucsacului** și de parametrii algoritmului.
  + Exemple:
    - C=30C = 30C=30: Fitness maxim = 500.
    - C=50C = 50C=50: Fitness maxim = 882.
    - C=70C = 70C=70: Fitness maxim = 1140.
* **Generații**:
  + În general, cele mai bune soluții sunt găsite după 30-100 generații pentru configurații standard.
  + Creșterea numărului de generații la 1000 a permis îmbunătățirea fitness-ului în scenarii constrânse (C=30C = 30C=30).

**Variații și impactul parametrilor**

1. **Elitism**:
   * **Rezultate**:
     + Elitism 0%0\%0%: Fitness = 882 (găsit rapid, generația 6).
     + Elitism 10%10\%10%: Fitness = 882 (generația 66).
     + Elitism 50%50\%50%: Fitness = 862 (generația 37).
   * **Concluzii**:
     + **Fără elitism**: Mai multă diversitate, soluții bune găsite rapid, dar uneori instabil.
     + **Elitism moderat (10%)**: Echilibru între diversitate și exploatare, convergență stabilă.
     + **Elitism ridicat (50%)**: Exploatare agresivă, dar cu riscul stagnării.
2. **Mutație**:
   * **Rezultate**:
     + Mutație 1%1\%1%: Fitness = 576 (generația 12, convergență rapidă dar soluție slabă).
     + Mutație 5%5\%5%: Fitness = 882 (echilibru între explorare și exploatare).
     + Mutație 20%20\%20%: Fitness = 824 (mai multă explorare, dar convergență mai lentă).
   * **Concluzii**:
     + Rata de mutație optimă este între 5-10%.
     + Rata prea mică (1%1\%1%) reduce explorarea și poate duce la stagnare.
     + Rata prea mare (20%20\%20%) introduce prea multă aleatorie, încetinind convergența.
3. **Dimensiunea populației**:
   * **Rezultate**:
     + Populație 25: Fitness = 868 (generația 46, convergență mai rapidă, dar diversitate redusă).
     + Populație 50: Fitness = 882 (mai multă diversitate).
     + Populație 100: Fitness = 882 (generația 83, timp de execuție crescut).
     + Populație 10,000: Fitness = 500 (convergență rapidă, dar limitată de C=30C = 30C=30).
   * **Concluzii**:
     + Dimensiuni mai mari ale populației cresc diversitatea, dar implică un cost computațional mai mare.
4. **Capacitatea rucsacului (CCC)**:
   * **Rezultate**:
     + C=30C = 30C=30: Fitness = 500 (soluții strict constrânse).
     + C=50C = 50C=50: Fitness = 882 (mai multe soluții fezabile).
     + C=70C = 70C=70: Fitness = 1140 (soluții mai puțin constrânse).
   * **Concluzii**:
     + Capacitatea influențează direct fitness-ul maxim posibil.

**5. Concluzii și observații generale**

1. **Cele mai bune rezultate** s-au obținut cu:
   * Populație 50−10050-10050−100.
   * Rata de mutație 5−10%5-10\%5−10%.
   * Rata de elitism 10%10\%10%.
   * Generații: 100−200100-200100−200.
   * Capacitate CCC: Mai mare permite soluții mai bune, dar depinde de problemă.
2. **Impactul parametrilor**:
   * Rata de elitism mică și mutație moderată oferă un echilibru bun.
   * Populații mai mari cresc diversitatea și stabilitatea, dar implică un timp mai mare de execuție.

Number of Items: 20

Elitism Rate: 10% (0.1)

Mutation Rate: 5% (0.05)

Population Size: 50

Generations: 100

Knapsack Capacity: 50

Best fitness: 882

Best chromosome: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1]

Found in generation: 66

Number of Items: 20

Elitism Rate: 0% (0.0)

Mutation Rate: 5% (0.05)

Population Size: 50

Generations: 100

Knapsack Capacity: 50

Best fitness: 882

Best chromosome: [0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1]

Found in generation: 6

Number of Items: 20

Elitism Rate: 50% (0.5)

Mutation Rate: 5% (0.05)

Population Size: 50

Generations: 100

Knapsack Capacity: 50

Best fitness: 862

Best chromosome: [1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1]

Found in generation: 37

Number of Items: 20

Elitism Rate: 10% (0.1)

Mutation Rate: 1% (0.01)

Population Size: 50

Generations: 100

Knapsack Capacity: 50

Best fitness: 576

Best chromosome: [1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0]

Found in generation: 12

Number of Items: 20

Elitism Rate: 10% (0.1)

Mutation Rate: 20% (0.2)

Population Size: 50

Generations: 100

Knapsack Capacity: 50

Best fitness: 824

Best chromosome: [0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1]

Found in generation: 50

Number of Items: 20

Elitism Rate: 10% (0.1)

Mutation Rate: 10% (0.1)

Population Size: 25

Generations: 100

Knapsack Capacity: 50

Best fitness: 868

Best chromosome: [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1]

Found in generation: 46

Number of Items: 20

Elitism Rate: 10% (0.1)

Mutation Rate: 10% (0.1)

Population Size: 100

Generations: 100

Knapsack Capacity: 50

Best fitness: 882

Best chromosome: [0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1]

Found in generation: 83

Number of Items: 20

Elitism Rate: 10% (0.1)

Mutation Rate: 10% (0.1)

Population Size: 50

Generations: 50

Knapsack Capacity: 50

Best fitness: 862

Best chromosome: [1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1]

Found in generation: 39

Number of Items: 20

Elitism Rate: 10% (0.1)

Mutation Rate: 10% (0.1)

Population Size: 50

Generations: 200

Knapsack Capacity: 50

Best fitness: 882

Best chromosome: [0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1]

Found in generation: 137

Number of Items: 20

Elitism Rate: 10% (0.1)

Mutation Rate: 10% (0.1)

Population Size: 50

Generations: 100

Knapsack Capacity:30

Best fitness: 468

Best chromosome: [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0]

Found in generation: 34

Number of Items: 20

Elitism Rate: 10% (0.1)

Mutation Rate: 10% (0.1)

Population Size: 50

Generations: 100

Knapsack Capacity:70

Best fitness: 1140

Best chromosome: [1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1]

Found in generation: 16

Number of Items: 20

Elitism Rate: 10% (0.1)

Mutation Rate: 10% (0.1)

Population Size: 10000

Generations: 100

Knapsack Capacity:30

Optimal solution not found. Best fitness: 500

Best chromosome: [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1]

Found in generation: 5

Number of Items: 20

Elitism Rate: 10% (0.1)

Mutation Rate: 10% (0.1)

Population Size: 50

Generations: 1000

Knapsack Capacity:30

Best fitness: 500

Best chromosome: [0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Found in generation: 210