**Reprezentacja:**

Do implementacji tego zadania postanowiłem użyć następującej reprezentacji: każde zadanie ma określone miejsce w kolejce i każde zadanie ma przydzielony zasób. Możemy osobno zmieniać zarówno kolejność zadań jak i przydział zasobu dla każdego zadania.

**Inicjalizacja:**

Inicjalizuje populacje przez generacje określonej ilości harmonogramów z losową kolejnością zadań i przypisanymi losowymi, ale posiadającymi odpowiednie umiejętności zasobami.

**Ocena:**

Jako funkcje oceny używam całkowitego czasu wykonania harmonogramu. W przypadku gdy poprzednik danego zadania jest w kolejce po nim przechodzimy do niego i dopiero wracamy do wykonywanego zadania.

**Selekcja:**

Użyłem selekcji turniejowej – wybieramy z populacji określoną liczbę osobników i wybieramy z nich najbardziej przystosowany, powtarzając ten algorytm aż do momentu osiągnięcia odpowiedniej ilości wybranych osobników.

**Mutacja:**

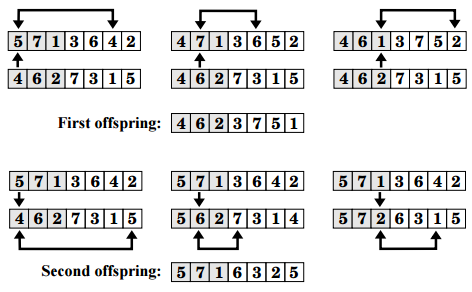
Zaimplementowane są dwie mutacje:

1. Dla mutacji kolejności wykonywania zadań zamieniam miejscami dwa zadania w kolejce
2. Dla mutacji przypisanych zasobów zamieniam przypisany zasób dla danego zadania na inny pasujący

**Krzyżowanie:**

Zaimplementowane są dwa krzyżowania:

1. Dla krzyżowania kolejności wykonywania zadań użyłem jednopunktowego PMX, która działa podobnie jak normalne krzyżowanie – w losowym miejscu dokonujemy krzyżowania, jednak wstawiając nowe zadania z innego osobnika, zamieniamy stare miejscami, aby każde zadanie występowało w kolejce tylko raz, najlepiej ilustruje to ten obrazek:



1. Dla krzyżowania przypisanych zasobów wykorzystałem proste krzyżowanie jednopunktowe

**Sąsiedztwo:**

Generujemy sąsiadów dla kolejności poprzez zamianę dwóch sąsiednich par (cyklicznie, dlatego możemy zamienić także pierwsza z ostatnią) lub dla zasobów poprzez zamianę jednego zasobu przydzielonego dla jednego zadania na sąsiedni w liście dozwolonych zasobów dla tego zadania.

**Taboo:**

W liście taboo przechowuje poprzednie genotypy rozwiązań.

**Simulated annealing:**

Jako temperaturę początkową przyjmuję różnicę pomiędzy maksymalnym a minimalnym czasem rozwiązania przemnożonym przez pewien mnożnik. Jako podstawę funkcji schładzania przyjmuje prostą funkcję geometryczną: , gdzie . Dodatkowo dla końcowej funkcji modyfikuje ją w zależności od różnicy funkcji oceny obecnego od najlepszego rozwiązania .

**Użyte parametry:**

* GA:
  + Wielkość populacji – 100
  + Ilość iteracji – 1 tys.
  + Wielkość turnieju – 10
  + Szansa na krzyżowania – 0.9
  + Szansa na mutacje – 0.01
* TS:
  + Wielkość populacji – 1
  + Ilość iteracji – 10 tys.
  + Wielkość sąsiedztwa – 10
  + Wielkość listy taboo – 100
* SA:
  + Wielkość populacji – 1
  + Ilość iteracji – 100 tys.
  + Wielkość sąsiedztwa – 1
  + Mnożnik zmniejszania temperatury – 0.99
  + Mnożnik temperatury początkowej – 1
* Każdy algorytm został uruchomiony 50 razy
* Ilość urodzeń wyniosła 100 tys.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | EA | | | TS | | | SA | | |
|  | Min | Avg | Std | Min | Avg | Std | Min | Avg | Std |
| 10\_3\_5\_3 | 93 | 93,6 | 0,966091783 | 93 | 93,3 | 0,483045892 | 93 | 93,5 | 0,527046277 |
| 10\_5\_8\_5 | 80 | 80 | 0 | 80 | 80 | 0 | 80 | 80 | 0 |
| 10\_7\_10\_7 | 104 | 104 | 0 | 104 | 104 | 0 | 104 | 104 | 0 |
| 15\_3\_5\_3 | 230 | 230 | 0 | 230 | 230 | 0 | 230 | 230 | 0 |
| 15\_6\_10\_6 | 102 | 102 | 0 | 102 | 102 | 0 | 102 | 102 | 0 |
| 15\_9\_12\_9 | 90 | 90 | 0 | 90 | 90 | 0 | 90 | 90 | 0 |
| 100\_5\_22\_15 | 484 | 486,4 | 1,173787791 | 486 | 487,8 | 1,398411798 | 486 | 487,7 | 1,159501809 |
| 100\_20\_46\_15 | 161 | 162,4 | 2,458545189 | 161 | 166,2 | 5,452828012 | 161 | 165,9 | 3,842742077 |
| 100\_20\_47\_9 | 125 | 130,3 | 2,311805451 | 134 | 137,1 | 3,414023368 | 125 | 129,9 | 2,846049894 |
| 200\_10\_50\_9 | 486 | 487,4 | 0,966091783 | 487 | 489,5 | 2,368778401 | 486 | 486,8 | 0,421637021 |
| 200\_20\_54\_15 | 259 | 262,9 | 1,595131482 | 264 | 270,1 | 6,118278625 | 260 | 261,3 | 1,636391694 |
| 200\_20\_97\_15 | 336 | 336 | 0 | 336 | 339,5 | 10,05816418 | 336 | 336 | 0 |
| Suma | 2550 | 2565 | 9,471453479 | 2567 | 2589,5 | 29,29353027 | 2553 | 2567,1 | 10,43336877 |