Rozwiązanie zaczyna się od zera, bez przerw pomiędzy kolejnymi.

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. Przykład:

Zadanie	1	2	3	4	5	6	7	8	9
Kolejność	7	4	1	5	6	8	3	2	9
Zasób	1	2	3	3	3	2	1	1	2

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.

Warunek stopu:

Warunkiem stopu jest osiągnięta liczba iteracji.

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:

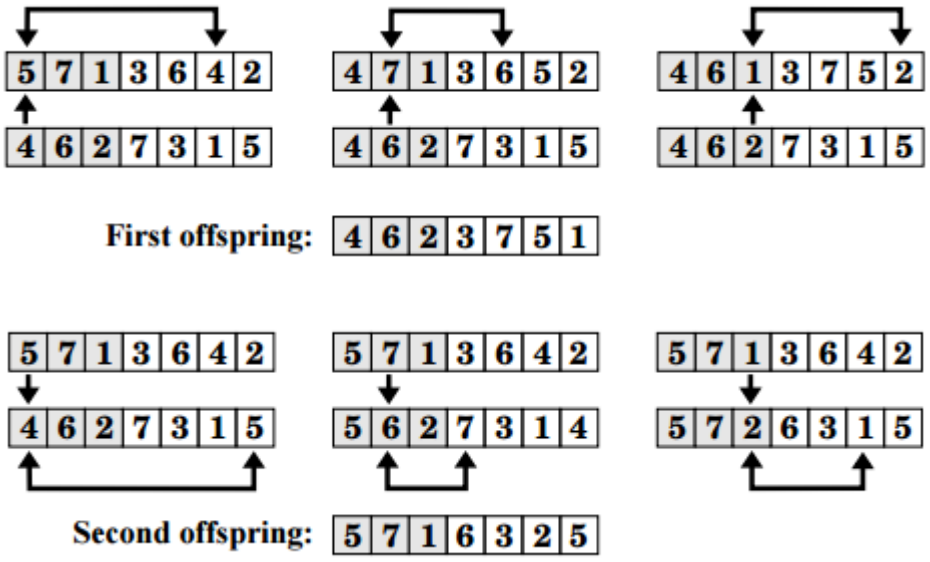
Zaimplementowane są dwie mutacje:

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:



2. 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. Parametrem jest także wielkość tej listy taboo.

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: $t_{n+1} = t_n * x$, gdzie $0 < x < 1$. Dodatkowo dla końcowej funkcji modyfikuje ją w zależności od różnicy funkcji oceny obecnego od najlepszego rozwiązania $t_k = t_n * (1 + \frac{f_{curr} - f_{min}}{f_{curr}})$.

[A Comparison of Cooling Schedules for Simulated Annealing, José Fernando Díaz Martín (University of Deusto, Spain) and Jesús M. Riaño Sierra (University of Deusto, Spain) Source Title: Encyclopedia of Artificial Intelligence, pages 344-352, Copyright: © 2009 |Pages: 9, DOI: 10.4018/978-1-59904-849-9.ch053]

http://what-when-how.com/artificial-intelligence/a-comparison-of-cooling-schedules-for-simulated-annealing-artificial-intelligence/

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. dla każdego uruchomienia

	EA			TS			SA		
	Min	Avg	Std	Min	Avg	Std	Min	Avg	Std
10_3_5_3	94	94,6	0,9660917	94	94,3	0,48304589	94	94,5	0,52704628
10_5_8_5	81	81	0	81	81	0	81	81	0
10_7_10_7	105	105	0	105	105	0	105	105	0
15_3_5_3	231	231	0	231	231	0	231	231	0
15_6_10_6	103	103	0	103	103	0	103	103	0
15_9_12_9	91	91	0	91	91	0	91	91	0
100_5_22_15	485	487,4	1,1737877	487	488,8	1,3984118	487	488,7	1,15950181
100_20_46_15	162	163,4	2,4585451	162	167,2	5,45282801	162	166,9	3,84274208
100_20_47_9	126	131,3	2,3118054	135	138,1	3,41402337	126	130,9	2,84604989
200_10_50_9	487	488,4	0,9660917	488	490,5	2,3687784	487	487,8	0,42163702
200_20_54_15	260	263,9	1,5951314	265	271,1	6,11827863	261	262,3	1,63639169
200_20_97_15	337	337	0	337	340,5	10,0581642	337	337	0
Suma	2562	2577	9,4714534	2579	2601,5	29,2935303	2565	2579,1	10,4333688
Średnia	213,5	214,75	0,7892877	214,916667	216,791667	2,44112752	213,75	214,925	0,8694474