Dana jest pojedyncza maszyna/procesor oraz zbiór *J*={1,…,*j*,…*n*} *n* niepodzielnych zadań. Każde z zadań j charakteryzuje się czasem wykonywania *pj*, terminem dostępności *rj* oraz czasem stygnięcia *qj*. Należy tak uszeregować zadania, aby czas zakończenia wykonywania wszystkich zadań był jak najkrótszy (minimalizujemy tzw. długość uszeregowania – makespan).

Rozwiązanie może być zapisane jednoznacznie za pomocą permutacji π={π (1), π (2),…, π (j),…, π (n)}, gdzie π (j) oznacza zadanie uszeregowane na pozycji j w permutacji π.

Czas wykonywania zadania π (j) to pπ (j)

Termin dostępności zadania π (j) to rπ (j)

Czas stygnięcia zadania π (j) to qπ (j)

Bez czasów stygnięcia:

Czas zakończenia wykonywania zadania π (1) to Cπ (1)=rπ (1)+pπ (1)

Ale czas zakończenia wykonywania kolejnego zadania to:

Cπ (2)=max{rπ (2), Cπ (1)}+pπ (2)

Dla zadania π (j) mamy Cπ (j)=max{rπ (j), Cπ (j-1)}+pπ (j)

Ale co z ogonkami (czasami stygnięcia)???

Możemy zrobić sobie dodatkową tablicę, albo zmienną, gdzie przechowujemy najdłuższy do tej pory pełen czas wykonywania zadania (ze stygnięciem), gdzie ten pełen czas wykonywania danego zadania to:

Dla pierwszego: π (1) to Cπ (1)q=rπ (1)+pπ (1)+qπ (1)

A w ogólności

Cπ (j)q=max{r{π (j), Cπ (j-1)}+pπ (j)+ qπ (j)

I musimy przechowywać najdłuższy C\_{π (j)}q, jaki do tej pory znaleźliśmy – to nasze kryterium