# Explicație completă și rezolvare manuală a algoritmului FAJSP (cu DEAP)

## 1. Prezentarea generală a problemei FAJSP

FAJSP (Flexible Assembly Job Shop Scheduling Problem) este o extensie a problemei clasice de Job Shop Scheduling, în care fiecare operație poate fi realizată de mai multe mașini alternative, iar între joburi pot exista constrângeri de precedență (ex: jobul 3 poate începe doar după finalizarea jobului 1).  
  
Scopul este să planificăm toate operațiile astfel încât să minimizăm timpul total de producție (makespan), să reducem încărcarea totală a mașinilor (load) și timpul pierdut (idle time).

## 2. Descrierea funcțiilor principale

### 2.1 Funcția parse\_fjs\_file

Această funcție primește un fișier .fjs și extrage:  
- numărul de joburi și mașini  
- lista operațiilor pentru fiecare job  
- timpul de procesare pentru fiecare operație pe fiecare mașină posibilă  
- eventualele constrângeri de precedență între joburi  
Ea returnează o structură de date ce va fi folosită de restul algoritmului.

### 2.2 Funcția create\_individual

Generează un individ format din:  
- un vector cu selecția aleatorie a mașinilor (MS)  
- o permutare aleatoare a operațiilor (OS)  
Acesta reprezintă o soluție potențială.

### 2.3 Funcția evaluate

Simulează executarea operațiilor conform individului dat:  
- respectă ordinea din OS și alocarea din MS  
- ține cont de constrângerile între operații și de disponibilitatea mașinilor  
- returnează 3 valori: makespan, total load, idle time.

### 2.4 Funcțiile de căutare locală/globală/hibridă

Modifică individul în scopul îmbunătățirii lui:  
- globală: permută 2 operații oriunde  
- locală: inversează o mică secvență de operații  
- hibridă: aplică mutație și apoi repară ordinea cu sortare topologică

### 2.5 Funcția run\_deap

Este funcția principală care:  
- inițializează populația  
- aplică selecție, crossover, mutație  
- aplică căutare locală adaptivă pe baza unui agent Q-learning  
- evoluează soluțiile timp de mai multe generații

# Exemplu complet – FAJSP cu DEAP (codul TA-MA)

## 1. Joburi și Constrângeri

J1 – 2 operații  
J2 – 2 operații  
J3 – 2 operații  
J4 – 2 operații  
J5 – 2 operații  
J6 – 2 operații  
J7 – 1 operație  
J8 – 1 operație  
  
Constrângeri între joburi:  
J6 ← J1, J2  
J7 ← J3, J4  
J8 ← J5, J6, J7

## 2. Niveluri de constrângere

J1 → nivel 1  
J2 → nivel 1  
J3 → nivel 1  
J4 → nivel 1  
J5 → nivel 1  
J6 → nivel 2  
J7 → nivel 2  
J8 → nivel 3

## 3. Individ selectat

MS (alocare mașini): [1, 3, 2, 1, 3, 1, 1, 2, 1, 3, 1, 3, 1, 3, 1, 3]  
→ fiecare poziție corespunde unei operații (indice global 0-15)  
  
OS (ordinea operațiilor): [6, 1, 3, 2, 4, 5, 0, 7, 8, 9, 10, 11, 12, 13, 14, 15]  
→ ordine topologică validă conform constrângerilor

## 4. Decodare pas cu pas

J6 - Op1 on M1: start 0, end 6  
J1 - Op1 on M3: start 0, end 9  
J2 - Op1 on M1: start 0, end 15  
J2 - Op2 on M2: start 15, end 35  
J3 - Op1 on M3: start 9, end 29  
J3 - Op2 on M1: start 29, end 44  
J6 - Op2 on M2: start 35, end 42  
J4 - Op1 on M2: start 0, end 24  
J5 - Op1 on M1: start 0, end 5  
J5 - Op2 on M1: start 5, end 12  
J7 - Op1 on M1: start 44, end 50  
J1 - Op2 on M3: start 9, end 18  
J4 - Op2 on M3: start 29, end 35  
J8 - Op1 on M3: start 50, end 58

## 5. Makespan final

58