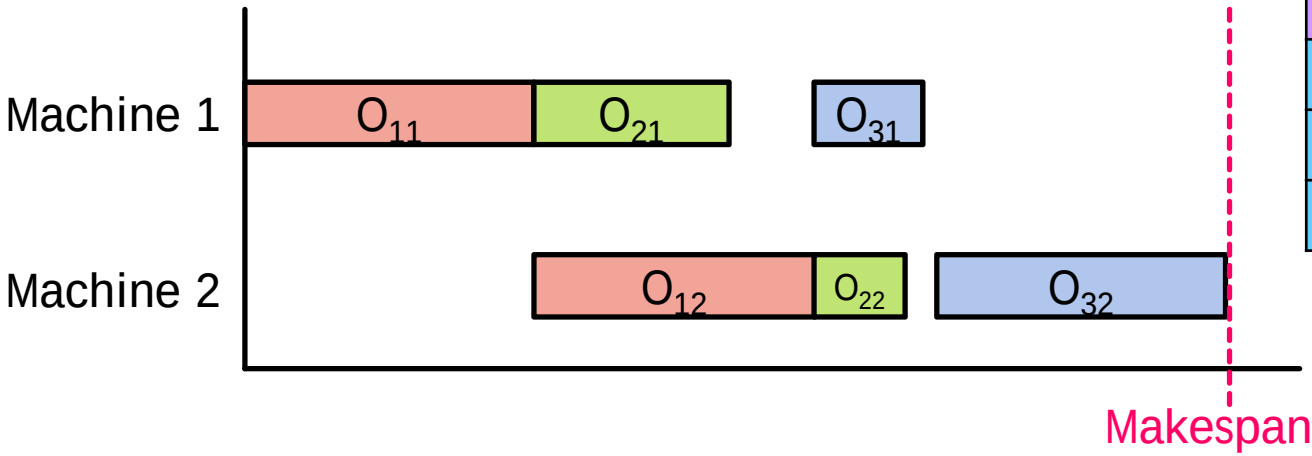
# Job Shop Scheduling

# Job shop vs. Flow shop scheduling

- Flow shop scheduling

所有job的加工機台順序**一樣**

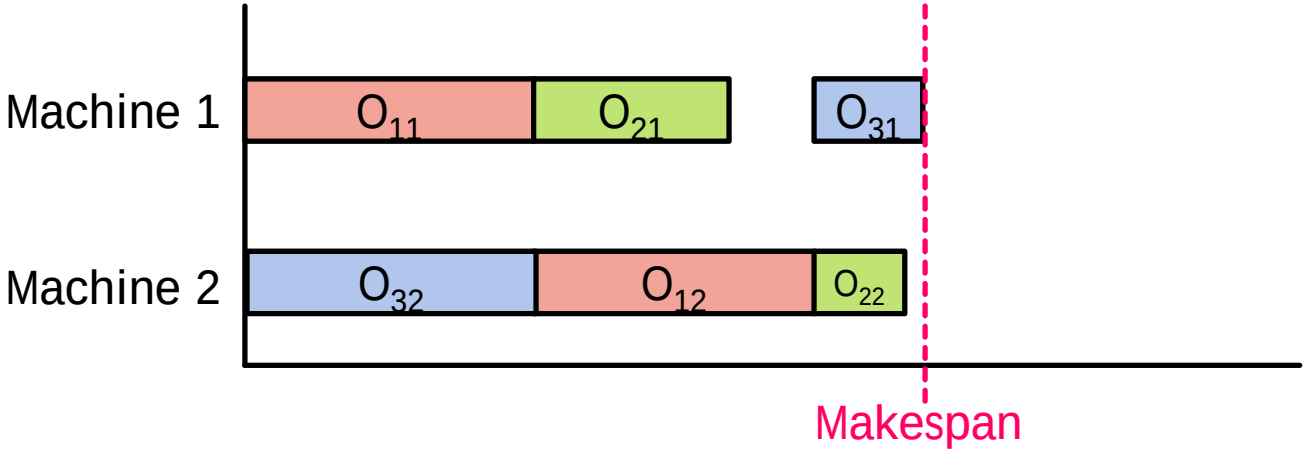
| | 加工1 | 加工2 |
|---|---|---|
| Job 1 | 機台1 | 機台2 |
| Job 2 | 機台1 | 機台2 |
| Job 3 | 機台1 | 機台2 |




- Job shop scheduling

job的加工機台順序**可不一樣**
**(但是是給定的)**

| | 加工1 | 加工2 |
|---|---|---|
| Job 1 | 機台1 | 機台2 |
| Job 2 | 機台1 | 機台2 |
| Job 3 | 機台2 | 機台1 |



# Different measures for performance

- Minimize the makespan (firm-oriented)
  - Find completion time $C_i$ of each job to minimize $\max_i \{C_i\}$
- Minimize the average flow time (customer-oriented)
  - $S_i$ = starting of the first operation of job $i \in \{1, \ldots, n\}$
  - Find $C_i$ of each job to minimize $(C_i - S_i)/n$
- Minimize the maximal tardiness
  - Given due date $d_i$ of each job $J_i$
  - Find $C_i$ of each job to minimize $\max_i \{C_i - d_i, 0\}$

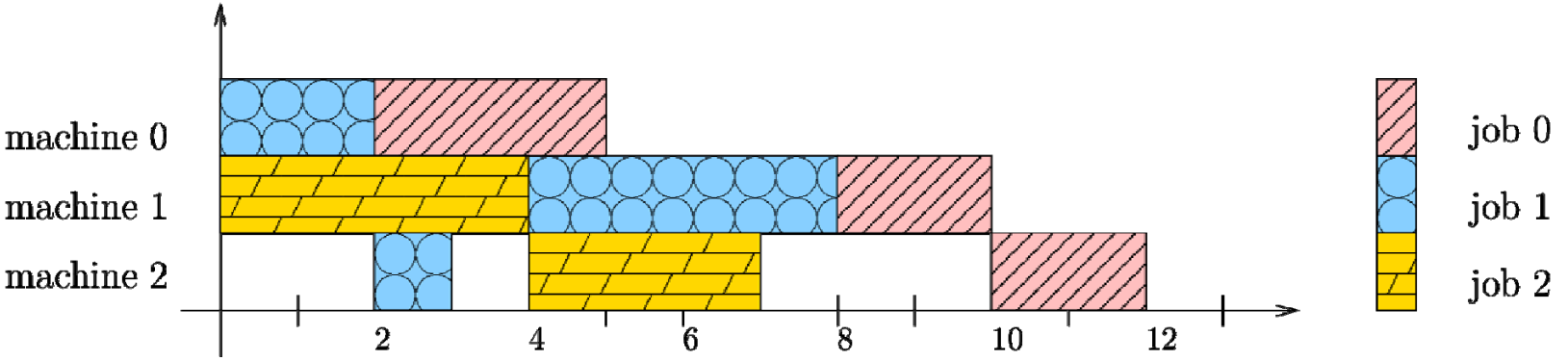
# 解的編碼

- 問題設定
  - job 0 = [(0, 3), (1, 2), (2, 2)]
  - job 1 = [(0, 2), (2, 1), (1, 4)]
  - job 2 = [(1, 4), (2, 3)]

| 加工順序 | 加工0 | 加工1 | 加工2 |
|---|---|---|---|
| Job 0 | 機台0 | 機台1 | 機台2 |
| Job 1 | 機台0 | 機台2 | 機台1 |
| Job 2 | 機台1 | 機台2 | |

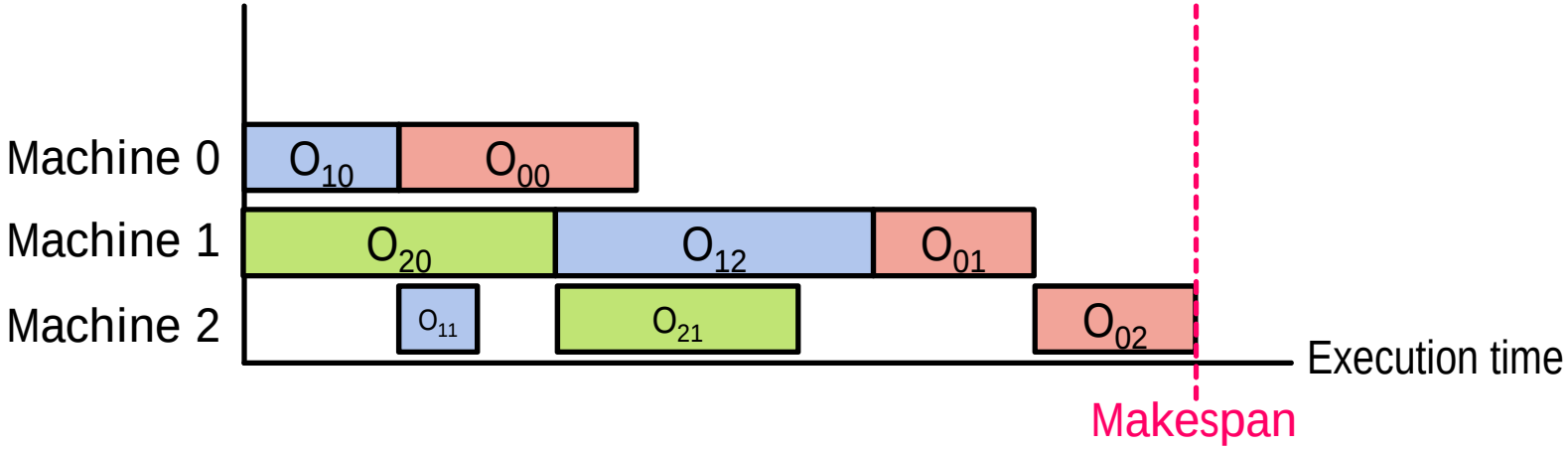
| 加工時間 | 加工0 | 加工1 | 加工2 |
|---|---|---|---|
| Job 0 | 3 | 2 | 2 |
| Job 1 | 2 | 1 | 4 |
| Job 2 | 4 | 3 | |

- 解的編碼
  - 00011122的隨機排列
    - ✓ 3個0分別代表job 0的3個加工機台
    - ✓ 3個1分別代表job 1的3個加工機台
    - ✓ 2個2分別代表job 2的2個加工機台
  - E.g., 12101200的答案如下：

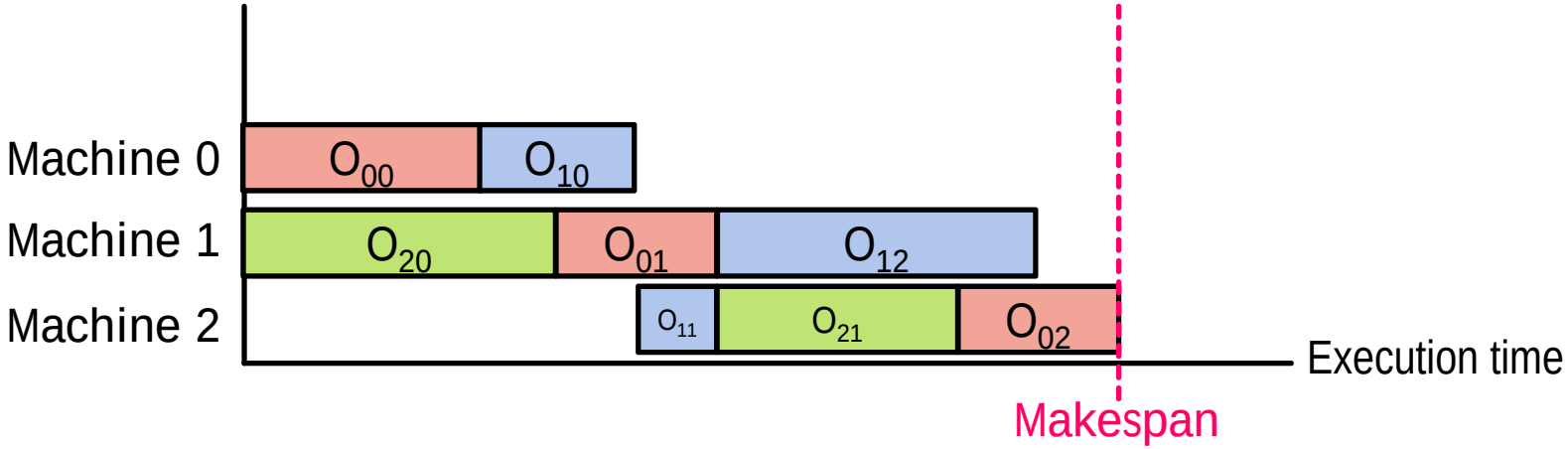


# Comparison of 2 JSP solutions

- 12101200 → $O_{10}$, $O_{20}$, $O_{11}$, $O_{01}$, $O_{12}$, $O_{21}$, $O_{01}$, $O_{02}$




- 20011210 → $O_{20}$, $O_{00}$, $O_{01}$, $O_{10}$, $O_{11}$, $O_{21}$, $O_{12}$, $O_{03}$




Better

# Python code for solution representation

- 解的編碼
  - 00011122的隨機排列

```
65 def initPop():                    # 初始化群體
66     p = []
67
68     # === 編碼 000111222 的排列  ===
69     for i in range(NUM_CHROME) :
70         a = []
71         for j in range(NUM_JOB):
72             for k in range(NUM_MACHINE):
73                 a.append(j)
74         np.random.shuffle(a)
75
76         p.append(a)
77
78     return p
```

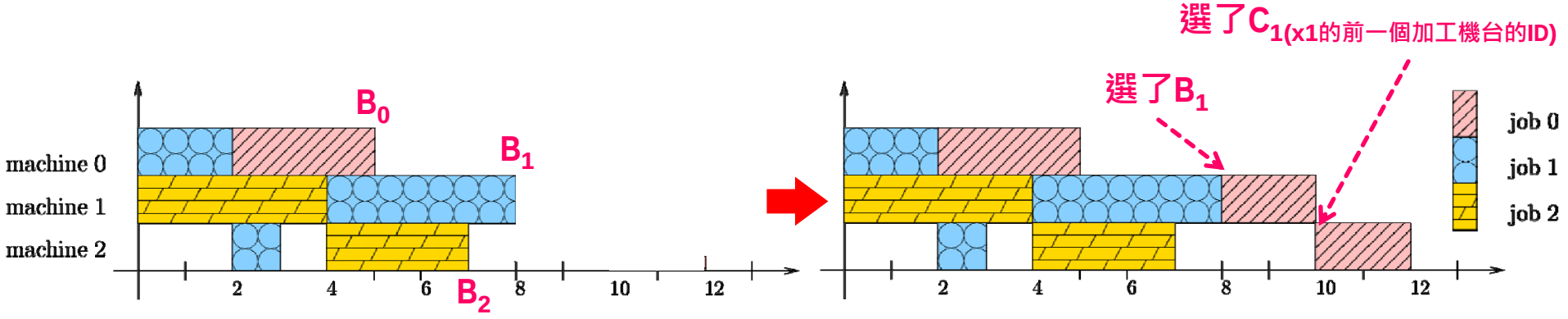
# 解碼

- 給定一個編碼 $x_0, x_1, \ldots, x_{n-1}$

| | |
|---|---|
| $S_{ij}$ | 表示 job i 在 machine j 的**開始**時間 |
| $P_{ij}$ | 表示 job i 在 machine j 的**加工**時間 |
| $C_{ij}$ | 表示 job i 在 machine j 的**完成**時間 |

- 解的解碼(盡可能把job靠左越緊越好)
  1. 定義 $B_j$ 表示 machine j 目前可開始的時間
  2. 考慮 jobs $x_0, x_1, \ldots, x_{n-1}$
     1. $S_{ij} = \max\{ B_j, C_{i(xi的前一個加工機台的ID)} \}$
     2. $C_{ij} = B_j = S_{ij} + P_{ij}$
  3. $makespan = \max_j \{B_j\}$

- E.g., 已排好 121012 → 接著排00變成12101200



# Python code for fitness evaluation

- 解的解碼(盡可能把job靠左越緊越好)

(自己練習看)

```
44 def fitFunc(x):              # 適應度函數
45     S = np.zeros((NUM_JOB, NUM_MACHINE))    # S[i][j] = Starting time of job i at machine j
46     C = np.zeros((NUM_JOB, NUM_MACHINE))    # C[i][j] = Completion time of job i at machine j
47 
48     B = np.zeros(NUM_MACHINE, dtype=int)    # B[j] = Available time of machine j
49 
50     opJob = np.zeros(NUM_JOB, dtype=int)    # opJob[i] = current operation ID of job i
51 
52     for i in range(NUM_BIT):
53         m = mOrder[x[i]][opJob[x[i]]]
54         if opJob[x[i]] != 0:
55             S[x[i]][m] = max([B[m], C[x[i]][mOrder[x[i]][opJob[x[i]]-1]]])
56         else:
57             S[x[i]][m] = B[m]
58 
59         C[x[i]][m] = B[m] = S[x[i]][m] + pTime[x[i]][opJob[x[i]]]
60 
61         opJob[x[i]] += 1
62 
63     return -max(B)             # 因為是最小化問題
```

# Exercise

- 用所提供的 “GA10-jobshop.py” 程式

  (原本是解決上述的3 jobs和3 machines的問題)

  解GA10-jobshop-abz7.txt問題

  (有20 jobs, 15 machines)，

  請調出最佳參數找出makespan < 800的解

  - FYI. 過去論文可找出makespan = 710

- (Option) 改目標函數為

  Minimize the average flow time $\Sigma_i (C_i - S_i)/n$