演算法作業說明

陳鉦鈞 (M113040018@student.nsysu.edu.tw)
林正昊 (M113040042@student.nsysu.edu.tw)
陳宥呈 (M113040112@student.nsysu.edu.tw)

Artificial Intelligence Lab@NSYSU

演算法作業3

Dynamic Programming 解 0/1 Knapsack problem

作業3- Dynamic Programming

- Simple Example

物品數量 (number of items)	4
背包最大容量 (capacity)	7
物品重量 (weight)	[1,4,5,7]
物品價值 (value)	[1,3,4,5]

			Capacity							
			0	1	2	3	4	5	6	7
item	$Value_i$	$Weight_i$	0	0	0	0	0	0	0	0
1	1	1	0							
2	4	3	0							
3	5	4	0							
4	7	5	0							

作業3- Dynamic Programming

- Simple Example

物品數量 (number of items)	4
背包最大容量 (capacity)	7
物品重量 (weight)	[1,4,5,7]
物品價值 (value)	[1,3,4,5]

			Capacity							
			0	1	2	3	4	5	6	7
item	$Value_i$	$Weight_i$	0	0	0	0	0	0	0	0
1	1	1	0	1	1	1	1	1	1	1
2	4	3	0							
3	5	4	0							
4	7	5	0							

作業3- Dynamic Programming

- Simple Example

物品數量 (number of items)	4
背包最大容量 (capacity)	7
物品重量 (weight)	[1,4,5,7]
物品價值 (value)	[1,3,4,5]

			Capacity							
			0	1	2	3	4	5	6	7
item	$Value_i$	$Weight_i$	0	0	0	0	0	0	0	0
1	1	1	0	1	1	1	1	1	1	1
2	4	3	0	1	1	4	5	5	5	5
3	5	4	0							
4	7	5	0							

作業3- Dynamic Programming

- Simple Example

物品數量 (number of items)	4
背包最大容量 (capacity)	7
物品重量 (weight)	[1,4,5,7]
物品價值 (value)	[1,3,4,5]

			Capacity							
			0	1	2	3	4	5	6	7
item	$Value_i$	$Weight_i$	0	0	0	0	0	0	0	0
1	1	1	0	1	1	1	1	1	1	1
2	4	3	0	1	1	4	5	5	5	5
3	5	4	0	1	1	4	5	6	6	9
4	7	5	0	1	1	4	5	7	8	9

作業3- Dynamic Programming

- Simple Example

物品數量 (number of items)	4
背包最大容量 (capacity)	7
物品重量 (weight)	[1,4,5,7]
物品價值 (value)	[1,3,4,5]

			Capacity							
			0	1	2	3	4	5	6	7
item	$Value_i$	$Weight_i$	0	0	0	0	0	0	0	0
1	1	1	0	1	1	1	1	1	1	1
2	4	3	0	1	1	4	5	5	5	5
3	5	4	0	1	1	4	5	6	6	9
4	7	5	0	1	1	4	5	7	8	9

演算法作業4

Genetic Algorithm 解 0/1 Knapsack problem

作業4-Genetic Algorithm

- Genetic Algorithm
  - 根據物競天擇的概念，選出較好的個體進行資訊交換

```
Initial population s
evaluation()
While(not termination):
    selection()
    crossover()
    mutation()
    evaluation()
```

作業4-Genetic Algorithm

- Evaluation：計算各解的 objective value

	item1	item2	item3	item4
profit	2	3	1	4
Weight	3	8	2	9

capacity: 13

```
Initial population s
evaluation()
While(eva < eva_max):
    selection()
    crossover()
    mutation()
    evaluation()
```

s[0]

1	1	0	0

<capacity →

s[0]

1	1	0	0

Objective value: 5

s[1]

1	0	0	1

s[2]

1	0	1	1

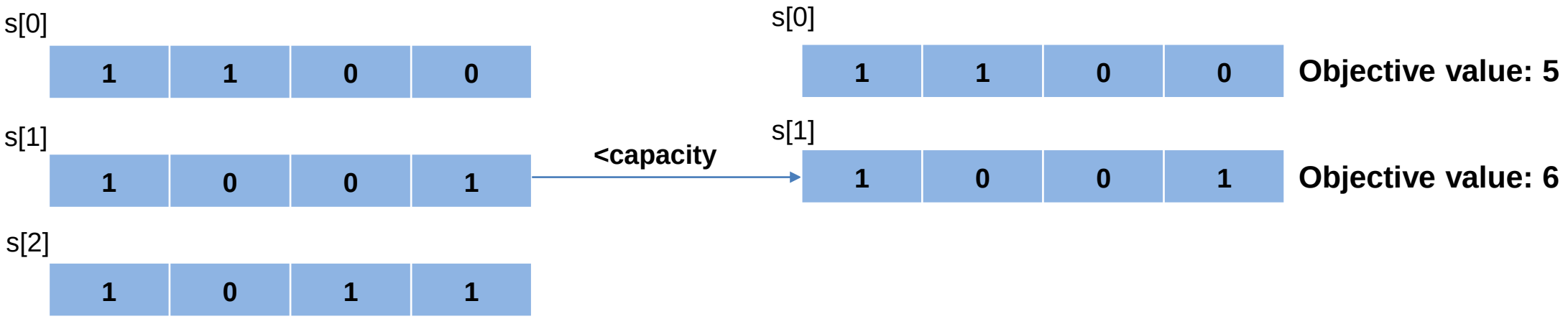
作業4-Genetic Algorithm

- Evaluation：計算各解的 objective value

	item1	item2	item3	item4
profit	2	3	1	4
Weight	3	8	2	9

capacity: 13

```
Initial population s
evaluation()
While(eva < eva_max):
    selection()
    crossover()
    mutation()
    evaluation()
```



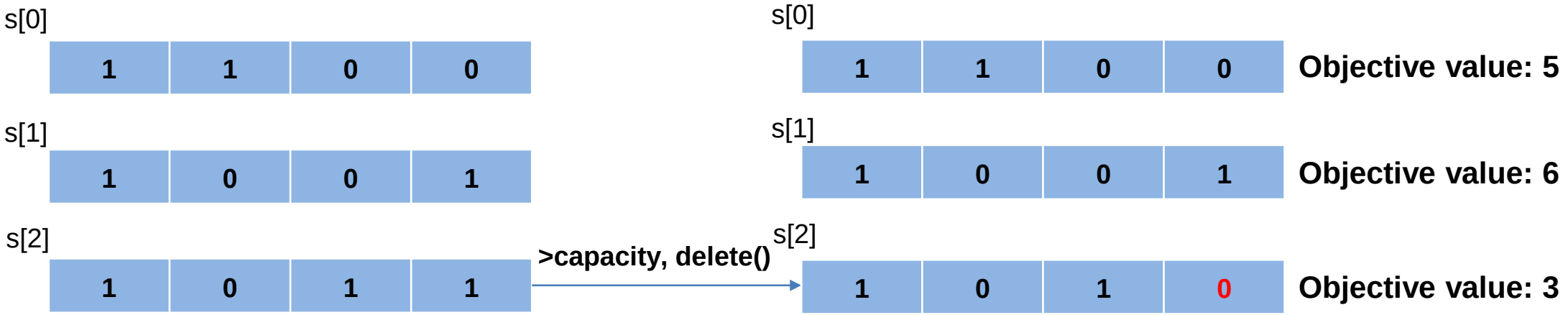
作業4-Genetic Algorithm

- Evaluation：計算各解的 objective value

	item1	item2	item3	item4
profit	2	3	1	4
Weight	3	8	2	9

capacity: 13

```
Initial population s
evaluation()
While(eva < eva_max):
    selection()
    crossover()
    mutation()
    evaluation()
```



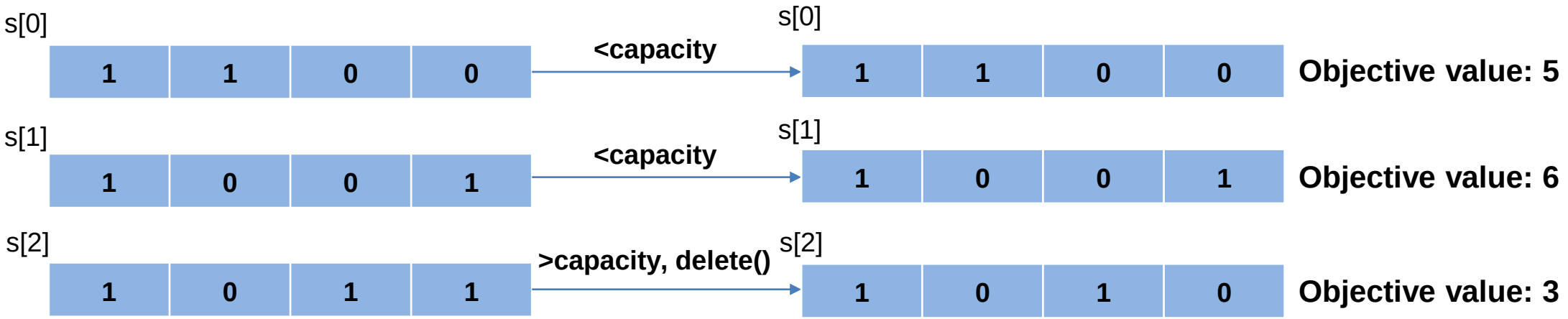
作業4-Genetic Algorithm

- Evaluation：計算各解的 objective value

	item1	item2	item3	item4
profit	2	3	1	4
Weight	3	8	2	9

capacity: 13

```
Initial population s
evaluation()
While(eva < eva_max):
    selection()
    crossover()
    mutation()
    evaluation()
```



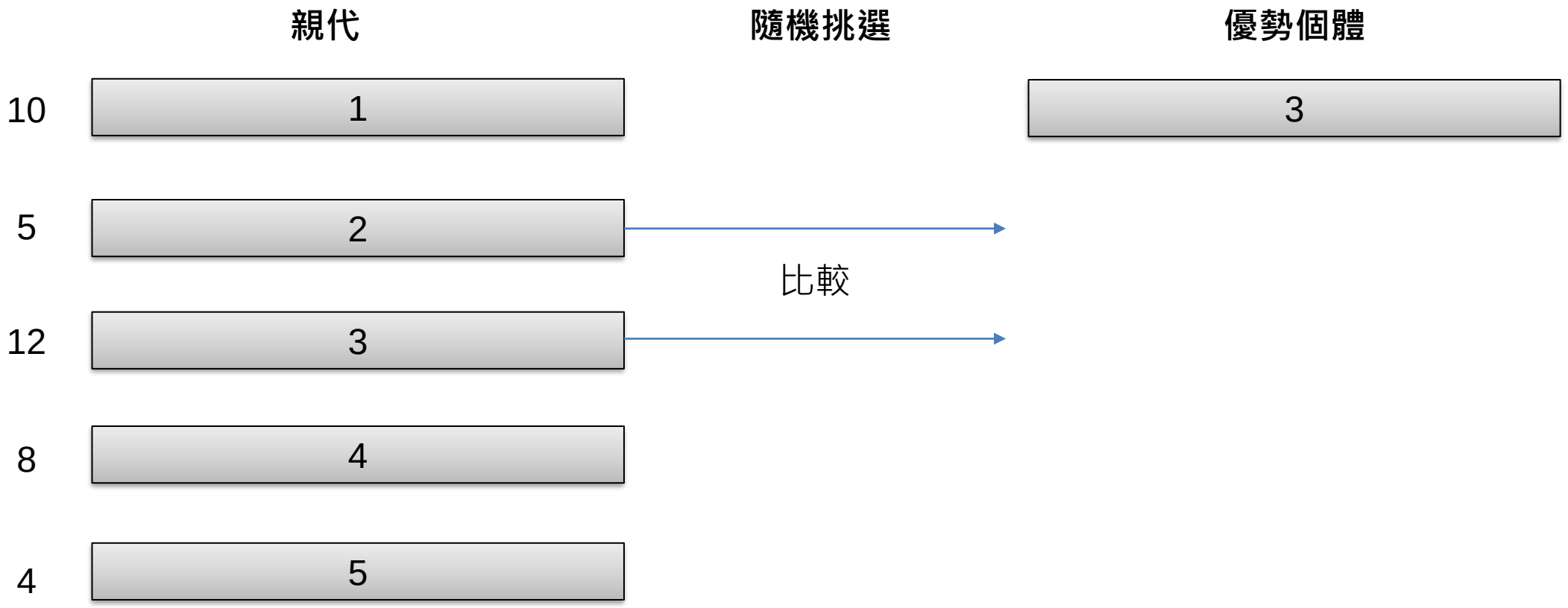
作業4-Genetic Algorithm

- Selection：選出能進行資訊交換(crossover)的解

- 方式：Tournament selection

作業4-Genetic Algorithm

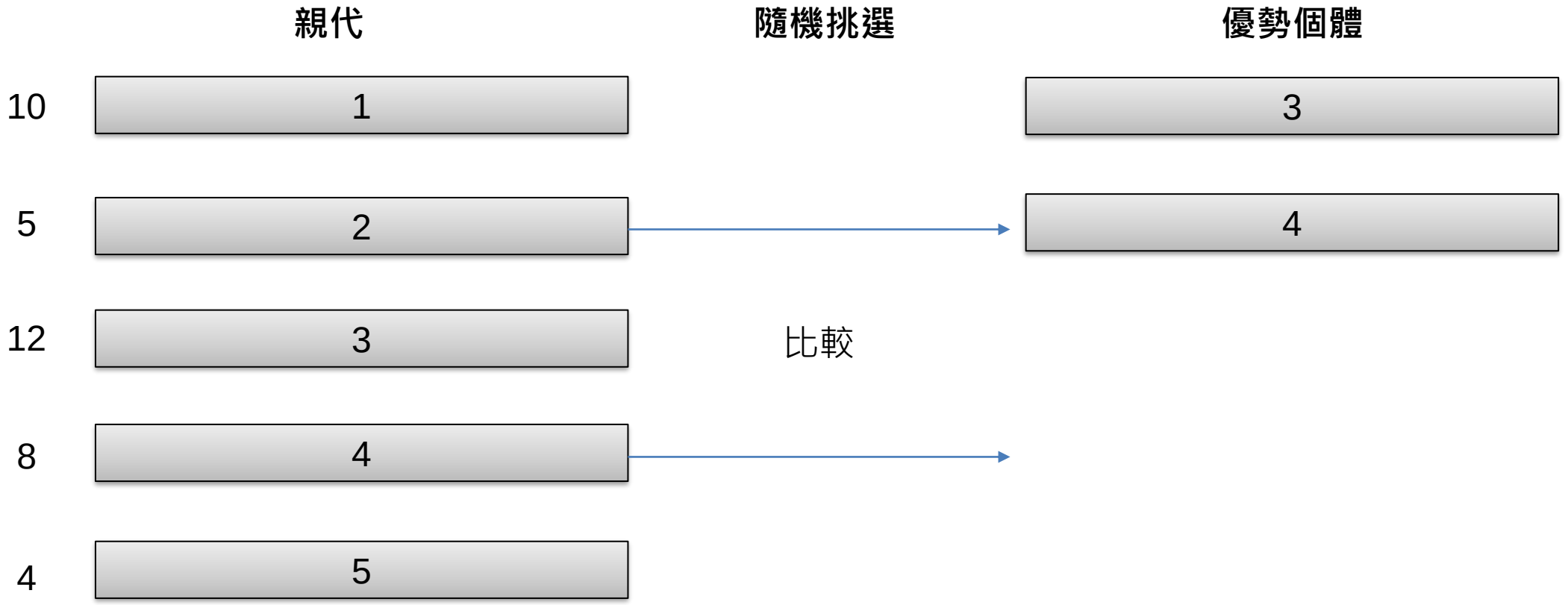
- Selection --- Tournament:
  - 隨機挑選數個解互相比較objective value，選擇最好的留下




```
Initial population s
evaluation()
While(eva < eva_max):
  selection()
  crossover()
  mutation()
  evaluation()
```

作業4-Genetic Algorithm

- Selection --- Tournament:
  - 隨機挑選數個解互相比較objective value，選擇最好的留下




```
Initial population s
evaluation()
While(eva < eva_max):
  selection()
  crossover()
  mutation()
  evaluation()
```

作業4-Genetic Algorithm

- Crossover：交換解之間的部分片段，影響彼此的搜尋資訊

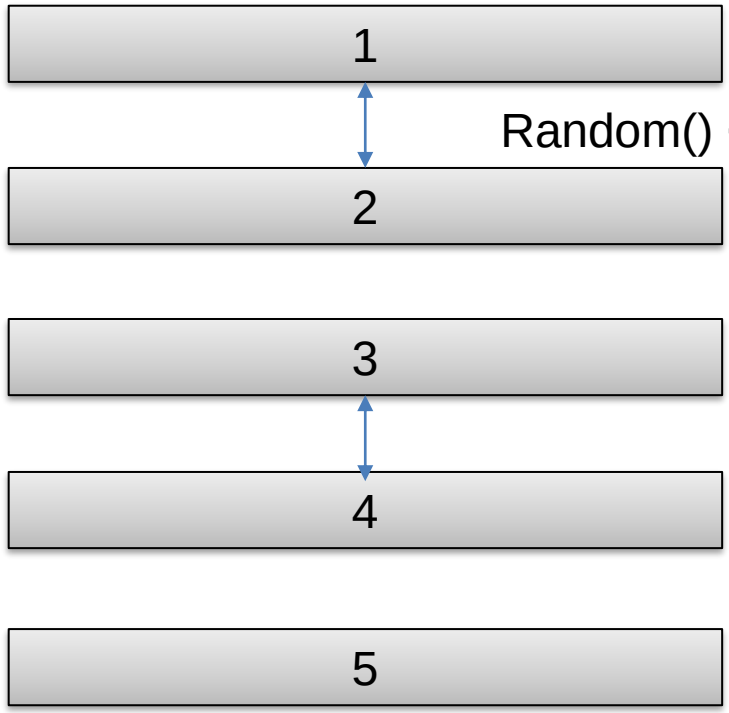
- 方式：
  - (**One-point crossover**)
  - (**Two-point crossover**)
  - (**Uniform crossover**)

作業4-Genetic Algorithm

- Crossover --- 交配率

假設交配率0.6，則每一對交配前都先隨機產生(0, 1)區間的亂數，小於0.6才進行crossover

```
Initial population s
evaluation()
While(eva < eva_max):
    selection()
    crossover()
    mutation()
    evaluation()
```

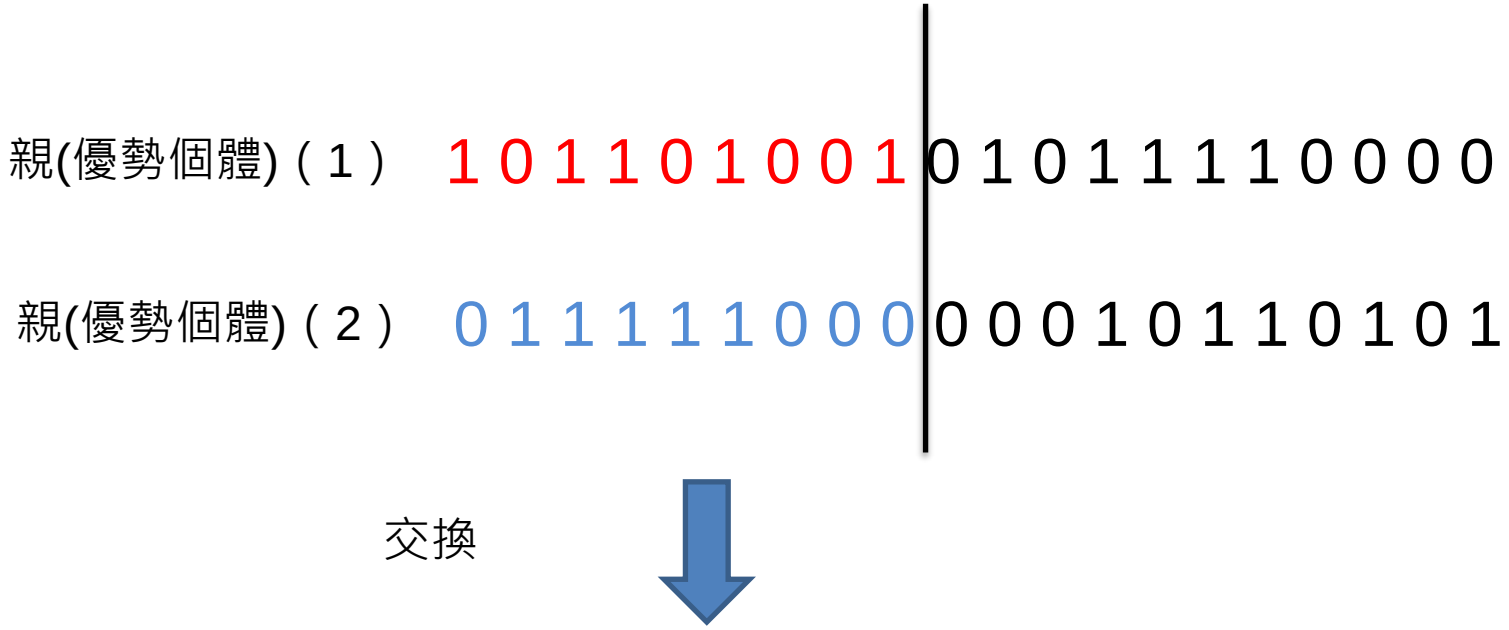


Random() < 0.6 => crossover

※若群體大小為奇數，缺交配對象的那組解可選擇直接進入下一代，或是再跟已經交配過的的某一組解交配

作業4-Genetic Algorithm

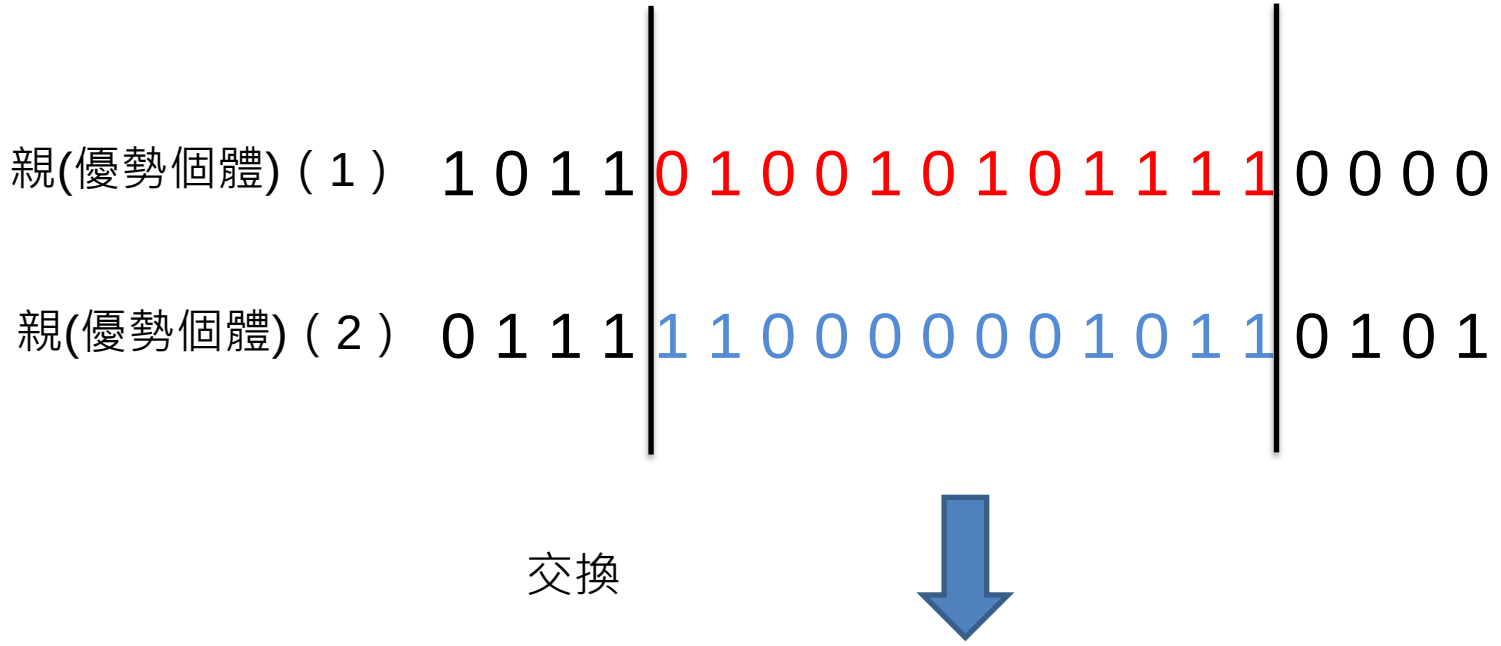
- Crossover --- **One-point** ：隨機選一點作為交換的切分點




```
Initial population s
evaluation()
While(eva < eva_max):
    selection()
    crossover()
    mutation()
    evaluation()
```

作業4-Genetic Algorithm

- Crossover --- **Two-point** ：隨機選兩點作為交換的區間




```
Initial population s
evaluation()
While(eva < eva_max):
    selection()
    crossover()
    mutation()
    evaluation()
```

作業4-Genetic Algorithm

- Uniform crossover

親(優勢個體)(1)　1 0 1 1 0 1 0

Mask　0 1 0 1 1 0 1

親(優勢個體)(2)　0 1 1 1 1 1 0

交換

子(1)　1 1 1 1 1 1 0

子(2)　0 0 1 1 0 1 0

```
Initial population s
evaluation()
While(eva < eva_max):
    selection()
    crossover()
    mutation()
    evaluation()
```

作業4-Genetic Algorithm

- Mutation : 存在一突變率，檢查每一組解是否需要突變。當 Random() < 突變率時，該組解會隨機做微小的變動

```
Initial population s
evaluation()
While(eva < eva_max):
    selection()
    crossover()
    mutation()
    evaluation()
```

0 1 1 1 1 1 0 0 0 **0** 0 0 1 0 1 1 0 0 0 0

0 1 1 1 1 1 0 0 0 **1** 0 0 1 0 1 1 0 0 0 0

作業4-Genetic Algorithm

- 到底用什麼方式比較好：視問題而定

各Operator都有自己的調整空間，掌握各Operator的設計概念為主

1. Selection：選出能改善搜尋能力的解，作為資訊交換對象
2. Crossover：解之間進行資訊交換，嘗試得到更好的解
3. Mutation：小幅度調整解，增加搜尋變化性

作業4-Genetic Algorithm

- 執行要求
  - 執行 30run, 每個 data_set 有相對應的 evaluation_max 次數
  - 執行完畢需輸出 30run 所得到的平均最佳解

	dt1	dt2	dt3
evaluation_max	1000	10000	1000000
best	309	1458	13549094
requested	290	1400	13545000

作業4-Genetic Algorithm

- 加分規則
  - 執行 30run, dt3 在 evaluation_max = 500000 每次都可以找到 best

	dt3
evaluation_max	500000
best	13549094
requested	13549094

作業4-Genetic Algorithm

- 參數參考
  - Population_size: 100
  - Crossover_rate: 0.6
  - mutation_rate: 0.01

```
Initial population s
evaluation()
While(eva < eva_max):
    selection()
    crossover()
    mutation()
    evaluation()
```

作業繳交規則

規則

- 繳交作業格式
  - 一律壓縮成：學號_hwx.zip (e.g. b103040000_hw1.zip)
  - 壓縮檔須包含 (請勿包含.exe)
  1. 程式碼 (C or C++)
  2. 輸出檔 (ans.txt)

- 繳交方式：上傳網路大學
- 繳交期限：12/22 (五) 23:59

- 實體demo位址：EC5009-1 (請自行攜帶電腦)
- 實體測驗時間：12/12,12/19,12/21,12/22,12/26
  下午1:30 ~5:00，依公告為主

評分標準

程式是否能正確執行?	30% (不能執行則全部拿0分)
答案是否正確?	20% (答案錯最多拿50分)
程式撰寫之結構與邏輯是否正確?	20%
輸出結果是否完整?	10%
清楚表達程式流程? (口頭 or 註解)	10%
繳交格式是否正確? (檔案名稱 and 檔案格式)	5%
是否能動態讀入readfile?	5%

※ 所有項目均為部分給分

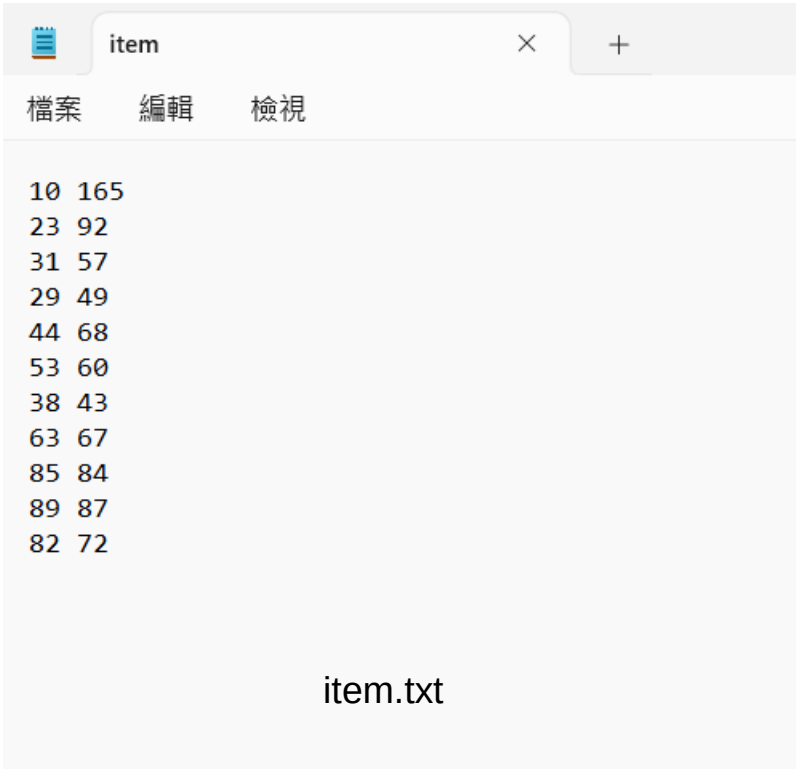
讀檔

- 3筆測試資料 (每筆資料物品數量不同)
  - item.txt : 物品重量、價值
  - ans.txt ：最佳解 (僅前兩筆測試資料提供)

讀檔範例

10 165
23 92
31 57
29 49
44 68
53 60
38 43
63 67
85 84
89 87
82 72

物品數量 (number of items)	10
背包最大容量 (capacity)	165
物品重量 (weight)	[23,31,29,44,53,38,63,85,89,82]
物品價值 (value)	[92,57,49,68,60,43,67,84,87,72]



item.txt

輸出

- 每筆測試資料皆獨立輸出
  - e.g. ans_dt01.txt, ans_dt02.txt, ans_dt03.txt

- 輸出規定
  1. 輸出背包所裝的所有物品總價值
  2. 以0和1表示物品是否被選擇

輸出範例

- max profit:309
- solution:1111010000

Thank You ;-)