**柔性計算法與應用 期末報告**

A genetic algorithm for unrelated parallel machine scheduling problem with sequence dependent setup time

學生: 賴春匠

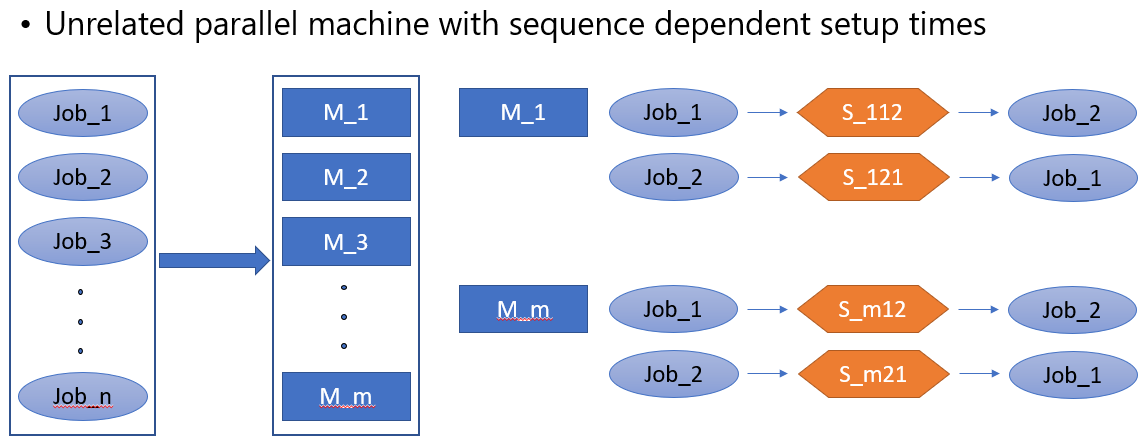
學號: r09546021

1. **前言**

在這次的期末報告中，我與自身研究興趣以及目前所遇到需要解決的問題結合，透過本門課程所學以及參考文獻，實做出一個以基因演算法求解無相關平行機台排程問題，此問題同時考慮了工作在不同機台上有不同加工時間，以及在每台機台上不同的加工順序導致工作間設置時間不同的情況，目標為最小化整體完工時間。求解的最大問題規模為 250 個工作與 30 台機台。

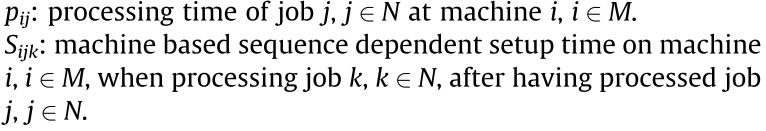
1. **問題與方法**

***問題定義***

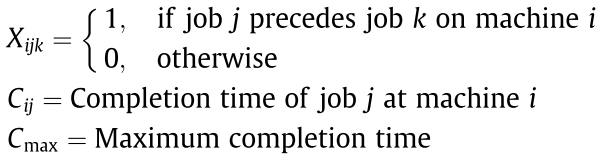
(示意圖)

有 n 個工作要分派到 m 台機台上進行加工，每台機器都可以加工任何工作，加工時間依工作、機台組合均不同。並且考慮了工作在機台上的設置時間，會依該機台加工的上一個工作而有所不同，且同樣一個加工順序，在不同的機器上也會有不同的設置時間。此問題可以建構為一混整數規劃模型 (MILP):

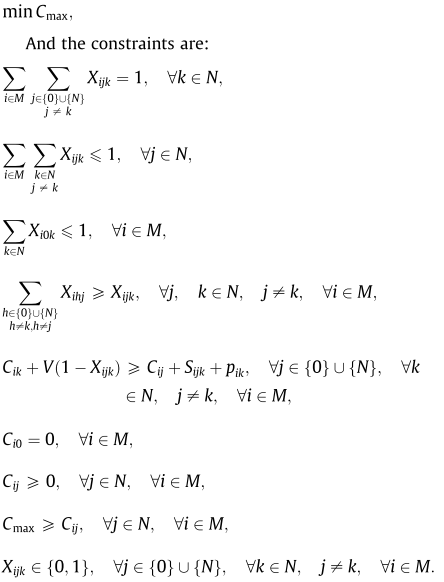
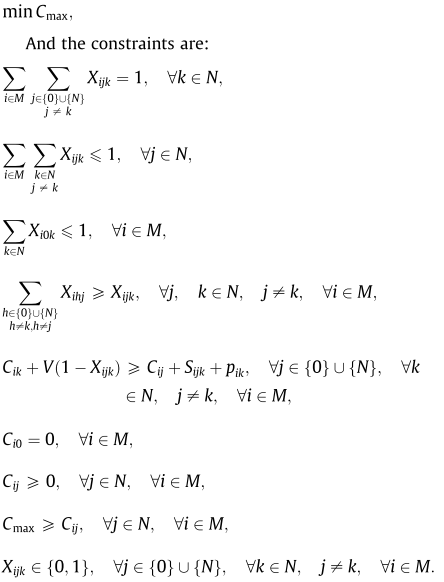
* **參數**



* **決策變數**



* **限制式**



***基因演算法***

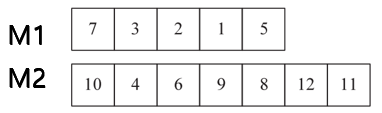
本篇報告所建構的基因演算法參考[Vallada and Ruiz (2011)](https://www.sciencedirect.com/science/article/pii/S0377221715002763?casa_token=1N2SFduKAdIAAAAA:pOe7Dr4_A9kc-qDwl8Q6CUbvDCOqqUm7VSW8JuX8VmWD8cO0gti8djqIH_wb9FL27Jn319Ks8oAk#bib0398)所提出之作法，其在框架上與本門課程所學相同，均為透過產生起始種群、選擇交配個體、交配、突變、更新種群來改善染色體 (解代理人)，然而其中解的表示方式與許多步驟均有其特別之處，以下進行說明:

* **染色體編碼**

將工作進行編號，使用二維陣列儲存一條染色體，其大小為:

Chromosome[機器數][工作數]，k = Chromosome[i][j] 代表在這個解中，機台 I 第 j 個加工編號為 k 的工作。而及台加工完所有工作後剩餘的空間全部設其值為 -1，以便於後續計算。

(ex.)

 Chromosome:

[ [7,3,2,1,5,-1,-1,-1,-1,-1,-1,-1],

[10,4,6,9,8,12,11,-1,-1,-1,-1,-1] ]

* **產生起始種群**

在本方法中，透過隨機分配的方式產生起始種群，之後隨機對一定比例的個體執行 Kurz and Askin(2001) 提出的 Multiple Insertion，此為一高效率演算法，透過將設置時間整合進加工時間，將工作依此時間由大到小進行排序。根據這個順序嘗試將每個工作插入任何一個可以插入的位置計算完工時間，並最終將其放入最好的位置。透過上述演算法可以使得起始種群就具有相當良好的個體，以改善整體求解效率。

* **N-Tournament Selection**

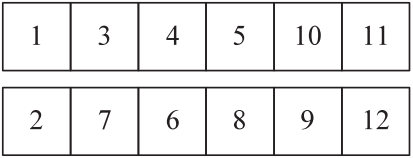
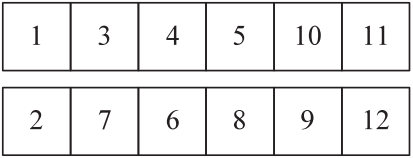
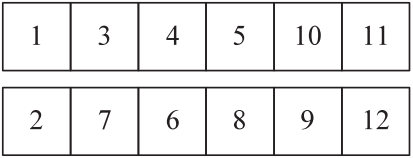
在選擇要進行交配的親代時使用 N-Tournament 方法，做法為給定一個參數 Pressure，從種群中隨機選出 Pressure% 個個體並計算其適合度，適合度最高的個體即被選中參與交配。重複執行兩次即可找到一對參與交配的親代 (後稱P1、P2) 並進入交配步驟。值得注意的是，在這裡可以很簡單地透過計算染色體的整體完工時間來得到其適合度排名，故為一個非常簡單且快速的做法。

* **交配方法 (local search enhanced)**

選定要進行交配的親代 (P1、P2) 後，有一定的機率進行交配產生兩個子代 (後稱 C1、C2)。首先在 P1 每一台機台的工作序列中，隨機找一點將該機台工作序列切分為兩段，前後半段分別複製成為C1、C2的起始工作序列。接下來觀察P2機台與工作指派資訊，不考慮已經存在於子代的工作，嘗試將尚未加入的工作插入相同機台的每一個可能插入位置，並最終將工作插入能使該機台自身完工時間最小的位置中。透過此交配方法能夠保存P1工作機台指派以及工作先後關係資訊，而對於P2則保存了其指派關係，對於無法保存的P2工作先後關係則透過local search方法使其插入較適合的位置。其操作方法如下頁所述:

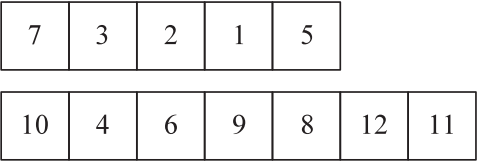
(示意圖)

P1 = 3

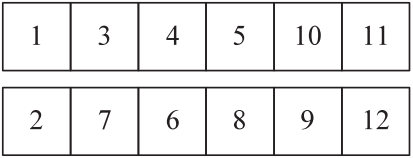
*Parent1 Parent2*

Parent 1



**

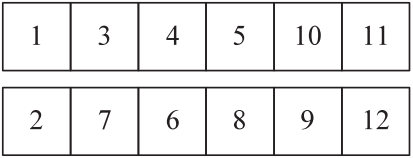
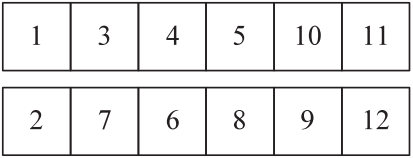
M1 M1

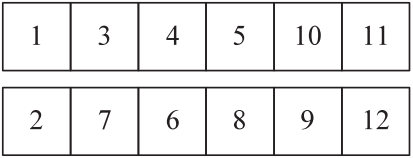


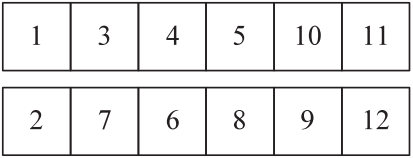
M2 M2

P2 = 4

*Child1 Child2*

**

M**1 (7, 3, 2, 1, 5) (7, 3, 2, 1, 5)

**

M2 (10, 4, 6, 9, 8, 12, 11) (10, 4, 6, 9, 8, 12, 11)

嘗試將示意圖中子代不重複工作 (黑色編號) 放入對應機台任何一個位置計算該機台完工時間，最後將其放入結果最小的位置。

* **突變**

當交配完成後，有一定機率進行突變，其做法為對要突變的染色體，隨機選擇一台機台並在其工作序列中隨機選擇一個工作，將其隨機放入同一台機台的任何一個位置中。

* **Local Search**

突變完成後，有一定機率對產生的子代與原種群中最好的個體進行Local Search，在這裡使用一個非常高效率且快速的做法: Inter-machine insertion neighborhood。對一個要進行Local Search的染色體，嘗試將每一台機器上的每一個工作，插入其他機器的任何一個可插入位置，判斷這次插入能否改善解，若可以則將工作插入並重新從頭執行一次，直到沒有任何插入能改善此解，即達到區域最佳解。判斷是否改善解的方法如下所述:

1. 若牽涉到工作移動的兩台機台完工時間都減少則為有改善
2. 若牽涉到工作移動的兩台機台完工時間都增加則為沒有改善
3. 若牽涉到工作移動的兩台機台完工時間一增一減則檢查增加量是否多過減少量以及此移動是否增加了整體完工時間，若皆為非則有改善。

值得注意的是在判斷的過程中沒有必要計算機台完工時間，只需要將工作移出與移入機台上相對應的設置時間以及工作加工時間進行加減，就足以判對是否有改善解，最多只需要進行四次加法與四次減法便能完成一次判斷，故此為一個相當快速的做法。

* **更新種群**

產生子代後比較子代與原先種群中最差的個體，若子代較為優良且獨特(種群中沒有一模一樣的個體)，則將最差的個體用子代替換掉。進入下一次迭代。

1. **系統功能**

* **讀取標竿問題**

在測試時使用<http://soa.iti.es>上此問題的標竿問題，故設計能夠讀取此檔案格式的方法，透過讀取可以取得工作數與機器數、工作指派到機器上的加工時間以及在不同機器上個工作間的設置時間。儲存資料型態如下:

加工時間: P = int[工作數, 機台數] 🡪 P[i, j] 代表機台j加工工作i所需的加工時間。

設置時間: S = int[機台數][工作數, 工作數] 🡪 S[i][j, k] 代表在機台i上工作j接工作k所需要的設置時間。

* **參數設定**

使用者可以調整基因演算法與Local Search中各項參數，包含:

PopulationSize、CrossoverRate、MutationRate、ItrationLimit等基本GA參數與

InitialMIHeuristicRate: 產生起始種群時有多少比例的個體要進行 Multiple Insertion來改善解。

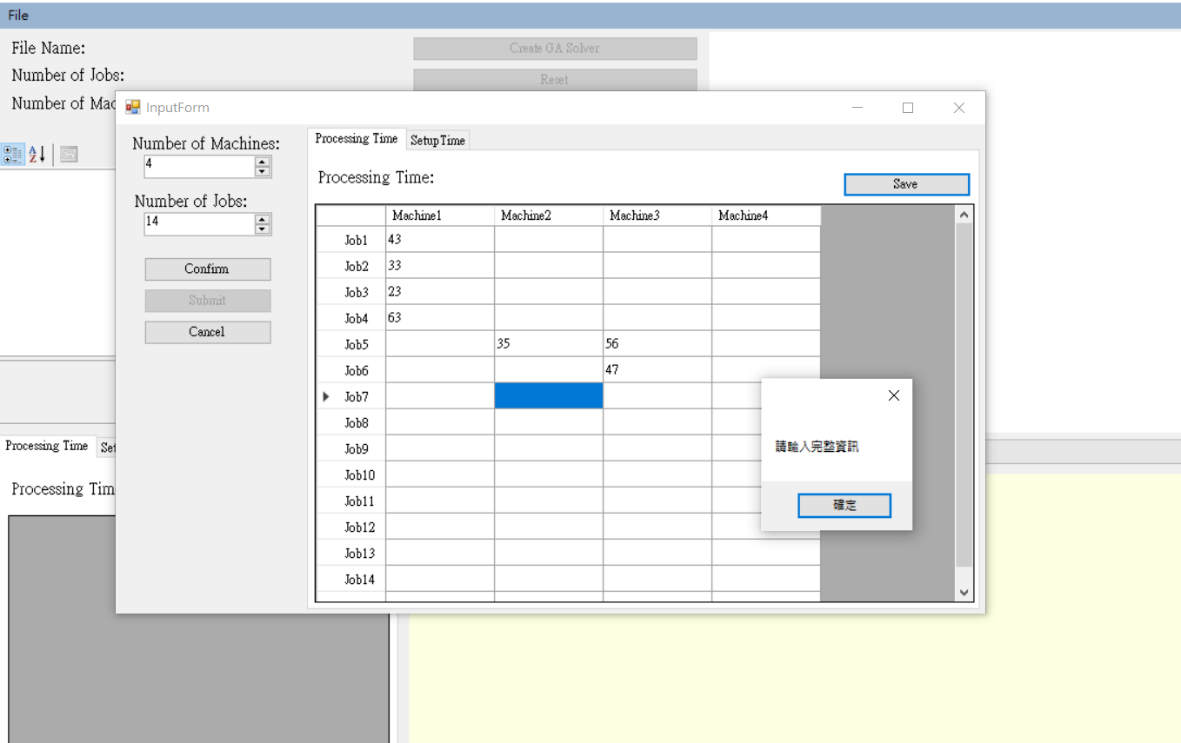
LocalSearch: 是否要使用Local Search

LocalSearchRate: 每個準備做Local Search的染色體中有多大的機率執行。

Pressure: 進行N-Tournament時隨機從種群中選出多少百分比的個體來挑出最好的成為交配候選人。

同時使用者也可以透過DataGridView調整標竿問題中各個Processing Time以及Setup Time的參數。

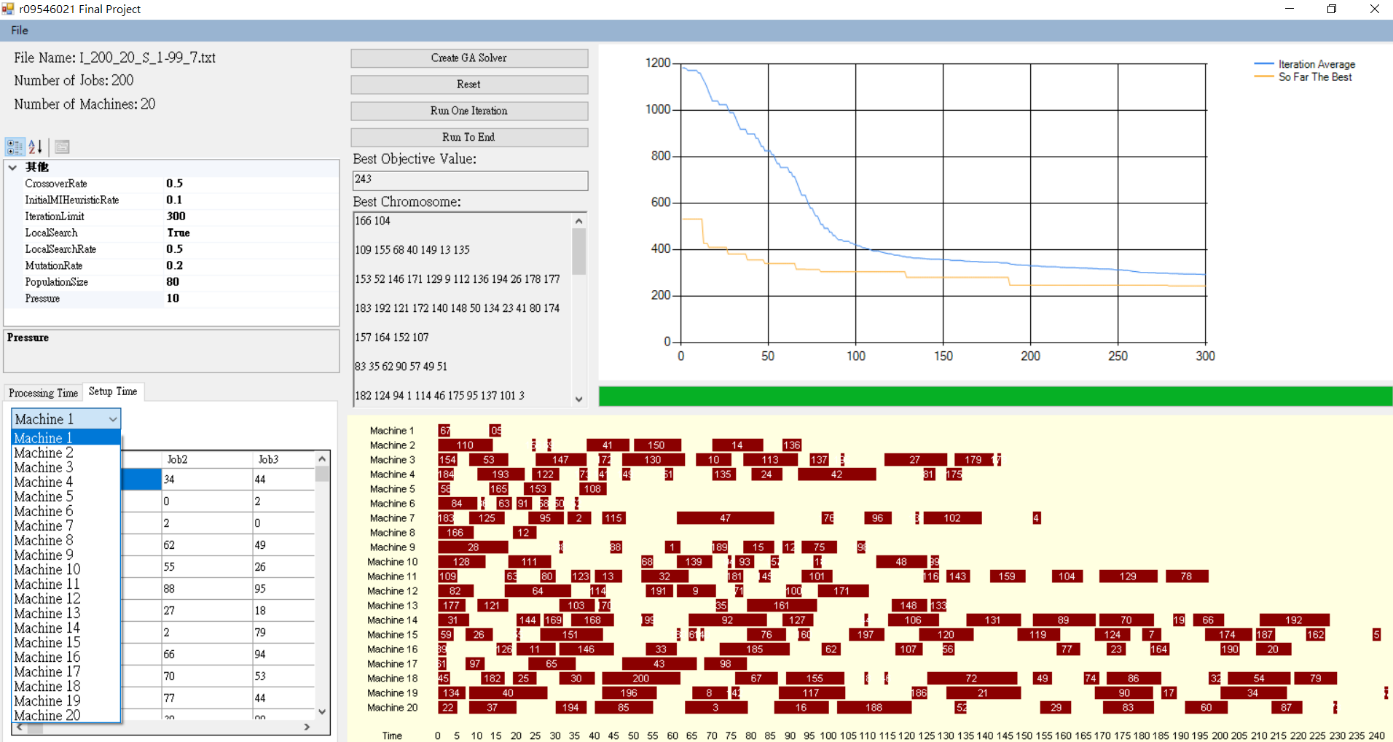
* **建立新檔案**

透過一個與MainForm連結的子表單，使用者可以新建符合自身情況的包含工作數、機台數、加工時間與設置時間等等，透過委派與主表單連結，新建完的檔案在子表單關閉後可直接在主表單進行求解。

* **儲存檔案**

觀察並模仿標竿問題的檔案格式，將使用者使用本系統新建立的問題，或讀取標竿問題進行修改後的問題參數儲存成與標竿問題相同的檔案格式，提供使用者未來讀取儲存的檔案、進行修改或與他人合作之用。

* **視覺化呈現結果**

使用DataGridView紀錄問題的各項參數(加工時間、設置時間)(A)，並使用折線圖呈現基因演算法隨代次增加演化的狀況(B)。最後使用畫布以長條表示工作的持續時間，刻畫出甘特圖以清楚的展示計算結果(C)。

C

A

B

1. **問題實作**

在[Vallada and Ruiz (2011)](https://www.sciencedirect.com/science/article/pii/S0377221715002763?casa_token=1N2SFduKAdIAAAAA:pOe7Dr4_A9kc-qDwl8Q6CUbvDCOqqUm7VSW8JuX8VmWD8cO0gti8djqIH_wb9FL27Jn319Ks8oAk#bib0398)有對此基因演算法的參數進行校正，故在實作問題時我參考其校正後的參數進行設定。

PopulationSize: 80

InitialMIHeuristicRate: 0.1

CrossoverRate: 0.5

MutationRate: 0.5

LocalSearchRate: 1

Pressure: 10

IterationLimit: {small: 3000, large: 1000}

並對以下問題參數進行求解:

Small Instance: 工作數: {6, 8, 10, 12} 機台數: {2, 5}

Large Instance: 工作數: {50, 100, 150, 200, 250} 機台數: {10, 30}

對上面每種組合都實作一個標竿問題，其結果如下頁所示:



其中以紅色表示找到已知的最佳解或優於原文作者求解相同問題的結果。僅有I\_250\_10\_S\_1-9\_1、I\_250\_30\_S\_1-9\_1分別求解了7分32秒與11分03秒，其餘實例求解時間均少於3分鐘。

1. **結論**

在本次的期末專案中，我選擇了一個感興趣、並且實際遇到的問題做為研究方向，透過文獻搜尋以及本學期所學進行結合，實作了一個應用基因演算法求解的平行機台排程系統。演算法中包含了一個Local search enhanced crossover operator以及快速且高效的Inter-machine insertion neighborhood Local search方法。同時運用本學期所學建構使用者介面，並以在類神經網路作業中使用的畫圖方法繪製出甘特圖以清楚的呈現結果。最終使用與參考文獻中所使用相同的標竿問題進行求解，均能夠在合理的時間內得出與原文作者相似乃至於更好的結果。

在求解標竿問題時，能夠發現即使是在代次更替時使加入的子代獨特以確保解的多樣性，演算法的解仍然會在很少的代次內快速收斂。這應是Local Search造成的結果，雖然可以在短時間內獲得相當好的解，然而在跳脫出Local Optimal的能力上仍有改進空間，或許能夠透過較大程度改變解的突變方法來嘗試改良。而在系統的完備性上，除了求解以及存檔讀檔以及建立新檔案以外，圖像化呈現解應該能更加清楚、精準、正規，包含可以使用實際時間或日期作為時間軸等等。這些是我未來能夠改良的方向。

1. **參考文獻**
2. **A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times,** European Journal of Operational **Research, 211 (3) (2011), pp. 612-622,** [**Vallada and Ruiz, 2011**](https://www.sciencedirect.com/science/article/pii/S0377221715002763?casa_token=1N2SFduKAdIAAAAA:pOe7Dr4_A9kc-qDwl8Q6CUbvDCOqqUm7VSW8JuX8VmWD8cO0gti8djqIH_wb9FL27Jn319Ks8oAk#bbib0398)
3. **Heuristic scheduling of parallel machines with sequence-dependent set-up times, International Journal of Production Research, 39 (2001), pp. 3747-3769, Kurz and Askin, 2001**