Minimizing Makespan in Permutation Flow Shops using Memetic Algorithms

戴宗明

*rf37535@gmail.com*

陳佑欣

*bazukaoc@gmail.com*

盧緯

Abc@ntnu.edu.tw

# **問題定義**

定序流線型工廠排程問題(*Permutation Flow shop Scheduling Problem*, *PFSP*) 假設有 *j* 份彼此獨立的工作，必須通過 *ｍ* 台彼此獨立的機器，每台機器處理每份工作有固定的執行時間，而且必須依照固定的機器順序處理，即每台機器只會接受上一台機器處理結束的工作，而且每台機器同時間只能處理一份工作，處理順序為先進先出。

問題的輸入為工廠中所有機器的效能矩陣，亦即每台機器處理每份工作的所需時間，輸出為一特定的工作序列，序列中出現的數字代表工作的編號，序列中的數字不會重複。問題的目標則是最佳化此序列，使得以序列中的工作排程進入生產線後能夠得到最快的處理時間，在[2]中，解答有*j!*種可能，問題等級被認定為是*NP*-*Complete*。

# **文化因子遺傳演算法**

|  |
| --- |
|  |
| 圖1: 文化因子遺傳演算法流程圖 |

## 目標函數與適應函數

由於此問題為求得最小之生產線處理時間，是一最小化的問題，故適應函數設定為目標函數之倒數。

## 初始解 (Initial Population)

1. 亂數(*Randomize*)

初始化族群裡每個染色體的基因，由工作編號小到大排列，且數字不重複，接著亂數打亂其順序。

1. 啟發式(*Heuristic*)

先以亂數初始化族群，再以區域搜尋中的反覆改善法對族群的1/3個體做一次搜尋。

## 親代選擇 (Mating Selection)

1. 競爭法(*Tournament selection*)

隨機取兩條染色體，並比較其適應值，適應值較高者則被取出為親代。重複以上程序直到取出的親代數目與族群大小相同。

## 交配 (Reproduction or Crossover)

交配的演算法皆為輸入兩條染色體，亦即親代，得到兩條新的染色體，為子代。

1. *Order Crossover* (*OX*)

隨機選取一個起始點，並隨機一段長度，從起始點向後算長度內的基因交換繼承至另一子代，再從起始點加上長度後的點繼續，將自己子代中缺乏的基因依序填入。

1. *Linear Order Crossover* (*LOX*)  
   由Falkenauer, E. and Bouffoix, S.在[3]中所提出，原先親代兩條染色體各保留一段基因作為遺傳，其餘部分交叉從頭部開始依序繼承另一方擁有但是自身尚未擁有的基因。
2. *Partially mapped crossover* (*PMX*)  
   由Goldberg, D. and Lingle, R.在[4]中所提出，親代染色體以對方保留基因的部分作為參考，若基因片段不同，則和自身相同基因交換；若基因片段相同，則不交換。
3. *Cycle crossover* (*CX*)

從其中一條染色體中隨機選取一個位置作為起始點，若另一條染色體相對位置的基因與之不同，則將起始點移動至原先染色體擁有同樣基因的位置，如此循環，直到找到一位置集合，擁有封閉的共同基因組，然後交換它們。

## 突變 (Mutation)

突變的演算法皆為輸入一條染色體，輸出為同一條受過親為擾動變化的染色體。

1. *Insertion*

隨機選取一個起始點作為操作基因，再隨機選取一個插入的位置，將操作基因移植插入的位置，並將其後的基因往後挪至直到填補原先操作基因位置留下的空缺。

1. *Swap*隨機選取兩個基因片段進行交換。
2. *Inverse*

隨機選取一個起始點，再隨機選取一段長度，將從起始點向後算起長度內的基因排列順序前後顛倒。

## 區域搜尋 (Local Search)

1. 反覆改善法(*Iterative Improvement*, *II*)  
   又稱為基本區域搜尋(*basic local search*, *BLS*)，是一種非常直觀的求解方式，其核心想法是，在鄰域函式中尋找比當下更好的解。以下是其演算法：

|  |
| --- |
| *s←GenerateInitialSolution()*  **repeat**  *s←Improve(N(s))*  **until** *no improvement is possible* |

這種直觀的求解方式，容易卡在區域最佳解而無法跳脫，因此需要新的機制以幫助跳脫。

1. 模擬退火法(*Simulated Annealing*, *SA*)

首先設定初始溫度與一初始解，並反覆從初始解生出一解，如此解的適應值優於初始解的適應值，則取代初始解，如劣於初始解的適應值，則當此條件：

成立時取代初始解，否則捨棄此解。並且於每次降低溫度，直到溫度到達停止溫度則停止。

初始溫度設為1000，冷卻率為0.99，停止溫度為2.42092e-322。

1. 禁忌搜尋法(*Tabu Search*, *TS*)

演算過程與反覆改善法雷同，差異在於擁有一個*Tabu-list*紀錄上一步地移動資訊，並且下次移動時不能移動至*Tabu-list*中曾記錄過的地點。這邊記錄的移動資訊為目標適應值。

## 更新區域搜尋解(Apply Local Search)

1. *Lamarckian*

區域搜尋結束後更新該條染色體的適應值，並且同時更新該條染色體的基因序列。

1. *Baldwinian*

區域搜尋結束後僅更新該條染色體的適應值。

## 環境選擇 (Environment Selection)

1. *Top-half*

從子代與親代中，由最佳的個體開始填入下一代的族群中，直到下一代族群的個數達到設定的族群大小為止。

1. *Generation Model*

子代直接取代親代，成為下一代的族群。

# **實驗與結果**

## 測試問題集

使用1993年E. Taillard發表的”Benchmarks for basic scheduling problems”[1]，作為測試資料。該測試資料是實作Bratley在[5]中提出的亂數產生器產生*M* = {5, 10, 20}數量的機器，對應於*N* = {20, 50, 100, 200, 500}數量工作的工作時間，並且經過挑選，原則如下：

1. 使用禁忌搜尋法求得的解距離最佳解越遠越好
2. 每次隨機的起點都盡可能找到不同區域最佳解

在本實驗中，只使用*M*={5, 10, 20}三種機器數量和*N*={20, 50, 100}三種工作數量作為測試資料。

## 實驗設定

計算環境：

中央處理器：Intel Core-i7 2.5GH

記憶體：32GB

作業系統：Windows 8.1

測試次數：100次

突變機率：20%

演化代數：200代

## 實驗結果

表 1: 表格標題放在上面 (8 點字)

|  |  |  |  |
| --- | --- | --- | --- |
|  | II | SA | TS |
| 20×5 | Min/Avg/Max |  |  |
| 20×10 |  |  |  |
| 20×20 |  |  |  |
| 50×5 |  |  |  |
| 50×10 |  |  |  |
| 50×20 |  |  |  |
| 100×5 |  |  |  |
| 100×10 |  |  |  |
| 100×20 |  |  |  |

# **結論**

##### 參考文獻

1. E. Taillard, “Benchmarks for basic scheduling problems,” *European Journal of Operational Research*, vol. 64, pp. 278–285, 1993.
2. Rinnooy Kan, A. H. G. (1976). Machine Scheduling Problems: Classification, Complexity and Computations. *Martinus Nijhoff, The Hague, The Netherlands.*
3. Falkenauer, E. and Bouffoix, S. (1991) A genetic algorithm for job shop. *IEEE ICRA*, pp. 824-829.
4. Goldberg, D. and Lingle, R. (1985) Allels, Loci and the traveling salesman problem, 1st *ICGA*, pp. 154-159.
5. P. Bratley, B.L. Fox & L.E. Schrage, “A guide to Simulation”, *Springer Verlag*, New-York, 1983.