智慧化企業整合 Individual Research Project

主題:SSO 演算法雙機排程程式

學生:鍾杰峰

學號:107034565

目錄

摘要

雙機排程問題為產業界所需面對的情況之一，若所有待加工任務工序相同，則可以強森法則(Johnson's Rule)進行排程。本研究旨在開發一便捷雙機排程程式，結合 SSO 演算法、網頁爬蟲程式及網頁系統，讓使用者可以在短時間內得到大量任務之排成結果。

第一章、介紹

1.1 研究背景及目的

生產排程為產業界每日所需處理的情況，排程優劣會對生產節奏造成影響。本研究假設問題情況為:所有任務加工順序相同，且共有兩個工作中心，此情況適用強森法則(Johnson's Rule)進行排程。本研究目的為建構一便捷雙機排程用途的程式，讓使用者可在短時間內得到工作排程結果。

1.2 文獻回顧

柔性計算概念在 1991 年首次被提出，有別於以往求精確解的方式，轉而強調藉由允許以不精確、不確定方式達成較低求解成本的目標。其中，SSO 於 2009 年首度由葉維彰老師所提出之柔性計算方式，用以改善 PSO 演算法求解離散問題時的短處，具有高效、彈性佳、簡單之優點。以下詳述 SSO 符號表示及更新機制:

$X_i^t = (x_{i1}^t, x_{i2}^t \dots\dots x_{i,Nvar}^t)$ 表示第 i 個解於第 t 個迭代，當中x_{ij}^t表示此解中第 j 個變數，每個解中有 Nvar 個變數，共有 Nsol 個解。

$$x_{ij}^t = \begin{cases} g_j \ \ if \ \ \rho \in [0, C_g) \\ p_{ij}^{t-1} \ \ if \ \ \rho \in [C_g, C_p) \\ x_{ij}^{t-1} \ \ if \ \ \rho \in [C_p, C_w) \\ x \ \ if \ \ \rho \in [C_w, 1] \end{cases}$$

圖 1.SSO 更新機制

SSO 具體演算步驟如下所示:

Step 0:隨機產生X_i^0並令其等於第 i 個解中表現最佳解，P_i，並以評估解表現

之適合度函數 F()計算F(X_i^0)，從而找到所有解中表現最佳者 g_{best}；另迭代數 t=1，解數 i=1..Nsol。

Step 1:另 i=1。

Step 2:將解X_i^{t-1}以上述機制更新至X_i^t，並計算F(X_i^t)。

Step 3:若F(X_i^t)表現優於F(P_i)，則使$P_i = X_i^t$。反之，至 Step 5。

Step 4:若F(P_i)表現優於F(gbest)，另 g_{best}=i。

Step 5:若 i<Nsol，則 i=i+1，回到 Step 2。

Step 6:若 t 等於迭代數 Ngen 或是達到設定執行時間上限，停止運算，反之，t=t+1 並返回 Step 1。

除此之外，研究者亦可依自身需求對 SSO 演算法進行修改，更衍伸出 ISSO、ESSO 等演算法。

第二章、研究方法

2.1 研究流程

本研究首先進行概念發想以確立研究主題，之後主要分別進行網頁、爬蟲及 SSO 程式撰寫。其中，SSO 程式撰寫又分為對 SSO 本身程式撰寫及依自身需求開發適合度函數此 2 部分。最後分別測試個子功能正常後再將其整合。整體研究步驟如下圖 2 所示。

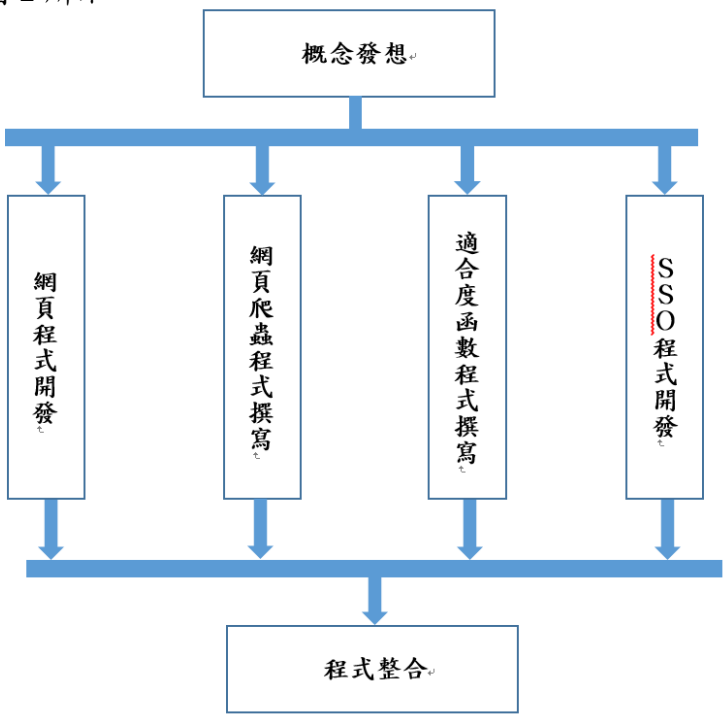

圖 2.研究流程圖

2.2 程式開發

關於本研究程式開發語言，在爬蟲程式、SSO 程式及適合度函數部分以 Python 撰寫，網頁部分則是結合 html、php 程式並配合 sql 程式碼以串接資料庫運行。

2.2.1 SSO 及適合度函數

於程式開發中之 SSO 程式碼部分，為了衡量雙機排程結果的表現，故以在 SSO 程式中的適合度函數作為衡量，透過評估每個解的總完工時間(makespan)來判斷此解的表現；在 SSO 程式運行中，為模擬實際情況中待排程的工作，故設定程式開始運行時會隨機產生 190 個加工時間不同，且須依序通過 a、b 兩機器的待排程任務。後續程式程碼則是依據上述所提及之 SSO 演算法運作步驟進行開發：首先須給定 SSO 程式運行的必要參數，並使其產生一定數量的初始解，以得到 P_i、g_{best} 的初始值，再以迴圈的方式實現 SSO 之更新機制；當中以適合度函數作為判斷解表現好壞之依據及決定是否替換 P_i、g_{best} 的內容，最後將迴圈運行結束後，g_{best} 最終值取出作為最後產出的排程結果。但若是面對於每個解中有極為大量變數等情況下，有時候須透過手動調整多個參數，嘗試不同參數組合才可使 SSO 演算法運行出結果更佳的解，本研究以網頁配合爬蟲方式來改善此方面問題。

2.2.2 網頁及爬蟲程式

如 2.2.1 所提及，為了改善使用 SSO 程式時，需嘗試多組參數組合情況下的測試效率，本研究撰寫一網頁系統，使使用者可以在事前預先在網頁上輸入 5 個 SSO 程式運行必備參數:**(1)**運行次數(nrun)、**(2)**迭代數(ngen)，**(3)**解數量(nsol)、**(4)**以 gbest 替換當前解的臨界值(cg)、**(5)**以 P_i 替換當前解的臨界值(cw)。透過事先輸入以上 5 個參數的不同組合，且將這些組合放入資料庫中，再抓出以表格方式顯示於另一網頁上，並以網頁爬蟲之程式碼將這些參數組合一一放進 SSO 程式中自動運行，避免了不斷重複調整參數、運行此一循環，進而提升測試多組參數時的操作效率。

第三章、研究結果

本研究程式運行流程如下圖 3 所示，詳述如下:**(1)**使用者於網頁中輸入多組欲測試之 SSO 參數組合，並以表格方式將所有參數組合顯示於另一網頁。輸入及顯示網頁分別示如圖 4、圖 5。**(2)**進入 Python 編輯器開始執行程式，程式碼會以爬蟲方式抓下網頁上所有參數組合，如圖 6、圖 7 所示。**(3)**以不同參數組合運行 SSO 程式進行雙機排程動作，當中總完工時間(makespan)最小之解作為

最後輸出解。由於程式中設計有在同一參數組合下仍會重複執行一定次數以確保求解品質，所有運行輸出結果皆會顯示於編輯器的控制面板(console)中，執行結果示如圖 8 所示，分別列出重複行次數、參數組合組數、排程結果及總完工時間。

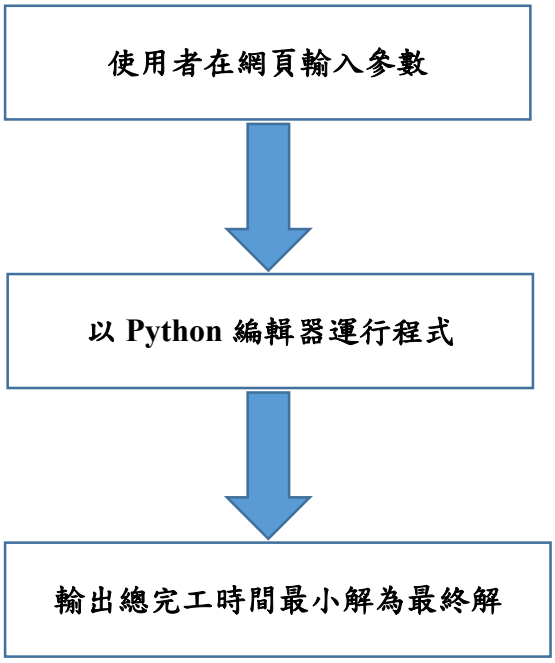

圖 3.研究流程圖

圖 4.輸入參數網頁

參數查看頁面

SSO參數				
nrun	ngen	nsol	cg	cw
1	20	100	0.2	0.4
2	150	20	0.1	0.6

圖 5. 顯示參數網頁

```
def get_contents(ulist,rurl):
 soup = BeautifulSoup(rurl,'lxml')
 trs = soup.find_all('tr')
 for tr in trs:
  ui = []
  for td in tr:
   ui.append(td.string)
  ulist.append(ui)
```

圖 6. 抓取爬蟲程式碼

```
[['1', '50', '70', '0.2', '0.7'], ['3', '50', '70', '0.1', '0.4']]
```

圖 7. 爬蟲抓取結果

```
the  1 times , 1 parameter combination's best solition is:
 [144.  30.  57.  13. 139. 101. 184.   9.  42. 185.  82.  88.  65.  36.
 189.  93.  46.  34. 109.  53.  97. 148.  17. 116. 160. 151. 107.  14.
  18. 120. 122. 172. 149.  86.  66. 140.  68. 169. 146.  72. 187. 156.
 166. 117.  23. 186.  22.  78.   1. 178.  58. 182. 167. 158. 173. 147.
 125.  98.   2. 118.  77. 129.  90.  52. 177.  69.   8.  89.  99.  96.
   3. 141. 121.  39. 174.  33. 164.  70. 126. 104. 123. 181. 136. 110.
  43.  64. 132. 155. 127. 114. 176.  49. 179. 188. 152.  80. 180. 171.
  38. 159.  60. 143.  21. 103. 150.  11.  45. 102. 175.  56. 113.  40.
  59. 130.   0.  32.  27.  94. 133. 108. 124. 157.  41.  47.  10. 165.
  12. 170.  91.  61. 106.  20.  44.  75.  84.  48.  73.  62.   5.  54.
  76. 119.  95.  50.  71.  19.  37. 105. 153.  67.  74.   6. 162.  15.
 168.  29. 100.  51. 135. 163.  87. 145. 161.  92.  35.  81. 112.  31.
  24.  85.  26.  79. 154. 131. 115. 142.  25. 137. 134.  16. 111.   4.
  55.  63.  28.   7. 128.  83. 183. 138.]
the  1 times , 1 parameter combination's  best solition makespane is:
 4818.6149502876415
the 1 times, 2 parameter combination's best solition  number is:
 2
the  1 times , 2 parameter combination's best solition is:
 [ 16.  19.  78.  18.  57.  14. 175.  12. 160. 185. 140.  72.  70.  88.
  42.  48. 110. 100.  27. 142. 183. 139. 166.  85.  58.  20. 162.  23.
 148.  67. 187.  31.  25.  56. 132. 157.  44. 151. 136.  86.  33. 123.
  91.   0.  45.   5. 167.  81. 133.  39. 164.  43. 102.   7. 128.  98.
   8.  53.  75. 154. 114. 111.  60. 176. 189.   6.   2.  68.  59.   1.
 126. 181. 112. 138. 178. 105.  17.  66. 103.  47.  22.  38.  40. 150.
 155.  35. 141.  28.  36. 144.  82. 173.  50.  93.  51.  95. 106.  61.
 130.  69. 171.  24.  11. 165. 153. 161.  13. 108. 146. 145. 124.  99.
 188.  34.  55. 125. 118.  46.  71.  74.  32.  30. 131. 159. 182.  80.
  26.  49.  89.  94.   4.  52. 116.  37.  29.  96. 169. 122. 172.  21.
 184. 135.  64. 113. 186.  79.  92.  65. 143.  84.   9. 101.  62.  77.
 119. 174.  87. 147. 163.  54. 104. 158.   3. 117.  63.  41. 129.  83.
 177. 156. 152. 180.  76. 170.  15. 168. 107.  97.  10. 134.  73. 121.
 115. 109. 149. 120. 179. 137.  90. 127.]
```

圖 8.程式運行結果

綜合上述，經由本研究所架構之系統，可產生一便捷方式進行雙機排程，具有以下優點:

1. 僅需花費數秒便可產生大量任務之排程結果。
2. 不需要重複手動測試不同參數組合的排程結果，僅需運行一次即可。
3. 結合橫跨 Python 程式、網頁及資料庫的運用。
4. 程式執行檔所占電腦儲存空間極小，不若許多程式會造成儲存空間負擔。

在未來，此系統仍有以下數點精進方向:

1. 使程式執行時僅需透過網頁操作及顯示結果，不需要再透過 Python 編輯器進行操做。
2. 可以結合實際接單系統進行排程。
3. 擴展使用範疇，不僅限於雙機排程使用。
4. 結合手機平台應用，讓使用者可於不同平台、裝置進行操作。

參考文獻

[1]Zadeh, Lotfi A., "Fuzzy Logic Neural Networks, and Soft Computing," *Communication of the ACM*, March 1994, Vol. 37, No.3, pages 77-84.

[2] Chyh-Ming Lai, W.-C. Y., Yen-Cheng Huang. (2017). Entropic simplified swarm optimization for the task assignment problem. Applied Soft Computing, 115-127.

[3] Peng-Yeng Yin, S.-S. Y., Pei-Pei Wang, Yi-Te Wang. (2006). A hybrid particle swarm optimization algorithm for optimal task assignment in distributed systems. Computer Standards & Interfaces, 441-450.

[4] Y.Daniel Liang(2016)。**Python程式設計入門指南。**台北市:碁峯。

[5] 李鈞(2017)。**生產管理進階。**自行出版