航　運　季　刊
第二十七卷　第一期
民國 107 年 3 月
頁 87～頁 111

Maritime Quarterly
Vol. 27 No. 1
March 2018
pp. 87–111

應用二階式啟發式演算法於動態及連續型船席指派

A Two-stage Heuristic for Solving the Dynamic and Continuous Berth Allocation Problem

徐賢斌 (*Hsien-Pin Hsu*)❶*、王曉晴 (*Shao-Chin Wang*)❷、陳紋惠 (*Wen-Hui Chen*)❷、歐陽亭 (*Yang-Ting Ou*)❷

摘要

國與國之貿易以海洋運輸為主，而海洋運輸則有賴港埠及碼頭基礎建設。如何提升碼頭效率及發揮港埠功能是一個重要的議題。本研究之重點在解決「動態及連續型」船席指派問題。經由船席之有效利用，以發揮港埠功能。此類型問題中，會同時考慮到達船以及在途船，並將碼頭視為連續泊線以供這些船舶停靠。本文中，提出一個二階式啟發式演算法來解決此類型問題。首先，產生船舶的隨機資料 (包含船舶長度、作業時間、預計到達時間及期望停靠點) 之後，在第一階段中本演算法會依據船舶預計到達時間，產生一個船舶置入順序。隨後，在第二階段中，本演算法會依序將船舶依其期望停靠點以及預計到達時間點置入時空圖中，如置入後發現重疊情形，則在考量偏移成本及相關限制 (船體、碼頭) 之下，進行船舶偏移以化解衝突，來產生最佳或近似最佳之可行解。本演算法以最小化總成本為目標。總成本則包含等待時間以及作業時間成本。本研究中，以 java 程式語言實作此演算法，並進行實驗，再經過對數據分析後，以探討解的品質以及該方法之可行性。實驗結果顯示，在目標函數下該演算法可求得最佳或近似最佳解，且其運算時間可應付實際作業需求。

關鍵字： 海洋運輸、動態及連續船席指派、二階式啟發式演算法

❶* 通訊作者，國立高雄科技大學供應鏈管理系副教授；聯絡地址：81157 高雄市楠梓區海專路 142 號，國立高雄科技大學供應鏈管理系；E-mail: hphsu@nkust.edu.tw。
❷ 國立高雄科技大學供應鏈管理系。

Abstract

Maritime transport is the major means for international trading, which depends heavily on port infrastructure, including berth. How to best utilize the berths of a port is an essential issue. This research focuses on dealing with one specific type of berth allocation problem (BAP)－the dynamic and continuous berth allocation problem (DCBAP)－in which both arrived and incoming ships are considered and a quay is entirely used as a continuous line to accommodate the calling ships. A two-stage heuristic has been proposed to solve the DCBAP. At the first stage, with the randomly generated ship data, the heuristic creates a ship placement sequence based on the estimated times of arrival (ETAs) the calling ships. At the second stage, it places ships into a time-space diagram one by one according to their desired berthing locations and ETAs. While placing a ship and this causes an overlap of ships the heuristic moves the ship to resolve the overlap based one the estimated costs of three different moving direction (i.e. up, down or right). Then, the least-cost moving direction is the first priority used to resolve the overlap. The aim is to minimize the total cost consisting of sub-costs that includes waiting and handling. Java language was used to implement this heuristic. Our experimental results showed that the heuristic was able to find the optimal/near-optimal solution with reasonable time.

Keywords: Maritime transport, Dynamic and continuous berth allocation problem, Two-stage heuristic.

壹、緒論

1.1 研究背景與動機

海洋運輸是國與國之間貿易的重要運輸方式。相較於空運，海洋運輸具有低成本及大量運輸之優勢。近年來，海運之發展與進步，眾所周知。海運的持續成長使得港口及碼頭資源之供給日漸窘迫，提升船邊作業 (seaside operations) 之效率，以維持港口之競爭力，有其重要性。學者 Song and Yeo (2004) 指出，有許多可行策略可應用於維持港口優勢，例如擴增港埠設施、現代化碼頭裝卸設備、專屬碼頭之承租、匯集航線網路、提供優惠費率以及強化效率等。上述策略中，硬體設備擴充

建置需時較長，而強化港口之效率則短期可見成效。

國際貿易有助於促進一個國家經濟之繁榮。其衍生之貨物流通，則可由海運或空運來達成。兩者相較，空運速度快，但運量小而成本高，故主要以運送高單價商品為主。反之，海運速度慢，但運量大而單位運輸成本低，故主要用來運送低單價及大量貨物。基於成本考量，國際貿易仍以海運為主。近年來，全球經濟雖有起伏，但海運之運量 (尤其是貨櫃運輸) 維持成長，致使世界上許多港口面臨壅塞問題。如何解決此項問題，並在激烈競爭的環境下來提升效率並降低成本，是各個港口所面臨重要的生存議題 (陳儀安，2011)。

海洋運輸又以貨櫃運輸為重，徐賢斌 (2014) 指出，貨櫃碼頭包含以下三種主要作業：船邊作業、儲區作業及陸上作業。此三者作業良窳會直接影響碼頭之整體效率。三者之中，尤以船邊作業之影響最大，因為其使用到碼頭兩項稀有資源，亦即船席及橋式起重機。因此，提升碼頭船邊作業之效率，不僅可以縮短船舶停靠碼頭之時間，舒緩碼頭壅塞，更可改善碼頭之整體作業績效。此外，黃獻治 (2009) 指出貨櫃碼頭的裝卸設備 (例如岸肩橋式起重機、跨載機與堆高機等) 之密切配合，可提升港口的裝卸效率。近年來，貨櫃運輸的高效率化促使貨櫃運輸的持續成長，其重要性更有增無減。但貨櫃運輸有賴貨櫃碼頭，其為國家的重要資產及基礎建設，是國內市場和國際市場聯繫的重要橋樑，其效率更是整體供應鏈成功的關鍵 (Narges et al., 2012)。有鑒於此，本研究聚焦於貨櫃碼頭作業之改善。

貨櫃碼頭船邊作業的問題包含船席指派問題 (Berth Allocation Problem, BAP)、橋式起重機問題 (Quay Crane Assignment Problem, QCAP) 以及橋式起重機排程問題 (Quay Crane Scheduling Problem, QCSP)(徐賢斌、王冠晴，2016；徐賢斌，2017)。以上這些作業均與船席或橋式起重機有關。韋又琳 (2013) 指出，海洋運輸過程中包含不同的環節作業，而各環節作業的良窳會影響整體運輸績效。碼頭船席的有效利用，其效益包含降低船舶等待時間、改善港口壅塞、並提高碼頭運轉效率。另外，楊逢新 (2011) 指出，船席指派是港務作業之最前線且最重要的一環，其包含兩個重要考慮的構面。一為最小化所有船舶服務時間，另一為服務水準之維繫 (Imai et al., 2007)。由於船席為稀少資源，短期難以擴建，學者紛紛強調其重要性，因此有效的船席利用吸引本研究之關注。

船席指派問題，依碼頭空間之使用情形，可分為「離散型」與「連續型」。「離散型船席指派問題」將碼頭泊線分割為固定大小之席位以供船舶停靠，一次只能停靠一艘船舶。但由於可能導致部分空間之浪費，故此種船席利用方式漸少。「連續型船席指派問題」則將碼頭視為連續泊線，可

讓多艘船舶同時停靠，可較充分使用船席空間，因此，此類型船席研究漸增 (楊逢新，2011)。本研究亦專注於「連續型船席指派問題」。再者，依船舶是否已抵達港口，船席指派問題可再細分為「靜態型」與「動態型」。「靜態型船席指派問題」只考慮已到港口之待泊船，而「動態型」船席指派問題則會再考慮在途船。兩者之中，以「動態型」船席指派問題較為實務，故本研究以「動態型」船席指派問題為主。如同時考慮兩者因素，本研究之問題可界定為「動態及連續型」船席指派問題。

求解船席指派問題的方法主要包含數學模式以及啟發式演算法。數學模式一般使用整數規劃或混合整數規劃法。整數規劃法中之決策變數其值域為 0 或 1 之整數；而混合整數規劃之決策變數其值域則可包含非 0 或 1 之整數。數學模式一般用來求取最佳解，但由於在求解實務問題時常面臨時間太久 (時間將隨問題規模之變大而呈指數成長)，故難以符合實務應用。目前實務應用上仍多仰賴啟發式演算法來取得可行解。

啟發式演算法可概分為自然式啟發式 (Nature-inspired) 演算法與非自然式啟發式 (Non-nature-inspired) 演算法兩種 (Zang et al., 2010)。自然式啟發式仿效大自然現象，發展演算理論。粒子群演算法、基因演算法、螞蟻演算法、蜂群演算法等為此類。非自然式啟發式 (Non-nature-inspired) 演算法則主要是依據學者經驗而得，可快速解決實務問題。本研究旨在發展一個非自然式之啟發式演算法，名為二階式啟發式演算法。該演算法首先將一週內預計到達之船舶依其 ETA 及期望停靠點轉換為時空圖中之座標值，並將船舶依序置入圖中時，利用條件判斷式找出相互重疊的船舶 (船舶在時空圖中所占之面積重疊)，進而化解重疊船舶以產生可行解。本研究中假設時空圖中之縱座標代表空間而橫座標代表時間軸。在化解船舶衝突可將船舶向上、向下或向右偏移。向上及向下偏移為化解空間衝突，向右偏移為化解時間衝突。這些偏移都會導致額外之作業成本或等待成本，可經由公式計算，經過成本比較後，以選擇船舶挪動方向。本研究提出之演算法選擇往成本較小之方向來偏移船舶，則最後所得的解為近似最佳解或最佳解。

1.2 研究目的

影響碼頭營運之因素包含如下：碼頭大小、碼頭吞吐量、碼頭船席設計位置、水深、長度等 (韋又琳，2013)。這些因素中，碼頭硬體設施之改善需時較長，若以管理之軟體手段來提升碼頭效率，則短期可見成效。因此，本研究之主要目的著重在發展啟發式演算法來協助船席指派，經由船席之有效利用以發揮港埠功能。本研究之目的列示如下：

1. 探討港埠船邊作業之船席指派規劃問

題。

2. 發展啟發式演算法來進行船席指派，以充分應用港埠稀有資源。

本研究中其他節次安排如下。第二節中進行相關文獻之回顧；第三小節中進行數學模式之建立；第四小節中提出二階式啟發式演算法；第五小節中提供數值範例；第六小節中則進行結論及建議。

貳、文獻回顧

2.1 船席指派問題類型

船席指派問題旨在安排靠泊碼頭位置及時段，以完成貨物裝卸作業。依船舶是否已抵達港口，船席指派問題可再細分為離散型與連續型，詳述如下：

1. **離散型船席指派問題 (Discrete Berth Allocation Problem, DBAP)**：碼頭之靠泊空間分割為固定大小，同一時間只能停入一艘船舶。此船席利用方式，若船舶過大，則無法停入船席；若船舶過小，則造成船席部分空間浪費。
2. **連續型船席指派問題 (Continuous Berth Allocation Problem, CBAP)**：由於船邊設施及其整體建置與離散型船席不同，其作業方式相較之下也較為彈性，其基礎是建立於直線型的船席空間上，同時間可服務多艘船舶，不但可減少空間浪費的問題，也可提升調派效率，故有利實務應用。

依船舶進港方式分為靜態模型與動態模型：

1. **靜態型船席指派問題 (Static Berth Allocation Problem, SBAP)**：船席指派對象為已經到港之船舶，所有船舶皆無抵達先後順序之問題，故調配上較為簡易，容易求得最佳解，但是較不符合實務之運作。
2. **動態型船席指派問題 (Dynamic Berth Allocation Problem, DBAP)**：船席指派對象亦包含在途船舶，須推估其到港之時間，並進行船席指派。由於必須在船舶抵達前完成指派作業，故此調派問題較為困難，但較符合實務狀況。此類問題近年來吸引較多學者投入研究。

本研究主要是探討「動態及連續型船席指派問題」(Dynamic Berth Allocation Problem of Continuous Berth)。林岱暘 (2011) 指出，連續型船席為近年來港口普遍的使用方式，並且將船舶視為動態到達較符合實務。因此本研究以此類問題為主，並於下節中進行相關文獻的回顧。

2.2 動態連續型船席指派之研究

Bierwirth and Meisel (2010) 對近年來有關於船邊作業之研究進行文獻回顧，包含了上節中所描述的各類型的船席指派問題。在另一篇文獻回顧中，Bierwirth and Meisel (2015) 更進一步分析了船邊作業規

劃所使用到的方法。該文獻中顯示，動態及連續型船席指派規劃的演算法中，基因演算法 (Genetic Algorithm, GA) 是最為廣泛使用的方法，其次為混合整數線性規劃 (Mixed Integer Linear Programming, MILP) 及貪婪隨機調適搜尋法 (Greedy Randomized Adaptive Search Procedure, GRASP)。模擬退火法 (Simulated Annealing, SA)、禁忌搜尋演算法 (Tabu Search, TS)。其他簡單式啟發式演算法 (Rule Based Heuristics)，例如先到先服務 (First Come First Serve, FCFS)，最小處理時間優先 (Shortest Processing Time, SPT) 也曾被使用來應付動態連續型船席指派問題。例如，Pratap et al. (2018) 發展決策支援系統，其包含作業法則用以支援決策者進行決策。

Lim (1998) 將連續型船席指派問題視為一個二維布料剪裁問題，並發展一個啟發式演算法來最小化船舶所需之碼頭長度。之後，Park and Kim (2002) 以及 Kim and Moon (2003) 則相繼提出次梯度法 (Subgradient Methods) 與模擬退火法 (Simulated Annealing, SA) 來進行船席指派，並以最小化船舶滯港時間為目標。Guan and Cheung (2004) 則提出搜尋樹 (Tree Search) 法來進行連續型船席指派，其目標在最小化船舶之靠泊總時間。在解決動態及連續型船席指派問題時，許多研究假設船舶之處理時間是固定不變的。

繼 2001 年完成「離散型船席指派」問題之探討，Imai et al. (2005) 轉進研究「連續型船席指派」問題。在假設船舶停靠的位置對船舶的處理時間有影響後，他們提出一個啟發式演算法，來最小化船舶完成時間。第一階段中，該演算法先產生船舶停靠位置的初始解，並在第二階段中將船舶重新定位，以產生可行解。Wang and Lim (2007) 則發展隨機探照搜尋法 (Stochastic Beam Search, SBS) 來求得船席指派之最小總成本解。最後他們發現該搜尋法較傳統的探照搜尋法 (Beam Search) 更為精確及有效，且該方法可以處理高達 400 艘船舶。Lee and Chen (2009) 則提出一個鄰域搜尋法 (Neighborhood Search) 來解決動態連續型船席指派問題。該研究中並考慮了先到先服務原則 (FCFS) 以及船舶的間距因素。他們指出好的決策應該考量到此些因素。

由上述之研究可知，在處理船席指派問題時，許多學者將目標放在減少船舶靠泊的時間 (成本) 來最大化碼頭之使用效率。此儼然為近年的研究趨勢。

2.3 船席指派成本因素

船席指派之良窳會反映在船舶的在港時間，其可細分為兩部分。一為船舶等待碼頭的時間 (waiting time) 另一為船舶停靠碼頭的時間 (berthing time)。船舶在港時間成本即為船舶等待時間成本與船舶處理時間成本之總合 (UNCTAD, 1985)。所有船

舶的在港時間成本可以反映出一個船席指派規劃之良窳，其可以用時間成本乘以各船的在港時間 (等待時間與處理時間之和) 來估算，如公式 (1) 所示。一或數個碼頭所有靠港船舶的在港時間成本亦可以相同方式估算。

所有船舶之在港時間成本

$$= \sum_{j=1}^{N} (c_1 W_j + c_2 H_j) \quad (1)$$

W_j：船舶 j 之等待時間成本

H_j：船舶 j 處理時間成本

c_1：船舶等待時間成本

c_2：船舶處理時間成本

2.3.1 等待時間成本

船舶等待時間是航商評估港埠服務的一項重要指標。等候時間長則營運成本高。因此，減少船舶等候時間為航商與港口管理機關之關注議題。過長的船舶等候時間，會影響船公司船期安排規劃，導致船公司需要增加船舶數量以維持定期航班服務，甚至將船舶改停到其他港口，此都會增加航商成本。王大瑾等人 (2012) 指出，若增加港埠之船席來減少等候時間，可能導致船席平均使用率降低。亦即，如果碼頭未經濟使用，碼頭之營運成本高於經濟使用下的成本，則這些高出之成本除有損港埠經營者之競爭力外，最終將轉嫁到航商身上。

2.3.2 處理時間成本

航商考量其船舶之最佳裝卸位置及裝卸機具之位置，會產生一個最佳停靠點，亦即期望停靠點。當船舶愈接近其期望停靠點時，可獲致愈短的船舶處理時間，亦即降低碼頭作業成本。若偏離此點，貨櫃拖車需要更多的搬運時間，因此增加船舶之處理時間成本。

根據 Meisel et al. (2009) 之研究，本文以公式 (2) 來估算船舶 j 當其偏離期望停靠點時，所需之處理時間。

$$H_j = h_j + \left(\left(\left(\frac{\Delta b_j}{100} \right) \cdot m_j \right) / 60 \right) \quad (2)$$

h_j：船舶 j 在期望停靠點之所需處理時間 (時間單位為小時)

Δb_j：船舶 j 偏離期望停靠點之距離 (距離單位為公尺)

m_j：拖車每移動 100 公尺所增加之時間 (時間單位為分)

估計船舶一百公尺對拖車之處理時間約增加 2 分鐘，因此本研究設定 $m_j = 2$。此外，在 Meisel et al. (2009) 之研究中設定橋式起重機每小時需時間成本為 1,000 美金，因此本研究以每小時 1,000 美金作為偏移位置之處理成本。

不同於過去之研究，本研究利用「先到先服務」及「往最小成本方向移動」之原則，發展出一個二階式演算法。該演算法可以在合理之時間內找到最佳或最佳近似解，以應付實務之需求。

參、數學模式

雖然整數規劃數學模式及混合整數規劃模式不適合用來求解大型實務問題，但數學模式仍有助瞭解問題之本質。過去許多研究雖然發展演算法，但仍然先建立數學模式。同理，本研究中首先對「動態及連續型」船席指派問題建立其數學模式。

3.1　假設與參數設定

在建立數學模式之前，首先建立一些假設如下：

1. 以先到先服務為原則。
2. 假設碼頭一開始皆為閒置狀態。
3. 假設碼頭水深足夠，船舶靠泊無虞。
4. 不考慮橋式起重機指派問題。
5. 船舶作業中，不再變動其停泊位置。
6. 船舶間之最小安全距離已考慮在船舶長度中。
7. 拖車拖運貨櫃至船邊時間，納入船舶之整體作業時間。
8. 每艘船都有一個期望停靠點。

下列表 1 定義了數學模式中所使用到的指標、參數及決策變數。

3.2　數學模式建立

以下為船席指派所建構之數學模式：

$$Min\ Z = \sum_{j=1}^{N}(c_1\Delta W_j + c_2\Delta H_j) \qquad (3)$$

表 1　指標、參數及變數之定義

指標	
j	第 j 條船舶
k	第 k 條船舶
i	第 i 順位
參數	
L	碼頭長度 (公尺)
T	規劃總時間 (小時)
N	船舶總數
a_j	船舶 j 到達碼頭時間
d_j	船舶 j 的期望停靠點
H_j	船舶 j 靠泊期望點之處理時間
A_j	船舶 j 之實際處理時間
l_j	船舶 j 的長度
c_1	船舶等待時間成本 (1,000 美金 / 小時)
c_2	船舶處理時間成本 (1,000 美金 / 小時)
決策變數	
t_j	船舶 j 的規劃靠泊時間
b_j	船舶 j 的規劃靠泊位置

S.t.

$$l_j \le L \quad \forall 1 \le j \le N \qquad (4)$$

$$\Delta W_j = t_j - a_j \quad \forall 1 \le j \le N \qquad (5)$$

$$\Delta W_j \ge 0 \quad \forall 1 \le j \le N \qquad (6)$$

$$\Delta H_j = A_j - H_j \ge 0 \quad \forall 1 \le j \le N \qquad (7)$$

$$\Delta H_j \ge 0 \quad \forall 1 \le j \le N \qquad (8)$$

$$b_j \le L - l_j \quad \forall 1 \le j \le N \qquad (9)$$

$$L_j \ge 0 \quad \forall 1 \le j \le N \qquad (10)$$

公式 (3) 為目標函數，旨在最小化所有船舶偏離其最佳停靠點及 ETA 時所增加之處理及等待成本。公式 (4) 表示為任一船舶長度需小於或等於碼頭長度。公式

(5) 用以計算船舶之增加的等待時間。公式 (6) 要求任一船舶之增加的等待時間必須大於或等於 0。公式 (7) 計算船舶之增加的處理時間。公式 (8) 要求增加的船舶之處理時間需大於或等於 0。公式 (9) 及 (10) 限定船舶之有效停靠點的範圍。

肆、二階式啟發式演算法

4.1 最佳解或近似最佳解

依據 3.2 節，本研究之數學模式的目標函數旨在最小化所有船舶所增加之等待時間成本及處理時間成本。當每艘船舶都安排在其預計到達時間點以及期望停靠點時，並且都不互相重疊，此時目標函數之總成本為零，故為最佳解。但是如果船舶之間互有時空衝突 (重疊)，則必須化解重疊以產生可行解。本演算法之基本做法是，將船舶往最小的成本方向偏離其最佳時空點，故最後所得的解可視為最佳解或近似最佳解 (因為在最佳解附近找尋可行解)。為化解重疊，做法上可以將船舶偏移其預計到達時間點 (向右移動) 或期望停靠點 (船舶向上或向下移動)。在考量三個不同之移動方向所產生之成本，本演算法將船舶往可行且成本最小之方向移動，以求得最佳或近似最佳解。

4.2 二階段之詳述

本研究提出之啟發式演算法主要包含兩階段。第一階段在產生一個順序將船舶置入時空圖中；第二階段重點在如何判斷船舶間重疊，並化解衝突 (重疊) 船舶。

4.2.1 第一階段

如 3.1 節所述，本研究假設船舶以先到先服務為原則。故於第一階段中，依到訪船舶之 ETA 由小到大產生一個將船舶置入時空圖之順序。於第二階段中，則將船舶依此順序一一置入時空圖中。此方式預期可以降低船舶之等待時間 (成本)，並簡化船舶重疊化解之過程 (若一次全部放入時空圖中，再化解重疊，過程會過於複雜)。在依據 ETA 順序將船舶一一安排進入時空圖時，可立刻判斷是否與之前排入的船舶重疊，並立刻化解，此可簡化處理過程。

4.2.2 第二階段

1. 判斷重疊船舶

要判斷兩船舶是否時空重疊，其原理如同於判斷兩矩形是否相交。若相交，則代表船舶在時空圖中有衝突，要化解衝突則必須偏移船舶位置或增加船舶等待時間；若不相交，則代表船舶之間並無衝突，故為可行解，故不需要再進行船舶靠泊之空間或時間之偏移。

要判斷船舶是否重疊可利用條件判斷式。本研究發現，如圖 1 所示，可利用兩矩形 (船舶) 之左下角點 a 與右上角座標 c 之關係式，即可判斷出兩船舶是否重疊。該圖中 X 軸代表時間軸，Y 軸代表空間 (碼頭長度)。

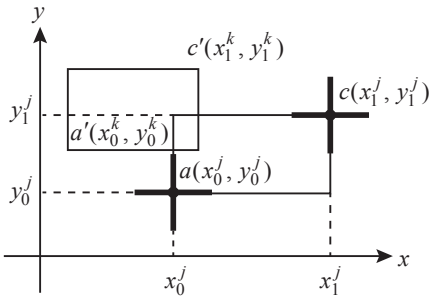

圖 1　船舶座標概念

此概念是來自於座標象限，當以船舶 j 的 c 點當作座標原點時，若 j 與 k 兩船舶重疊，則船舶 k 點 a' 必定位於船舶 j 的點 c 之第三象限 (−, −) 時，且船舶 k 的點 c' 必定位於船舶 j 點 a 之第一象限 (+, +)。依此，本研究利用判斷式 (11) ~ (14) 來判斷兩船舶是否重疊 (當此四條件式同時成立時則兩船舶重疊；否則不互相重疊)。(11) ~ (14) 中下標之 0 代表為船舶左下角點座標，而下標 1 代表為船舶右上角點座標。

$$x_0^j < x_1^k \tag{11}$$

$$y_0^j < y_1^k \tag{12}$$

$$x_0^k < x_1^j \tag{13}$$

$$y_0^k < y_1^j \tag{14}$$

2. 挪動重疊船舶

船舶在時空圖中可以經由向上、向下及向右三種方向之挪動來化解時空衝突。向上及向下挪動為化解空間衝突，但因偏離船舶之期望停靠點 d_j 因此會增加處理成本。向右挪動則可化解時間衝突，但會延後船舶靠泊時間，使其偏離預計到達時間 (ETA) 而產生等待時間成本。本演算法會預先估算船舶往各方向挪動所增加之成本，並選擇往最小成本方向挪動。公式 (2) 為計算偏離期望停靠點時，所增加之處理時間。公式 (15) 用來估計向上移動所增加之處理時間 (移動距離每 100 公尺增加 2 分鐘)，再乘上單位處理時間成本 c_2 = 1,000 美金 / 小時。公式 (16) 用來估算向下移動所增加之處理時間 (100 公尺 /2 分鐘)，再乘上 c_2。公式 (17) 為計算向右移動所增加之等待時間，再乘以每小時等待時間成本 c_1 = 1,000 美金 / 小時，則可得到偏移成本。

$$\Delta H_j \times c_2 = \left(\frac{2 \mid d_j - y_0^k \mid}{100 \times 60} \right) \times 1000 \tag{15}$$

$$\Delta H_j \times c_2 = \left(\frac{2 \mid y_0^k - (d_j + l_j) \mid}{100 \times 60} \right) \times 1000 \tag{16}$$

$$\Delta W_j \times c_1 = (\mid x_0^j - x_1^k \mid) \times 1000 \tag{17}$$

經評估並確定移動方向後，可利用公式 (18)、(19) 或 (20) 來重新修正船舶在時空圖中之角點 a 及 c 的座標值。公式 (18) 計算向上移動時 y_0^j 及 y_1^j 需加上的移動距離，以及 x_1^j 需加上所增加之處理時間。公式 (19) 計算向下移動時 y_0^j 及 y_1^j 需減去的移動距離，以及 x_1^j 需加上所增加之處理時間。公式 (20) 計算向右移動時 x_0^j 及 x_1^j 需加上的移動時間。

$$a:(x_0^j,\ j_0^j+|y_0^j-y_0^k|),$$
$$c:\left(x_0^j+h_j+\left(\frac{2\times|d_j-y_0^k|}{100\times60}\right),\ y_1^j+|y_0^j-y_0^k|\right) \quad (18)$$

$$a:(x_0^j,\ j_0^j-|y_1^j-y_0^k|),$$
$$c:\left(x_0^j+h_j+\left(\frac{2\times|y_0^k-(d_j+l_j)|}{100\times60}\right),\ y_1^j-|y_1^j-y_0^k|\right) \quad (19)$$

$$a:(x_0^j+|x_0^j-x_1^k|,\ y_0^j)$$
$$c:(x_1^j+|x_0^j-x_1^k|,\ y_1^j) \quad (20)$$

4.3 啟發式演算法流程

本研究中所提出之二階式啟發式演算法其流程圖如圖 2 所示。其流程與步驟詳細說明如下：

步驟一：本步驟中先隨機產生 N 條船舶相關資料，包含船舶預計到達時間 a_j、期望停靠點 d_j、船舶全長 l_j 與船舶處理時間 h_j 等，以便於本演算法演算。

步驟二：依先到先服務為原則，將各船舶依其 a_j 產生一個由小到大排列之置入時空圖順序。

步驟三：假設 j 與 k 為船舶置入順序指標。將 $j=1$ 的船舶依角點座標資料置入時空圖。由於是第 1 艘船此時無船舶重疊問題。設定 $k=1$。

步驟四：置入下一艘船舶 j ($j=j+1$)。而船舶 j 的座標則依其預計到達時間 a_j 與期望停靠點 d_j，各分別加上處理時間 h_j 與船舶全長 l_j，得到船舶 j 的 a 及 c 點座標為 $(a_j,\ d_j)$ 及 $(a_j+h_j,\ d_j+l_j)$，將其置入時空圖之中。

步驟五：利用公式 (11)、(12)、(13) 及 (14) 來判斷出船舶 j 是否與已排定之船舶 $j-k$ 重疊 (此時設定 $i=1$; i 為移動方向順位指標)；若無重疊則 $k=k-1$ 進行步驟九；若船舶 j 與 k 重疊則進行步驟六。

步驟六：依公式 (15)、(16) 及 (17) 計算船舶 j 向上、向下及向右之偏移成本。

步驟七：當移動後船舶再次衝突到先前已排除之船舶 k 時，則選擇下一移動順位。因此當船舶 j 移開船舶 k 時之移動順位依增加成本由小到大排序儲存，以便於之後改變移動方向。

步驟八：往下一順位方向挪動船舶 j 來化解與船舶 k 之衝突，並依據公式 (18)、(19) 及 (20) 修正船舶 j 的角點 a 及 c 的座標值。

步驟九：當 $k=j$ 時表示已完成目前置入船舶 j 與之前所有置入船舶是否重疊之判斷，故此時可繼續置入下一艘船舶於時空圖中。否則，必須回到步驟五繼續檢 查前一艘船舶。

步驟十：移動後船舶可能因此再次衝突到

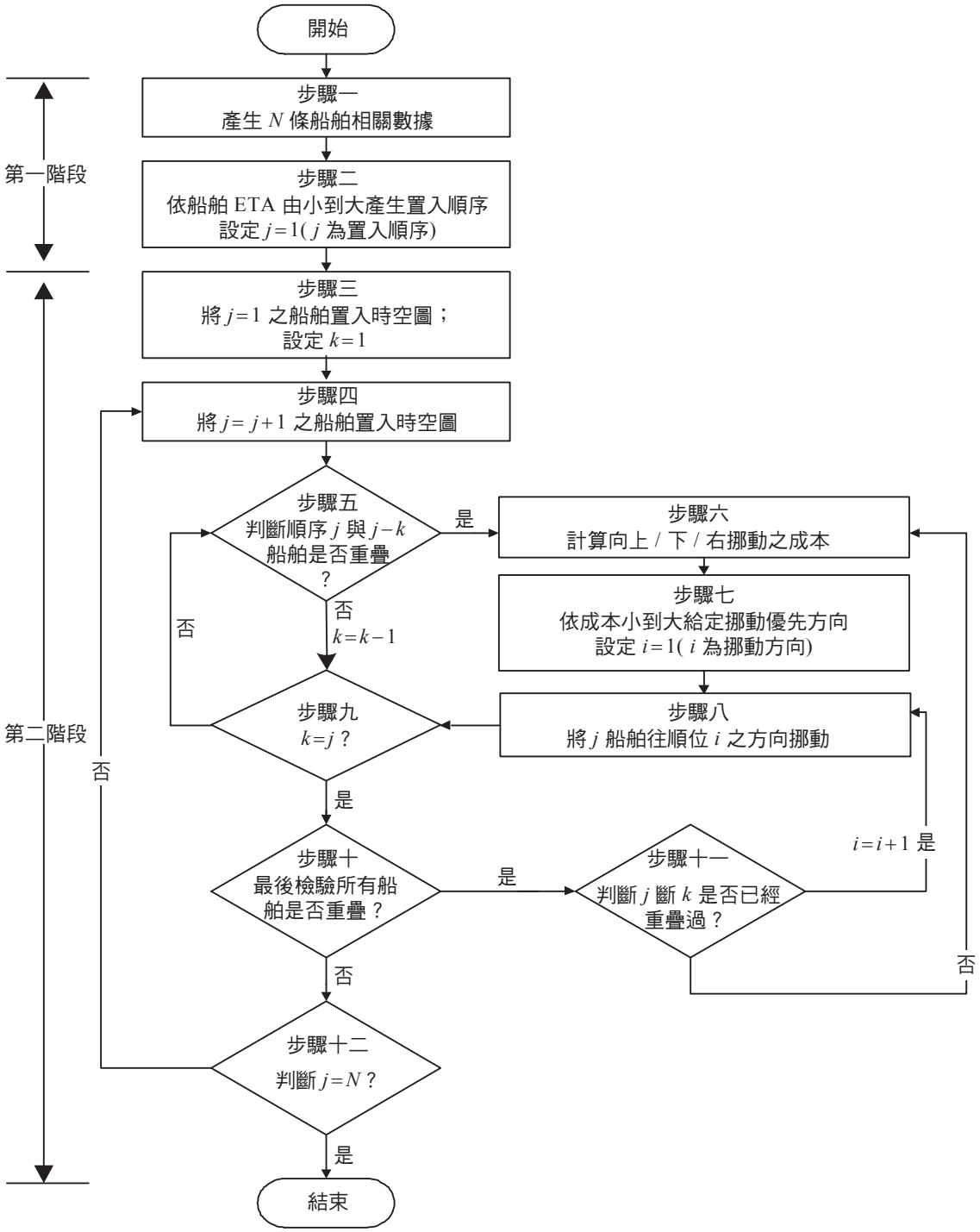

圖 2　二階式啟發式演算法流程圖

先前已排除之船舶 k，因此需判斷船 j 排除衝突後是否還有重疊？如否則進行步驟十二，否則進行步驟十一。

步驟十一：若仍有重疊則需判斷船舶 j 與 $j-k$ 是否曾經移動過，若是則進行步驟八依下一順位方向移動，以避免無限循環，此時 $i=i+1$；否則進行步驟六。

步驟十二：在置入下一艘船舶前需判斷 j 是否 $j=N$（表示是最後一艘船舶）？如是，則結束；否則回步驟四。

4.4 範例說明

本範例中假設時空圖中之縱座標為碼頭空間，其總長度為 1,000 公尺（每一單位為 100 公尺）；而橫軸座標為規劃期間，共 15 小時。假設共有五艘來船（$N=5$），其基本資料如表 2 所示，其中包含隨機產生的預計到達時間（a_j）、期望停靠點（d_j）、船舶全長（l_j）、處理時間（h_j）等資料。如依據表 2 之資料，將船舶依照其 a_j 及 d_j 置入時空圖中，則最終會產生圖 3。但因為有船舶互相重疊之情形，可知圖 3 為不可行解。

表 2 船舶基本資料

	船舶 j				
	1	**2**	**3**	**4**	**5**
a_j	1	3	4	6	10
d_j	2	6	1	7	4
l_j	3	4	3	2	6
h_j	6	2	4	5	1

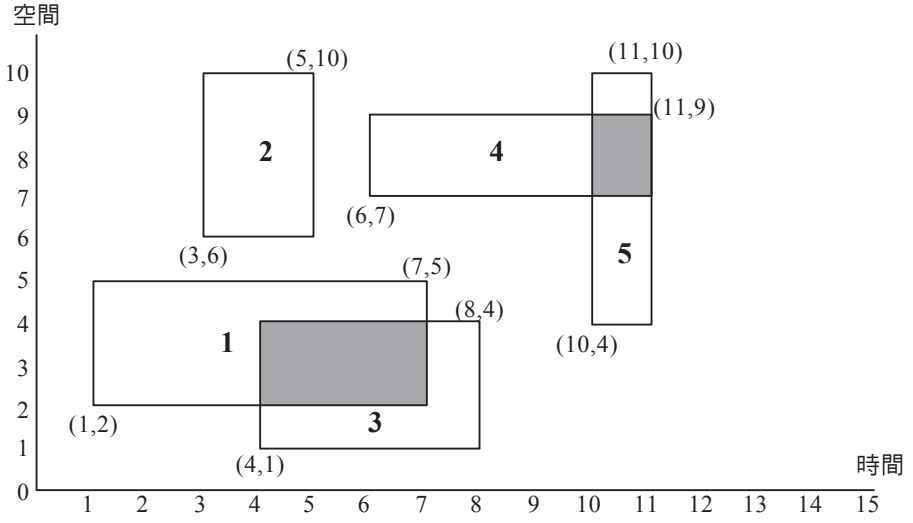

圖 3 第一階段時空圖

以下說明如何應用本啟發式演算法來求得一個最佳或近似最佳解。依循圖 2 所示之演算法流程，在步驟二中，本演算法首先依船舶之 ETA 由小到大，產生一個 1, 2, 3, 4, 5 的船舶置入順序。於步驟三中，將船舶 1 置入時空圖中，此時並無重疊問題。於步驟四中，置入船舶 2，接著在步驟五中，依四項條件式來判斷船舶 2 與 1 是否重疊。因為不同時滿足 (11) 至 (14) 之關係式，故可知該兩艘船舶並不互相重疊。在步驟九中，可知船舶 2 已完成與所有他船隻重疊性之檢查，並於步驟十中再次確認。在步驟十二可知，尚有其他船舶待置入時空圖中，故返回步驟四，繼續置入船舶 3 及進行重疊檢查。由於船舶 3 與 2 存在 $y_0^k > y_1^j$ (6 > 4) 關係，不滿足公式 (14)，故可知兩船並不互相重疊。在步驟九中，由 $k \neq j$ 可知尚有船舶待檢，故回到步驟五，繼續進行船舶 3 與 1 之重疊性檢查。由於該兩船舶同時滿足 (11) 至 (14) 條件式，故知其互相重疊，因此進入步驟六，估算船舶三個方向之挪動成本計算。依公式計算可得表 3。在步驟七中，初步選定向上為第一順位移動方向，因為其成本最少。

接著於步驟八中，將船舶 3 向上偏移，並以公式 (18) 修正其 a 及 c 點角點座標為 (4, 5) 及 (8.033, 8)。於步驟九中，發現已判斷完船舶 3 ($k=j$) 與所有船舶之重疊性檢查。但於步驟十之再次檢查，卻發現船舶 3 向上移動後，會與船舶 2 重疊。在步驟十一中，首先檢查船舶 3 與 2 之前是否產生過重疊，以避免重覆選擇到之前移動的方向而造成無限循環；如有上述情形則應改選下一挪動方案 (亦即 $i=i+1$)，再執行步驟八；否則到步驟六。本範例進入步驟六，在計算船舶 3 偏移船舶 2 之偏移成本後可得表 4。

依表 4，選擇第一順位往上，因不可行，故選擇第二順位將船舶 3 往下移動。

表 3　船舶 3 偏移船舶 1 之偏移成本

方向	移動成本	順位
上	$(\lvert 1-2 \rvert)\times 2 \div 60 = 0.0333 \text{ hr} \times 1{,}000 = 33.3$	1
下	$(\lvert 2-(1+3) \rvert)\times 2 \div 60 = 0.067 \text{ hr} \times 1{,}000 = 67$ (不可行)	2
右	$(\lvert 7-4 \rvert) = 3 \text{ hr} \times 1{,}000 = 3{,}000$	3

表 4　新船舶 3 偏移船舶 2 之偏移成本

方向	移動成本	順位
上	$(\lvert 5-6 \rvert)\times 2 \div 60 = 0.0333 \text{ hr} \times 1{,}000 = 33.3$ (不可行)	1
下	$(\lvert 6-(3+1) \rvert)\times 2 \div 60 = 0.0666 \text{ hr} \times 1{,}000 = 66.6$	2
右	$(\lvert 4-5 \rvert) = 1 \text{ hr} \times 1{,}000 = 1{,}000$	3

並於步驟八中，修正船舶 3 之 a 及 c 的角點座標為 (4, 3) 及 (8.033, 6)，並知道其為可行解 (因為 $1 \le Y \le 10$)。於步驟九中，發現並未全部判斷完畢，故回到步驟五，繼續判斷船舶是否重疊。但在步驟十中，發現向下偏移後會造成船舶 3 與 1 再次重疊，經計算三種方向之偏移成本後可得表 5。如選擇向上偏移 (第 1 順位) 則可能產生無限循環 (之前已選用過此偏移方向)，但若選擇向下偏移 (第 2 順位) 則會產生不可行解 (超出碼頭空間)，故最後選擇向右偏移船舶 3，並於步驟八中修改其 a 及 c 的角點座標為 (7, 3) 及 (11.067, 6)。

於步驟九中可知已判斷完所有先前船隻，故進行步驟十再次無船舶重疊情形。於步驟十二發現船舶 3 非最後一艘，故回到步驟四，繼續置入船舶 4，並繼續船舶重疊之判斷。由於船舶 4 與 3 存在 $y_0^j > y_1^k$ $(7 > 6)$ 關係，不符合條件判斷式 (12)，可知兩船舶不互相重疊。於步驟九中發現尚未判斷完所有船舶，故進入步驟五繼續船舶 4 與 2 之重疊判斷。由於兩船存在 $x_0^j > x_1^k$ $(6 > 5)$ 之關係，不符合條件判斷公式 (10)，可知兩船不相重疊，於步驟九中 $k \neq j$ 可知未完成所有船舶判斷，故再回到步驟五，繼續進行船舶 4 與 1 之重疊判斷。由於該兩船存在 $y_0^j > y_1^k$ $(7 > 5)$ 之關係，不符合條件判斷式 (12)，故知兩船舶不相重疊。步驟九發現已判斷完所有先前船隻後進入步驟十再次確認，發現並無重疊情形。在步驟十二中，可知船舶 4 不是最後一艘船，故回到步驟四繼續置入下一艘船舶 5，並判斷是否與他船重疊。由於船舶 5 與船舶 4 符合四項條件判斷式，可知其互相重疊，於步驟六中計算三種偏移方向所產生之成本，結果如表 6。

由表 6 知，向上為不可行解故船舶 5 向下偏移且為可行解 (因為 $1 \le Y \le 10$)，將其向下偏移並修正其 a 與 c 角點座標為 (10, 1) 及 (11.1, 7)。在步驟九中發現尚

表 5　新船舶 3 偏移船舶 2 之成本

方向	移動成本	順位
上	$(\lvert 1-2 \rvert) \times 2 \div 60 = 0.0333 \text{ hr} \times 1{,}000 = 33.3$ (已重覆)	1
下	$(\lvert 6-(3+1) \rvert) \times 2 \div 60 = 0.0666 \text{ hr} \times 1{,}000 = 66.6$ (不可行)	2
右	$(\lvert 4-7 \rvert) = 3 \text{ hr} \times 1{,}000 = 3{,}000$	3

表 6　船舶 5 偏移船舶 4 之成本

方向	移動成本	順位
上	$(\lvert 4-7 \rvert) \times 2 \div 60 = 0.1 \text{ hr} \times 1{,}000 = 100$ (不可行)	1
下	$(\lvert 7-(4+6) \rvert) \times 2 \div 60 = 0.1 \text{ hr} \times 1{,}000 = 100$	2
右	$(\lvert 10-11 \rvert) = 1 \text{ hr} \times 1{,}000 = 1{,}000$	3

未判斷完所有船舶，故回到步驟五繼續檢查。發現船舶 5 與 3 同時符合四項條件判斷式，故知其互相重疊，於步驟六計算其三種偏移方向之成本，結果如表 7。

表 7 顯示向上移動 (第一順位) 有最低成本，但因其 *c* 點 *y* 座標值為 12 超出時空圖 ($1 \le Y \le 10$)，故不可行。向下移動 (第二順位) 則其 *a* 點 *y* 座標值為 –3 超出時空圖 ($1 \le Y \le 10$)，亦不可行。因此選擇向右移動，並於步驟八中修正船舶 5 之 *a* 及 *c* 角點座標為 (11.067, 1) 及 (12.167, 7)。步驟九顯示需回到步驟五，再進行船舶 5 與 2 之重疊判斷。由於該兩船存在 $x_0^j > x_1^k$ (11.067 > 5) 之關係，不符合條件判斷式 (11)，故知兩船舶不相重疊。依步驟九，進入步驟五中再進行船舶 5 與 1 之重疊判斷。由於船舶 5 與 1 存在 $x_0^j > x_1^k$ (11.067 > 7) 之關係，不符合條件判斷式 (11)，故知兩船不相重疊。當完成所有船舶之判斷，進入步驟十再次確認。最後於步驟十二中，可知船舶 5 為最後一艘置入船舶，且所有船舶皆無重疊情形，故結束作業。最終結果如圖 4 所示。

表 7　新船舶 5 偏移船舶 3 之成本

方向	移動成本	順位
上	$(\lvert 4-1 \rvert) \times 2 \div 60 = 0.1 \text{ hr} \times 1{,}000 = 100$ (不可行)	1
下	$(\lvert 3-(4+6) \rvert) \times 2 \div 60 = 0.583 \text{ hr} \times 1{,}000 = 583$ (不可行)	2
右	$(\lvert 10-11.067 \rvert) = 1.067 \text{ hr} \times 1{,}000 = 1{,}067$	3

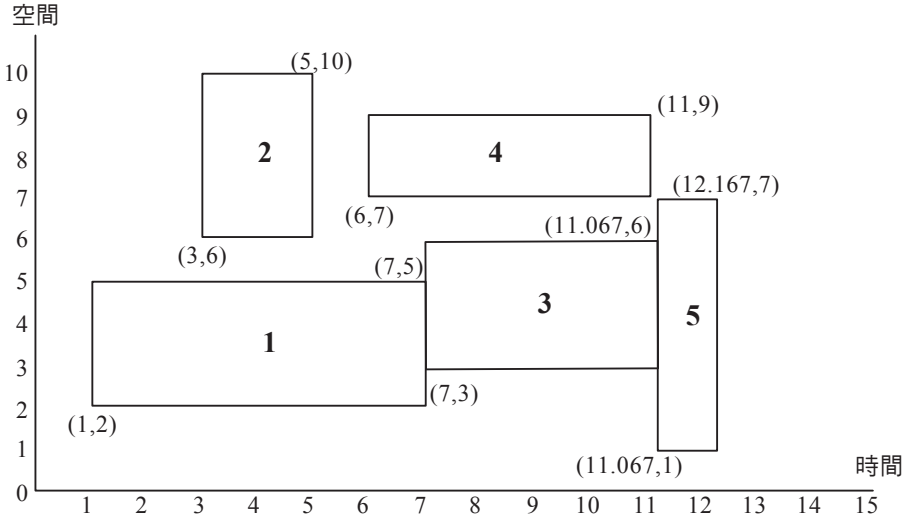

圖 4　化解後之時空圖

伍、數值範例

本研究以 java 程式語言實作二階式演算法來進行電腦化船席指派。

5.1 基本參數設定

首先我們以小範例 (N=10 艘船) 來驗證本演算法求解是否正確，是否符合邏輯及預測結果，並判斷該演算法是否可求得最佳或近似最佳解。接著我們以大範例 (N=50 艘船) 來瞭解本演算法之演算時間是否可以應付實際狀況。

本研究考量一週內到達的船舶，故將每艘船舶之 ETA 隨機取值於 [0, 168] 小時之間。本研究另假設碼頭長度 L=2,000 公尺；船舶編號為 1 到 N，N 為船舶總數；船舶之長度 (l_j) 介於 [30, 200] 公尺之間。船舶 j 的期望停靠點 (d_j) 介於 [0, $L-l_j$] 之間。處理時間 (h_j) 介於 [5, 48] 小時之間。上述資料由電腦隨機產生。

5.2 小範例

表 8 為小範例之船舶基本資料。

表 8 船舶之基本資料 (N=10)

	船舶編號	船舶長度	預計到達時間	期望停靠點	處理時間
1	8	194.0	25.5	1017.4	31.9
2	1	152.0	35.3	800.7	23.4
3	3	187.0	41.7	1008.0	30.5
4	5	188.0	44.4	1678.7	22.4
5	2	121.0	45.1	813.0	46.1
6	9	44.0	55.7	389.0	45.1
7	6	35.0	79.3	419.4	47.6
8	7	89.0	95.6	243.3	18.7
9	4	130.0	149.3	610.0	30.6
10	10	163.0	167.2	298.7	36.6

如將以上船舶置入時空圖中將產生如圖 5 之不可行解，因為包含時空衝突之船舶。經應用本研究所提出之二階式演算法求解，將重疊船舶化解後可得船舶之新座標資料如表 9。

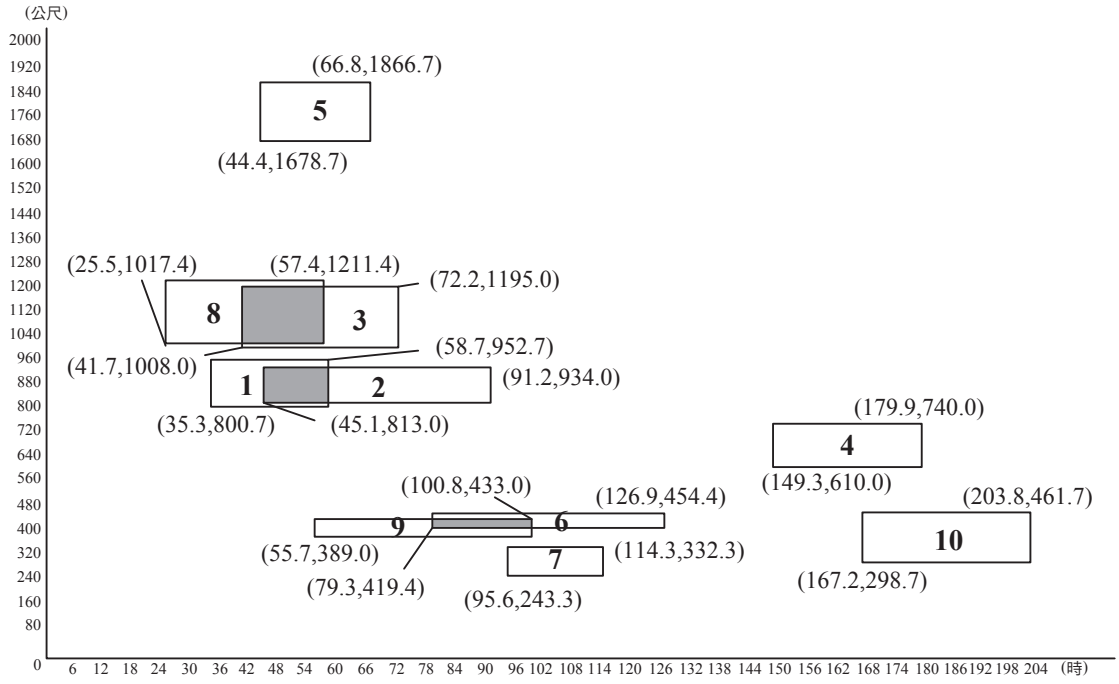

圖 5　第一階段時空圖

表 9　船舶 $N=10$ 化解船舶重疊後之船舶角點座標資料

序號	船舶編號	船舶長度	預計到達時間	期望停靠點	處理時間	左下角點		右上角點		等待時間成本增加量	處理時間成本增加量
						x	*y*	*x*	*y*		
1	8	194.0	25.5	1017.4	31.9	25.5	1017.4	57.4	1211.4	0	0
2	1	152.0	35.3	800.7	23.4	35.3	800.7	58.7	952.7	0	0
3	3	187.0	41.7	1008.0	**30.6**	41.7	**613.7**	**72.3**	**800.7**	0	131.4
4	5	188.0	44.4	1678.7	22.4	44.4	1678.7	66.8	1866.7	0	0
5	2	121.0	45.1	813.0	46.1	**58.7**	813.0	**104.8**	934.0	13600	46.6
6	9	44.0	55.7	389.0	45.1	55.7	389.0	100.8	433.0	0	0
7	6	35.0	79.3	419.4	47.6	79.3	**433.0**	126.9	**468.0**	0	4.5
8	7	89.0	95.6	243.3	18.7	95.6	243.3	114.3	332.3	0	0
9	4	130.0	149.3	610.0	30.6	149.3	610.0	179.9	740.0	0	0
10	10	163.0	167.2	298.7	36.6	167.2	298.7	203.8	461.7	0	0
總計										13600	182.5

註：粗體為更動資料。

將表 9 資料顯示於時空圖可得圖 6。　　解。

由於圖 6 中未包含重疊船可知其為一可行

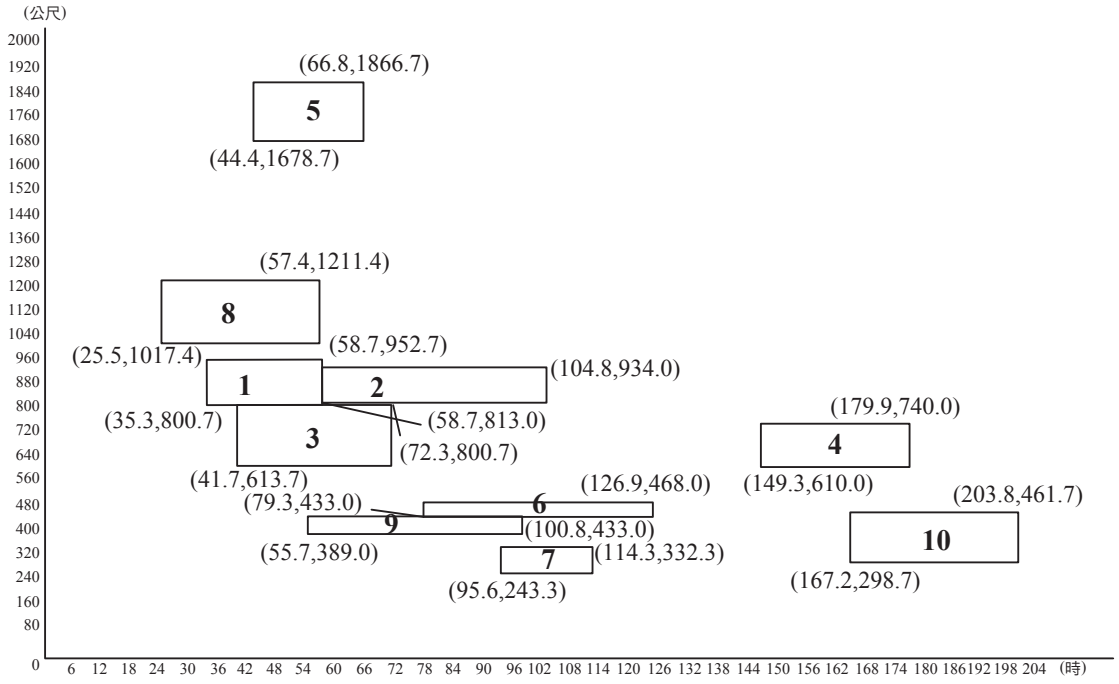

圖 6 化解後之時空圖

5.3 大範例

本節中以大範例 N=50 船並以本演算法來進行求解，以瞭解本研究中所提出之演算法之效力。如表 10 所示，在產生 50 艘船之隨機資料後，本演算法進行衝突排解後可得可行解，如表 11 所示。表中之粗體數字表示，該船舶經過偏移並修正過後之角點座標，如船舶 3、7、9、12 等均經過偏移以化解衝突。欄位 $\Delta W_j c_1$ 顯示了所增加之等待時間成本，欄位 $\Delta H_j c_2$ 則顯示了所增加之處理時間成本。經由加總合後，可知此解之總成本為 192,270.5+2,495.9=194,766.4。

圖 7 顯示在不同之船舶數目之下 (N=10, 20, 50, 100, 150, 200)，本演算法所需之運算時間。由該圖知所需之運算時間隨船舶數目之增加而緩步增加。在船舶數目為 200 艘之情形下，所需之運算時間僅為 53.569 秒。由於所需之運算可行，顯示該演算法具有實用性。

利用 java 程式執行不同船舶數量，各運作五次之平均時間表如表 12 及圖 7 所示。

5.4 分析與討論

針對上述數值實驗資料，討論結果如下：

表 10　船舶 $N=50$ 基本資料

	船舶編號	船舶長度	預計到達時間	期望停靠點	處理時間	左下角點		右上角點	
						x	y	x	y
1	16.0	52.0	1.5	1042.7	10.4	1.5	1042.7	11.9	1094.7
2	49.0	185.0	2.0	126.2	46.4	2.0	126.2	48.4	311.2
3	21.0	181.0	3.7	987.6	**27.9**	3.7	**987.6**	**31.6**	**1168.6**
4	5.0	74.0	8.5	1720.3	24.6	8.5	1720.3	33.1	1794.3
5	18.0	64.0	15.4	1601.2	46.2	15.4	1601.2	61.6	1665.2
6	4.0	127.0	23.0	912.7	25.5	23.0	912.7	48.5	1039.7
7	15.0	159.0	35.5	793.6	**36.7**	35.5	**793.6**	**72.2**	**952.6**
8	41.0	134.0	39.8	1096.4	47.8	39.8	1096.4	87.6	1230.4
9	10.0	107.0	43.0	709.9	**40.0**	43.0	**709.9**	**83.0**	**816.9**
10	30.0	142.0	43.1	161.3	**41.8**	43.1	**161.3**	**84.9**	**303.3**
11	39.0	50.0	43.4	68.8	28.6	43.4	68.8	72.0	118.8
12	50.0	140.0	43.5	1072.0	**44.2**	43.5	**1072.0**	**87.7**	**1212.0**
13	47.0	162.0	47.8	1598.5	**25.6**	47.8	**1598.5**	**73.4**	**1760.5**
14	20.0	63.0	48.2	169.6	**39.0**	48.2	**169.6**	**87.2**	**232.6**
15	8.0	179.0	48.6	763.4	**16.1**	48.6	**763.4**	**64.7**	**942.4**
16	13.0	111.0	48.9	1572.8	**40.8**	48.9	**1572.8**	**89.7**	**1683.8**
17	19.0	169.0	52.8	1221.6	**47.6**	52.8	**1221.6**	**100.4**	**1390.6**
18	11.0	74.0	54.0	116.7	**35.0**	54.0	**116.7**	**89.0**	**190.7**
19	28.0	156.0	57.5	242.7	**17.9**	**57.5**	**242.7**	**75.4**	**398.7**
20	33.0	199.0	63.7	144.5	**29.5**	**63.7**	**144.5**	**93.2**	**343.5**
21	36.0	156.0	64.4	774.9	**11.8**	64.4	**774.9**	**76.2**	**930.9**
22	35.0	69.0	66.3	1868.1	**45.1**	**66.3**	**1868.1**	**111.4**	**1937.1**
23	23.0	170.0	70.8	1175.9	**24.3**	70.8	**1175.9**	**95.1**	**1345.9**
24	26.0	77.0	71.2	133.4	**24.0**	**71.2**	**133.4**	**95.2**	**210.4**
25	43.0	144.0	72.0	954.5	**34.0**	**72.0**	**954.5**	**106.0**	**1098.5**
26	46.0	67.0	78.5	396.6	**8.6**	78.5	**396.6**	**87.1**	**463.6**
27	29.0	165.0	87.8	997.5	**9.3**	87.8	**997.5**	**97.1**	**1162.5**
28	34.0	61.0	88.0	873.5	**24.2**	88.0	**873.5**	**112.2**	**934.5**
29	1.0	65.0	95.7	1104.7	**19.7**	95.7	**1104.7**	**115.4**	**1169.7**
30	42.0	144.0	102.7	418.5	31.1	102.7	418.5	133.8	562.5
31	31.0	38.0	103.1	1957.3	24.7	103.1	1957.3	127.8	1995.3
32	32.0	125.0	111.1	1328.8	44.6	111.1	1328.8	155.7	1453.8
33	6.0	154.0	112.0	1356.3	**30.2**	112.0	**1356.3**	**142.2**	**1510.3**
34	44.0	76.0	113.2	1182.8	46.0	113.2	1182.8	159.2	1258.8
35	40.0	173.0	113.9	1012.8	**42.1**	113.9	**1012.8**	**156.0**	**1185.8**
36	25.0	136.0	115.4	1140.2	**38.4**	**115.4**	**1140.2**	**153.8**	**1276.2**
37	27.0	194.0	120.0	701.4	**41.8**	120.0	**701.4**	**161.8**	**895.4**
38	17.0	182.0	121.5	81.9	**20.2**	**121.5**	**81.9**	**141.7**	**263.9**
39	37.0	194.0	128.3	904.1	**38.8**	**128.3**	**904.1**	**167.1**	**1098.1**
40	9.0	141.0	131.6	250.2	**43.5**	**131.6**	**250.2**	**175.1**	**391.2**
41	2.0	70.0	134.3	1075.0	**20.0**	134.3	**1075.0**	**154.3**	**1145.0**
42	3.0	71.0	135.0	559.2	**31.5**	135.0	**559.2**	**166.5**	**630.2**
43	7.0	55.0	137.0	1152.0	**34.8**	137.0	**1152.0**	**171.8**	**1207.0**
44	48.0	155.0	139.9	1337.7	**34.6**	**139.9**	**1337.7**	**174.5**	**1492.7**
45	12.0	23.0	144.5	750.3	**47.9**	144.5	**750.3**	**192.4**	**773.3**
46	14.0	156.0	147.5	372.2	**10.4**	147.5	**372.2**	**157.9**	**528.2**
47	45.0	157.0	151.6	295.8	**40.9**	**151.6**	**295.8**	**192.5**	**452.8**
48	22.0	137.0	155.0	1577.0	**25.1**	155.0	**1577.0**	**180.1**	**1714.0**
49	38.0	63.0	161.1	761.5	**39.7**	161.1	**761.5**	**200.8**	**824.5**
50	24.0	122.0	165.8	479.5	**8.5**	165.8	**479.5**	**174.3**	**601.5**

註：粗體為須更動資料。

表 11　船舶 $N=50$ 更新資料

	船舶編號	船舶長度	預計到達時間	期望停靠點	處理時間	左下角點		右上角點		$\Delta W_j c_1$	$\Delta H_j c_2$
						x	y	x	y		
1	16.0	52.0	1.5	1042.7	10.4	1.5	1042.7	11.9	1094.7	0	0
2	49.0	185.0	2.0	126.2	46.4	2.0	126.2	48.4	311.2	0	0
3	21.0	181.0	3.7	987.6	**27.936**	3.7	**1094.7**	**31.6**	**1275.7**	0	35.7
4	5.0	74.0	8.5	1720.3	24.6	8.5	1720.3	33.1	1794.3	0	0
5	18.0	64.0	15.4	1601.2	46.2	15.4	1601.2	61.6	1665.2	0	0
6	4.0	127.0	23.0	912.7	25.5	23.0	912.7	48.5	1039.7	0	0
7	15.0	159.0	35.5	793.6	**36.713**	35.5	**753.7**	**72.2**	**912.7**	0	13.3
8	41.0	134.0	39.8	1096.4	47.8	39.8	1096.4	87.6	1230.4	0	0
9	10.0	107.0	43.0	709.9	**40.021**	43.0	**646.7**	**83.0**	**753.7**	0	21.1
10	30.0	142.0	43.1	161.3	**41.85**	43.1	**311.2**	**84.9**	**453.2**	0	50
11	39.0	50.0	43.4	68.8	28.6	43.4	68.8	72.0	118.8	0	0
12	50.0	140.0	43.5	1072.0	**44.253**	43.5	**1230.4**	**87.8**	**1370.4**	0	52.8
13	47.0	162.0	47.8	1598.5	**25.622**	47.8	**1665.2**	**73.4**	**1827.2**	0	22.2
14	20.0	63.0	48.2	169.6	**39.055**	48.2	**5.8**	**87.3**	**68.8**	0	54.6
15	8.0	179.0	48.6	763.4	**16.15**	48.6	**912.7**	**64.7**	**1091.7**	0	49.8
16	13.0	111.0	48.9	1572.8	**40.828**	48.9	**1490.2**	**89.7**	**1601.2**	0	27.5
17	19.0	169.0	52.8	1221.6	**47.802**	52.8	**1827.2**	**100.6**	**1996.2**	0	201.9
18	11.0	74.0	54.0	116.7	**35.001**	54.0	**118.8**	**89.0**	**192.8**	0	0.7
19	28.0	156.0	57.5	242.7	**17.929**	**89.0**	**155.2**	**106.9**	**311.2**	31500.7	29.2
20	33.0	199.0	63.7	144.5	**29.511**	**106.9**	**112.2**	**136.4**	**311.2**	43229.9	10.8
21	36.0	156.0	64.4	774.9	**11.895**	64.4	**490.7**	**76.3**	**646.7**	0	94.7
22	35.0	69.0	66.3	1868.1	**45.114**	**100.6**	**1827.2**	**145.7**	**1896.2**	34301.9	13.6
23	23.0	170.0	70.8	1175.9	**24.383**	70.8	**926.4**	**95.2**	**1096.4**	0	83.2
24	26.0	77.0	71.2	133.4	**24.033**	**87.3**	**35.2**	**111.3**	**112.2**	16054.6	32.7
25	43.0	144.0	72.0	954.5	**34.243**	**100.6**	**1683.2**	**134.8**	**1827.2**	28601.9	242.9
26	46.0	67.0	78.5	396.6	**8.619**	78.5	**453.2**	**87.1**	**520.2**	0	18.9
27	29.0	165.0	87.8	997.5	**9.333**	87.8	**1096.4**	**97.13**	**1261.4**	0	33
28	34.0	61.0	88.0	873.5	**24.202**	88.0	**865.4**	**112.2**	**926.4**	0	2.7
29	1.0	65.0	95.7	1104.7	**19.724**	95.7	**1031.4**	**115.4**	**1096.4**	0	24.4
30	42.0	144.0	102.7	418.5	31.1	102.7	418.5	133.8	562.5	0	0
31	31.0	38.0	103.1	1957.3	24.7	103.1	1957.3	127.8	1995.3	0	0
32	32.0	125.0	111.1	1328.8	44.6	111.1	1328.8	155.7	1453.8	0	0
33	6.0	154.0	112.0	1356.3	**30.233**	112.0	**1453.8**	**142.2**	**1607.8**	0	32.5
34	44.0	76.0	113.2	1182.8	46.0	113.2	1182.8	159.2	1258.8	0	0
35	40.0	173.0	113.9	1012.8	42.151	113.9	**858.4**	**156.1**	**1031.4**	0	51.5
36	25.0	136.0	115.4	1140.2	**38.431**	**115.42**	**1046.8**	**153.9**	**1182.8**	24.4	31.1
37	27.0	194.0	120.0	701.4	**41.812**	120	**664.4**	**161.8**	**858.4**	0	12.3
38	17.0	182.0	121.5	81.9	**20.252**	**136.4**	**236.5**	**156.7**	**418.5**	14940.6	51.5
39	37.0	194.0	128.3	904.1	**39.035**	**134.9**	**1607.8**	**173.9**	**1801.8**	6544.8	234.6
40	9.0	141.0	131.6	250.2	**43.556**	**136.4**	**418.5**	**178.0**	**559.5**	4840.6	56.1
41	2.0	70.0	134.3	1075.0	**20.061**	134.3	**1258.8**	**154.4**	**1328.8**	0	61.3
42	3.0	71.0	135.0	559.2	**31.5**	135.0	**559.5**	**166.5**	**630.5**	0	0.1
43	7.0	55.0	137.0	1152.0	**35.124**	137.0	**181.51**	**172.1**	**236.5**	0	323.5
44	48.0	155.0	139.9	1337.7	**34.755**	**145.7**	**801.8**	**180.5**	**1956.8**	5815.5	154.7
45	12.0	23.0	144.5	750.3	**47.936**	144.5	**641.4**	**192.4**	**664.4**	0	36.3
46	14.0	156.0	147.5	372.2	**10.516**	147.5	**25.5**	**158.0**	**181.5**	0	115.6
47	45.0	157.0	151.6	295.8	**40.99**	**158.0**	**24.5**	**199.0**	**181.5**	6415.6	90.4
48	22.0	137.0	155.0	1577.0	**25.135**	155.0	**1470.8**	**180.1**	**1607.8**	0	35.4
49	38.0	63.0	161.1	761.5	**39.732**	161.1	**858.4**	**200.8**	**921.4**	0	32.3
50	24.0	122.0	165.8	479.5	**8.561**	165.8	**296.5**	**174.3**	**418.5**	0	61
總計										192270.5	2495.9

註：粗體為挪動船舶位置後資料。

表 12　電腦運算時間

單位：秒

項次	船舶數量 (*N*)					
	10	**20**	**50**	**100**	**150**	**200**
1	0.031	0.040	0.181	9.944	18.636	61.317
2	0.035	0.040	0.578	8.886	27.443	50.774
3	0.031	0.051	0.580	7.225	23.787	53.178
4	0.021	0.047	0.232	5.039	27.078	52.921
5	0.031	0.028	0.201	5.847	18.441	49.656
平均	0.029	0.041	0.354	7.3882	23.077	53.569

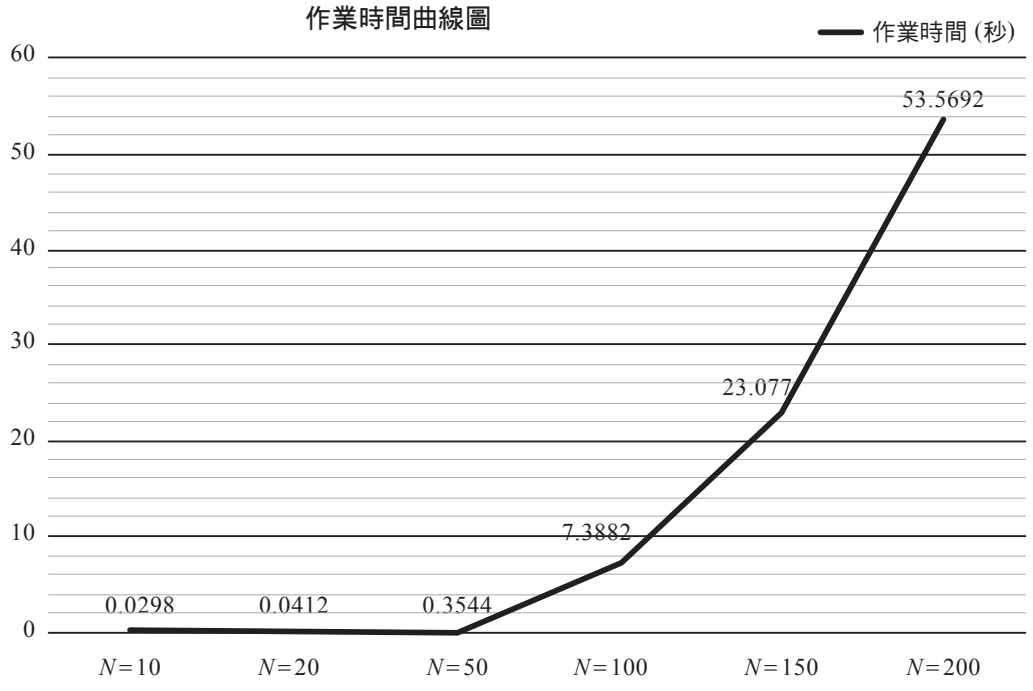

圖 7　電腦運算時間

1. 結果顯示向上或向下移動所增加之處理成本較向右移動所增加之等待成本為低。但由於碼頭空間有限，並非每次重疊都可靠向上或向下挪動來化解，向右挪動化解重疊 (增加等待時間) 有時是必要的。本研究提出之演算法基本是以最少成本來化解衝突。

2. 成本分析，可比較出符合空間限制且成本最低之移動方式。

3. 由表 7 可知，本研究所發展之程式可處理多達 200 艘船舶，且計算時間快速。此顯示本研究所發展之二階式演算法不僅能夠在可接受之時間內運算出結果，也能夠應付實務之需求。

陸、結論與建議

「動態及連續型」船席指派問題是貨櫃碼頭營運者每日面臨的重要課題。本研究針對此問題發展了一個二階式演算法來協助船席指派，其貢獻列示如下：

1. 本研究中針對「連續及動態型」船席指派問題發展一個混合整數規劃之數學模式。
2. 不同於過去之研究，本研究利用「先到先服務」及「往最小成本方向移動」之原則，發展出一個二階式演算法。該演算法可以在合理之時間內找到解，以應付實務之需求。並且，在定義的目標函數下，可推斷其為最佳或近似最佳解。
3. 本研究發展之演算法以 java 語言完成實作。可利用電腦進行快速之連續型船席指派。

由於時間及篇幅之限制，本研究開發之二階式演算法並未與其他演算法進行比較，未來之研究可朝此方向努力。雖然本研究所發展之二階式演算法，可以化解船隻時空衝突之情形，但以下幾點建議可作為未來研究方向，以進一步改善此演算法或擴大研究之成果：

1. 本研究是以先到先服務為原則，來建立船舶之置入順序，此方法理論上雖然可以減少船舶之等待時間，但是否為最佳之順序？或有其他更好之方法來決定船舶之置入順序，可作為未來之研究議題。
2. 本研究中假設並幾定船舶之作業時間(依船舶之期望停靠點及 ETA 停靠)，但實際上船舶之作業時間與橋式起重機之指派數量有關，本研究中並未同時考慮橋式起重指派問題。因此，後續之研究可同時探討橋式起重機指派問，以便對船邊作業做更整體之規劃。
3. 本演算法並未考慮船舶偏移後的連動效應，本演算法僅移動船舶 *j* 以化解與之前置入之船舶的衝突，並未考量移動船舶 j 後對後面進來船舶所造成之連動影響。所選擇的移動方向是目前最低的移動成本。因此，未來之研究可考慮連動效應所造成之增加成本。
4. 本研究中不考慮船舶加速情形，因此當船舶發生時空衝突時，不考量向左方進行偏移化解之情形。若航商能提高船舶航速提前到港，或許能產生更佳之船席利用，但船舶加速會增加耗油及產生空汙問題，是否可行，有待深入探討。

參考文獻

王大瑾、葉彥志、陳聰毅、吳苡媛，2012，船席調派策略適切性研究，*第十一屆離島資訊技術與應用研討會*，497-502。

林岱暘，2011，改良式 GRASP 演算法求解動態連續船席調配問題，國立交通大學

運輸科技與管理學系碩士論文，新竹市。

韋又琳，2013，比較基因演算法與角落空間演算法應用於船席指派規劃，國立高雄海洋科技大學運籌管理系碩士論文，高雄市。

徐賢斌，2014，應用混合式粒子群演算法同時求解貨櫃碼頭動態船席指派及橋式起重機指派問題，*國立高雄海洋科技大學學報*，第 28 期，97-118。

徐賢斌、王冠晴，2016，高雄港船席調派作業現況之研究，*航運季刊*，第 25 卷，第 2 期，73-95。

徐賢斌，2017，應用混合式粒子群演算法進行離散式船席及變動式橋式起重機指派，*運輸學刊*，第 29 卷，第 1 期，1-34。

陳儀安，2011，運用螞蟻演算法求解動態船席指派問題，國立交通大學運輸科技與管理學系碩士論文，新竹市。

黃獻治，2009，港埠貨櫃裝卸作業效率分析與探討 —— 以西岸貨櫃場為例，國立臺灣海洋大學航運管理學系碩士論文，基隆市。

楊逢新，2011，以粒子群演算法應用於船席調配問題之研究，國立交通大學運輸科技與管理學系碩士論文，新竹市。

Bierwirth, C. and Meisel F., 2010. A survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research*, 202(3), 615-627.

Bierwirth, C. and Meisel F., 2015. A follow-up survey of berth allocation and quay crane scheduling problems in container terminals. *European Journal of Operational Research*, 244(3), 675-689.

Guan, Y. and Cheung, R.K., 2004. The berth allocation problem: models and solution methods. *OR Spectrum*, 26(1), 75-92.

Imai, A., Sun, X., Nishimura, E. and Papadimitriou, S., 2005. Berth allocation in a container port: using a continuous location space approach. *Transportation Research Part B*, 39(3), 199-221.

Imai, A., Zhang, J., Nishimura, E. and Papadimitriou, S., 2007. The berth allocation problem with service time and delay time objectives. *Maritime Economics and Logistics*, 9(4), 269-290.

Kim, K.H. and Moon, K.C., 2003. Berth scheduling by simulated annealing. *Transportation Research Part B*, 37(6), 541-560.

Lee, Y. and Chen, C.Y., 2009. A heuristic for the train pathing and timetabling problem. *Transportation Research Part B: Methodological*, 43(8-9), 837-851.

Lim, A., 1998. The berth planning problem. *Operations Research Letters*, 22(2), 105-110.

Meisel, F. and Bierwirth, C., 2009. Heuristics for the integration of crane productivity in the berth allocation problem. *Transportation Research Part E*, 45(1), 196-209.

Narges K., Nathan, H. and Rahimian, S.K.,

2012. An efficient genetic algorithm for solving the quay crane scheduling problem. *Expert Systems with Application*s, 39(4), 13108-13117.

Park, K.T. and Kim, K.H., 2002. Berth scheduling for container terminals by using a subgradient optimization technique. *Journal of the Operational Research Society*, 53(9), 1054-1062.

Pratap S., Daultani, Y., Tiwari M.K. and Mahanty, B., 2018. Rule based optimization for a bulk handling port operations. *Journal of Intelligent Manufacturing*, 29(2), 287-311.

Song, D.W. and Yeo, K.T., 2004. A competitive analysis of Chinese container ports using the analytic hierarchy process. *Maritime Economics & Logistics*, 6, 34-52.

United Nations Conference on Trade and Development (UNCTAD), 1985. Manual on a uniform system of port statistics and performance indicators. Available at: http://r0.unctad.org/ttl/docs-un/unctad-ship-185-rcv2/cn/UNCTAD_SHIP_185_Rcv2c.pdf (accessed 12 July 12, 2017).

Wang, F. and Lim, A., 2007. A stochastic beam search for the berth allocation problem. *Decision Support Systems*, 42(4), 2186-2196.

Zang, H., Zhang, S. and Hapehsi, H., 2010. A review of nature-inspired algorithms. *Journal of Bionic Engineering*, 7, 232-237.