產品模組化設計及組裝順序之研究：元件組裝限制條件情況下

褚文明
興國管理學院
電子商務學系
chuwengming@gmail.com

曾懷恩
國立勤益科技大學工業工程與管理學系
hwaien@ncut.edu.tw

楊仲靖
國立勤益科技大學工業工程與管理學系
brinflow@gmial.com

許唐愷
國立勤益科技大學工業工程與管理學系
stk861138@yahoo.com.tw

摘要

由於資訊技術高速進展及企業經營進入微利時代，致使產品生命週期短促，企業須做快速回應以應市場之需求，方能大幅提升市場競爭力。產品模組化需考慮因素繁多，也較為複雜，且屬 NP-hard 問題。雖已有相當多啟發式演算法針對產品組裝順序與裝配干涉作研究分析，然而求解之效率往往因產品元件數量之增加而受到相當程度之影響，多半僅止於近似最佳模組化作求解。本研究擬提出一新設計之組件裝配限制條件下組裝程序演算法(Constrain-Based Construction Sequence Algorithm, CBCSA)來建立關聯網路的產品模組化之組裝上的限制，用以提升模組裝配順序最佳化的效果。本研究最後以釘書機為實體範例，說明上述演算法的求解之效率。

關鍵詞：產品模組化、產品設計、產品組裝、分群演算法。

Abstract

Today, the rapid flow of information and low-profit of enterprises result in the short product life cycle, the enterprises have to quickly respond to market demands in order to significantly increase market competitiveness under the environment of frequent product variation. In this study, we make use of modular product design to enhance the efficiency, where the products include components of the engineering information, materials and assembly structure in the modular product. In addition, we use the network models to present the relation between product components which is different from the general Cluster analyses. Due to considering of many factors in modular design, the above problems are complex and be taken as NP-hard problems.

Though, a considerable number of heuristic algorithms are used to do research and analysis of modular, the efficiency of their methods is invariable due to the increasing size of product components. Besides, most of them can only be near-optimal solutions. This study proposes a new algorithm called Constrain-Based Construction Sequence Algorithm (CBCSA) to to enhance the optimal design of module assembly sequence by considering the modular assembly constraints associated with each other in the products network.

Finally, this study will take a staple as a physical sample analysis to show the efficiency of our developed algorithm.

Keywords: Modular Product Design 、Product Design、Product Assembly、Product Clustering.

1. 前言

1.1. 研究背景

由於科技創新帶來無數新面向的產品設計概念，且不斷的推陳出新，先進的製程技術也讓產品生產得更有效率，品質也得到了相對的保障，消費水準及品味也更加提升，行銷策略因此也需要比以往更為靈活及有彈性，才能應付這瞬息萬變的消費市場，才能獲取相對的利潤。

另一方面，產品生產所需的總成本，在設計階段即已多半決定了。尤其是各種類的 3C

產品，其產品生命週期極其短促，競爭也非常之激烈，必須在產品設計階段就要將產品從生產到退場過程中所有的因素考慮進去，產品已無足夠的變異修改空間，即便有之，仍須強調快且精準。又因全球市場競爭激烈，Tseng 和 Jiao[1]認為消費者需求變動頻繁，企業為了鞏固市場地位及市場競爭優勢，勢必要提供較多的產品變異(Product Variety)，同時也為了要強調差異性及個性化，故產品客製化是普遍受消費者青睞的，也較能獲取商機。如何推陳出新來滿足消費者需求、如何在第一時間拔得頭籌、如何滿足多樣生產下的企業本身的應變能力，來因應消費者的消費需求，讓消費者增加選擇自身產品的機會，也同樣是現今企業發展之重要策略。因此上述兩種不同的產品生產策略間，的確存在互斥與取捨的問題，也是產品設計者的最大困擾。

基於上述之理由及消費市場多變的需求下，Ong 和 Wong [2]提出產品之組裝件(Component)、次組裝(Subassembly)間的組合、分離比以往更為複雜，而組合分離的複雜度造成了次組裝增殖(Proliferation)現象，同時也助長了生產管理的困難度，這對於生產週期短促的產品而言，無疑的更是雪上加霜，因此產品模組化也絕對是產品設計的必然趨勢及考量。模組化工作在整個產品設計過程中，乃是結合相同功能與特性之設計，以符合特定目的之實體。過去亦有文獻將模組化作定義，如下 Kusiak 和 Huang [3]所提出之定義：

1.介於產品實體與功能設計上的相似度。

2.實際零件間組合、拆解關係的最小程度。

Huang [4]以模組化的架構則產生諸多的優勢，如零組件共用的經濟規模效益、提高產品或部件變更設計的可能性、增加產品的多樣化、縮短訂貨的前置時間、工作分割並平行處理、產品易於升級、維修與回收。因此模組化架構相對於一體化架構有較多的優勢，Mckay 等人 [5]對於建立產品平台，以共享共用的原則，發展延遲化(postponement)的製造或裝配，滿足大量客製化的經濟規模。

本研究基於上述之理由，發展出一新的產品模組化設計及組裝順序演算法，該演算法不僅能利用本研究所研發之組件分群演算，先行將產品作適當的分群並作為產品模組化之基礎，同時將產品組件間之組裝限制條件考慮進去，設計出一有效率之產品組裝順序演算程序。

1.2. 研究架構

下圖 1 為產品組裝之整體架構，首先建立產品零件模型，該模型實際為產品元件之資料結構，資料內容為元件相關之「工程資訊」。元件間之連接關係可構成一網路結構圖，接續才進行產品零件的分群演算。產品組裝的過程中，模組與模組間在組裝上會出現干涉情形，因此在設計的過程中應該要將此問題考慮進去，方利於產生組裝的先行順序。然而本文僅針對元件間之分群作研究，該議題乃為本研究後續之重點。

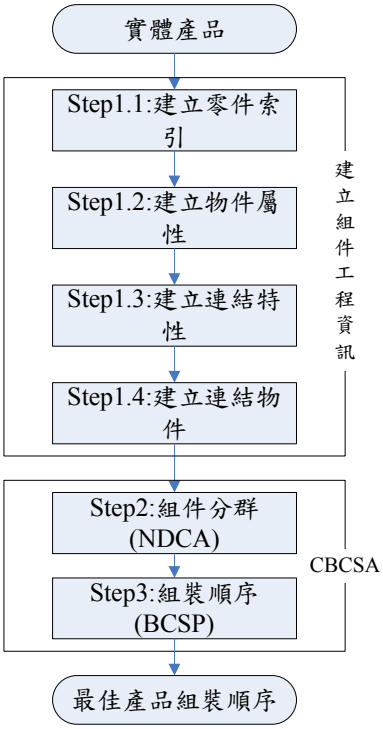


圖 1. 研究架構

本文擬於第 3 章節說明產品組件資料結構網路，第 4 章節介紹本研究所發展之演算法 CBCSA 執行步驟及範例，第 5 章節將以釘書機之實體範例，進行組件組裝順序最佳化之探討，最後章節則為本研究之結論。

2. 文獻探討

從上述背景描述的內容可以很清楚的理解，產品模組化的過程，對於現代的產品設計有著舉足輕重的角色，能夠因應競爭激烈生命週期短促的產品投入市場。已有相當多的研究針對模組化產品作設計，但大多數的演算法均以啟發式或圖解式的方法來進行分析與設計，如 Stone 等人 [6]所發展的啟發式模組識別法，以產品的能量流、材料流、與訊號流，再

依據三種判斷準則，分別是支配性流（dominant flow）、分支流（branching flow）與轉換-傳輸流（conversion-transmission flow），來進行模組的規劃。Kusiak 和 Huang [3]亦發展一啟發式的演算法來模組產品架構，以辨別零組件交換模組、匯流排模組與零組件共用模組，同樣著重於零組件的關係探討與模組規劃。Salhieh 和 Kamrani [7]則利用矩陣的方法來分析零組件間的相似程度，再經由 p-median model 來進行群集的作業。Huang 和 Liang [8]則從同步工程的角度，發展一網際網路為基礎並以模組組合來設計（design with module）的模組式設計公式，以求解模組組合的最佳化。

因此近年來在分群的研究領域中，為了應付複雜的問題及提升分群的效能及效率之方法，因此非常多的分群研究採用啟發式的演算法，包括 Al-Sultan [9]、Xu [10]運用禁忌演算法；Brandyopadhyay 和 Maulik [11]運用退火模擬演算法；Tseng [12]、Maulik 和 Bandyopadhyay [13]、Saha 和 Bandyopadhaya [14]、Chiou 和 Lan [15]以其基因演算法；Handl 等人 [16]、Shelokar 等人 [17]、Kao 等人[18]利用螞蟻最佳化演算法；Merwe 和 Engelbrecht [19]以粒子最佳化演算法，來解決資料分群上的問題。針對上述的啟發式分群法作了實驗比較，也初步獲得其針對問題求解的優劣及特性，但是任何啟發式演算法仍有其先天上的限制條件：第一是不能保證任何問題都能夠得到滿意的收斂及全域最佳解；第二是當問題範圍變數增大時，啟發式演算法的求解效率便會大打折扣。

除了模組最佳化問題外，探討產品之組裝順序也是該考慮的議題之一。其模組與模組間的組裝順序上，受限於組裝的干涉情形將會是影響較佳組裝順序的要素。Smith and Smith [20]提出考慮2維(+x,-x,+y,-y)零件間組裝方向矩陣計算組裝順序轉換方向次數結合基因演算法解決自動組裝規劃。拆解零件的優先順序上，Zhang 等人 [21]運用表現出產品零件組裝優先順序的矩陣與次組裝組裝優先順序矩陣，求解產品的組裝順序。Xing 等人 [22]運用組裝優先關係矩陣探討連續性組裝、非連續性組裝以及綜合性組裝的組裝順序問題。除此之外，也有其它學者提出針對組裝順序上的求解方法 Sundaram 等人 [23]利用動態計畫的方法，來解決組裝次序的搜尋，Hidefumi 等人 [24]也提出了一個拆卸支援系統來解決組裝/拆卸次序的問題，Lanham 等人 [25]利用組裝狀態的向量表示法來求出組裝次序，Mascle 等人 [26]則是提出利用組裝和拆卸給整合起來，對於高價位的每個組裝動作、修理、維修以及元件、組件和產品的在利用來說，是非常有幫助的一種方式，Imamura 等人 [27]研究中提到利用多代理協力的方式來搜尋組裝次序，不僅可以找出最佳的次序，而且此種方法也可以應付在組裝時所會發生的任何狀況。另 Cheng 和 Tseng [28]也提出以 3D 空間概念分析拆解活動，需不斷轉換方向求解產品拆解之特殊產品。

將會在第三章說明組件資料結構網路，第四章則介紹 CBCSA 之執行步驟與範例。最後，第五章將以釘書機之實體範例進行組件組裝順序最佳化之探討。

3. 組件資料結構網路圖

本研究將以 DeFazio 和 Whitney [29]所提出之 Liaison Graph 為基礎，以網路圖的型態表達零件之間的連結關係，再進一步將組件工程上的資訊紀錄於圖中節點的資料結構內，形成組件資料結構網路圖 (Component Data Structure Network, CDSN)，該資料結構內容將詳述於本章節中。本研究之 CDSN 與一般文獻所採用之抽象結合觀念及圖論表達法不同，組件資料結構網路圖藉由結點之資料結構的描述，除能表達出產品彼此連結的關係外，亦能將組件間連結的工程特性、型態及功能描述出來。這將對產品設計、產品改善及成本分析，提供設計者足夠之資訊及參考。

3.1.1. 網路節點資料結構介紹

本研究以 DeFazio 和 Whitney [29]所提出之 Liaison Graph 為基礎，以圖形的方式來表達組裝順序與實體結構，如圖 2，由節點(Node)代表組件與弧(Arc)代表組件和組件之間的關係。

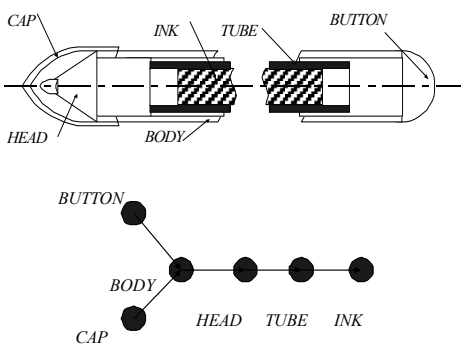


圖 2. 筆的 Liaison Graph

在 *G(N,A)*關聯網路圖之集合中，*N* 表示為節點(Node)的資料結構屬性，其中內部儲存組件(Component)除了本身之外，以及節點(Node)的工程資訊。

3.1.2. 組件資料結構屬性介紹

若將網路節點視為產品待裝配之組件，其網路組件中之資料結構(Data Structure)包含：組件標號(Index Number)、組件連結關係、組件構建屬性、組件結合特徵屬性及組件裝配資訊屬性。其中組件構建屬性包括：組件材質及組件型態，組件結合特徵屬性包括：接觸面特性、結合方向、結合方式及工具性，組件裝配資訊屬性則包括：組裝時間及組裝成本等因素。

3.1.3. 組件連結關係

連結物件集合用來表達組件與組件間的結合關係，並且包含組件的結合順序。另一方面組件連結關係也透露出，組件與其它組件之間的相依關係，若依存關係越高代表此組件在產品中扮演不可或缺的角色，也是在產品設計時所需注意的關鍵零件，所以連結物件集合是方便資料結構做搜尋的重要資訊。

3.1.4. 組件構建屬性

組件構建屬性包括組件主要材質與組件型態的特性，一般組件材質會考慮結合強度、物料生產成本與未來產品回收等相關利益，近年來可替代性的環保材質已漸為設計所納入考量的重要因素，以降低產品對環境所造成的衝擊，因此組件該屬性的考慮變得很重要。另針對組件型態的特性，根據 Mikkola 和 Gassmann [30]所述，將其分類成共用件、標準件和專用件四種類，本研究取其共用件、標準件和專用件三種型態類別，並將其定義彙整於下表 1 內。

表 1. 組件型態類別

	描述
共用件	能夠分享於多種不同類型產品之相同型態和功能上作替代，以避免及減少重覆設計新零件的時間，進而加強零件的管理及降低成本。
標準件	非新設計之零件，可供同類型之產品所使用，能夠提升零件之利用率。
專用件	專門為唯一種產品所設計之新型零件，用來提升產品結構，以及提供研發技術的提升。

3.1.5. 組件結合特徵屬性

組件結合特徵與組件間之接觸面特性和彼此間之連接方式有關，本研究參考 Cheng 和 Tseng [28]所提之接觸面特性、工具性之零件工程資訊，以及結合方向、結合方式納入組件資料結構中。

(1)接觸面特性

表示出組件與組件之間接觸面或接觸點的型態，當接觸面愈廣或是接觸點愈多時，則表示二組件之間的密接程度越高；換言之，此兩組件間的結合也更牢固。反之，組件與組件間之結合介面設計較容易，且易於拆裝，本研究將 Cheng 和 Tseng [28]所提出之接觸面特性整理如圖 3 所示。

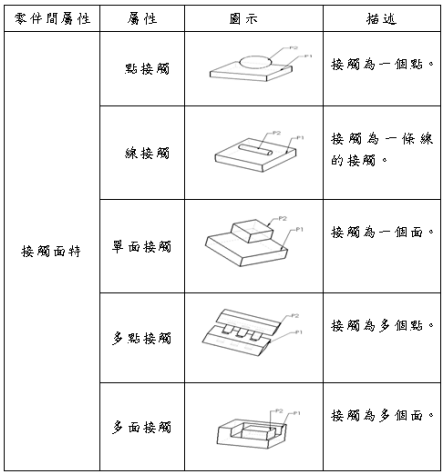


圖 3. 接觸面特性

(2)結合方向

組件結合方向，乃探討組件與組件間的可行組合方向，過去像是 Smith 和 Smith [20]曾提出考慮以 2 維(+X,-X,+Y,-Y)零組件間組裝方向矩陣，計算組裝順序轉換方向次數之研究。本研究以 3 個維度空間為基準，共計 6 個座標結合方向，故組件間可形成一個組合方向矩陣(Combined Direction Matrix, CDM)，藉此方向矩陣瞭解組件間結合方向關係。矩陣內的元素為一組 6 個方向之座標向量(+X、-X、+Y、－Y、+Z、-Z)，每一座標值均為整數 0 或 1，0 表示兩組件間在該方向沒有結合關聯，1 則表示該方向有結合關聯。CDM 矩陣對角線均為令為 0，矩陣的上三角

與下三角數值相對稱。若以三個組件 P1、P2 及 P3 為例，表 2 為其 CDM 矩陣，其中 P1 與 P2 組件間的向量數值為(+X,-X,+Y,-Y,+Z,-Z)=(1,1,0,0,0,0)，表示 P1 可經由+X 或是-X 之方向與 P2 進行結合。

表 2. 方向矩陣

Part	P1	P2	P3
P1	0	(1,1,0,0,0,0)	(1,1,1,1,1,1)
P2	(1,1,0,0,0,0)	0	(1,1,1,0,0,0)
P3	(1,1,1,1,1,1)	(1,1,1,0,0,0)	0

(3)結合方式

Akagi [31]提出以扣接(Fastener)方法，將兩組件間之結合方式分為以下四種型態：如圖 4 所示，並參考 Tseng 和 Li [32]之結合方式，本研究將其結合方式分類如下表 3 內。

1. 固定扣接能夠被拆解(Disassembled)，如：螺紋(Screwed)或鍵(Key)等等。
2. 固定扣接不能夠被拆解(Don't disassembled)，如：鉚釘(Riveted)或焊接(Welding)等等。
3. 移動扣接能夠被拆解(Disassembled)，如：彈簧(Spring)。
4. 移動扣接不能夠被拆解(Don't disassembled)，如：滾珠軸承(Balls Bearing)。

型態		範例
固定式 Fastener	可以拆解	螺絲(Screw)、螺栓(Bolted Joint)、鍵(Key)、栓槽(Spline)、契(Wedge)
	不能拆解	緊配合(Pressing)、鉚釘(Riveted Joints)、焊接(Welding)
移動式 Fastener	可以拆解	扣環(Snap Ring)、軸承(Bearing)、彈簧(Spring)
	不能拆解	軌道及滾珠軸承(Races and Ball-Bearing Balls)

圖 4.之組件結合方式分類

表 3. 組件結合方式分類

連接器	材料	特性
C1	鐵	表示鉚釘(rivet)。
C2	鐵	表示彈簧(spring)。
C3	鐵	表示螺絲(screw)。
C4	錫	表示焊接(welding)。
C5	-	表示緊密結合。
C6	-	表示扣接。
C7	鐵	鍵(Key)。
C8	-	插入。

(4)工具性

工具主要參考自 Cheng 和 Tseng [28]所提出之 5 項工具型態，其中的考量屬性有徒手、小工具、特殊之工具及大型工具等。研究中將用以表示操作程度越困難其作業時可能會影響到組裝上的效率，以下圖 5 是其工具性與其說明。

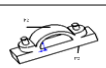
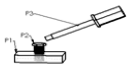
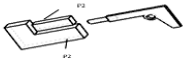
零件間屬性	屬性	圖示	敘述
工具性	手		僅需要雙手，即可將兩個零件結合起來。
	螺絲起子		兩個零件結合時，需要螺絲起子才可以結合。
	小工具		兩個零件結合時，需要小型工具才可以結合，如：十字螺帽與螺絲需透過十字螺絲起子才可順利結合。
	使用特別的工具		兩個零件結合時，需要特殊工具才可以結合。
	大型工具		兩個零件結合時，需要大型工具才可以結合，如：兩個零件需要透過焊接才可順利連結。

圖 5. 工具性分類

3.1.6. 組件裝配資訊屬性

組件間裝配相關資訊主要是提供組件裝配的過程中，由內、外在環境中提供或消耗資源量等相關資料項目，通常與組件裝配時間或成本有關係。該屬性資料可作為組件裝配過程中計算組裝時間、組裝成本或求解最佳化問題的參考依據，對於產品開發，設計及分析等作為有很大的幫助，本研究僅以組裝時間及組裝成本兩項納作為考量因素。最後將組成如圖 6 釘書機組件資料結構表。

組件節點	組件構建屬性		組件連結關係	組件結合特徵屬性				組件裝配資訊屬性	
	組件材料	組件型態		接觸面特性	connecter 屬性			組裝時間	組裝成本
					結合方向	結合方式	工具性		
1	鐵	-	2	單面接觸	(0,0,0,0,0,1)	C5	手	-	-
			4	單面接觸	(0,0,0,0,0,1)	C5		-	-
2	鐵	-	1	單面接觸	(0,0,0,0,1,0)	C5	手	-	-
4	鋼	-	1	單面接觸	(0,0,0,0,1,0)	C5	手	-	-
			5	單面接觸	(0,0,0,0,0,1)	C6	小工具	-	-
			8	單面接觸	(0,0,0,0,0,1)	C5	手	-	-
5	鋼	-	4	單面接觸	(0,0,0,0,1,0)	C6	小工具	-	-
			6	單面接觸	(0,0,0,0,0,1)	C5	手	-	-

圖 6. 釘書機組件資料結構表

4. 組件安裝限制條件下組裝程序演算法(CBCSA)

本研究所發展之組件安裝限制條件下組裝程序演算法(Constrain-Based Construction Sequence Algorithm)，乃針對網路密度分群演算法(Network Density Clustering Algorithm, NDCA)，在網路中的所有節點間相互連結的密

度作為分群的依據，即網路中任一節點與其相鄰周邊節點間之連結路徑數愈多，表示該區域的節點間連接密度愈高。若將節點視為裝配之組件，則連結密度愈高的組件形成群組(Cluster)的機會就愈高。

網路節點連結密度可針對某部份之節點群並令數目為 N，再將這些節點間之連結路徑數算出並令為 L，密度 L/N 即可計算出來。因此本研究之分群演算重點工作，首先必須在廣域網路的節點分佈中，找出適當的節點群組，該節點群組內的節點數及其間相連結的路徑數所構成的密度，須比網路中其它節點群組要相對的高才行。很明顯的，網路中的節點群組可以擁有無數的組合型態，且會隨網路節點數增加而讓節點分群更加複雜，若要從網路中決定出最佳的節點群組合是非常困難的，因此網路節點分群乃為一 NP-Hard 的問題，且不容易處理。

本研究擬將所有網路節點連結關係以一關聯矩陣 A 表示，再利用矩陣運算的機制找出節點間連結的路徑數；針對矩陣運算所產生的數據資料，並透過本研究所發展之演算分析，找出能夠代表網路各群組的重點連結路徑(Moment Link)，該重點連結路徑兩端的節點以 $node_i$ 及 $node_j$ 表示，群組會以 $node_i$ 及 $node_j$ 間所有相連的路徑數與節點數形成同一群組，該群組之節點密度若與其它網路節點群組比較，會相對的高。演算法產生第一個分群必定是發生在整體網路的某定區域，其他區域的節點與該群組比較沒有關係，因此演算法可進一步針對其他網路區域執行上述相同的演算，找出網路節點第二個群組；依此類推，最後網路可以形成數個群組出來，完成網路節點第一層(Layer)的分群。同一群組內的節點可以結合成單一的節點，針對組裝而言，即將裝配組件組合併成一個模組(Module)之意，我們可將模組視為一個新組裝完成的組件(Component)，該組件亦能擁有其相關之組件屬性，包括組件間的聯結關係。因此可持續上述分群演算之機制，執行第二層(Layer)的分群演算，求出模組間之分群結果。

另外，本研究也考慮到組件組裝上限制之斥資訊，組裝的元件彼此相接連的方是有許多種類型，但也須考慮到元件間相接連之互斥關係，即部分的元件的安裝存在順序性，若未按此順序執行，則部分元件就無法進行安裝。若演算法重覆上述之步驟，進行多層次的分群演算，所有組件最後會裝配成一成品。本研究所提出之分群演算法完整的流程，擬詳述於下節中。

4.1. 演算法符號介紹

在介紹本研究之分群演算法之前，先將演算中使用的符號表示如下：

1.網路包含 n 個節點，其構成的 $n \times n$ 的關聯矩陣為 A^1，其中

$$A^1 = \begin{bmatrix} a_{11}^1 & a_{12}^1 & \cdots & a_{1n}^1 \\ a_{21}^1 & a_{22}^1 & \cdots & a_{2n}^1 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1}^1 & a_{n2}^1 & \cdots & a_{nn}^1 \end{bmatrix} \quad (1)$$

A^1 矩陣中的元素可表示為 $a_{ij}^k, 1 \le i, j \le n$，$k=1$，表示 $node_i$ 能夠透過 $k(k=1)$步距離到達 $node_j$ 的可行路徑數目。

2.令 $A^k = A^{k-1} \times A^1$，$k > 1$，其中 A^k 內的元素 a_{ij}^k 可計算如下所列：

$$a_{ij}^k = \sum_{q=1}^{n} (a_{iq}^{k-1} \cdot a_{qj}^1), \; k > 1, \; 1 \le i, j \le n \quad (2)$$

3. P_{ij}^v 表示為 $node_i$ 至 $node_j$ 可到達之第 v 條路徑的節點集合。

4. D_s 表示網路節點鐘第 s^{th} 個群組之步階密度矩陣，其中，

$$D_s = \begin{bmatrix} 0 & d_{12} & \cdots & d_{1n} \\ d_{21} & 0 & \cdots & d_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ d_{n1} & d_{n2} & \cdots & 0 \end{bmatrix} \quad (3)$$

5. CN_r 表示針對網路節點中第 r^{th} 個網路群組，從 $node_i$ 出發到 $node_j$ 結束，中間所有經過的節點集合。

6. CN_number 為 CN_r 內節點數目。

7. CA_number 為 CN_r 內節點間路徑步階數目。

8. θ_r 表示第 r^{th} 個網路群組密度。

9. $Cluster_r$ 網路節點第 r^{th} 個群組節點集合。

10.網路包含 n 個組件，建構成以 E^k 來表達，探討組裝限制之矩陣，其矩陣形式與關聯矩陣相似，其表示如下:

$$E^1=\begin{bmatrix} e_{11}^1 & e_{12}^1 & \cdots & e_{1n}^1 \\ e_{21}^1 & e_{22}^1 & \cdots & e_{2n}^1 \\ \vdots & \vdots & \ddots & \vdots \\ e_{n1}^1 & e_{n2}^1 & \cdots & e_{nn}^1 \end{bmatrix} \quad (4)$$

11.令 $E^k = E^{k-1} \times E^1, k>1$，其中 E^k 內的元素 e_{ij}^k 可計算如下所列：

$$e_{ij}^k = \sum_{q=1}^{n} (e_{iq}^{k-1} \cdot e_{qj}^1) \ , \ k>1, \ 1 \le i,j \le n \quad (5)$$

12.c_{ij}:表示在 $node_i$ 與 $node_j$ 進行組裝時，會造成干涉之組件集合。

4.1.1. NDCA 執行步驟

Step_1: 1 步階關聯矩陣運算(*k=1*)

令由 n 個網路節點構成 $n \times n$ 矩陣 A^1，該矩陣即為 1 步階關聯矩陣，1 步階關聯矩陣可表達網路節點間之連結關係；矩陣中的元素 $a_{ij}^1, 1 \le i,j \le n$ 可表示為 $node_i$ 經由 1 步階連結到 $node_j$ 的可行路徑數目。當 $a_{ij}^1 = 1$ 表示 $node_i$ 與 $node_j$ 有 1 條相鄰的連結，反之，$a_{ij}^1 = 0$ 表示 $node_i$ 與 $node_j$ 間沒有任何連結存在。矩陣中 $a_{ij}^1 = 1$, $i = j$ 的情況表示節點連結回自己，雖然矩陣運算過程中為合理，亦屬 1 步階之連結，但該情況表示 $node_i$ 在原地踏步，對演算法並無任何作用，反而會擾亂後續節點間步階數之計算，因此演算法將所產生之矩陣，A^1 其中令 $a_{ij}^1 = 0$, $i=j$, $1 \le i,j \le n$，A^1 可修改如下所列：

$$A^1=\begin{bmatrix} 0 & a_{12}^1 & \cdots & a_{1n}^1 \\ a_{21}^1 & 0 & \cdots & a_{2n}^1 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1}^1 & a_{n2}^1 & \cdots & 0 \end{bmatrix} \quad (6)$$

若將節點視為待裝配的組件，$a_{ij}^1 = 0$ 表示組件 i 與組件 j 間沒有存在任何連結機制，反之 $a_{ij}^1 > 0$，則表示至少存在一條組合之方法，依前述之定義，若以圖 7 為範例，可得而 A^1 如下：

$$A^1=\begin{bmatrix} 0&1&1&0&0&0&0&0&0 \\ 1&0&1&1&0&0&0&0&0 \\ 1&1&0&1&0&0&0&0&0 \\ 0&1&1&0&1&0&0&0&0 \\ 0&0&0&1&0&1&0&0&0 \\ 0&0&0&0&1&0&1&1&1 \\ 0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0&0 \end{bmatrix} \quad (7)$$

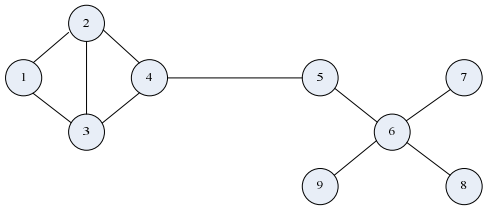


圖 7. 組件關聯圖例

Step_2: 2 步階關聯矩陣運算(*k=2*)

2 步階關聯矩陣 A^2 可表達網路節點間，可經由兩步階到達之節點連結關係；矩陣中的元素 a_{ij}^2 , $1 \le i,j \le n$ 表示 $node_i$ 經由 2 步階連結到 $node_j$ 的可行路徑數目。另外，節點走 1 步至相鄰節點後，再走回原節點亦屬 2 步階之連結，該情況對演算法問題分析並無助益，也同樣會擾亂後續節點間步階數之計算，因此該狀況亦須排除之，A^2 可修正表示如下：

$$A^2=\begin{bmatrix} 0 & a_{12}^2 & \cdots & a_{1n}^2 \\ a_{21}^2 & 0 & \cdots & a_{2n}^2 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1}^2 & a_{n2}^2 & \cdots & 0 \end{bmatrix} \quad (8)$$

以圖 2.1 為例，執行二步階矩陣運算，其所得到的運算結果可表示如下：

$$A^2=\begin{bmatrix} 0&1&1&0&0&0&0&0&0 \\ 1&0&1&1&0&0&0&0&0 \\ 1&1&0&1&0&0&0&0&0 \\ 0&1&1&0&1&0&0&0&0 \\ 0&0&0&1&0&1&0&0&0 \\ 0&0&0&0&1&0&1&1&1 \\ 0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0&0 \end{bmatrix} \times \begin{bmatrix} 0&1&1&0&0&0&0&0&0 \\ 1&0&1&1&0&0&0&0&0 \\ 1&1&0&1&0&0&0&0&0 \\ 0&1&1&0&1&0&0&0&0 \\ 0&0&0&1&0&1&0&0&0 \\ 0&0&0&0&1&0&1&1&1 \\ 0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0&0 \end{bmatrix}$$
$$= \begin{bmatrix} 0&1&1&2&0&0&0&0&0 \\ 1&0&2&1&1&0&0&0&0 \\ 1&2&0&1&1&0&0&0&0 \\ 2&1&1&0&0&0&0&0&0 \\ 0&1&1&0&0&1&0&0&0 \\ 0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&1&1 \\ 0&0&0&0&0&0&1&0&1 \\ 0&0&0&0&0&0&1&1&0 \end{bmatrix} \quad (9)$$

參考 2.7 式，$a_{14}^2=1$ 表示 $node_1$ 到 $node_4$ 存在一條 2 步階連結，本文以 P_{14}^1=<1-2-4>表示該連結路徑，其中 $node_2$ 為 $node_1$ 到 $node_4$ 之間隔節點。以圖 7 為例，A^2 為 2 步階矩陣運算結果可以圖 8 表示如下：

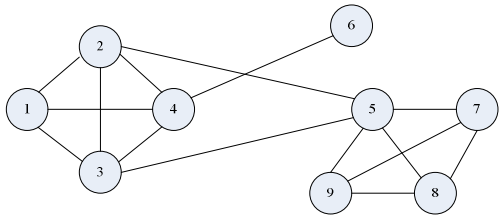


圖 8. 二階段組件關聯圖

Step_3: 計算步階密度矩陣

接續要算出網路任意兩節點間所存在的路徑步階數，這是一件很重要的計算，因為該項訊息能夠初步判斷出“網路中有那些區域可能存在密度較高的節點群！”兩節點間步階數目愈大，則表示該兩節點之相鄰周邊可能存在密度高的節點群。

$$
\begin{aligned}
D_s &= A^1+2\times A^2 \\
&= \begin{bmatrix} 0 & a_{12}^1 & \cdots & a_{1n}^1 \\ a_{21}^1 & 0 & \cdots & a_{2n}^1 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1}^1 & a_{n2}^1 & \cdots & 0 \end{bmatrix} + (2\times \begin{bmatrix} 0 & a_{12}^2 & \cdots & a_{1n}^2 \\ a_{21}^2 & 0 & \cdots & a_{2n}^2 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1}^2 & a_{n2}^2 & \cdots & 0 \end{bmatrix}) \\
&= \begin{bmatrix} 0 & a_{12}^1+2\cdot a_{12}^2 & \cdots & a_{12}^1+2\cdot a_{12}^2 \\ a_{21}^1+2\cdot a_{21}^2 & 0 & \cdots & a_{2n}^1+2\cdot a_{2n}^2 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1}^1+2\cdot a_{n1}^2 & a_{n2}^1+2\cdot a_{n2}^2 & \cdots & 0 \end{bmatrix} \\
&= \begin{bmatrix} 0 & d_{12} & \cdots & d_{1n} \\ d_{21} & 0 & \cdots & d_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ d_{n1} & d_{n2} & \cdots & 0 \end{bmatrix}
\end{aligned} \tag{10}
$$

$d_{ij},\ i,j\in[1,...,n]$ 為步階密度矩陣內的元素，表示 $node_i$ 到 $node_j$ 所有路徑步階的數目，數目愈大則表示 $node_i$ 到 $node_j$ 的路徑，愈有機會形成前文所提的重點連結路徑(Moment Link)。

Step_4：計算群組內節點數目

自 D_s 中找出最大的 $d_{ij},\ i,j\in[1,...,n]$，透過 2 步階關聯矩 A^2，找出所有自 $node_i$ 出發到 $node_j$ 結束，將中間所有經過的節點紀錄為 CN_r 集合，如下式：

$$
\begin{aligned}
CN_r &= \{node_q \mid q\in P_{ij}^v,\ v=1,...,p\} \\
&= \{node_i,\ node_{q_1},...,\ node_{q_p},\ node_j\},\ p=a_{ij}^2
\end{aligned} \tag{11}
$$

P_{ij}^v 是 $node_i$ 到 $node_j$ 存在 2 步階的路徑集合，共有 $p=a_{ij}^2$ 條路徑；CN_r 則為這些路徑中，所有經過的節點集合，其中當然也包括 $node_i$ 到 $node_j$，同時也令 CN 內的節點數目為 CN_number。

Step_5：計算群組內節點相互連結密度

本步驟將針對 CN_r 內所有的節點，其內所有相互連結的路徑數目，並將其令為 CA_number。首先針對 CN_r 內所有的節點，令其構成 $(p+2)\times(p+2)$ 之矩陣 B，

$$
\begin{aligned}
B &= \begin{bmatrix} 0 & b_{12} & \cdots & b_{1(p+1)} & b_{1(p+2)} \\ b_{21} & 0 & \cdots & b_{2(p+1)} & b_{1(p+2)} \\ \vdots & \vdots & 0 & \vdots & \vdots \\ b_{(p+1)1} & b_{(p+1)2} & \ddots & 0 & b_{(p+1)(p+2)} \\ b_{(p+2)1} & b_{(p+2)2} & \cdots & b_{(p+2)(p+1)} & 0 \end{bmatrix} \\
&= \begin{bmatrix} 0 & a_{iq_1}^2 & \cdots & a_{iq_p}^2 & a_{ij}^2 \\ a_{q_1i}^2 & 0 & \cdots & a_{q_1q_p}^2 & a_{q_1j}^2 \\ \vdots & \vdots & 0 & \vdots & \vdots \\ a_{q_pi}^2 & a_{q_pq_1}^2 & \ddots & 0 & a_{q_pj}^2 \\ a_{ji}^2 & a_{jq_1}^2 & \cdots & a_{jq_p}^2 & 0 \end{bmatrix}
\end{aligned} \tag{12}
$$

矩陣 B 內所有的元素數值，表示 CN_r 內所有的節點間之 CA_number，但因矩陣 B 之上矩陣及下矩陣是對稱的關係，表示兩節點間的路徑數目會重複乙次，因此 CA_number 可計算如下式：

$$
CA\_number = \frac{\sum_{b_{ij}\in B} b_{ij}}{2},\ \forall i,j \tag{13}
$$

根據上述之計算，若將 CN_r 內所有節點構成的群組，其網路節點密度令為 θ_r，則該值計算如下：

$$
\theta_r = \frac{CA\_number}{CN\_number} \tag{14}
$$

紀錄 θ_r 其相對應之 CN_r。

Step_7：決定出網路第 r^{th} 個分群

自 D_s 中找出下一個次大的 $d_{ij},\ i,j\in[1,...,n]$，重複 Step_4 及 Step_5，並計算出 θ_r，同時也記錄其相對應之 CN_r。最後比較所有求得的 θ_r 值，找出最大值並將其相對應 CN_r 內之節點視為網路中第 r^{th} 個群組 $Cluster_r$(初始為 $r=1$)。其

後，CN_r 內所有的節點也會從矩陣 A^1 中移除。

Step_8：找尋網路節點下一個群組

令 $r=r+1$，針對下一個網路節點群組 $Cluster_r$ 作求解；根據前一步驟，A^1 已將前一個群組所屬的節點移除，需再次重複 Step_1 到 Step_7 之執行步驟，針對 A^1 剩下的節點則持續做分群的動作。

Step _9：網路節點第 m=1 層分群的結束

重複 Step_8 執行的結果，會產生網路第 1 層所有的群組，並將其令為 $Cluster_1$~$Cluster_w$。本步驟執行結束的條件是：當關聯矩陣 A^1 已無任何節點存在為止，如圖 9 所列。

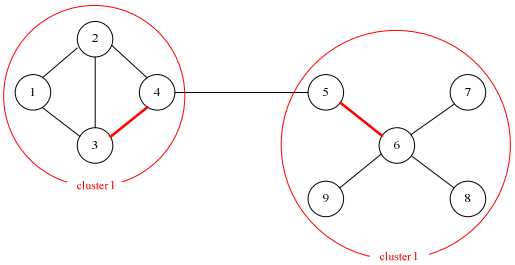


圖 9. 將零件分成為二個群組

Step_10:針對第 *m* 層內所有群組產生組裝順序

以下圖的螺栓與螺帽之組合為例，共構成四個元件，分別為螺栓 1 及三個螺帽 2、3、4；

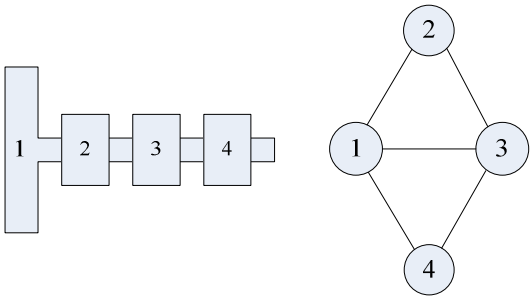


圖 10. 螺栓與螺帽酸組合範例

元件安裝的順序非常的重要，若螺帽 2 未能先比螺帽 3 或 4 裝在螺栓上，則就無法進行安裝，螺帽 3 也有同樣的問題。因此圖 10 為四個元件的網路節點圖，從圖中無法判斷出元件組裝的順序，因此網路節點圖僅能顯示元件間的連結關係，但無法提供組裝順序的資訊。由圖 10 為例，可得到之互斥訊息如下所列:

$c_{13}=c_{31}=\{2\}$

$c_{14}=c_{41}=\{2,3\}$

表示 $node_2$ 與 $node_1$ 安裝完後才能安裝 $node_3$，若 $node_1$ 與 $node_3$ 先組裝，$node_2$ 即無法安裝。同理若 $node_1$ 與 $node_4$ 先組裝，$node_2$ 與 $node_3$ 亦無法安裝。有了上述的資訊後，即可以下列演算法求出網路節點組裝程序。但是，當組裝順序所限制之零件集合，其集合內之組件已先行被排入組裝順序時，此組裝順序方為成立，即不屬受限於互斥條件。

Step_10.1：產生第 r^{th} 群組之組裝順序的 1 步階矩陣 A^1

由 $node_i$ 與 $node_j$ 以其組成之關聯性，所構成之 A^1 矩陣，如下式(15):

$$E^1=\begin{bmatrix}0&1&1&1\\1&0&1&0\\1&1&0&1\\1&0&1&0\end{bmatrix} \tag{15}$$

Step_10.2：2 步階關聯矩陣運算($k=2$)，並考慮組件組裝限制

參考組裝元件間相連互斥之條件，產生 2 步階矩陣 E^2。如下公式 14，其中 $e^2_{31}=2$，用以表示為有 2 條可行的 2 步階組裝順序。

$$E^2=E^1\times E^1=\begin{bmatrix}0&0&1&1\\1&0&1&2\\2&0&0&1\\1&1&0&0\end{bmatrix} \tag{16}$$

在矩陣中每一個元素 a^2_{ij} 的計算均要將元件間相連互斥之條件考慮進去，其計算如下公式(17):

$$a^2_{ij}=\sum_{q=1}^{4}(a^1_{iq}\cdot a^1_{qj}),\ 1\le i,j\le 4 \tag{17}$$

e^2_{ij} 值算出後再檢查 P^v_{iq}, $\forall v$ 路徑中的最後一個節線，是否與任一個組裝限制的組件組合 C 內的元素 x_{ij} 相同？若相同，則令 $a^2_{ij}=0$，否則就以原計算 a^2_{ij} 為之。以 $e^2_{32}=\sum_{q=1}^{4}(e^1_{3q}\cdot e^1_{q2})$, $1\le i,j\le 4$ 為例，得 $e^2_{32}=e^1_{31}\times e^1_{12}=1$；然而

P_{31}^{1} =< 3 − 1 > ，存在與 C 內的元素 x_{31} 相同的節線，因 x_{31}={2}，表示 $node_3$ 與 $node_1$ 間應該存在 $node_2$，但 P_{31}^{1} 內的節點排列順序違反了此限制條件，因此令 $e_{32}^{2}=0$ 。

Step_10.3：持續執行 *k=k+1* 步階運算

依據上述 **step_10.2** 同樣條件執行 3 步階矩陣求解，3 步階矩陣內的元素除了組裝元件間相連互斥之條件檢查外，亦不得有折返路徑的情況存在。

Step_10.4：完成第 r^{th} 群之裝配順序

最後，運算至所有矩陣內元素皆為 0 或達到最大之運算 *k* 值時，即完成第 r 群之裝配順序。下列則將最終 3 步階之結果呈現在公式(18):

$$E^3 = E^2 \times E^1 = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 2 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad (18)$$

以圖 10 為例，最後僅存在路徑<1-2-3-4>、<2-3-1-4>、<2-1-3-4>、<3-2-1-4>或<4-3-2-1>。

Step_10.5：結束第 *m* 層之運算

若網路節點為 n，演算法的執行停止條件如下:

1. 當 *k* 步階的矩陣所有的元素皆為 0(表示節點已無進一步的路徑可連結)。
2. 最多執行 *n-1* 步階矩陣為止(表示所有的元件為循序組裝的組件)。

上述圖例僅說明了當所有的組裝件彼此相接連存在互斥關係的連結順序，但組裝件通常是部分組件沒有互斥關係，另部分存在互斥關係，則上述組件相依順序演算法在分析的過程中，是否能夠清楚的分辨所有組件的組裝順序。

Step_ 11：持續網路節點第 *m=m+1* 層分群

原尚未執行分群的網路節點可視為網路第 *m=1* 層，節點經執行上述分群步驟後，所形成的群組連結型態，可視為網路第 *m = 2* 層。其中每一群組可組合成一個單體組件，該單體組件繼承所有節點的屬性，因此亦可將上一層的群組視為下一層的節點。故本演算法仍可持續執行如下：

Step_11.1：將群組以節點作為表示

將上一層分群後的群組，均視為一節點，並將其相關節點屬性表列之，最後建構出一相互連結之網路圖。依圖 9 為例，其第二層的節點網路連結情況如圖 11 所列。

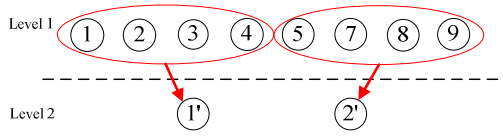


圖 11. 分群後第二層節點網路圖

Step_11.2：重覆執行 Step_1~Step_10

求解出第二層分群之結果。依圖 12 為例，其第三層的節點網路連結情況如圖 13 所列。

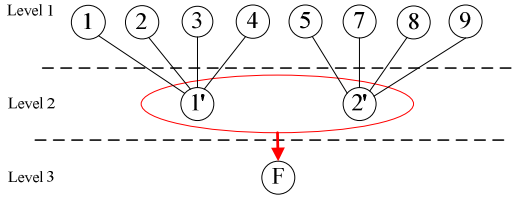


圖 12. 分群後第三層節點網路圖

Step12：重覆 Step_11 步驟

重覆 Step_11 的步驟，網路節點經過多層次的分群後，最後終會形成單一群組，該最後的群組即為根節點(Root node)；若將節點視為組件，則表示最終產品組裝完畢，分群演算法亦到此終結。整個分群的過程，各層節點關係可表示成一樹狀結構，以圖 7 為例，該結構表示如圖 13。

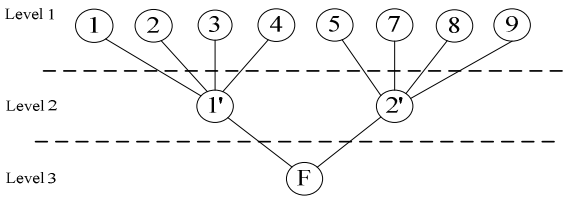


圖 13. 節點分群後之樹狀結構示意圖

本研究將上述之 CBCSA 之程序步驟以流程圖表達如下圖 14 所示，以及 CBCSA 虛擬碼如下:

Algorithm Pseudo codes：

Algorithm1

u=1；

$Layer_u = \{node_1^1,\ node_2^1, ..., node_n^1\}$ ；

While ($|Layer_u| \neq 1$**) Do**

k=2；

While ($k \le k_{\max}$**) Do**

c=1；

$Cluster_c$=$Empty$ ；

Setup the initial

matrix $A^1,\ a_{ij}^1 \in A^1,\ i, j \in [1,...,n]$ ；

While ($A^1 \neq Empty$**) Do**

c=c+1；

Record all $P_{ij}^1(1)$**;**

For (*z*=2; *z*=*k*; *z*++)

$A^z = A^{z-1} \times A^1,\ a_{ij}^z = 0, \forall i = j$;

If (z > 2) Then

For each value of $a_{ij}^z \in A^z,\ i \neq j$ ，

regulate them by procedures as:

For (*g*=1; *g*=$|v|$**; g++)**

For (*h*=1; *h*=*n*; *h*++)

If ($P_{hj}^1(1) \in P_{ih}^{z-1}(g)$) Then

a_{ij}^z=$a_{ij}^z - 1$ ；

End

End

End

End

End

$D = \sum_{i=1}^{k} k \times A^k$ ；

r=1；

$D_{temp} = Empty$ ；

While ($D \neq Empty$ **) Do**

d_{xy}=$\max\{d_{ij} \in D,\ \forall i, j,\ d_{ij} \notin D_{temp}\}$ ；

$CN_r = \{node_q \mid q \in P_{xy}^k(v),\ v = 1,..., a_{xy}^k\}$ ；

CN_Num_r=$|CN_r|$ ；

$B = Matrix(CN_r)$ ；

$CA\_Num_r = \frac{\sum_{\forall i,j} b_{ij}}{2},\ b_{ij} \in B$ ；

$\theta_r = \frac{CA\_Num_r}{CN\_Num_r}$ ；

$D_{temp} = D_{temp} \mid + d_{ij}$ ；

r++；

End

$Cluster_c$=$Matrix(CN_q)$ ；

$\theta_q = \max\{\theta_r,\ \forall r\}$;

$A^1 = A^1 \mid\sim Cluster_c$;

End

k++；

End

u++；

Algorithm2;

Algorithm3;

End

Algorithm2

Record all constraint *cluster*$_c$;251

$Layer_u = \{node_c^u \mid node_c^u = Cluster_c,\ \forall c\}$ ；

Algorithm3

Setup the initial matrix

$E^1, e_{ij}^1 \in E^1, i, j \in [1,...,n]$;

while($|Layer_u| \neq 1$**)Do**

Record all constraint *cluster*$_c$;

c_{ij}=$\{node_h\}, i \neq j$, i, j=$1,...,n$;

while($k \le k_{\max}$ **or** $A^k \neq Empty$ **)Do**

For (*z*=2; *z*=*k*; *z*++)

$A^z = A^{z-1} \times A^1,\ a_{ij}^z = 0, \forall i = j$;

For each value of $a_{ij}^z \in A^z,\ i \neq j$**;**

regulate them by procedures as:

For (*g*=1; *g*=$|v|$**; g++)**

For (*h*=1; *h*=*n*; *h*++)

If ($P_{hj}^1(1) \in P_{ih}^{z-1}(g)$ **)**

e_{ij}^z=$e_{ij}^z - 1$ ；

Else If($P_{hj}^1(1) \in c_{ij}$ **)**

e_{ij}^z=$e_{ij}^z - 1$ ；

End

End

End

End

End

k++;

End
u++;
End

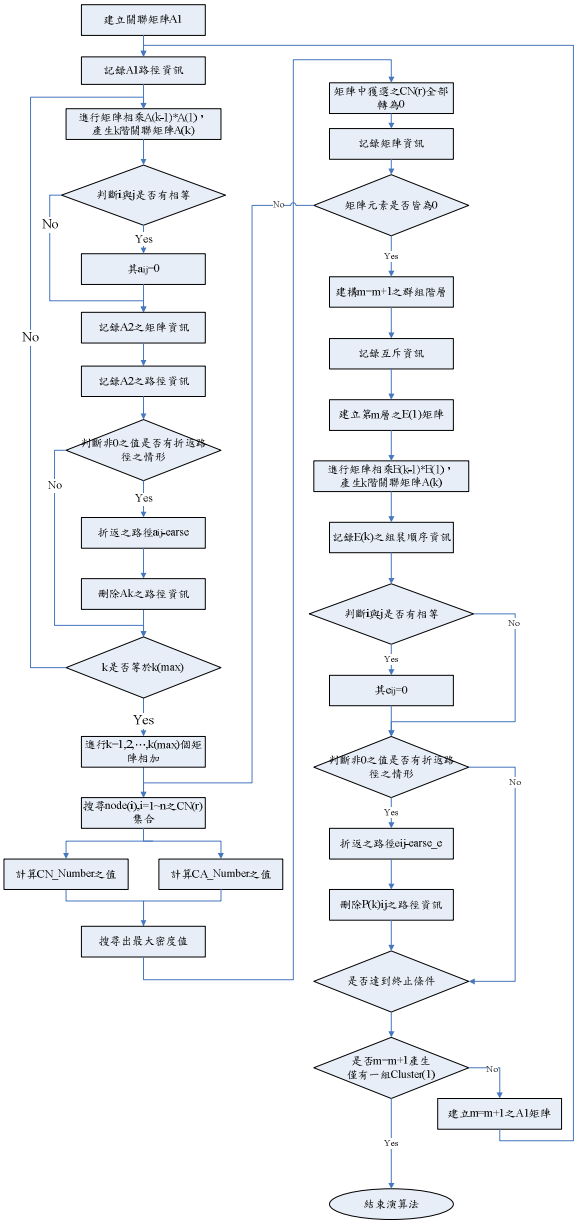


圖 14. CBCSA 演算法流程

5. 實例分析

5.1. CBCSA 演算法

本章以釘書機之 12 個零組件實例來分析 NDCA 分群演算法之效果，實例如下圖 16 釘書機之 3D 爆炸圖以及圖 17 為零組件的關聯網路圖做為表示，進行零組件模組化之分群。在密切度分群演算法的參數設定方面，研究中將以步階數為 2 進行模組化結構的階層表示。

首先針對釘書機之零組件工程屬性，建立釘書機結構之料結構表，表示如下圖 15。其運算的步驟說明如下：

組件節點	組件構建屬性		組件連結關係	組件結合特徵屬性				組件裝配資訊屬性	
	組件材料	組件型態		接觸面特性	connecter 屬性			組裝時間	組裝成本
					結合方向	結合方式	工具性		
1	鐵	-	2	單面接觸	(0,0,0,0,0,1)	C5	手	-	-
			4	單面接觸	(0,0,0,0,0,1)	C5		-	-
2	鐵	-	1	單面接觸	(0,0,0,0,1,0)	C5	手	-	-
4	鋼	-	1	單面接觸	(0,0,0,0,1,0)	C5	手	-	-
			5	單面接觸	(0,0,0,0,0,1)	C6	小工具	-	-
			8	單面接觸	(0,0,0,0,0,1)	C5	手	-	-
5	鋼	-	4	單面接觸	(0,0,0,0,1,0)	C6	小工具	-	-
			6	單面接觸	(0,0,0,0,0,1)	C5	手	-	-
6	鋁	-	5	單面接觸	(0,0,0,0,1,0)	C6	手	-	-
			8	單面接觸	(0,0,0,0,0,1)	C5		-	-
			9	單面接觸	(1,0,0,0,0,0)	C8		-	-
7	鐵	-	9	單面接觸	(1,1,0,0,0,0)	C8	手	-	-
8	鋼	-	4	單面接觸	(0,0,0,0,1,0)	C5	手	-	-
		-	6	單面接觸	(0,0,0,0,1,0)	C5		-	-
		-	12	單面接觸	(0,0,0,0,1,0)	C1		-	-
9	鐵	-	6	單面接觸	(0,1,0,0,0,0)	C8	手	-	-
		-	9	單面接觸	(1,1,0,0,0,0)	C8		-	-
7	鐵	-	9	單面接觸	(1,1,0,0,0,0)	C8	手	-	-
8	鋼	-	4	單面接觸	(0,0,0,0,1,0)	C5	手	-	-
		-	6	單面接觸	(0,0,0,0,1,0)	C5		-	-
		-	12	單面接觸	(0,0,0,0,1,0)	C1		-	-
9	鐵	-	6	單面接觸	(0,1,0,0,0,0)	C8	手	-	-
		-	9	單面接觸	(1,1,0,0,0,0)	C8		-	-
11	鋁	-	12	單面接觸	(0,0,0,0,0,1)	C5	手	-	-
12	鋼	-	8	單面接觸	(0,0,0,0,0,1)	C1	手	-	-
		-	11	單面接觸	(0,0,0,0,1,0)	C5	手	-	-
		-	14	線接觸	(0,0,0,0,0,1)	C5	小工具	-	-
		-	15	多面接觸	(0,1,0,0,0,0)	C1	手	-	-
14	鋼	-	12	線接觸	(0,0,0,0,1,0)	C5	手	-	-
15	鋼	-	12	多面接觸	(1,0,0,0,0,0)	C5	手	-	-
16	混合	-	17	多面接觸	(0,0,0,1,0,0)	C6	手	-	-
17	混合	-	16	多面接觸	(0,0,1,0,0,0)	C6	手	-	-
		-	18	多面接觸	(1,0,0,0,0,0)	C8	手	-	-
		-	19	點接觸	(1,1,0,1,0,0)	C1	手	-	-
18	混合	-	17	多面接觸	(0,1,0,0,0,0)	C8	手	-	-
19	混合	-	19	點接觸	(1,1,1,0,0,0)	C1	手	-	-

圖 15 釘書機組件資料結構

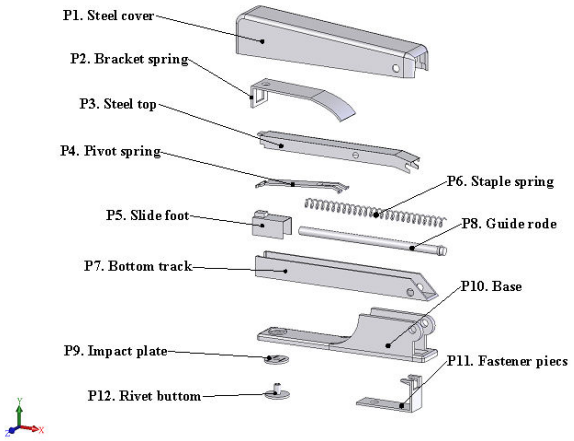


圖 16. 釘書機 3D 爆炸圖

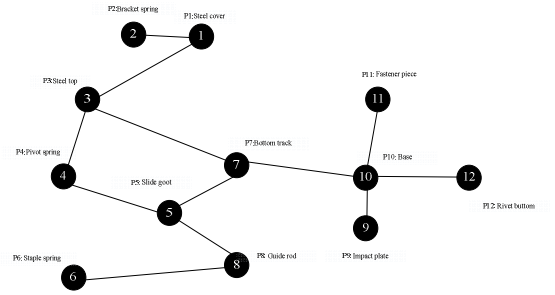


圖 17. 組件關聯圖

Step_1: 一步階關聯矩陣運算(*k=1*)

由 12 個節點所構成之12×12矩陣 A^1 即為一步階關聯矩陣，一步階關聯矩陣可表達網路節點間之前後連結關係，矩陣中的元素 a_{ij}^1, $1 \le i, j \le 12$ 亦可表示為 $node_i$ 經由 1 步連結到 $node_j$ 的可行路徑數目，如公式(19)所示。

$$A^1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \quad (19)$$

Step_2: 二步階關聯矩陣運算(*k=2*)

A^2 可經由一步階之矩陣的相乘運算 $A^1 \times A^1$ 獲得，由於上述 17 式已將一步階矩陣節點原地踏步的情況排除，即 $a_{ij}^1 = 0$, i=j, $1 \le i, j \le n$，故二步階陣節點原地踏兩步的情況，在矩陣運算的過程中也會排除之，其二步階之計算結果與二步階網路圖分別，如下公式(20)及圖 18 所示。

(20)式之下列矩陣，其中 a_{35}^2=2 表示 $node_3$ 到 $node_5$ 存在二條二步階連結，分別為 P_{35}^1=<3-4-5>與 P_{35}^2=<3-7-5>此二條連結路徑，其中節點 4 與 7 分別為 $node_3$ 到 $node_5$ 所存在二條之間隔節點。

$$A^2 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad (20)$$

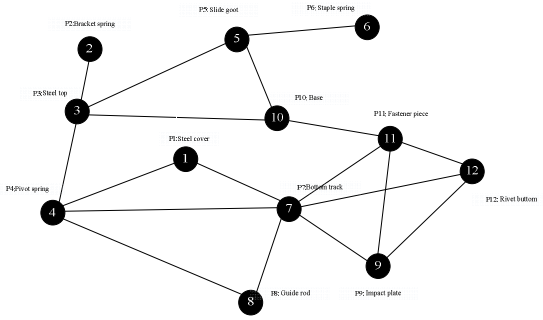


圖 18. 二階組件關聯圖

Step_3: 計算步階密度矩陣

我們可利用上述 1 步及 2 步關聯矩陣間的運算第 1 個群組之步階密度矩陣 Ds 產生如下 21 式。

$$D_1 = A^1 + 2 \times A^2$$

$$= \begin{bmatrix} 0 & 1 & 1 & 2 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 0 & 1 & 4 & 0 & 1 & 0 & 0 & 2 & 0 & 0 \\ 2 & 0 & 1 & 0 & 1 & 0 & 4 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 1 & 0 & 2 & 1 & 1 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 1 & 4 & 1 & 0 & 0 & 2 & 2 & 1 & 2 & 2 \\ 0 & 0 & 0 & 2 & 1 & 1 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 1 & 2 & 2 \\ 0 & 0 & 2 & 0 & 2 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 2 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 2 & 1 & 2 & 0 \end{bmatrix} \quad (21)$$

Step_4：計算群組內節點數目

自 D_s 中找出最大的 d_{ij}, $i, j \in [1,...,12]$，透過 2 步階關聯矩陣 A^2，找出所有自 $node_i$ 出發到 $node_j$ 結束，將中間所有經過的節點紀錄為 CN_r 集合如下表 4 所列:

表 4. 所有 CN_number 之計算結果

CN_1={1,2,3,4,7}	*CN_number*=5

$CN_2=\{2,1,3\}$	$CN\_number=3$
$CN_3=\{1,2,3,4,5,7,10\}$	$CN\_number=7$
$CN_4=\{1,3,4,5,7,8\}$	$CN\_number=6$
$CN_5=\{3,4,5,6,7,8,10\}$	$CN\_number=7$
$CN_6=\{5,6,8\}$	$CN\_number=3$
$CN_7=\{1,3,4,5,7,8,9,10,11,12\}$	$CN\_number=10$
$CN_8=\{4,5,6,7,8\}$	$CN\_number=5$
$CN_9=\{7,8,10,11,12\}$	$CN\_number=5$
$CN_{10}=\{3,5,7,9,10,11,12\}$	$CN\_number=7$
$CN_{11}=\{7,9,10,11,12\}$	$CN\_number=5$
$CN_{12}=\{7,9,10,11,12\}$	$CN\_number=5$

Step_5：計算群組內節點相互連結密度

本步驟將針對 CN_r 內所有的節點，其內所有相互連結的路徑數目，並將其令為 CA_number，計算結果如下表 5:

表 5. 所有 CA_number 之計算結果

$CA\_number=\frac{28}{2}=14$	$CA\_number=\frac{8}{2}=4$
$CA\_number=\frac{50}{2}=25$	$CA\_number=\frac{50}{2}=25$
$CA\_number=\frac{50}{2}=25$	$CA\_number=\frac{8}{2}=4$
$CA\_number=\frac{84}{2}=42$	$CA\_number=\frac{28}{2}=14$
$CA\_number=\frac{32}{2}=16$	$CA\_number=\frac{52}{2}=26$
$CA\_number=\frac{32}{2}=16$	$CA\_number=\frac{32}{2}=16$

根據上述之計算，若將 CN_r 內所有節點構成的群組，其網路節點密度令為 θ_r，則該值計算如下表 6：

表 6. 所有 θ_r 之密度值計算結果

$\theta_1=\frac{5}{14}=0.357$	$\theta_1=\frac{3}{4}=0.75*$
$\theta_3=\frac{7}{25}=0.28$	$\theta_4=\frac{6}{25}=0.24$
$\theta_5=\frac{7}{25}=0.28$	$\theta_6=\frac{3}{4}=0.75*$
$\theta_7=\frac{10}{42}=0.238$	$\theta_8=\frac{5}{14}=0.357$
$\theta_9=\frac{5}{16}=0.5$	$\theta_{10}=\frac{7}{26}=0.269$
$\theta_{11}=\frac{5}{16}=0.313$	$\theta_{12}=\frac{5}{16}=0.313$

Step_7：決定出網路第 1 個分群

自 D_s 中找出下一個次大的 $d_{ij}, i,j\in[1,...,n]$，重複 Step_4 及 Step_5，並計算出 θ_r，同時也記錄其相對應之 CN_r。計算後之結果 $\theta_{\max}=\{\theta_2,\theta_6\}$，因此隨機選擇 θ_2 的元素 $CN_2=\{2,1,3\}$ 集合會作為 $Cluster_1$，並取 $node_2$ 至 $node_2$ 之一步階為群組中心線，其群組 1 之結果表示如圖 19。且將 CN_r 內所有的節點也會從矩陣 A^1 中移除，移除後之矩陣結果如(22)式。

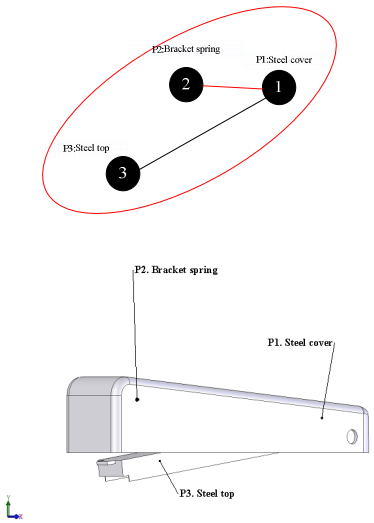


圖 19. 群組 1 與實體模組模型

$$D_2=\begin{bmatrix}0&0&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&4&0&0&0&0\\0&0&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0&0&0\\0&0&0&4&0&0&0&0&2&1&2&2\\0&0&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&2&0&0&1&2&2\\0&0&0&0&0&0&1&0&1&0&1&1\\0&0&0&0&0&0&2&0&2&1&0&2\\0&0&0&0&0&0&2&0&2&1&2&0\end{bmatrix} \quad (22)$$

Step_8：找尋網路節點下一個群組

令 $r=r+1$，針對下一個網路節點群組 $Cluster_r$ 作求解，將求得重複 Step_1 到 Step_7 之執行步驟，直至所有零組件皆有群組歸屬，其求解之結果分別為 $r=2$ 時 $\theta_6=3/4=0.75$ 為最大值，所以將 θ_6 作為 $Cluster_2$ 之分群 CN_2

={5,6,8}；r=3 時 $\theta_4 = 2/4 = 0.5$ 為最大值,因此 $Cluster_3$ 為 CN_2 ={4,7}；r=4 時 $\theta_9 = \theta_{10} = \theta_{11} = \theta_{12} = 4/9 = 0.444$ 皆為最大值,因此我們隨機選取 θ_9 的連結集合 CN_2 ={9,10,11,12} 作為 $Cluster_4$，如此一來便完成所有零組件之分群作業。

Step _9：網路節點第 m=1 層分群的結束

重複 Step_8 執行的結果，會產生網路第 1 層所有的群組，並將其令為 $Cluster_1$~$Cluster_w$。其最後完成 m=1 階層的分群，以下圖 20 為釘書機總共分作四個模組區塊。

Step_10：針對第 *m=1* 層內所有群組產生組裝順序

Step_10.1：產生第 *r=4* 群組之 E^1 矩陣

例如由 4 項組件所構成之 $Cluster_4$ 模組，其關聯資訊建立成為 1 步階矩陣 E^1 如下:

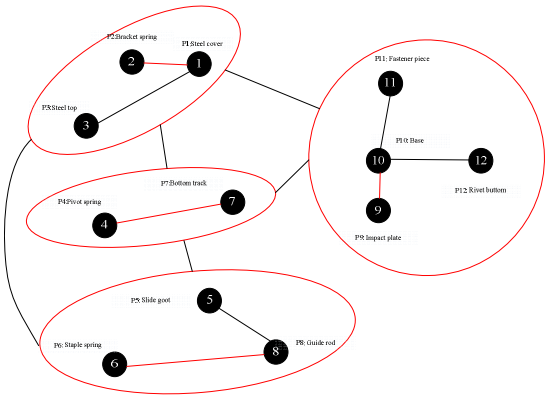


圖 20. 釘書機分群結果

$$E^1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \quad (23)$$

經由組件組裝取上獲得模組間組件互斥資訊，以下為 $Cluster_4$ 模組互斥資訊:

x_{912}=x_{219}={10}

Step_10.2：*k=2* 步階關聯矩陣運算，並考慮組件組裝限制

參考組裝元件間相連互斥之條件 c_{912}=c_{219}={10}，產生 2 步階矩陣 E_2，其結果如下 24 式。

$$E^2 = E^1 \times E^1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \quad (24)$$

由上式之 E^2 結果排列出之二階組裝順序如下:

$P_{12}^2(1)$=<1-2-3>、$P_{13}^2(1)$=<1-2-4>、

$P_{31}^2(1)$=<3-2-1>、$P_{34}^2(1)$=<3-2-4>、

$P_{41}^2(1)$=<4-2-1>、$P_{43}^2(1)$=<4-2-3>。

Step_10.3：完成 1~4 群組之裝配順序

組裝元件間檢查程序與第 3 階段之矩陣

$$E^3 = E^2 \times E^1 = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (25)$$

由於在 3 步階時，矩陣內元素 eij=0 時，則上一階之矩陣為最後之組裝順序，由於此模組內擁有 4 個組件，其最佳之關聯路徑因為 4-1=3，但此模組內之最長路徑數僅擁有 2 步階的長度，我們必需將未分配之裝配順序的第 4 個組件加入其組裝順序中，使第 4 個組件作為最後的裝配程序。如下述<1-2-3-4>、<1-2-4-3>、<3-2-1-4>、<3-2-4-1>、<4-2-1-3>、<4-2-3-1>。最後，將 *r=1~4* 之群組的所有 3 步階可行之裝配順序列於下表:

表 7. 所有群組 3 步階組裝順序

$Cluster_1$	<3-1-2>、<2-1-3>
$Cluster_2$	<1-2-3>、<3-2-1>
$Cluster_3$	<1-2>、<2-1>
$Cluster_4$	<1-2-3-4>、<1-2-4-3>、<3-2-1-4>、<3-2-4-1>、<4-2-1-3>、<4-2-3-1>

Step_10.5：結束第 1 層之組裝順序運算

由於第 1 層所有之零組件皆已排列出組裝順序，意即 3 步階之矩陣內元素 $e_{ij}=0$。

Step_ 11：持續網路節點第 *m=m+1* 層分群

原尚未執行分群的網路節點可視為網路第 m=1 層，節點經執行上述分群步驟後，所形成的群組連結型態，可視為網路第 $m = 2$ 層。

Step_11.1：將群組以節點作為表示

將上一層分群後的群組，均視為一節點，並將其相關節點屬性表列之，最後建構出一相互連結之網路圖。下圖為第二層的節點網路連結情況如圖 21 所列。

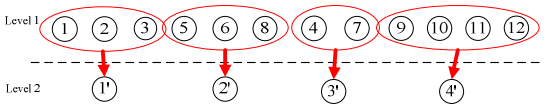


圖 21. 第二層之分群結果

Step_11.2：重覆執行 Step_1~Step_10

重覆執行 Step_1~Step_10，求出第二層分群之結果。如下圖 23 所列為第三層的節點網路連結情況以及圖 22 為第三層之四塊模組之裝配順序。

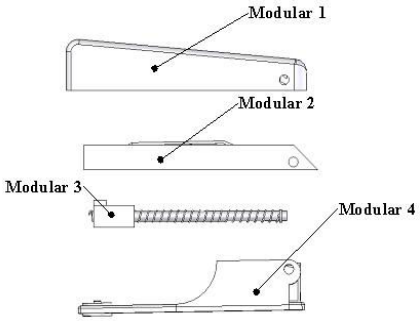


圖 22. 釘書機四大模組塊

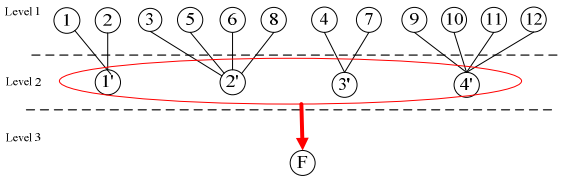


圖 23. 第三層之分群結果

最後，由此四個模組所排列成的四個可行組裝順序，如下述:

<2-3-1-4>、<3-1-2-4>、<3-1-2-4>、<4-2-3-1>

Step12：重覆 Step_11 的步驟

重覆 Step_11 的步驟，網路節點經過多層次的分群後，最後終會形成單一群組，該最後的群組即為根節點(Root node)。整個分群的過程，各層節點關係可表示成一樹狀結構，該結構表示如圖 24。

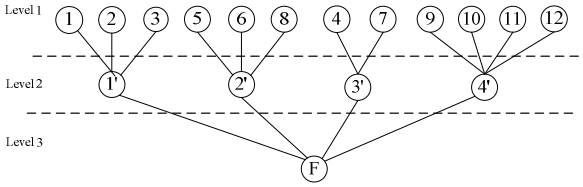


圖 24. 最終之樹狀結構

最後，經由一連串 CBCSA 演算法步驟，可獲得 192 條的可行路徑，其中模組 1 有 2 種可行的組裝次序，模組 2、模組 3 和模組 4，分別有 2 種、2 種與 6 種不同的組裝次序。將其組裝次序結果彙整如下表 8：

表 8. 組件至成品的完整 BCAS

	編號	BCAS	路徑數
半成品	M1	<3-1-2>、<2-1-3>	2
	M2	<5-6-8>、<8-6-5>	2
	M3	<4-7>、<7-4>	2
	M4	<9-10-11-12>、<9-10-12-11>、<11-10-9-12>、<11-10-12-9>、<12-10-9-11>、<12-10-11-9>	6
最終成品	F	<M2-M3-M1-M4>、<M3-M1-M2-M4>、<M2-M1-M3-M4>、<M4-M2-M3-M1>	4

6. 結論

產品模組化設計的分群演算法是非常重要的，若干複雜的產品其組裝元件動則上百甚至上千件，這類產品往往更需要模組化的設計，若採用一般啟發式演算法是無法在實務上有所貢獻的。本研究所發展之NDCA分群演算法亦以此作為研究動機，期能符合產品設計的條件要求，主要貢獻有下列兩項內容：

1. 能針對任何種型態的組裝元件作分群演算；就如同前文所述，組裝元件並非一般的資料或數值，它存在許多工程上的屬性，要考慮到實務面的組合情況，因此需要分析的面向非常廣，因此在分群的計算分析上，考量的層面很廣也較為複雜及多元。由其最重要的，所有組裝元件的屬性紀錄還要能與網路結構圖結合，這也是產品設計分群的重要步驟之一。
2. 探討產品組件限制下裝配順序最終要求能提供該產品所有組裝元件的組裝干涉資訊，以順遂後續生產製程、行銷、綠色設計、同步工程及供應鏈管理。因此就實務而言，產品裝配順序最主要之目的是達成產能最大、裝配時間最短以及最小之成本等績效。

文章最後以已訂書機為例，作為本文 CBCSA 演算之範例，依限於範例篇幅，雖然該裝置的元件不算多，但是仍能依本文之預期，有效率的達成釘書機組件最佳之模組裝配次序。其較佳之結果提供未來可與其它最佳化裝配次序演算法作比較，以分析優缺點。同時能夠納入其它原件各種工程資訊，發揮 CBCSA 在產品組裝上各種類型之貢獻，這也是本研究後續之未來發展方向。

參考文獻

[1] Tseng, M. M. and Jiao, J. Merchant, M. E., "Design for mass customization," ***Annals of the CIRP***, Vol. 45, No. 1, pp. 153-156, 1996.

[2] Ong, N. S. and Wong, Y. C., "Automatic subassembly detection from a product model for disassembly sequence generation," ***International Journal of Advance Manufacturing Technology***, Vol. 15, No. 6, pp. 425-431, 1999.

[3] Kusiak, A., and Huang, C. C., "Development of modular products," ***IEEE Transactions on Components, Packaging, and Manufacturing Technology-Part A***, Vol. 19, No. 4, pp. 523-538, 1996.

[4] Huang, C. C., "Overview of modular product development", ***Proc. Natl. Sci. Counc. ROC(A)***, Vol. 24, No. 3, pp. 149-165, 2000.

[5] Mckay, A., Erens, F., and Bloor, M. S., "Relating product definition and product variety", ***Research in Engineering Design***, Vol. 8, No. 2, pp. 63-80, 1996.

[6] Stone, R. B., Wood, K. L., and Crawford, R. H., "A heuristic method for identifying modules for product architectures", ***Design Studies***, Vol. 21, No. 1, pp. 5-31, 2000.

[7] Salhieh, S. M., and Kamrani, A. K., "Macro level product development using design for modularity", ***Robotics and Computer Integrated-Manufacturing***, Vol. 15, No. 4, pp. 319-329, 1999.

[8] Huang, C. C., and Liang, W. Y., "A formalism for designing with modules", ***Journal of the Chinese Institute of Industrial Engineers***, Vol. 18, No. 3, pp. 13-20, 2001.

[9] Al-Sultan, K. S., "A tabu search approach to the clustering problem", ***Pattern Recognition*** , Vol. 28, No. 9, pp. 1443-1451, 1995.

[10] Xu, H. B., Wang, H. J., Li, C. C., "Fuzzy tabu search method for the clustering problem", ***IEEE Proceedings International Conference on Machine Learning and Cybernetics ICMLC-02***, pp. 876 - 880, 2002.

[11] Bandyopadhyay, S., Maulik, U., MK Pakhira, "Clustering using simulated annealing with probabilistic redistribution" ***International J. Patt. Recog. and Artif. Intell***, Vol. 15, No. 2, pp.269–285, 2001

[12] Tseng, L. Y., "Genetic algorithm for clustering, feature selection and Classification", ***1997 IEEE International Conference on Neural Networks***, Vol. 3, pp. 1612-1616, 1997.

[13] Maulik, U. and Bandyopadhyay, S., "Genetic algorithm-based clustering technique," ***Pattern Recognition***, Vol. 33, No. 9, pp.1455-1465, 2000.

[14] Saha, S. and Bandyopadhyay, S., "A fuzzy genetic clustering technique using a new symmetry based distance for automatic evolution of clusters," ***Theory and Application ICCTA2007***, pp. 309-314, 2007.

[15] Chiou, Y. C., and Lan, L. W., "Genetic clustering algorithms," ***European Journal of Operational Research***, Vol. 135, No. 2, pp. 413-427, 2001.

[16] Handl, J. and Meyer, B., "Improved Ant-based Clustering and Sorting in a Document Retrieval Interface," ***Proceedings of the Seventh International Conference on Parallel Problem Solving from Nature (PPSN VII) 2002***, LNCS 2439, pp. 913-923, 2002.

[17] Shelokar, P. S., Jayaraman, V. K. and Kulkarni, B. D., "An ant colony approach for clustering," ***Analytica Chimica Acta*** Vol. 509, No.2, pp. 187-195, 2004.

[18] Kao Y. and Cheng K., "An ACO-Based

Clustering Algorithm," ***ANTS 2006***, LNCS 4150 ,pp. 340-347, 2006.

[19] van der Merwe, D. W. and Engelbrecht, A. P. "Data clustering using particle swarm optimization," ***Proceedings of IEEE Congress on Evolutionary Computation 2003***, pp. 215-220, 2003.

[20] Smith G. C. and Smith S. S. F., "An enhanced genetic algorithm for automated assembly planning," ***Robotics and Computer Integrated Manufacturing***, Vol. 18, No.5-6, pp. 355-364, 2002

[21] Zhang Y. Z., Ni J., Lin Z. Q. and Lai X. M., "Automated sequencing and sub-assembly detection in automobile body assembly planning ", ***Journal of Materials Processing Technology***, Vol. 129, No. 1-3, pp. 490-494, 2002.

[22] Xing Y., Chen G., Lai X., Fin S. and Zhou F., "Assembly sequence planning of automobile body components based on liaison graph", ***Assembly Automation***, Vol. 27, No. 2, pp. 157-164, 2007.

[23] Sundaram, S. Remmler, I. and Amato, N. M. "Dissassembly Sequencing Using a Motion Planning Approach", ***IEEE International Conference on Robotics and Automation***, Vol. 2, pp. 1475-1480, 2001.

[24] Wakamatsu, H. Tsumaya, A. Shirase, K. and Arai, E., "Development of Disassembly Support System for Mechanical Parts and Its Application to Design Considering Reuse/Recycle," ***IEEE Environmentally Conscious Design and Inverse Manufacturing***, pp. 372-377, 2001.

[25] Lanham, J. and Dialami, F., "The Assembly State Vector: a new approach to the generation of assembly sequences," ***IEEE International Symposium on Assembly and Task Planning***, pp. 37-42, 2001.

[26] Mascle, C. and Balasoiu, B. A., "Disassembly-Assembly sequencing using feature-based life-cycle model", ***IEEE Ieternational Symposium on Assembly and Task Planning***, pp. 31-36, 2001.

[27] Imamura, S. Masaki, H. and Nakazawa, Y., "Disassembly Planning for Machine Maintenance with Multi-Agent Cooperation," ***IEEE International Symposium on Assembly and Task Planning***, pp. 214-219, 2001.

[28] Tseng H. E. and Chang C. C. and Cheng C. C., "Disassembly-oriented assessment methodology for product modularity," ***International Journal of Production Research***, pp.1-24, 2009.

[29] De Fazio, L. T. and Whitney, D. E., "Simplified generation of all mechanical assembly sequences", ***IEEE Journal of Robotics and Automations***, Vol. 3, No. 6, pp. 640-658, 1987.

[30] Mikkola, J. H. and Gassmann O., "Managing modularity of product architectures: Toward an integrated theory," ***IEEE Transactions on Engineering Management***, Vol. 50, No. 2, pp. 204-218, 2003.

[31] Akagi, F., H. Osaki, and S. Kikuchi, "The method of analysis of assembly work based on the fastener method," ***Bulletin of the JSME***, Vol. 23, No. 184, pp. 1670-1675 , 1980.

[32] Tseng, H. E. Chang, C. C. and Li, J. D., "Modular design to support green life-cycle engineering", ***Expert Systems with Applications***, Vol. 34, No. 4, pp. 2524-2537, 2008.