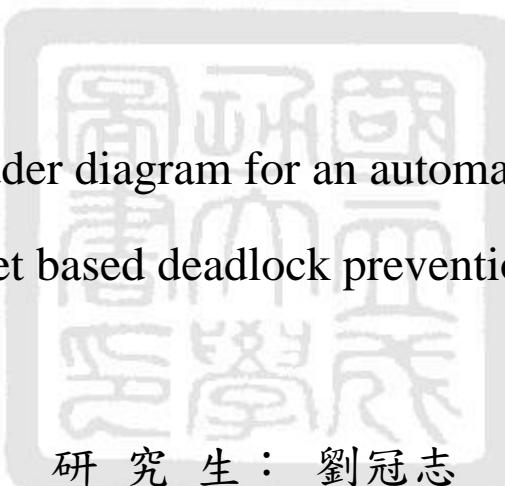


國立成功大學
工業與資訊管理研究所
碩士論文

利用裴氏圖之鎖死預防策略探討自動化系統之階梯圖編程方
法

On generating a ladder diagram for an automatic system based on
a Petri net based deadlock prevention policy



研究 生： 劉冠志

指 導 教 授： 王逸琳 博士

中華民國一百零三年六月

國立成功大學
碩士在職專班論文

利用裴氏圖之鎖死預防策略探討自動化系統之階梯圖
編程方法

On generating a ladder diagram for an
automatic system based on a Petri net based
deadlock prevention policy

研究生：劉冠志

本論文業經審查及口試合格特此證明

論文考試委員：

王逸琳
林東昌 李子欣

指導教授：王逸琳

系(所)主管：
管宇綱

中 華 民 國 103 年 6 月 5 日

摘要

自動化系統常見於日常生活中，可程式邏輯控制器(Programmable Logic Controller, PLC)為其控制手段之一，階梯圖(Ladder Diagram, LD)則為常用的PLC 編程方法。一個自動化系統必須歷經規格、驗證、撰碼、測試與偵查五項步驟，然而階梯圖不易系統化建構與偵錯，為加速其開發流程，我們可先以易於描述離散型系統動態行為的裴氏圖(Petri Net, PN)建構，經過分析驗證後再轉換為對應的階梯圖。

在階梯圖的設計過程中，常會因編程人員的疏失而導致系統中的某個或多個工件彼此佔住資源，互相等待其它工件釋放資源而本身卻不釋放資源的「鎖死」(Deadlock)現象。此時若將階梯圖轉換為裴氏圖即可較易進行「鎖死分析」(Deadlock Analysis)，以找出並解決鎖死現象。基於裴氏圖的鎖死分析有裴氏圖的結構分析(基於 Siphon 的方法)與可達圖(Reachability Graph)分析等兩種手法，本研究擬應用區域理論(Theory of Region)與可達圖來分析鎖死問題，並運用裴氏圖化簡方法，降低產生可達圖時因裴氏圖的規模增加而導致的狀態數爆增問題。最後透過裴氏圖軟體 Petri LLD 將無鎖死的裴氏圖再轉換回階梯圖，以塑膠射出成型機為例，實現自動化系統建構的程序。

關鍵字：

階梯圖(Ladder Diagram)、裴氏圖(Petri Net)、鎖死分析(Deadlock Analysis)、區域理論(Theory of Region)

On Generating a Ladder diagram for an Automatic System Based on a Petri Net Based Deadlock Prevention Policy

Author: Guan-Zhi Liu

Advisor: I-Lin Wang

Industry and Information Management Department & National Cheng Kung University

SUMMARY

The Ladder diagram is the most popular programming language for the programmable logic controller (PLC) that is commonly used in controlling an automatic system. The Ladder diagram programmers usually suffer from the problem of deadlocks in which two or more competing processes waiting for each other to finish, and thus none of them continues. The deadlock problem of a ladder diagram is difficult to be detected and resolved, yet it is easier to conduct the deadlock analysis over the Petri net associated with a ladder diagram. We use the theory of region and the reachability graph to analyze deadlocks. By techniques of Petri net reduction, we could avoid the problem of state explosion for large-scale Petri nets. After resolving the deadlocks, we convert a deadlock-free Petri net back to a Ladder diagram. We illustrate how these procedures work by several real-world examples.

Key words: Ladder Diagram, Petri net, Deadlock analysis, Theory of Region

INTRODUCTION

Programmable logic controllers (PLC) are widely used in automatic systems. And the ladder diagram is the most popular programming language for PLC. The Ladder diagram programmers usually suffer from the problem of deadlocks in which two or more competing processes waiting for each other to finish, and thus none of them continues. The deadlock problem of a ladder diagram is difficult to be detected and

resolved, yet it is easier to conduct the deadlock analysis over the Petri Net associated with a ladder diagram. Therefore, it may be beneficial to convert a Ladder diagram to its corresponding Petri Net, analyze and resolve the deadlock in Petri Net, and then convert it back to a deadlock-free Ladder diagram.

MATERIALS AND METHODS

There are two approaches of Petri net based deadlock analysis, one is based on reachability graphs, and the other is Petri net structure analysis (siphon approach). Using the reachability graph to handle deadlock states in Petri net could get the maximum liveness solution, but it may be time consuming for generating the reachability graph. Here we reduce the size of Petri net to shorten the computational time, and then use the reachability graph and theory of regions to handle the deadlock states.

Figure 1 illustrates procedures of our research. First step we convert our Ladder diagram into a Petri net by a software called “Snoopy”; Second, we use another software called “Integrated Net Analyzer (INA)” to reduce the Petri net size and generate its reachability graph; Third, we conduct deadlock analysis. If a deadlock is detected, we generate its corresponding CMTSI equations by Visual Basic .net, and solve the equations by a mathematical programming software called “Gurobi”. Finally, we resolve the deadlocks, and convert the new Petri net to a Ladder diagram by a software called “Petri LLD”.

Previous researches either can only convert the Euler marked graph to Ladder diagrams [Huang & Liang, 2010], or can only convert the tokens bounded Petri net to Ladder diagram [Chen & Liang, 2013]. That is why we use the Petri LLD by Brusey [2006], since it can directly convert a signal interpreted Petri net to a Ladder diagram, and it is also adaptive to the structure of PLC.

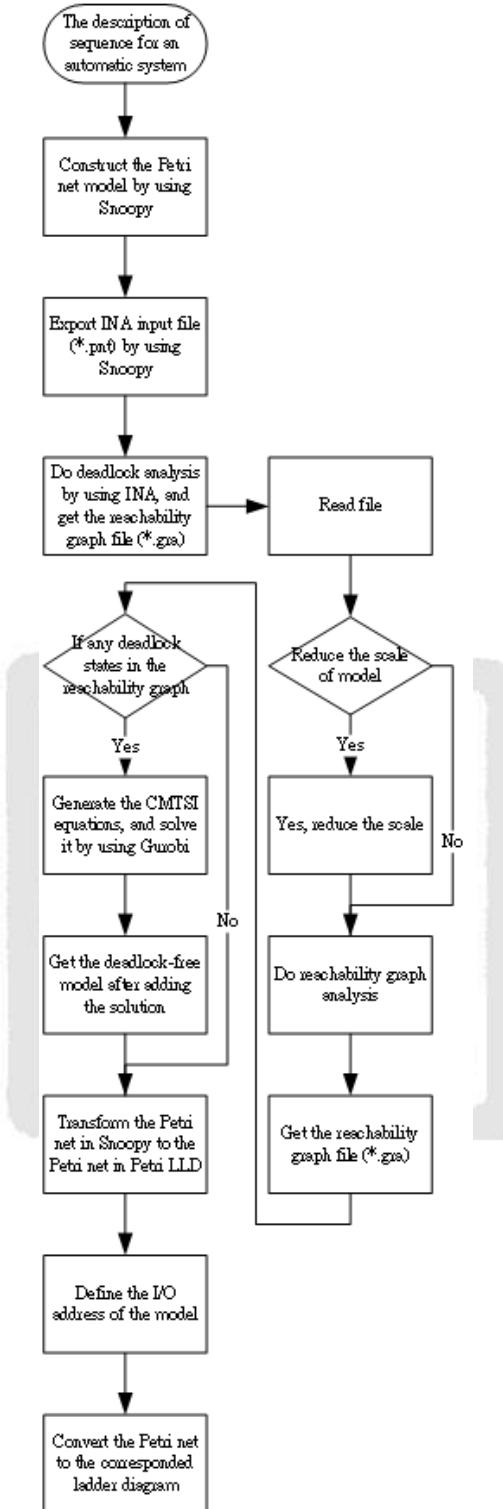


Figure 1. The building procedure of an automatic system

To make the entire procedures more automatically, we have designed a program called “Petri net deadlock solver in PLC”, as shown in Figure 2, which reads the reachability graph file and circuit file (all were generated from a software named

“Integrated Net Analyzer, INA”), generates the CMTSI equations automatically, and then solve the CMTSI equation by Gurobi optimizer (import the gurobi56.net.dll). Because Petri net was not the programming language for PLC, we have to convert the deadlock-free Petri net to its corresponded Ladder diagram, which can be done by using Petri LLD.

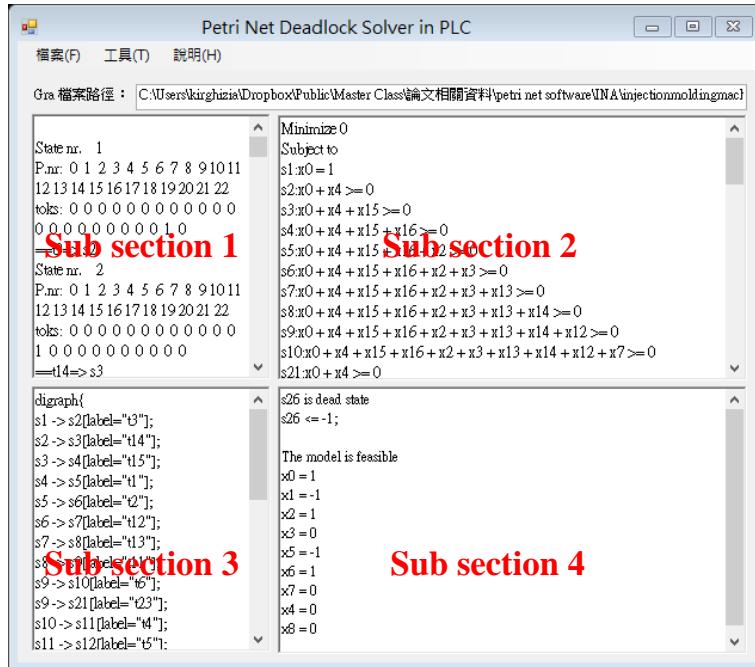


Figure 2. The program – Petri net deadlock solver in PLC

RESULTS AND DISCUSSION

The deadlock analysis approached used in this thesis are based on theory of region and CMTSI algorithm. We also apply some size reduction techniques to avoid state explosion in the analysis. The advantages of this approach cover three objectives: it could get the max reachable states, non-redundant control states and minimum calculating cost.

We have tested and demonstrated the entire building procedures on several real-world applications. The results indicate that our procedures could save much time in debugging to design Ladder diagrams free of deadlocks for automatic systems. This work thus speeds up the processes of Ladder diagram designs.

To suggest topics for future research, we have the following comments: First, among those Petri net data formats used by different softwares, the format of INA input file

is the most convenient one since it is simple, clear, and easy to analyze. Second, many PLC programmers are not familiar with Petri net, thus we suggest to develop the software that can convert Ladder diagrams or instruction lists to Petri net more conveniently. Such a software will shorten the development period of deadlock-free Ladder diagrams. Finally, we suggest to design some remote control software, using the vb.net and java output file of Petri LLD, and integrate it with some hydraulic simulation software (e.g. “Festo FluidSim”) to construct virtual simulation machine. The related research could refer to Li, Liu, and Sun [2007].



誌謝

在研究所期間，由於在職之故，並未能如一般學生全心投學業與論文著述，而能完成本論文，首先要感謝指導教授 王逸琳老師的悉心指導與鼎力協助。在繁忙的教學與研究外，尚撥冗予以逐字檢查斧正，使得學生在繁忙的工作壓力下，得以清楚掌握論文方向與架構，讓本論文在期限前完成。在此致上最誠摯的感謝與敬意。

口試期間，承蒙口試委員成功大學 李宇欣博士和成功大學 林東盈博士的不吝指導，提供本論文許多寶貴的意見，使本研究更臻於完善，在此深表謝意。在繁重的課業期間，感謝公司的同仁與主管，能因應我的狀況給予協助與照顧。並且要感謝研究所期間的學長、姐與同學們和學弟、妹，大家共同為學業努力奮鬥，與一起玩樂的歡樂時光。

最後要感謝我的家人和朋友們，感謝您們在我背後的支持，讓我得以無後顧之憂的往自己的目標前進。

劉冠志 謹誌

民國 103 年 8 月

目錄

摘要	1
Abstract	2
誌謝	7
1. 緒論	12
1.1 研究背景與動機	12
1.2 研究目的.....	15
1.3 研究範圍與架構	16
2. 文獻探討	18
2.1 自動化系統編程的建構方法	18
2.1.1、使用階梯圖進行自動化系統編程的建構程序	19
2.1.2、使用順序流程圖進行自動化系統編程的建構程序	23
2.2 裴氏圖.....	27
2.2.1 裴氏圖的基本元件	27
2.2.2 裴氏圖的基本性質	28
2.2.3 裴氏圖的鎖死問題分析	33
2.2.4 裴氏圖的化簡	44
3. 研究方法	46
3.1 建構裴氏圖：使用裴氏圖軟體 Snoopy	48
3.2 鎖死狀態處理：區域理論與關鍵標記/移轉分離性質(CMTSI)演算法	50
3.3 裴氏圖化簡與可達圖的生成：使用裴氏圖軟體 INA	62
3.4 CMTSI 方程組求解：使用數學規劃軟體 Gurobi.....	64
3.5 裴氏圖轉換為階梯圖：使用裴氏圖軟體 Petri LLD	74
4. 實例說明	77
4.1 自動搬運車系統	77
4.2 塑膠射出成型機簡介	80
4.3 塑膠射出成型機動作程式	84
5. 結論與後續研究建議	96
5.1 結論	96
5.1.1 裴氏圖鎖死處理	96
5.1.2 裴氏圖轉換為階梯圖	102
5.2 後續研究建議.....	103
參考文獻	104
附錄	108

表目錄

表 2-1、行動資格符號說明表	24
表 2-2、記號圖與非記號圖	29
表 2-3、循環式等待規則範例	36
表 2-4、裴氏圖鎖死問題研究的相關文獻整理 [陳音帆, 民 97]	40
表 2-5、裴氏圖的簡化法則	44
表 3-1、圖 3-4(a)裴氏圖的關聯矩陣	53
表 3-2、新增暫存點 P_c 在各狀態下浮標數的方程式表示	53
表 3-3、彈性製造系統鎖死狀態解	60
表 4-1、裴氏圖在 Snoopy 和 Petri LLD 的對照	93

圖目錄

圖 1-1、實體模擬台	13
圖 1-2、日本 Star Denshi PLC 階梯圖程式編輯器	14
圖 1-3、論文架構	17
圖 2-1、文獻探討架構	18
圖 2-2、自我保持迴路範例	20
圖 2-3、互鎖迴路範例	20
圖 2-4、歐姆龍 PLC 中的 SFC	26
圖 2-5、階梯圖、記號圖與法則式專家系統之關係 [梁高榮, 民 98]	29
圖 2-6、裴氏圖的分類[梁高榮, 民 100]	30
圖 2-7、自動搬運車系統的裴氏圖與對應的狀態圖 [梁高榮, 民 98]	31
圖 2-8、自動搬運車系統的可達圖 [梁高榮, 民 98]	31
圖 2-9、鎖死問題發生的範例	34
圖 3-1、本研究提出的自動化系統編程上的建構程序	47
圖 3-2、裴氏圖軟體 Snoopy	48
圖 3-3、裴氏圖與階梯圖之邏輯對應關係 [Lee & Hsu, 2004]	49
圖 3-4、標記/移轉分離性質(MTSI)範例說明一 [潘彥良, 民 100]	51
圖 3-5、標記/移轉分離性質(MTSI)範例說明二 [潘彥良, 民 100]	54
圖 3-6、彈性製造系統的示意圖與裴氏圖	58
圖 3-7、彈性製造系統的可達圖	58
圖 3-8、彈性製造系統裴氏圖解除鎖死狀態比較	61
圖 3-9、彈性製造系統裴氏圖的可達圖解除鎖死狀態比較	61

圖 3-10、裴氏圖分析軟體 INA	63
圖 3-11、裴氏圖分析軟體 INA 生成的可達圖中的迴圈資訊	63
圖 3-12、數學規劃軟體 Gurobi 執行程序圖	68
圖 3-13、本研究撰寫的裴氏圖鎖死求解軟體	69
圖 3-14、本研究撰寫的裴氏圖鎖死求解軟體執行流程圖	69
圖 3-15、子程式區塊 1 中顯示的可達圖輸出檔(*.gra)	70
圖 3-16、圖形語言軟體 Graphviz 的 gvedit 操作介面與範例說明	71
圖 3-17、子程式區塊 2 中顯示的可達圖的 Graphviz 程式碼	71
圖 3-18、子程式區塊 3 中顯示的 CMTSI 方程組	72
圖 3-19、將 INA 輸出的可達圖中迴圈資訊整理為循環方程式	73
圖 3-20、子程式區塊 4 中顯示的 MTSI 方程組求解結果	74
圖 3-21、Petri LLD 的範例：鑽孔機	75
圖 3-22、經由訊號詮釋裴氏圖轉換後的階梯圖	76
圖 4-1、自動搬運車系統的裴氏圖	78
圖 4-2、自動搬運車系統的可達圖(有鎖死狀態)	78
圖 4-3、自動搬運車系統的 CMTSI 方程組解	79
圖 4-4、自動搬運車系統的裴氏圖比較	79
圖 4-5、自動搬運車系統的可達圖比較	80
圖 4-6、塑膠射出成型機外觀圖	83
圖 4-7、塑膠射出成型機標準動作流程圖(自行整理)	83
圖 4-8、標準動作程式的動作流程圖與具有鎖死狀態的裴氏圖	85
圖 4-9、裴氏圖軟體 INA 的標準動作程式輸入檔	86
圖 4-10、裴氏圖分析軟體 INA	87
圖 4-11、裴氏圖分析軟體 INA 生成的可達圖中的迴圈資訊	87
圖 4-12、標準動作程式裴氏圖的求解結果與轉移點的編號和名稱對照	89
圖 4-13、標準動作程式「具鎖死狀態的裴氏圖」與「無鎖死狀態的裴氏圖」比較	91
圖 4-14、標準動作程式「具鎖死狀態的可達圖」與「無鎖死狀態的可達圖」比較	92
圖 4-15、轉換前與轉換後的裴氏圖比較	94
圖 4-16、經由裴氏圖軟體 Petri LLD 轉換後的標準動作程式階梯圖	95
圖 5-1、CMTSI 演算法與試誤法處理裴氏圖鎖死狀態的比較	99
圖 5-2、鑽孔機裴氏圖解除鎖死狀態前後的比較	100
圖 5-3、鑽孔機鎖死狀態求解結果與裴氏圖中的轉移點編號與名稱對照	100

圖 5-4、鑽孔機解除鎖死狀態前後的可達圖比較	101
圖 5-5、鑽孔機裴氏圖解除鎖死狀態前後的裴氏圖比較	101
圖 5-6、鑽孔機裴氏圖解除鎖死狀態後的可達圖(以試誤法解除) ..	102
圖 A-1、引用 Gurobi 的 DLL，進行數學規劃求解的程式碼	108
圖 A-2、裴氏圖軟體 INA 分析後的標準動作程式可達圖輸出檔(有鎖 死狀態)	109
圖 A-3、塑膠射出成型機動作程式的可達圖生成的 CMSTI 方程組	112
圖 A-4、塑膠射出成型機動作程式的可達圖生成的 CMSTI 方程組	113



1. 緒論

本章共有三個小節，分別是 1.1 節「研究背景與動機」、1.2 節「研究目的」和 1.3 節「研究範圍與架構」。

1.1 研究背景與動機

自動化是人類社會未來的趨勢，不論在工作、家庭等生活環境中，到處都可見自動化設備的蹤跡。現今自動化設備的控制方式由國際電工協會 (International Electro technical Commission, IEC)所訂定，可程式邏輯控制器 (Programmable Logic controller, PLC)是其中廣泛使用的方法。可程式邏輯控制器源自於美國通用汽車對生產線能夠快速變更的需求，先前的繼電器邏輯實體配線方式在變更設計時過於複雜與耗費成本，因此希望藉由可程式邏輯控制器，以軟體程式變更的方式取代硬體配線變更[宓哲民, 民 101]。

可程式邏輯控制器的編程標準由國際電工協會訂定，稱為 IEC 61131-3，包含五種編程語言：階梯圖(Ladder Diagram, LD)、順序流程圖(Sequential Function Chart, SFC)、基本指令(Instruction List, IL)、功能區塊圖(Function Block Diagram, FBD)與結構化文字(Structured Text, ST)五種規格，其中因所有的可程式邏輯控制器皆提供階梯圖編程方法，因此階梯圖最為廣泛使用。階梯圖是一種圖形化程式語言，類似傳統的繼電器電路，容易學習與變更設計。然而因為階梯圖的性質，輸出端的值是依據輸入端的值與計時器、計數器狀態所決定的，即使輸入端的值固定不變，輸出端的值還是會因持續的掃瞄而產生變化而形成不同的路徑。因為外部事件的變化與掃瞄率的關係，階梯圖的路徑很難藉由傳統的測試方法偵測到 [Aiken, Fahndrich, & Su, 1998] 。Grafcet PLC 程式語言的出現是第一次把裴氏圖(Petri net)運用在 PLC 上的試驗。然而 Grafcet 有著許

多限制，例如不能建構非程式性的流程。雖然 Grafcet 被 IEC 61131-3 重新命名為 Sequential Function Chart (SFC)，但是 Grafcet 只在法國盛行，階梯圖依舊是最主要的控制方式[Cohen, Wang, & Bidanda, 2010]。

由於 PLC 用來取代傳統實體繼電器配線，當機器動作有修改時，程式就必須修改，這種情況經常因應客戶需求發生；PLC 的階梯圖程式亦有容量限制，常見的容量為 8k Steps，亦即一個階梯圖程式僅可容納 8000 個動作指令(如：常開接點、常閉接點和輸出線圈等)；此外，階梯圖不像一般高階程式語言(如：C 語言、Visual Basic 等)可以用註解符號將用不到的程式碼忽略，只要是在階梯圖程式中撰寫的指令，就會經過掃瞄的程序。因此若是要在既有的階梯圖程式新增動作，可能就必須重新調整整個程式的架構，將不需要的程式碼刪除，以便能有多的空間寫入新的程式碼。實務上，我們會先將編寫好的階梯圖，如圖 1-2，安裝到圖 1-1 的實體的模擬台進行測試，如果有發現錯誤便進行修改，這是一個不斷試誤(Try & Error)的過程，並且易受限於實體模擬台的數量，使得新專案開發的速度降低，成本提高。



空氣壓縮機



日本星電機 Star Denshi PLC 模擬台

圖 1-1、實體模擬台

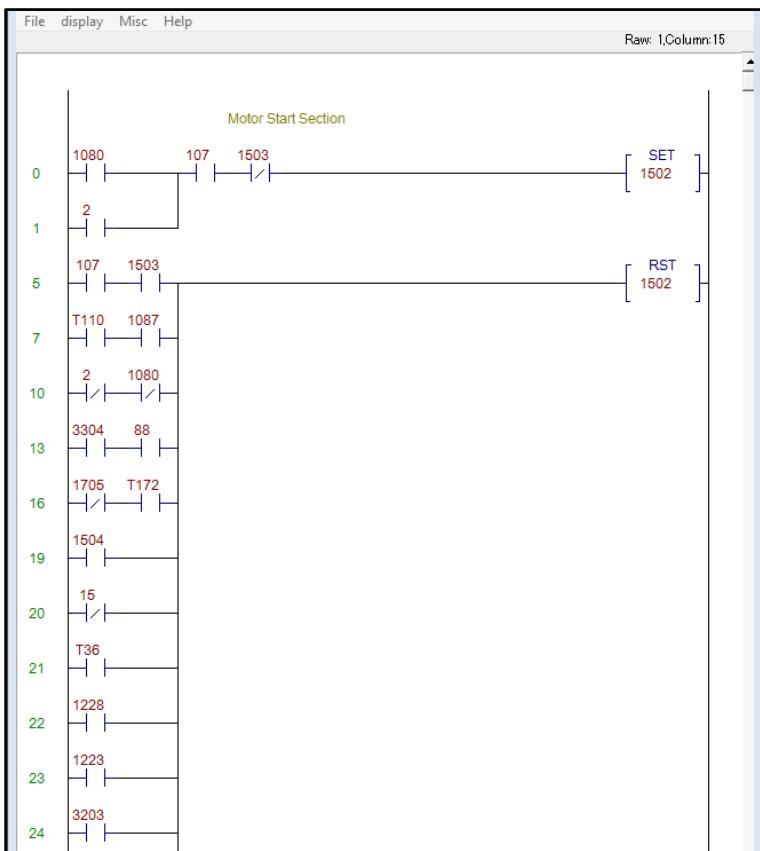


圖 1-2、日本 Star Denshi PLC 階梯圖程式編輯器

為了達到全面性的整體規劃，自動化系統的設計與實作必須經歷五大步驟：規格(Specification)、驗證(Validation)、撰碼(Coding)、測試(Testing)和偵查(Monitoring)[林潔好, 民 97]。目前提出整體自動化系統編程上的建構程序與階梯圖自動生成的相關文獻有[Han, 2010]：基於裴氏圖的方法、基於有限狀態機的方法和基於流程圖的方法。PLC 語言的實現方法有傳統的指令動作式、節點分析法、裴氏圖(Petri net)塑模法與 UML 塑模法[楊鎮宇, 何佳育, & 朱智煒, 民 99]等。由於裴氏圖適合用來描述動態系統的離散事件，並且可表達成矩陣模式，具備數學分析的性質。任何只要有流程特性的資料，皆可利用裴氏圖來表達。階梯圖本身具備流程特性，因此若將階梯圖轉換成裴氏圖將較易進行分析。規範測試可由裴氏圖推導出狀態可達圖(Reachability Graph)檢測系統中每個狀態

的轉換，確認程式設計與現場實際運作情況的一致性，並檢驗實際運作狀態轉換間的順暢，避免自動化系統發生鎖死(Deadlock)現象。

目前鎖死問題的解決策略可分為**鎖死預防(Deadlock Prevention)**、**鎖死偵測(Deadlock Detection)**、**鎖死避免(Deadlock Avoidance)**與**鎖死復原(Deadlock Recovery)**四種，而基於裴氏圖的解決方法則有三種：一是限制進入系統工件的數量，使裴氏圖保持活性(Liveness)，從而原則上保證系統不會發生鎖死情況，但是此法較為保守，會降低系統的產能與資源使用率；二是控制對資源的申請，使系統避免鎖死；三是改變裴氏圖的結構(基於 Siphon 的方法)，使系統不會鎖死[潘彥良, 民 100]。基於本研究的對象 - 塑膠射出成型機的特性屬於危險機工具，且具備重複製造系統與彈性製造系統的性質，故以第一項「限制進入系統工件的數量，使得原則上保證系統不會發生鎖死情況」作為本研究鎖死處理的方法。

1.2 研究目的

裴氏圖由於涉及多個浮標的互動行為，不容易直接目視分析。在此情況下，裴氏圖常轉成可達圖來分析其性質，可達圖是追蹤裴氏圖所有浮標流向的紀錄。然而運用可達圖來解決彈性製造系統的鎖死問題，是相當耗時且不易處理的，這也是為什麼現今大多數的專家學者均採取結構分析的方法(Siphon Control) [潘彥良, 民 100]。但以區域理論(Theory of Regions)與可達圖為基礎的鎖死分析，最大的優勢在於它能夠有效地求得擁有最大許可行為(可達狀態數)的控制方法 [李開南, 2010]。因此本研究擬藉由可達圖來進行基於區域理論的鎖死問題分析，並以關鍵標記/移轉分離性質(Crucial Marking / Transition Separating Instance, CMTSI) 演算法[潘彥良, 民 100]求解處理鎖死狀態，處理的過程為：首先依據自動化系統規格的文字描述、動作時序圖或是動作程序圖等，建構自

動化系統的裴氏圖，並藉由裴氏圖化簡方法來降低生成可達圖時，隨圖的規模增加所造成的狀態個數暴增之問題，再生成其相對應的可達圖。若可達圖中存在鎖死狀態，則以本研究撰寫的程式處理：首先依據區域理論與關鍵標記/移轉分離性質(CMTSI)演算法，將可達圖中的各個節點逐一列出，形成一組或多組CMTSI 方程組(依據可達圖中的鎖死狀態數量而定)。接著以數學規劃軟體 Gurobi 進行求解，並將求解結果代入原具有鎖死狀態的裴氏圖，得到一無鎖死狀態的裴氏圖。

在裴氏圖轉換為階梯圖方面，黃柏勳 [民 99]提出三層式架構解裴氏圖/階梯圖轉換問題，但僅能將歐氏記號圖轉換為階梯圖。之後陳書皓 [民 102]提出五層式架構解歐氏記號圖/階梯圖轉換，可將浮標守恆裴氏圖轉換為階梯圖。Brusey [2006] 提出的裴氏圖軟體 Petri LLD 則是將訊號詮釋裴氏圖 (Signal Interpreted Petri Net, SIPN) 轉換為階梯圖，因訊號詮釋裴氏圖是專為可程式邏輯控制器的架構(將裴氏圖中的暫存點分為數位輸入(Digital Input, DI)、數位輸出(Digital Output, DO)和暫存繼電器)發展，因此是較好的轉換方法。因此本研究擬以裴氏圖軟體 Petri LLD 將無鎖死狀態的裴氏圖轉換為階梯圖，實現整個自動化系統編程上的建構程序。

1.3 研究範圍與架構

由於重複性自動化系統的裴氏圖模型建構通常以記號圖(Marked Graph)的方式來表示，而彈性製造系統則以選項裴氏圖(Conflict Petri net)表達 [方本欣, 張弘, & 梁高榮, 民 100]。本研究所探討的自動化系統為塑膠射出成型機，具有執行重複動作且資源共用及選擇的特性，因此將著重於記號圖與選項裴氏圖的探討，接著則是將藉由區域理論與可達圖來處理自動化系統鎖死問題，最後透過裴氏圖軟體 Petri LLD 進行將無鎖死狀態的裴氏圖轉換成階梯圖的程序，

以便能透過裴氏圖的優異數學性質分析，有效地建構階梯圖，降低自動化系統的編程上的建構時間。如圖 1-3 所示，本論文的架構為：第 1 章說明研究背景、動機、目的與範圍、第 2 章為文獻探討，將先探討自動化系統編程上的建構程序的相關文獻，並介紹階梯圖與裴氏圖，最後探討裴氏圖轉換階梯圖的相關文獻。第 3 章為研究方法，將介紹本研究提出的自動化系統編程上的建構程序，包含：能建構裴氏圖的軟體 Snoopy、能進行裴氏圖化簡與生成相對應可達圖的軟體 INA、能處理裴氏圖鎖死狀態的區域理論與關鍵標記/移轉分離性質(CMTSI) 演算法、能進行 CMTSI 方程組求解的數學規劃軟體 Gurobi、能將裴氏圖轉換為階梯圖的軟體 Petri LLD，第 4 章為實例說明，以塑膠射出成型機的動作流程為研究，提出實例說明，最後於第 5 章提出結論與後續研究建議。

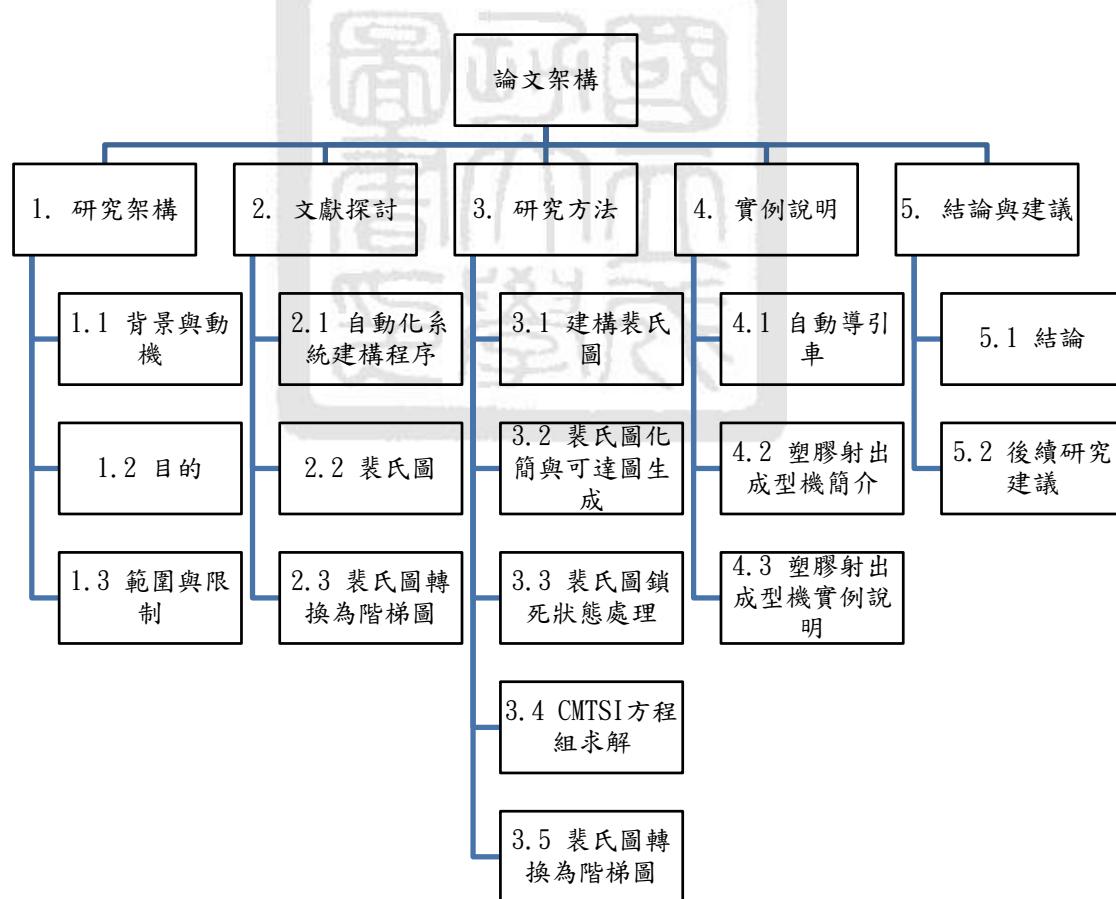


圖 1-3、論文架構

2. 文獻探討

本研究主要探討自動化系統編程上的建構程序，經過眾多文獻的探討後發現，以裴氏圖建構自動化系統後，經過裴氏圖化簡與鎖死分析，最後再轉換回階梯圖，是最好的建構程序，並能夠有效地降低建構系統的成本與提升效率。因此本章將先於 2.1 節回顧自動化系統編程上建構程序的相關文獻，接著在 2.2 節說明裴氏圖的基本元件、基本分析鎖死問題分析和裴氏圖化簡方式。2.3 節則探討裴氏圖轉換為階梯圖的相關文獻。本章的架構如圖 2-1 所示。

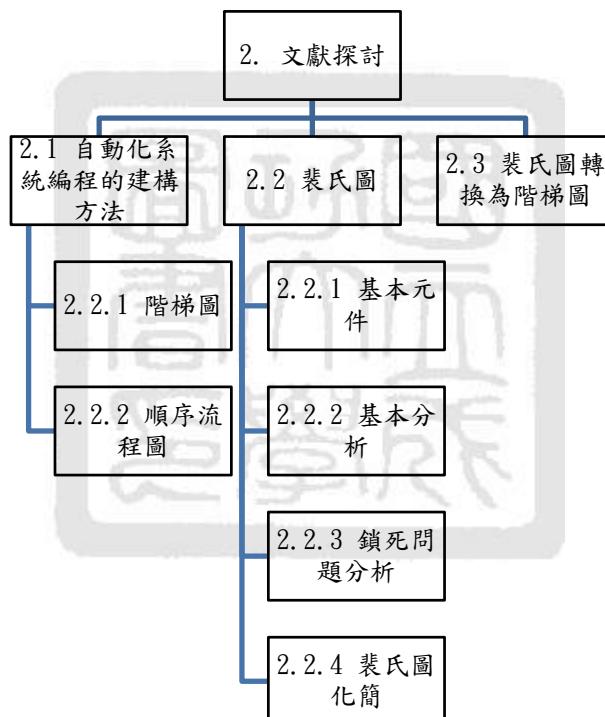


圖 2-1、文獻探討架構

2.1 自動化系統編程的建構方法

基於可程式邏輯控制器的編程方法為：階梯圖(LD)、順序流程圖(SFC)、基本指令(IL)、功能方塊圖(FBD)和結構化文字(ST)。本研究主要是希望找

到一個自動化系統編程上的建構程序，能夠降低開發時間和成品的方法，而 PLC 語言的實現方法有傳統的指令動作式、節點分析法、裴氏圖(Petri net)塑模法與 UML 塑模法[楊鎮宇 et al., 民 99]等。因此本研究將著重探討透過上述編程方法的自動化系統建構程序的優劣。

2.1.1、使用階梯圖進行自動化系統編程的建構程序

階梯圖源自於繼電器控制電路的電路圖，因圖形像階梯故以此命名。階梯圖是現在可程式邏輯控制器最常使用的編程方法。階梯圖有許多型態的元素：繼電器類型的元素(Relay Instruction)，包含常開接點(Normal Open Contact, NO)和常閉接點(Normal Closed Contact, NC)、計時器(Timer)或是計數器(Counter)類型的元素、資料轉移類型的元素(Transfer Instruction)、算數操作、資料比較與程式控制類型的元素 [Aiken et al., 1998]。

一個階梯圖在第一次執行時，會先讀取所有外部輸入裝置的狀態，並且從程式的最頂端，由左而右往下掃瞄。某些程式控制區塊可能被略過(因為發生的條件沒有被觸發)或是重覆執行。直到最後一行程式掃瞄完後，所有從輸出端連結的實體外部裝置會被更新狀態，這三個階段的執行稱為一個掃瞄(Scan)。程式在掃瞄過後開始執行，直到程式被中斷。在掃瞄期間輸入端的值可能會因上次掃瞄後修正或是外部感測器被觸發而改變。在階梯圖中，由於輸出端的值是依據輸入端的值與計時器、計數器狀態所決定的，因此即使輸入端的值固定不變，輸出端的值還是會因持續的掃瞄而產生變化，因而形成不同的路徑。因為外部事件的變化與掃瞄率的關係，階梯圖的路徑很難藉由傳統的測試方法偵測到 [Aiken et al., 1998]。

在階梯圖中使用 OR / AND 迴路達到控制目的，並可用布爾邏輯式表達。OR 迴路是指並聯迴路，當其中一個接點成立時，則此組電路為通路。AND 電路為串聯迴路，必須所有接點都成立，此組電路才為通路。以下以三菱可程式邏輯控制器的階梯圖書寫方式舉例說明 [三菱電機股份有限公司, 民 97]：

(一)、自我保持迴路

如圖 2-2 所示，自我保持迴路係指藉由接點 X020 導通後讓輸出線圈 Y000 觸發，進而讓接點 Y000 自我觸發輸出線圈 Y000，並保持住此狀態。

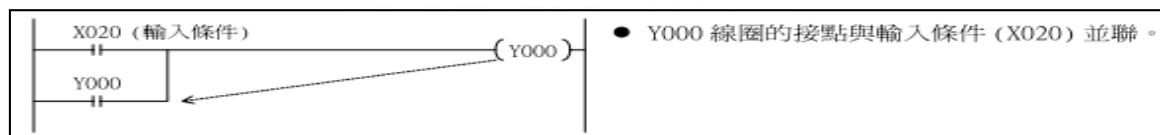


圖 2-2、自我保持迴路範例

(二)、互鎖迴路

如圖 2-3 所示，互鎖迴路係指讓輸出線圈 Y000 和 Y001 在相互的線路上以常閉接點(Normal Closed, NC)的形式出現，假如任何一個輸出線圈觸發後，另一個輸出線圈一定不會觸發。

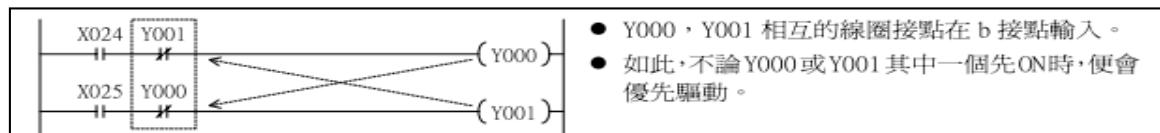


圖 2-3、互鎖迴路範例

在自動化系統的編程與氣壓迴路設計過程中，一般使用串級法(Cascade)作為順序動作控制的設計方法，串級法是一種很嚴謹的設計方式，幾乎所有的順序動作控制與氣壓迴路設計用此種方式都可以迎刃而解[傅振棻, 民 97]。串級法在設計時必須把氣壓的動作序列加以分組，動作愈複雜就分愈多組，但是受

限於氣源供應的問題，分組之組數不宜超過四組以上。串級法概略可分為下列幾個步驟：

1. 了解各支氣壓缸動作序列的前後動作關係，並以串級法的分組原則(同一支氣壓缸的兩個動作，必須分為不同組)分組。
2. 弄清楚各支氣壓缸在移動到前後端點時，是分別碰觸到哪些極限開關。
3. 在每個極限開關被碰觸後，所產生的氣壓訊號被送至哪個控制元件，控制氣壓缸的動作。
4. 寫出氣壓缸和極限開關之間的邏輯方程式。
5. 把邏輯方程式轉化為氣壓迴路圖。

然而串級法並不適用於設計彈性製造系統上，因為就其設計步驟的第一點而言，無法確定各支氣壓缸動作序列的前後動作關係。因此大部分的彈性製造系統未採取此設計方法進行實作。

Aiken et al. [1998]提出將階梯圖轉換成限制式，再藉由限制式解題引擎找出階梯圖的錯誤。作者認為使用限制式來表達階梯圖而不是布爾邏輯表達式，因為階梯圖中有些元素難以用布爾邏輯表達，且以限制式來表達較有彈性與效率。限制式解題引擎主要包含兩個部分：傳統的資料與控制流分析，用以獲取階梯圖轉換成限制式後的初始狀態；擁有將階梯圖的實際輸入值的限制式添加到初始的限制式系統中。上述中第一個部分只有執行一次，但是第二個部分則是以隨機選擇的輸入點多次執行。限制式解題引擎藉由部分生成初始的限制式，並且藉由部分提昇限制式解題引擎的績效。根據限制式生成的規則，階梯圖會被轉換成一個摘要文法樹(Abstract syntax tree, AST)，限制式為AST的節點，並且與經由傳統程式分析得到的基底系統(base system)進行比較。接著則是對階梯圖進行轉換的路徑進行多次掃瞄的分析，並瞭解各個路徑的輸出值。給定

輸入端一個隨機的值，並將相對應的限制式加入基底系統，該階梯圖在最終掃瞄時輸出端的值便能夠一清二楚。並且有些輸出端也可能同時為輸入端，因此可將上一次掃瞄時輸出端的值做為下次掃瞄時指派給輸入端的值。當掃瞄週期交替時，輸出端值的改變便能偵測到一條路徑。但上述方法在建構限制式時相當地複雜，且僅針對階梯圖進行驗證，沒有提供整體的自動化系統編程上的建構程序，並且以裴氏圖進行階梯圖驗證的方法相對比較容易。

Han [2010]認為在自動生成階梯圖的方法中，主要有三種面向，並且對相關的文獻進行分類：基於裴氏圖的方法、基於有限狀態機的方法和基於流程圖的方法。在這三種面向中，第一種和第二種面向會隨著控制邏輯的複雜度增加而有爆炸性的陳述問題。第三種面向則是藉由使用它的順序和直覺的控制邏輯程式，相對容易使用。然後藉由此種方法生成的階梯圖不同於工廠自動化工程師所撰寫的階梯圖。因此對於工廠自動化工程師而言並不合用，而且也不易瞭解自動生成的階梯圖。於是她提出一個使用延伸 UML 的物件導向階梯圖程式設計框架，來整合設計、驗證與自動生成階梯圖。然而該研究未著墨於如何處理鎖死狀態的過程。

林楊祥 [民 101]認為可程式邏輯控制器的廠牌眾多，如：三菱(Mitsubishi)、歐姆龍(Omron)、西門子(Siemens)和飛斯妥(Festo)等，且各廠牌的程式編輯軟體與程式語法差異大，因此他提出步進設計法規劃動作流程，以設置指令(SET)和重置指令(RST)為主體程式指令的程式設計方法。然而該研究僅針對編寫階梯圖程式時「重複執行的輸出(Out)指令，會造成系統動作錯誤」的問題進行改進，未考慮整體自動化系統編程上的建構，多個工件互動時發生的鎖死問題，且不能進行非一般程式架構的動作流程。

2.1.2、使用順序流程圖進行自動化系統編程的建構程序

SFC(Sequence Flow Chart，順序流程圖)原本是法國特利美(Telemecanique)公司為自家的 PLC 系統設計的一種編程語言，名為 Grafset，由於一般順序控制的問題，大多數都可以用 SFC 來規劃解決，所以三菱與歐姆龍公司也先後將這種方法納入他們的 PLC 整合發展環境軟體內[王元昌, 民 98]。國際電工協會(IEC)後來將 Grafset 修改後命名為 SFC，並納入 PLC 的編程規範 IEC 61131-3 中。SFC 構成的要素定義如下列敘述，整體程序建構後如圖 2-4 所示：

1. 步進點[Roch & Starke]：

步進點是 SFC 的基本要素，代表整個流程中的單一程序。當一個步進點被啟用時，表示隸屬於該步進點的行動區塊執行中。SFC 中的第一個步進點通常稱為初始步進點。

2. 行動區塊(Action Block)：

對於單一步進點而言，一個行動區塊包含行動資格和區塊名稱。

3. 行動資格(Action Qualifier, AQ)：

用以表明這段行動區塊的時間特性，以及保留每次行動狀態的優先權特性，詳細說明見表 2-1。

4. 區塊名稱(Block Name)：

針對每個行動區塊，我們可以直接指定布林邏輯變數(接點)來命名或者另給行動區塊一個專屬名稱。

5. 轉移條件(Transition)：

亦即將轉移條件前的步進點轉移到之後步進點的轉換條件

6. 分岐並進 / 並進合流

分岐並進(Parallel Divergence)是指當轉移條件成立後，下方連接的所有並進步進點將同時啟動。並進合流(Parallel Convergence)則是將轉移條件前的並

進步進點合併為單一步進點。

7. 選擇性分歧 / 合流

分歧(Divergence)是指當一個轉移條件成立時，它下方連結的步進點將啟動；

合流(Convergence)則是將多個轉移條件匯合連接到一個步進點。

表 2-1、行動資格符號說明表



AQ 符號	原名	功能	設定定時器預設值
D	延遲 (Delay)	從該步進點被啟動開始計時，一旦用戶所指定的時間計時完成，則該行動再次執行。如果在用戶指定時間計時完成前，該步進點未被啟動，則該行動將不被執行。	需要
DS	延遲設定 (Delay Set)	其操作大致與「SD」相同，不同點在於如果在用戶指定的延遲時間計時完成前，該步進點未被啟動時，則該行動將不被執行。本行動可用「R」終止。	需要
L	限時 (Limit)	當該步進點被啟動，則行動執行。並且當用戶指定的時間計時完成時，結束行動。如果在用戶指定的時間計時完成前，步進點將停用，該行動也將中止。	需要
N	正常的 (Normal)	只要該步進點被啟動，則行動就會執行(若未指定 AQ 時，此為 Omron PLC 整合開發環境軟體 CX-One 預設值)。	
P	脈波 (Pulse)	如果啟用最後一次掃描之選項，一旦該步進點被啟動，則行動將執行 2 個週期時間(等於 P0+P1)。如果關閉最後一次掃描，則當該步進點被啟動時，該行動將只執行一次週期時間(等於 P1)。	
P1	脈波上升緣 (Rising Pulse)	當步進點被啟動的時候，立即執行行動一個週期時間。	
P0	脈波下降緣 (Falling Pulse)	當步進點不被啟動時，立即執行行動一個週期時間。	
R	重置 (Reset)	如果本項行動是執行在「S」、「SL」、「SD」或是「DS」形式之下的 AQ，則停止並且重置該行動。如果是在其他 AQ 情況下，執行中的行動將被重置，但不是停止。當一行動被重置時，PLC 的 CPU 將進行下列操作： <ul style="list-style-type: none"> ● OUT / OUT NOT 指令：OFF ● TIM / TIMH 指令：重置 ● 其他計時器、計數器和移位暫存器：保持原狀 	
S	設定 (Set)	當該步進點被啟動，則行動執行，並且即使當該步進點被停用後，但行動仍繼續執行，可用「R」AQ 中止。	

SD	設定延遲 (Set Delay)	從該步進點被啟動後開始計時，一旦當用戶所指定的時間已完成計時，則行動執行。並且即使在該步進點被停用後，行動將繼續執行。行動可用「R」AQ 中止。	需要
SL	設定限時 (Set Limit)	當步進點被啟動後，行動執行。並且當用戶指定的時間計時完成時，行動結束。不同於「L」的是，即使在該步進點被停用後，行動將繼續執行。行動可用「R」AQ 中止。	需要

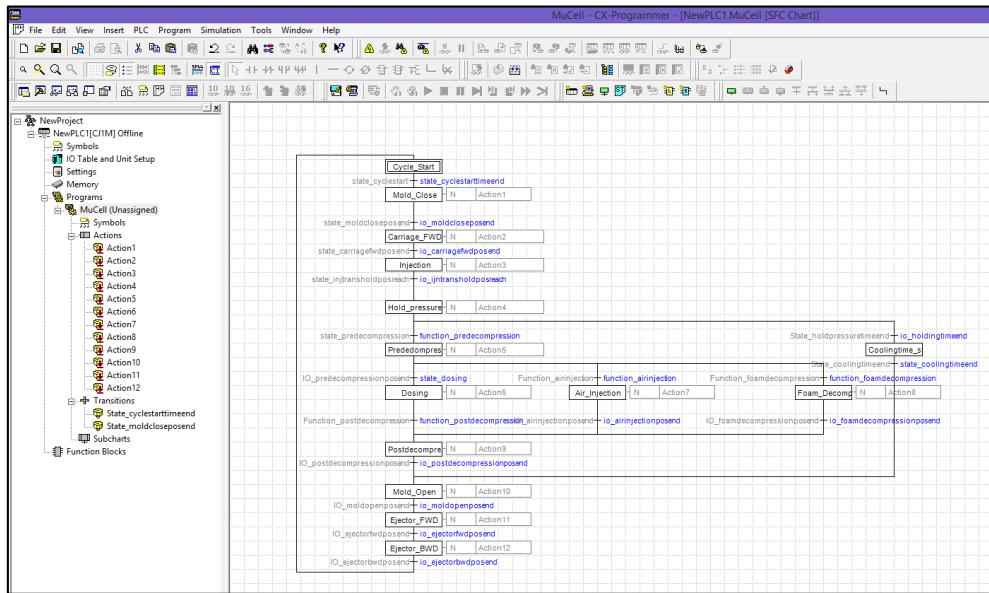


圖 2-4、歐姆龍 PLC 中的 SFC

吳添財 [民 91] 提出整合裴氏圖 / SFC / STL(步進階梯圖，為三菱 PLC 特有的程式語法)的建構程序，但沒有說明處理鎖死問題的過程。廖崧富 [民 93] 提出整合 IDEF / CTPN(彩色時間裴氏圖) / SFC 的建構程序，但在處理鎖死狀態時，僅表示使用裴氏圖軟體 Artifec 解決，沒有詳細說明處理的過程；曾呂國 [民 94] 提出整合 IDEF / 物件導向裴氏圖的建構程序，也沒有說明處理鎖死問題的過程。因此本研究將著墨於說明自動化系統建構時處理鎖死的過程。

2.2 裴氏圖

裴氏圖 [梁高榮, 民 98]於 1962 年由德國 Carl Adam Petri 發表，用來描述動態系統的離散事件，並且可表達成矩陣模式，因此具備數學分析的性質。任何只要有流程特性的資料，皆可利用裴氏圖來表達。由於階梯圖本身具備流程特性，因此將先建構裴氏圖再轉換成階梯圖。首先說明裴氏圖的基本元件和性質，接著說明裴氏圖的鎖死分析。

2.2.1 裴氏圖的基本元件

裴氏圖包含暫存點 P(Place Nodes)、轉移點 T(Transition Nodes)、具方向性的弧 F(Directed Arc)、弧的權重 W(Weight)、初始狀態(Mo)。裴氏圖的定義如下 [林潔好, 民 97]：

裴氏圖的五個元件， $PN = \{P, T, F, W, Mo\}$

- $P = \{p_1, p_2, \dots, p_n\}$ ，為暫存點的有限集合
- $T = \{t_1, t_2, \dots, t_m\}$ ，為轉移點的有限集合
- $F \subseteq (P \times T) \cup (T \times P)$ ，為弧的組合(流程關聯)
- $W: F \rightarrow \{1, 2, 3, \dots\}$ ，為弧的權重函數
- $Mo: P \rightarrow \{0, 1, 2, 3, \dots\}$ ，暫存點的初始狀態

裴氏圖圖形介紹，由以下四個圖形化元素組成：

- 暫存點(\circ)(Place)：表示系統狀態
- 轉移點(\blacksquare)(Transition)：代表系統的觸發事件
- 方向弧(\rightarrow)(Arc)：系統狀態改變的方向性
- 浮標(\odot)(Token)：代表系統正處於何種狀態

裴氏圖藉由事件和狀態的觀念來描述系統模型，將離散系統狀態的變化視為一個事件的發生。其中暫存點表示狀態，轉移點表示事件，一個轉移點可以包含許多的輸入暫存點與輸出暫存點代表事件發生前的狀態與事件發生後的狀態。方向弧用來連結暫存點與轉移點，方向弧若由暫存點指向轉移點，稱此暫存點為輸入暫存點(Input Place)；反之，方向弧若由轉移點指向暫存點，則稱此暫存點為輸出暫存點(Output Place)。暫存點內可以同時擁有很多個浮標(Token)，一個暫存點內若存有 K 個浮標，代表有 K 筆項目或資源可供利用。裴氏圖中浮標在暫存點的分布狀況稱之狀態(Marking)，存有浮標的暫存點代表系統正處於此暫存點的狀態；反之，沒有浮標的暫存點代表系統正並未處於此暫存點的狀態 [梁高榮, 民 98]。

2.2.2 裴氏圖的基本性質

2.2.2.1 記號圖和選項裴氏圖(Conflict Petri net)

記號圖是裴氏圖的特例，如表 2-2 所示，其定義為：記號圖中每個暫存點必須只有一個輸入端轉移點及輸出端轉移點。此特性使得記號圖的總浮標數不會改變。記號圖也能用矩陣型式表達，根據記號圖的特性只會有單一輸入端與單一輸出端，因此在法則矩陣當中暫存點所存在的行中只有一個「1」值和一個「-1」值。在此特性下，暫存點其每個行的值加總各會為 0 [鄭仲元, 民 99]。

圖 2-5 顯示階梯圖、記號圖與法則式專家系統的相互關係。這三者雖各自以不同的形式表達，但本質上描述的是同樣的一件事。這表示我們可以以比較簡單、清楚的裴氏圖來描述階梯圖或是法則式專家系統所表達的事物。

表 2-2、記號圖與非記號圖

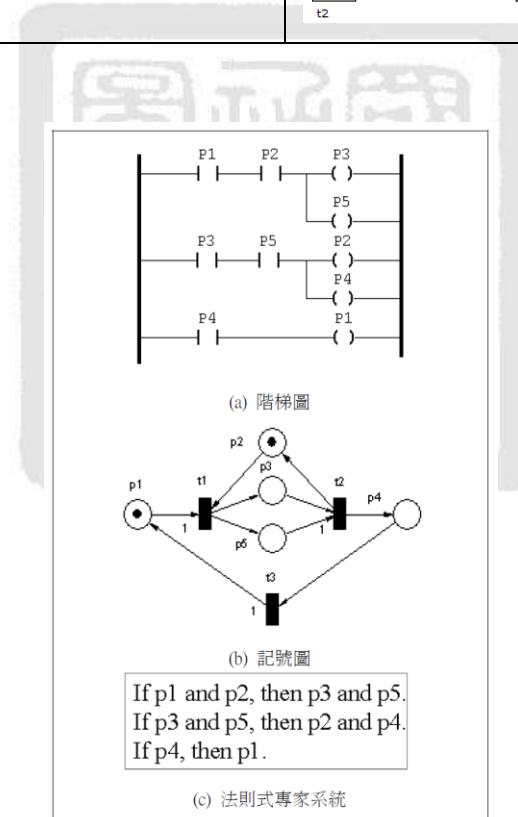
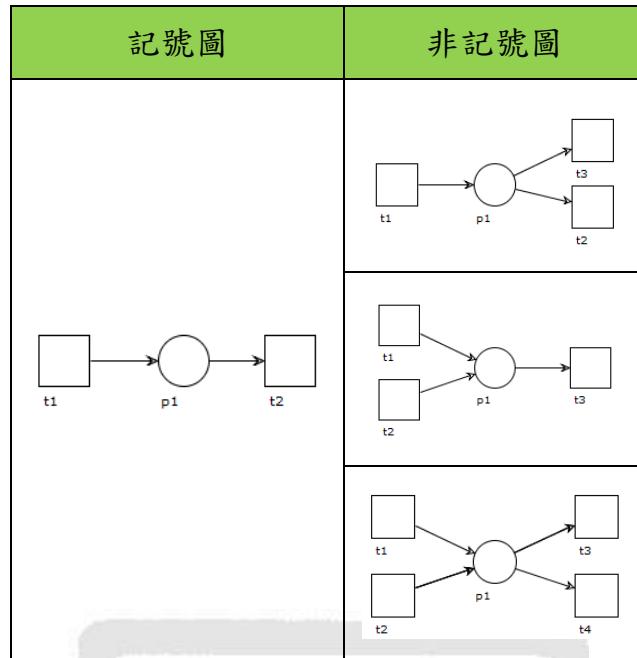


圖 2-5、階梯圖、記號圖與法則式專家系統之關係 [梁高榮, 民 98]

選項裴氏圖是指含有多輸出選項暫存點的裴氏圖。若這選擇是操之在人且轉移點受最多一個選項暫存點控制，則稱為簡易裴氏圖(Simple Petri net)，如圖2-6(b)。被動離開的浮標意味著它是一種資源，而它的離開代表有製造程序要

使用該資源，所以簡易裴氏圖常出現於資源共用問題；若這選擇是操之在我，則稱為自由選項裴氏圖(Free-Choice Petri net)，如圖 2-6(c)。又選項暫存點可能為二選項(Binary Conflict)、三選項(Temary Conflict)、四選項(Quaternary Conflict)等，故可依此命名其對應的裴氏圖，例如最多只含二選項的暫存點的裴氏圖，稱為二選項裴氏圖(Binary Conflict Petri net)[梁高榮，民 100]。

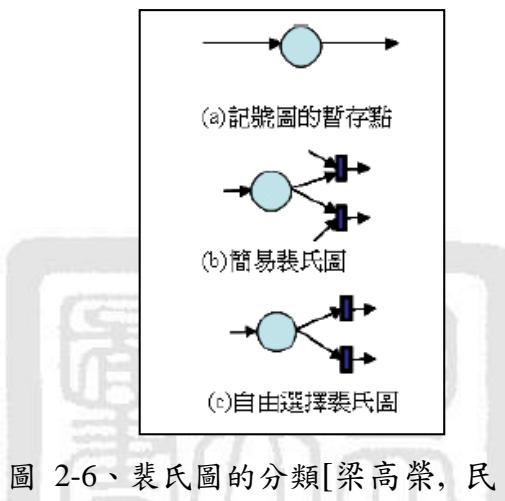


圖 2-6、裴氏圖的分類[梁高榮，民 100]

2.2.2.2 可達圖(Reachability graph)

裴氏圖由於涉及多個浮標的互動行為，不容易直接目視分析。在此情況下，裴氏圖常轉成可達圖來分析其性質。採用可達圖分析時，鎖死分析與派工分析(dispatching analysis)是兩個最常的分析方法。對可達圖來說，這是追蹤裴氏圖所有浮標流向的紀錄。追蹤圖 2-7 的裴氏圖可得到其對應的狀態圖，共有 16 列，且每一列由 12 個暫存點組成。例如：初始狀態 M0 的浮標出現在暫存點 P1、P2 和 P7 上，所以該列的 P1、P2 和 P7 的值皆為 1，而其餘暫存點的值為 0。將這 16 個狀態各視為一個節點，再依各狀態的前後互動關係而連接對應的節點，此連接的節點圖就稱為可達圖，如圖 2-8。從該圖中可以看出含有兩個鎖死節點 M2 與 M15，也就是說當裴氏圖的浮標進入 M2 或 M15 的狀態時，整個裴氏圖就鎖死而無法移動 [梁高榮，民 98]。

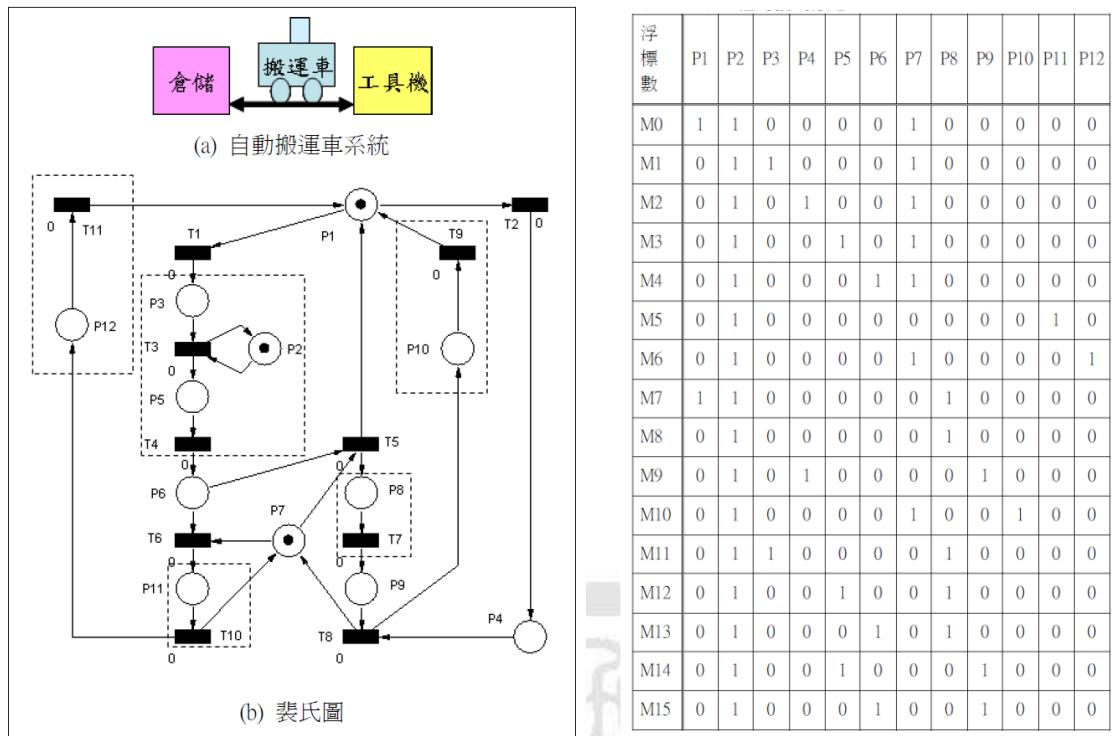


圖 2-7、自動搬運車系統的裴氏圖與對應的狀態圖 [梁高榮, 民 98]

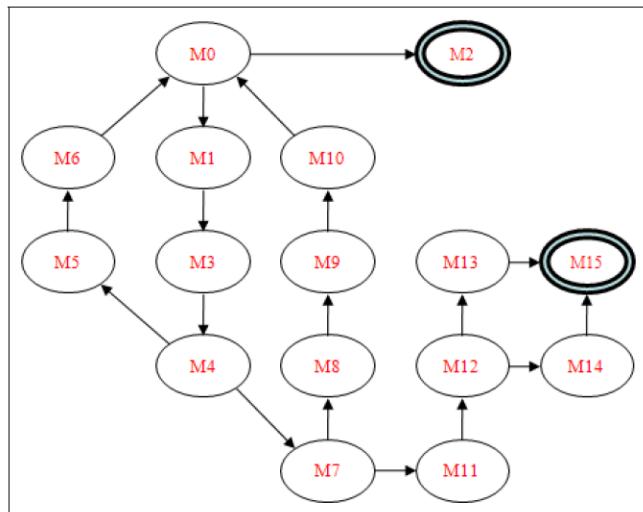


圖 2-8、自動搬運車系統的可達圖 [梁高榮, 民 98]

2.2.2.3 裴氏圖的激發規則

下列為裴氏圖中轉移點(Transition)的激發規則(Firing Rule)，用以瞭解系統

狀態(State)或標記狀態(Marking)的變異情況 [潘彥良, 民 100] :

1. 若某一個轉移點 t 的輸入暫存點 p 皆具有至少與權重值相等的浮標(Token)數，則稱此轉移點 t 為可致能(Enabled)。
2. 若激發一個可致能的轉移點 t ，則會自轉移點 t 的每一個輸入暫存點 p 移出等於 $w(p, t)$ 權重值的浮標數，且增加與 $w(t, p)$ 權重值相等的浮標數至轉移點 t 的每一個 輸出暫存點 p 。
3. 一個可致能的轉移點可能或也許不會激發 (視事件是否實際發生而定)。

2.2.2.4 裴氏圖的基本特性

一般探討運用裴氏圖塑模的模型具備二項主要特性：結構特性(Structure Property)與行為特性(Behavior Property)。結構特性探討裴氏圖組成架構的邏輯合理性，故與初始標記狀態(Initial Marking)無關；行為特性強調系統的動態行為模擬與狀態變遷情況的探討，故與給定的初始標記狀態相關。下列說明研究中常見的裴氏圖的基本特性 [廖扶西, 民 89; 潘彥良, 民 100] :

1. 可達性(Reachability)：

可達性是研究系統動態行為的重要特性。若存在由一初始標記狀態轉換至某個標記狀態之發生序列，則標記狀態稱之為從初始標記狀態至某個標記的可達性(reachable)。

2. 限制性(Boundedness)：

若一裴氏圖 N ，對其任何可達的標記(Marking)狀態而言，任一暫存點上的標記數量均未超過一有限數量 K ，則稱此裴氏圖 N 為 K -bounded 或是 simply bounded，亦即($M(p) \leq K$)；若裴氏圖 N 為 1-bounded，即 K 值為 1，則稱其為安全的。

3. 活性(Liveness)：

若一裴氏圖 N ，不論其可達的任何標記狀態，最終可能藉由一些更進一步

的發生序列而激發任何轉移點，則稱此裴氏圖 N 為活的。

4. 可逆性(Reversibility)：

若一裴氏圖 N，對於其可達的任何標記(marking)狀態，皆可再回復至其初始的標記狀態(initial marking)，則稱此裴氏圖 N 為可逆的(reversible)。

5. 強連結性(Strong Connectedness)：

對於裴氏圖 N 中的任一節點(node)，均存在至任一節點之指向路徑(directed path)，則稱此裴氏圖 N 為強連結(strong connected)。

6. 覆蓋性(Coverability)：

一裴氏圖 N 中可達的標記(marking) M，若存在一標記(marking) M'，且 $M'(p) = M(p)$ ，則稱此裴氏圖 N 為可覆蓋的(Coverable)。

7. 持續性(Persistence)：

若一裴氏圖 N 中任二個可激發(enabled)的轉移點，其中一個轉移點發生(occur)後，仍將使另一個轉移點為可激發，則稱此裴氏圖 N 為持續的(persistent)。

2.2.3 裴氏圖的鎖死問題分析

陳音帆 [民 97] 認為鎖死現象是指製造系統中的一個工件或是多個工件彼此佔住資源，等待其他工件釋放資源，但本身卻不釋放資源的現象。鎖死問題可能導致整個系統停滯，更嚴重甚至使產品或系統損毀。鎖死的現象不易察覺，可達圖為鎖死問題的分析方式之一。事先進行狀態分析，修正系統路線以確保系統的可行性與安全性。

Silberschatz, Galvin, and Gagne [2005] 對鎖死問題的定義為系統中存在一组行程陷入「互相等待對方所擁有的資源」，造成所有的行程皆無法往下執行，

使得系統資源的利用率及產能大幅降低。以圖 2-9 為例，假設一個系統中有兩個資源 R_1 和 R_2 ，而且該系統產生兩個行程 P_1 和 P_2 。 R_1 已配置給 P_2 ， R_2 已配置給 P_1 ，但這兩個行程在執行過程中又向對方要求對方正在使用中的資源，因此造成系統打結。

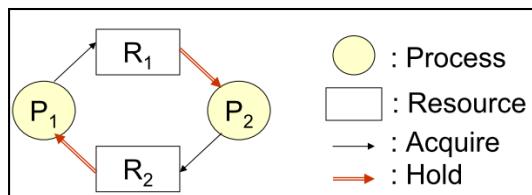


圖 2-9、鎖死問題發生的範例

Silberschatz et al. [2005]認為形成鎖死問題有四個必要條件，缺一不可：

- 互斥(mutual exclusion)：

某些資源在同一時間點，最多只能被一個行程使用，不能有多個行程同時使用此資源。其他欲使用此資源的行程則必須等待，直到該資源被釋放為止。

- 持有並等待(hold and wait)：

行程持有部分資源，且在等待其他行程所持有的資源。

- 不可搶奪(no preemption)：

行程不可任意搶奪其他行程所持有的資源。必須等待其他行程自願釋放這些資源才可以使用。

- 循環式等候(circular waiting)

系統中存在一組行程{ P_0, P_1, \dots, P_n }，其 P_0 正在等待 P_1 所持有的資源， P_1 正在等待 P_2 所持有的資源，……， P_n 正在等待 P_0 所持有的資源， $P_0 \sim P_n$ 形成循環式等候。

該研究表明前三個條件實際上是由系統和資源的物理特性決定的。亦即對於一個給定資源集合的系統而言，前三個條件之成立與否在一開始即已知，不會隨著時間而變化。而第四個條件卻可因對資源的請求、分配和釋放隨時間而變化。只要發生鎖死，這四個條件必然滿足。反之，只要有一個條件不滿足，系統就不會發生鎖死 [甘清華, 2010]。一般而言解決鎖死策略可分為四種：鎖死預防(Deadlock Prevention)、鎖死偵測(Deadlock Detection)、鎖死避免(Deadlock Avoidance)與鎖死復原(Deadlock Recovery)。其中鎖死避免設計最困難，它是以動態監督下一步的狀態避免之，能有最高的資源使用率。不過鎖死避免並無法保證一定不會有鎖死現象發生，所以常與鎖死復原配合使用。而鎖死偵測難度與資源使用率則介於鎖死防止與鎖死避免之間。鎖死預防是一種靜態策略，只考慮非鎖死路徑的設計，實作後可確保系統無論何時皆能處於安全狀態 [陳音帆, 民 97]。

Silberschatz et al. [2005] 認為鎖死問題的處理方式可從下列方向思考：

1. 鎖死預防(Deadlock Prevention)，分別從上述四點鎖死問題形成的必要條件進行分析：
 - (1). 互斥：
互斥是某些資源與生俱來的特質，很困難從這方面解決。
 - (2). 持有並等待，可以採取下列兩種方法其中之一：
 - A. 規定「除非行程可以一次取得完成工作所需的全部資源，才允許行程持有資源，否則行程不准持有任何資源。」，若採取此規定時，則「持有」不會成立，但系統產能低。
 - B. 行程在執行之初可持有部分資源，但若要在申請資源之前，必須先釋放手中所有的資源才可以提出申請。若採取此規定時，則「持有」不會成立，但系統產能低。

(3). 不可搶先：

可以將規則改成可以搶奪(preemptive)，如此頂多會出現飢餓問題(starvation)，但是不會出現鎖死問題，並且飢餓問題可以用老化技術(aging)解決。

(4). 循環式等待：

可以規定系統採取下列措施：為每個不同類型的資源賦於一個獨一無二的資源編碼(unique ID)。規定行程必須按照「資源編號遞增」的方式提出資源申請。如表 2-3 所示，當行程 A 申請資源 R5 時，因其僅握有資源 R1 和 R2，依據上述規則，可申請資源 R5。但行程 B 申請 R1 時，因擁有資源 R3，不符合上述規則，故不能申請資源 R1。同理，行程 C 因擁有資源 R1、R2 和 R6，申請資源 R4 時，須先放棄資源 R6 才可允許申請。

表 2-3、循環式等待規則範例

行程編號	行程持有的資源	申請資源	可/否	原因
A	R1, R2	R5	可	
B	R3	R1	否	需先釋放 R3 才可申請
C	R1, R2, R6	R4	否	需先釋放 R6 才可申請

總結：

雖然鎖死預防可以保證系統不會存在鎖死問題，但是資源的利用率低，且系統產能低。

2. 鎖死避免(Deadlock Avoidance)

當行程提出對資源的申請時，系統會根據以下資訊執行「銀行家演算法

(Banker's algorithm)」，來判斷系統在「假設」核准申請後是否處於安全狀態 (safe state)。若是，則核准其請求；否則否決此次申請，行程必須再等待一段時間，下一次再提出申請。

3. 鎖死偵測與復原(Deadlock Detection and Recovery)

偵測所有的行程是否有鎖死的情況發生。雖然鎖死偵測演算法可以偵測出系統是否存在鎖死情況。如果有鎖死情況，還可以知道哪些行程陷入鎖死。但是此演算法必須花費 $O(n^2m)$ 的時間複雜度，其中 n 為行程的數量，m 為資源種類的數量。

鎖死復原(Deadlock Recovery)可以分成下列兩種方法：

(1). 終止行程

- A. 刪除所有行程
- B. 每次只終止一個行程，直到鎖死情況被解決。
 - (A). 每刪除一個行程後皆須再執行鎖死偵測演算法，判斷是否存在鎖死情況。
 - (B). 若刪除一個行程後鎖死情況依然存在，表示該行程不應該刪除。
 - (C). 執行成本很高(偵測和刪除都需要成本)。

(2). 資源搶奪

- A. 程序：
 - (A). 選擇要犧牲的行程
 - (B). 剝奪其資源
 - (C). 恢復至此犧牲行程原先無該資源的狀態，表示系統必須紀錄每一個行程每次資源使用狀況。
- B. 需考量飢餓問題的發生，但可以把被剝奪的次數列入選擇犧牲行程

的考量因素。

黃泰霖 [民 102]認為關於解決製造系統鎖死的相關研究，主要可分為排程(Schedule)、環狀鍊(Circuit)與循環鍊(Cycle)、加入控制器(Controller)避免鎖死。以下分別詳述此三項鎖死相關研究以及學者提出的文獻作探討。

1. 排程(Schedule)：利用生產排程的概念，妥善安排工單以及加工機台的順序，以避免鎖死，以下為運用排程於鎖死相關文獻。

- Gang and Wu [2004]利用裴氏圖建模並運用邏輯搜尋方法找出無鎖死狀態的最佳化排程，此邏輯搜尋方法運用基因演算法與裴氏圖結構分析，但是此方法並不考慮替代機台的途經。
- Yoon and Lee [2004]提出無鎖死狀態的排程方法用於半導體製造業的跟催系統上，採用兩項程序：首先定義潛在鎖死以及安全鎖死的預防策略，之後應用此策略於線上的甘特圖中，產生一個最佳化或接近最佳解的無鎖死狀態的排程。

2. 環狀鍊(Circuit)與循環鍊(Cycle)：利用限制策略的方式，避免製造系統進入環狀鎖死與循環鎖死兩種鎖死情形，以下為相關文獻探討。

- 關於環狀鍊鎖死，Moorthy, Wee, Ng, and Teo [2003]認為先前的無人搬運車鎖死策略皆只適用於小型系統中，於是發展一個預防產生環狀鎖死的策略用於大型的無人搬運車區域控制上，此研究的鎖死是針對環狀鍊產生的鎖死，採用動態預測的方式檢測每一部無人搬運車的現存位置以及下一步新的位置，以用來偵測無人搬運車的環狀週期，做出預防策略。
- 關於循環鍊鎖死，Lee and Tilbury [2007]提出避免高階鎖死的控制方法。當系統佈置為多機台串行與並行時，製造系統就可能會出現循環鎖死的情況，此篇研究用來避免鎖死的方法是分析潛在可能會產生鎖

死的狀態，建立限制策略，以限制策略控制系統中浮標的轉移，避免系統進入循環鎖死狀態。

3. 加入控制器(Controller)：利用控制器防止裴氏圖觸發關鍵的轉移點，預防裴氏圖進入鎖死狀態，以下為預防鎖死控制器相關文獻探討。

- Mohan, Yalcin, and Khator [2004]研究彈性製造單元的預防鎖死控制器與績效評估，以顏色裴氏圖建立初始未受控制的系統模型，並且採用基於預防鎖死控制策略與及時資源配置決策的系統控制器於系統模型中。加入控制器的系統績效評估是採用控制邏輯的有效性與適用性為評估準則，且與最佳化控制策略的控制績效做比較。
- Dohi, Nomura, Shimoda, and Murakoshi [1996]提出利用硬體裝置於預防鎖死，且可以不改變裴氏圖，此硬體單元基於預防鎖死控制器，利用於偵測會導致鎖死的不安全狀態，並且預防對應的轉移點觸發，此硬體裝置包含三個記憶體與邏輯單元，利用邏輯控制的概念預先設置鎖死情境，藉由控制預防發生鎖死情境。
- Xing, Jin, and Feng [2005]探討多資源服務的自動化製造系統鎖死問題，此篇提出建立可控制的暫存點於每個鎖死的結構上，對於在鎖死結構上的工件數量做限制，以此預防鎖死。鎖死的結構是藉由裴氏圖做描述，但某些加入的控制器並不能完全預防鎖死，因此本研究的控制器並未達到最少數量。往後研究可以針對最大允許與最小數量的控制器做研究。
- Aybar and Iftar [2008]利用延展方法(Stretching)於擁有可控制與不可控制轉移點的時間裴氏圖，此裴氏圖稱為延展裴氏圖(Stretched Petri net)，首先建構延展裴氏圖取代原始的時間裴氏圖，則可輕易地設計監督控制器於裴氏圖上，然後加入此控制器至初始的時間裴氏圖上，控制裴氏圖避免鎖死。

表 2-4、裴氏圖鎖死問題研究的相關文獻整理 [陳音帆, 民 97]

作者	裴氏圖類型	處理鎖死狀態的策略	考量裴氏圖的規模	使用彩色裴氏圖	使用的方法	效益比較
Viswandham, Narahari, and Johnson [1990]	一般裴氏圖	鎖死預防			事前找安全路徑	完全預防
		鎖死避免	◎		動態偵測安全路徑	非完全預防，但資源利用率高
Ezpeleta, Colom, and Martinez [1995]	特殊裴氏圖 S ³ PR	鎖死預防		◎	裴氏圖結構分析，並加入限制	完全預防，資源利用率低
Uzam [2002]	特殊裴氏圖 S ³ PR	鎖死預防		◎	裴氏圖結構分析，並加入限制	完全預防，資源利用率低
Uzam [2004]	特殊裴氏圖 S ³ PR	鎖死預防	◎		裴氏圖的可達圖分析，並加入限制	完全預防，資源利用率高

表 2-4 為[陳音帆, 民 97]整理的基於裴氏圖方法鎖死處理的文獻彙整。表中各學者提出的研究內容將於下文詳細敘述。Viswandham et al. [1990]等學者針對一般類型的裴氏圖提出了鎖死避免(Deadlock Avoidance)的方法，第一階段是先從裴氏圖生成可達圖，第二階段則提出前瞻(look-ahead)的線上控制器(on-line controller)方法，例如：在進行資源分配決策之前，系統會評估、計算步數。然而只有一些能無限地前瞻(infinite look-ahead)的系統可以保證鎖死的避免。當鎖死不能被避免時，將會進行鎖死復原(Deadlock Recovery)。

Ezpeleta et al. [1995]提出了一種基於控制庫所(monitor)的活性裴氏圖控制器的設計方法。這一方法被認為是將裴氏圖的結構分析技術應用於自動製造系

統鎖死預防的經典之作。他們提出了一種用於製造系統建模的裴氏圖子類，稱為 S3PR (System of Simple Sequential Processes with Resources)，研究了這類網系統中虹吸管(siphon)和系統鎖死的關係，對裴氏圖中的嚴格極小虹吸管(strict minimum siphon)，提出了一種添加控制庫所的方法，最終得到了活性的裴氏圖控制器。這種方法的優點在於成功地分離了系統模型和控制器，使得它們之間有了清晰的界限。Ezpeleta 等人的鎖死控制方法的不足之處有：行為許可性、計算複雜性和結構複雜性。行為許可性即系統的許可行為受到了較大的限制。這是因為添加的控制庫所的輸出弧指向了裴氏圖模型的源轉移點，從而在很大程度上限制了本來可以進入系統進行加工的零件數目。其結構複雜性是指活性裴氏圖控制器中控制庫所的個數和網規模在理論上是指數關係，因為每一個嚴格極小虹吸管需要添加一個控制庫所。活性裴氏圖控制器的結構複雜性增加了系統控制實現 [李志武 & 周孟初, 2009]。

目前文獻上有關的鎖死避免演算法大多採定性結構分析方法(structural analysis)，如此的控制法雖可達到預防成效，然而過於保守，導致活性變小而降低系統的生產能量。自 2002 年開始，國內外學者紛紛尋求最大可達數的控制理論，藉以取得系統最大的活性程度，因此區域理論(theory of regions)便被運用於彈性製造系統的鎖死問題。其中較具代表性的研究方法可分成三類 [潘彥良, 民 100]：

1. 以鎖死區域觀念來解決鎖死問題：

Uzam [2002] 首先提出運用區域理論來解決彈性製造系統的鎖死問題，其理論背景為限制住所有可能會進入系統鎖死的所有路徑，此一所有可能會進入鎖死的路徑所形成的區域稱為鎖死區域(deadlock zone)，該學者宣稱只要能限制住所有進入此一區域的路徑，便可得到最大的控制解並使系統獲得

最大的活性。此外該學者宣稱此一控制論是現今最佳的控制演算法。經實證分析後，在小型的裴氏圖所塑模彈性製造系統中，其所發展的演算法則可藉由封鎖所有會造成鎖死區域的路徑確實可以達到最大的可達數的最佳控制。但是在大型或較複雜的彈性製造系統，其限制住所有路徑的方法，反而造成過高的運算成本。

2. 以標註移轉分離性質觀念來解決鎖死問題：

Ghaffari, Rezg, and Xie [2003] 等學者亦採用區域理論來解決彈性製造系統的鎖死問題，更清楚定義各種標記，如禁止的標記(MF)、危險標號(MD)、合法標號(ML)和**標記移轉分離性質(MTSI)**來探究最大可達數存在之原因，以及如何在控制鎖死系統的最佳化。其控制策略為限制住所有會進入禁止標記的路徑，便可解決鎖死問題。然而經驗證其計算結果後發現，該鎖死預防策略確實可以得到最大的可達數，但是其缺點為在計算所有的 MTSI 中，須耗損計算成本。尤其當系統愈大，其 MTSI 的數量通常以指數函數遞增。

3. 以結構分析方法結合標記移轉分離性質來解決鎖死問題：

Li, Zhou, and Wu [2008] 等學者提出將**定性結構分析方法(structural analysis)**與**圖像分析方法(graph analysis)**結合的控制方法，也就是採用結合**基本虹吸控制論(elementary siphon control)**及**區域理論(theory of region)**的鎖死預防策略。他們的方法是採取先控制系統的虹吸現象，再進一步使用區域理論。其論點為當系統的虹吸現象一旦被控制，鎖死的情況便會大幅地被解決，其目的是縮小其限制範圍，後續再利用區域理論，便可以節省許多的計算成本並且可以得到最佳的控制結果。然而就分析而言，此一理論在第一階段採用基本虹吸控制來減少 MTSI 的數量，但在第二階段期採用的演算法仍是依循上述 Ghaffari et al. [2003]等學者所提出的演算法，表示其效能但有

待改善。

總結來說，無論上述策略為何，目前以區域理論為研究基礎的控制論中，現階段已發表的相關文獻均是採用阻絕所有進入鎖死標記的路徑。因此潘彥良 [民 100]認為找出能阻絕進入鎖死區域的關鍵路徑路徑的狀態就能減少 MTSI 的計算，並解除鎖死狀態。因為就裴氏圖塑模的彈性製造系統而言，其最終的鎖死狀態(**final deadlock state**)才是造成一個系統失控的主要因素。若以系統的可達數分布圖的觀點，就是指最終的壞死標記(dead marking)。其次，一旦合法的標記(ML)進入最終的壞死標記的路徑受到限制，便能解除系統的鎖死狀態。最後再針對所找出的壞死標記的路徑，個別地加以節制，進而使系統保存最大活性(**maximally permissive markings**)，並保證系統不會有鎖死問題發生。

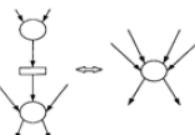
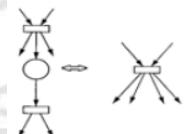
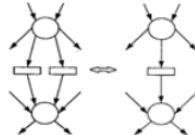
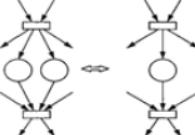
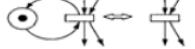
本研究採用潘彥良 [民 100]處理裴氏圖鎖死問題的方法：以區域理論與CMTSI 的方法求出控制解，因此可得到最大的可達數，並以裴氏圖的化簡方法縮減裴氏圖的規模，能夠降低計算成本。依據潘彥良 [民 100]的研究，Ezpeleta et al. [1995]和 Li and Zhou [2004] 在處理裴氏圖鎖死問題的方法均無法達到保留最大可達數 (Maximally Permissive)的目標，且均僅能使用在具有特殊結構裴氏圖的 S3PR 的彈性製造系統模式當中。Huang, Xie, and Chung [2006]提出的方法突破了演算法 Ezpeleta et al. [1995]及 Li and Zhou [2004] 被侷限在 S3PR 模型的缺點。但其採用迭代方式可能需要更多的時間及其方向弧是含有權重的則為其缺點。Uzam [2002]、Uzam and Zhou [2007]、Ghaffari et al. [2003]與潘彥良 [民 100] 均藉由區域理論來獲得具鎖死彈性製造系統最大合法活性。其中就演算法 Uzam [2002] 及 Uzam and Zhou [2007] 來看，後者採用迭代手段來獲得所有的控制庫所，而前者似乎又比後者來得有效益。而潘彥良 [民 100]採用結合簡化法、CMTSI 及限制理論為基底的預防鎖死策

略來獲得最大活性的監控器，並宣稱其方法是目前裴氏圖鎖死處理的最佳方法。

2.2.4 裴氏圖的化簡

裴氏圖的化簡的使用是為了將複雜的系統模型簡單化，但仍保留系統原本的特性。表 2-5 列出六種簡化法則，使用後仍可保留裴氏圖原先的活性、安全性和限制性 [林潔好, 民 97]。

表 2-5、裴氏圖的簡化法則

1. 合併連續的暫存點	
2. 合併連續的轉移點	
3. 合併平行的暫存點	
4. 合併平行的轉移點	
5. 清除自循環的暫存點	
6. 消除自循環的轉移點	

上述 6 種簡化法則，在裴氏圖軟體 INA 中可以輸入「E(縮減)」開始裴氏

圖的化簡程序，主要的化簡功能有[Roch & Starke, 1999]：

- 將相同的節點融合(Fusion of congruent nodes, F)
- 合併等價的暫存點(Merging of equivalent places, M)
- 消除 $F(P^F) = p$ -places (Elimination of $F(P^F) = p$ -places, A)
- 消除 $(F^P)F = p$ -places (Elimination of $(F^P)F = p$ -places, B)
- 消除具迴圈的暫存點 (Elimination of looping places, C)
- 刪除具迴圈的轉移點 (Deletion of looping transitions, U)
- 刪除單一輸出的轉移點 (Deletion of single output transitions, V)
- 刪除 PTP 序列 (Deletion of PTP-sequences, W)

首先輸入「P(編輯化簡程序)」後，輸入「FMABCUVW」，再輸入「Q(結束)」，最後輸入「N(儲存成縮減後的裴氏圖)」即完成裴氏圖的化簡。



3. 研究方法

本章將介紹本研究提出的自動化系統編程上的建構程序，包含繪製裴氏圖的軟體 Snoopy、能進行裴氏圖化簡與生成裴氏圖相對應的可達圖與鎖死狀態的裴氏圖軟體 Integrated Net Analyzer (INA)、能處理裴氏圖鎖死狀態的區域理論與關鍵標記/移轉-分離性質(Crucial Marking / Transition Separating Instance, CMTSI)演算法、能求解 CMSTI 方程組的數學規劃軟體 Gurobi optimizer、與能將裴氏圖轉換為階梯圖的軟體 Petri LLD。本研究提出的自動化系統編程上的建構程序如圖 3-1 所示。



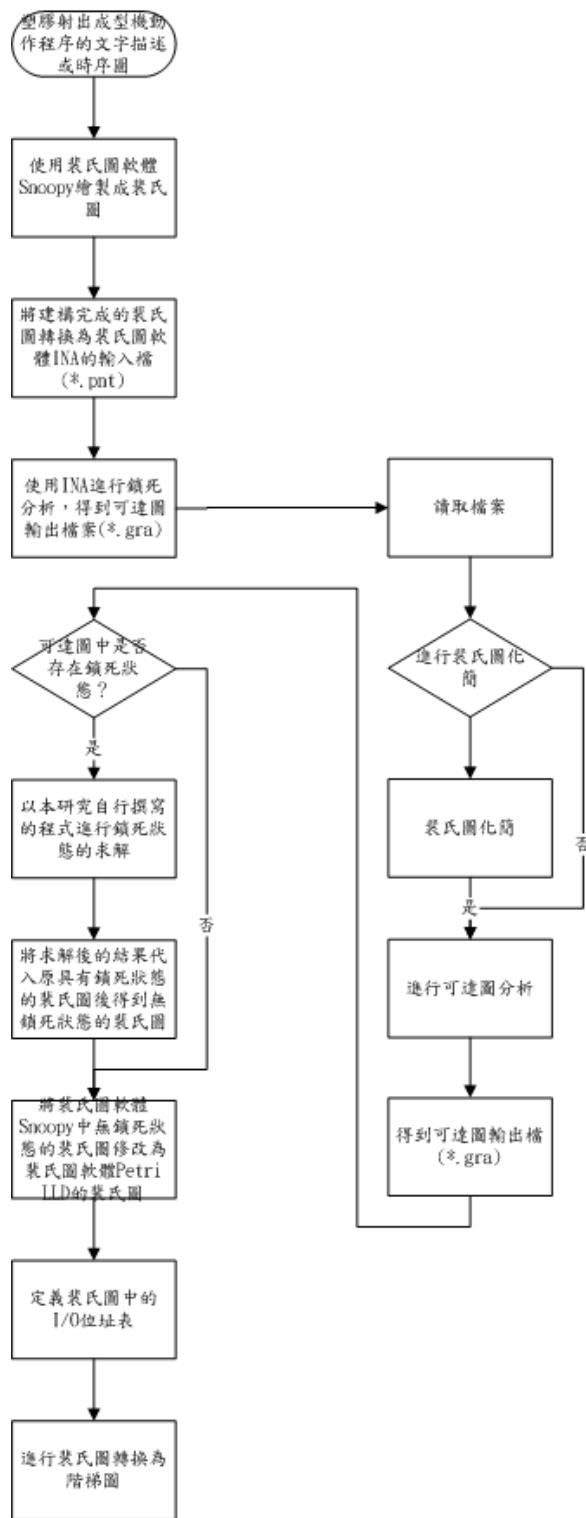


圖 3-1、本研究提出的自動化系統編程上的建構程序

3.1 建構裴氏圖：使用裴氏圖軟體 Snoopy

裴氏圖發展歷史已有半個世紀左右，相關軟體也已漸趨完備。其中因本研究進行裴氏圖化簡與裴氏圖相對應的可達圖生成，需借助分析能力卓越的裴氏圖軟體 Integrated Net Analyzer (INA)，故以可匯出 INA 輸入檔案 (*.pnt) 的裴氏圖軟體 Snoopy 進行建構。Snoopy 是由德國布蘭登堡大學(Brandenburg University)學者 Monika Heiner 等人發展，目前最新版本為 20140401 [Heiner, Herajy, Liu, Rohr, & Schwarick, 2012]。開啟裴氏圖軟體 Snoopy 後，選擇新增「Petri net」檔案後，開始建構裴氏圖，軟體的操作介面如圖 3-2 所示。

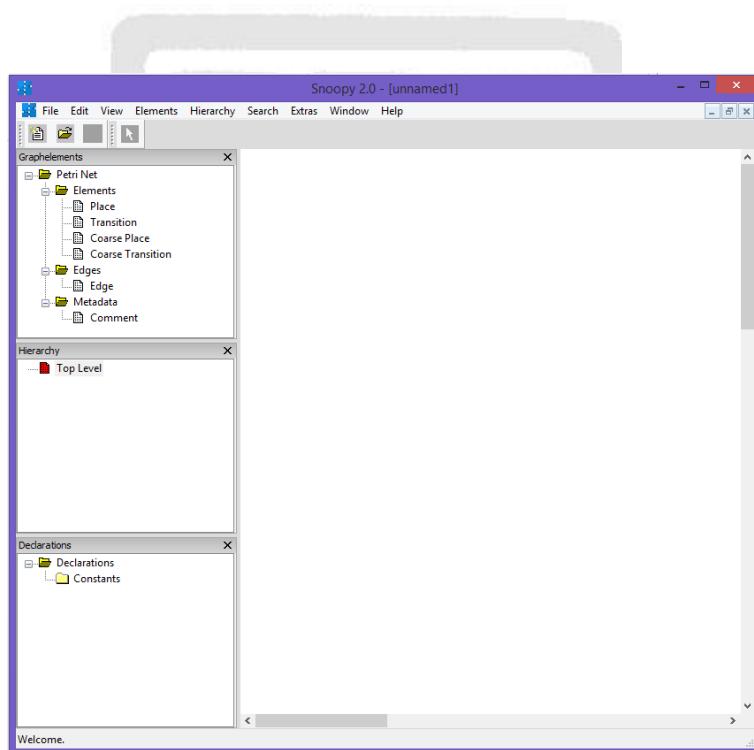


圖 3-2、裴氏圖軟體 Snoopy

在建構裴氏圖的時候，可能因對裴氏圖不熟悉，而無法順利建構系統。由於階梯圖的基本元素為接點(contact)、計時器(timer)、計數器(counter)、輸出線圈(coil)和連接線(wire)。PN 的基本元素則是有暫存點(place)、轉移點

(transition)和弧(arc)。雖然兩者的節點和弧有著不同的實際意義，可從「把 LD 和 PN 轉換成 IF-THEN 規則」的方法進行建構 [Lee & Hsu, 2004]，如圖 3-3 所示。

基本上，IF-THEN 規則包含交集與聯集符號，並且可歸納為 4 種類型的型式：

- 型式一：IF(A and B) THEN C 或者表示為 $(A \cap B) \rightarrow C$
- 型式二：IF A THEN (C and D) 或者表示為 $A \rightarrow (C \cap D)$
- 型式三：IF (A or B) THEN C 或者表示為 $(A \cup B) \rightarrow C$
- 型式四：IF A THEN (C or D) 或者表示為 $A \rightarrow (C \cup D)$

其中，型式二可分裂成兩個簡化的規則： $A \rightarrow C$ 和 $A \rightarrow D$ 。型式三則是等價於 $A \rightarrow C$ 和 $B \rightarrow C$ 兩個規則。型式四則是因為沒有實際的意涵，並且經常導致誤解，因此並不適用於順序控制的實際應用上，並予以排除。計時器和計數器也能夠在表達式中呈現，例如：條件 A 可以用來表示一個預設時間的延遲，狀態 C 則可以用來表示一個計數器增加或減少一單位。

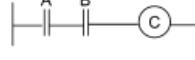
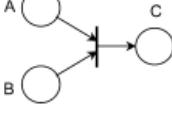
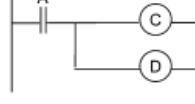
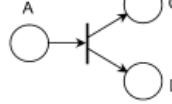
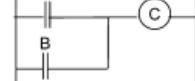
IF-THEN rules	LLD	PN
IF A and B, THEN C $A \cap B \rightarrow C$		
IF A, THEN C and D $A \rightarrow C \cap D$		
IF A or B, THEN C $A \cup B \rightarrow C$		

圖 3-3、裴氏圖與階梯圖之邏輯對應關係 [Lee & Hsu, 2004]

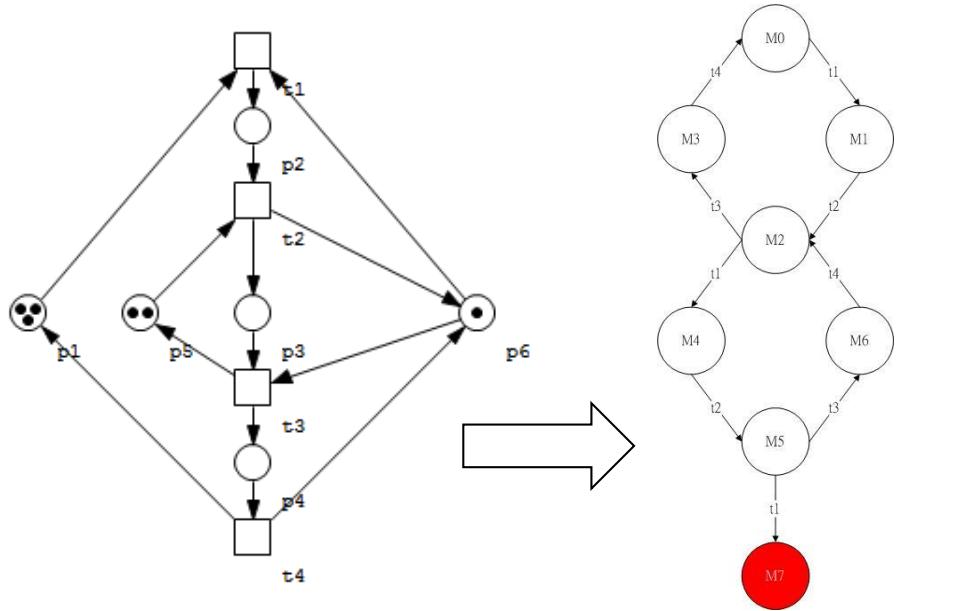
3.2 鎖死狀態處理：區域理論與關鍵標記/移轉分離性質

(CMTSI)演算法

區域理論(Theory of Regions)最早是從數學家的論點所提出來的，初期應用在過渡系統(Transition System, TS)中，其主要是用來定義過渡系統。對一主系統而言，若包含有不同的次系統，則其各次系統之間的相互關係如：裡面、外面、進入及離開等意義，並依此定義其元素在此主系統與各系統之間的動態情況，以確認元素是處於何種情況。此外區域理論可將離散動態事件之元素，如標記(Marking)及激發(Firing)的狀態模型顯示出來，故可將其應用在裴氏圖中[潘彥良, 民 100]。

Ghaffari et al. [2003]等人應用區域理論於裴氏圖的鎖死預防策略中，並提出標記/移轉-分離性質(MTSI)的演算法則：就自動製造系統的裴氏圖而言，若系統存在鎖死時，其轉移點必定含有壞的轉移點(bad transition)，因此當系統的暫存點經由壞的轉移點激發後，在可達圖上就會形成危險的標記(dangerous marking)。如果進一步將標記區分為危險的標記(MD)及合法的標記(ML)兩種，故此種標記所形成的交集必為空集合($M_L \cap M_D = \emptyset$)，所以其標記分離事件轉移集合就是找出所有會經由不可控制的轉移點進入的標記，再限制所有不好的路徑，即可找出控制解。因此數學上標記/移轉-分離性質(MTSI)集合定義為：

$$MTSI(\Omega) = \{(M, t) | M[t] \in M' \cap M \in M_L \cap M' \notin M_L\}.$$



(a) 具有鎖死狀態的裴氏圖

(b) 其相對應的可達圖

圖 3-4、標記/移轉分離性質(MTSI)範例說明一 [潘彥良, 民 100]

圖3-4(a)是一個具鎖死狀態的裴氏圖，而圖3-4(b)為其可達圖，從圖3-4(b)可以清楚看出當標記 M_5 經由轉移點 t_1 激發到標記 M_7 時，此系統產生鎖死情況。表示此系統的標記/移轉-分離性質等於 $\{M_5, t_1\}$ ，因此只要能阻絕此一標記分離事件轉移便可解除系統鎖死狀態。為限制住標記 M_5 經由不可控制移轉 t_1 激發到標記 M_7 這條路徑，Ghaffari et al. [2003]提出的解決之道如下：

數學符號說明[李志武 & 周孟初, 2009]：

- $M_0(P_c)$ ：裴氏圖中新增暫存點在初始狀態下的浮標數。
- $[N](P_c, t_n)$ ：新增暫存點連接到轉移點 t_n 的關聯向量。
- $[N]([N](P_c, \cdot))$ ：新增暫存點在裴氏圖的關聯向量(行向量)。

MTSI 方程組：

$$M_0(P_c) \geq 0 \quad (3.1)$$

$$M_1(P_c) = M_0(P_c) + [N](P_c, t1) \geq 0 \quad (3.2)$$

$$M_2(P_c) = M_0(P_c) + [N](P_c, t1) + [N](P_c, t2) \geq 0 \quad (3.3)$$

$$M_3(P_c) = M_0(P_c) + [N](P_c, t1) + [N](P_c, t2) + [N](P_c, t3) \geq 0 \quad (3.4)$$

$$M_4(P_c) = M_0(P_c) + 2[N](P_c, t1) + [N](P_c, t2) \geq 0 \quad (3.5)$$

$$M_5(P_c) = M_0(P_c) + 2[N](P_c, t1) + 2[N](P_c, t2) \geq 0 \quad (3.6)$$

$$M_6(P_c) = M_0(P_c) + 2[N](P_c, t1) + 2[N](P_c, t2) + [N](P_c, t3) \geq 0 \quad (3.7)$$

$$M_7(P_c) = M_0(P_c) + 3[N](P_c, t1) + 2[N](P_c, t2) \leq -1 \quad (3.8)$$

$$t1+t2+t3+t4=0 \quad (3.9)$$

再者，由圖 3-4(b) 可觀察出，系統有兩個指向性的循環 (cycle)：

$t1 \rightarrow t2 \rightarrow t3 \rightarrow t4 \rightarrow t1$ ，此為系統的循環方程式，故可得為一個循環方程式(3.9)。

依上述方程式組及標記分離事件轉移集合方程式，可求出其控制解為

$$[N][N](P_c, \cdot) = (-1, 0, 1, 0) \text{ 及 } M_0(P_c) = 2 \text{，如圖3-5所示。}$$

上述方程式處理的過程將於下列步驟說明：

- 在裴氏圖中新增一個暫存點 P_c ，並假設暫存點 P_c 的浮標數為變數 $x0$ 。
- 當暫存點 P_c 經過一連串的轉移點觸發後，暫存點 P_c 的浮標數，變數 $x0$ 的值會根據圖 3-5(a)裴氏圖的關聯矩陣，如表 3-1，進行加減運算而變化。並假設新增暫存點與圖 3-5(a)裴氏圖中的轉移點 $t1, t2, t3$, 和 $t4$ 的連接關係依序表示為變數 $x1, x2, x3$ 和 $x4$ 。因此新增暫存點經過一連串轉移點觸發後的各狀態下的浮標數，即為表 3-2 中的方程式所示。
- 根據圖論(Graph Theory)中的定義，在一個有向圖中存在一個鎖死節點，表示該節點的內分支度(In degree)和外分支度(Out degree)不相等。並且就

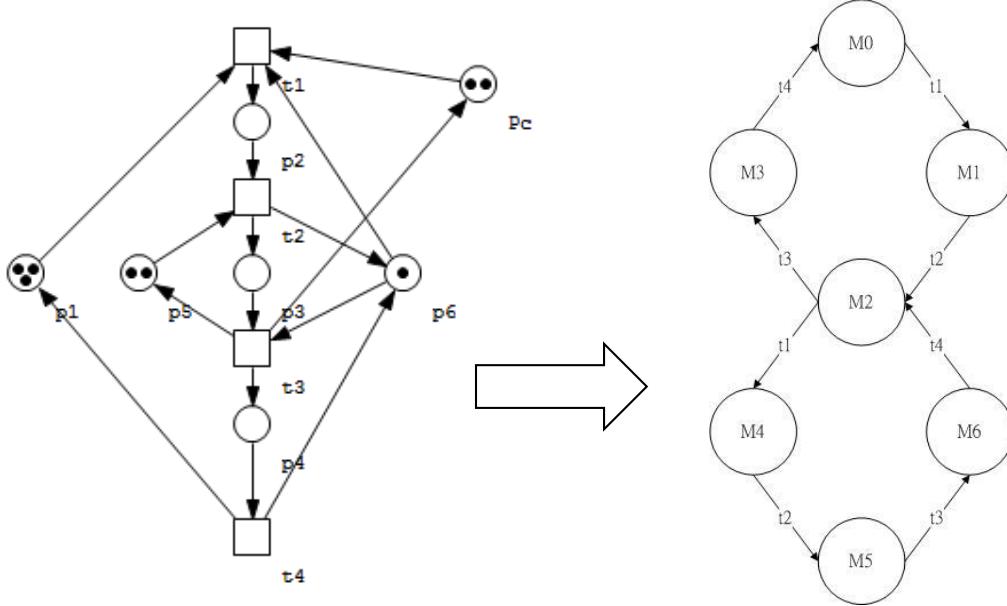
圖 3-4(b)的可達圖而言，該鎖死節點的狀態為「只進不出」，因此新增暫存點在該鎖死狀態下的方程式可表示為： $x_0+3x_1+2x_2 \leq -1$ 。而在其他非鎖死狀態下方程式的右手邊則表示為 ≥ 0 。

表 3-1、圖 3-4(a)裴氏圖的關聯矩陣

	P1	P2	P3	P4	P5	P6	Pc
T1	-1	1	0	0	0	-1	$x_1(-1)$
T2	0	-1	1	0	-1	1	$x_2(0)$
T3	0	0	-1	1	1	-1	$x_3(1)$
T4	1	0	0	-1	0	1	$x_4(0)$

表 3-2、新增暫存點 Pc 在各狀態下浮標數的方程式表示

狀態	P1	P2	P3	P4	P5	P6	Pc
0	3	0	0	0	2	1	$x_0 \geq 0$
1	2	1	0	0	2	0	$x_0+x_1 \geq 0$
2	2	0	1	0	1	1	$x_0+x_1+x_2 \geq 0$
3	2	0	0	1	2	0	$x_0+x_1+x_2+x_3 \geq 0$
4	1	1	1	0	1	0	$x_0+2x_1+x_2 \geq 0$
5	1	0	2	0	0	1	$x_0+2x_1+2x_2 \geq 0$
6	0	1	2	0	0	0	$x_0+2x_1+2x_2+x_3 \geq 0$
7	1	0	1	1	1	0	$x_0+3x_1+2x_2 \leq -1$



(a).加入控制庫所後的裴氏圖

(b).其相對應的可達圖

圖 3-5、標記/移轉分離性質(MTSI)範例說明二 [潘彥良, 民 100]

從圖3-5可看出，系統在加入控制庫所後其鎖死狀態即立刻獲得解除，並且在解除鎖死狀態後，仍可維持系統最大合理的6個可達數。上述控制方法即為運用區域理論及MTSI所建立的控制策略。然而在大型的裴氏圖中，MTSI可能不只一個，如果所有的MTSI都要計算，該系統監控的計算時間將以指數倍數增加。因此潘彥良 [民100]提出關鍵標記 / 移轉分離性質演算法(Cruical Marking / Transition Separation Instance, CMTSI)來解決，他認為無需找出所有的MTSI，只要找出關鍵的MTSI加以預防便可處理系統的鎖死狀態。他將可達圖中的標記(狀態)分成三種類型：合法標記(Legal Markings)、準鎖死標記(Quasi-Dead Markings)和鎖死標記(Dead Markings)，其中準鎖死標記的定義為「其轉移點激發路徑方向終究是朝向某一個鎖死標記，無任何轉移點激發路徑可供其回到初始狀態。」；合法標記的定義則為可達圖中，去除鎖死標記和準鎖死標記，即為合法標記。並且經過研究證實，如果一個鎖死系統能被計算出的合法標記數量(等於其初始可達圖中所有的標記數量，減去所有準鎖死標記數量和鎖死標記

數量)，便是其可以獲取的最大活性行為，亦稱之為最佳控制 [Ramadge & Wonham, 1989]。

潘彥良提出的關鍵標記 / 移轉分離性質共有兩種型式，分別為：

- Type I CMTSI (數學符號為 Ω')：

$$\Omega' = \{(M, t) | M \in M_L, t \in T, \text{and } \exists M' \in M_D, \text{Such That } M[t > M']\}$$

- Type II CMTSI(數學符號為 Ω'')：

$$\Omega'' = \{(M, t) | M \in M_L, t \in T, \text{and } \exists M' \in M_Q, M'' \in$$

M_D and a Firing Sequence $\sigma = \sigma^*(M'')$ from M' to M'' such that $M[t > M']$ and $M'[\sigma > M'']\}$

上述定義說明：一個含有鎖死狀態的自動化系統，其相對應的可達圖中如果存在有一個合法標記(M_L)，對其而言，其中一條路徑可以直接一次就被激發到一個相對應的第一類鎖死標記(M_D)，並且同時存在有另一條路徑可以被激發回合法區域內，這就是Type I CMTSI；對於鎖死標記，沒有存在 Type I CMTSI，其鎖死標記便會存在 Type II CMTSI。Type II CMTSI 則說明：對於鎖死標記而言，沒有存在有一個合法標記可以經由單一個轉移點就可以被激發到鎖死標記，因此必須由第二種型式的鎖死標記尋找出最短的轉移點激發序列。

總結來說，一個鎖死標記一定有其相對應的關鍵性的標記 / 移轉分離性質(CMTSI)，並且這個CMTSI一定是Type I CMTSI或是Type II CMTSI的其中一種。Type I CMTSI可以被視為Type II CMTSI的特例，因為在最短轉移點激發序列为0的情況下，其合法標記直接被激發到鎖死區域，而不經過準鎖死區

域。因此當一個鎖死標記存在兩種CMTSI可以選擇時，應優先選擇並執行Type I CMTSI(因其轉移點激發序列長度為0)。並且如果一個具鎖死狀態的可達圖，其第一類型的鎖死標記，同時存在兩個以上的最短路徑的CMTSI時，僅需控制其中一個。假使一個具鎖死狀態的可達圖中，若其第二類型的鎖死標記同時存在兩個以上的最短路徑的CMTSI，到達同一個路徑最短的準鎖死標記，則僅需控制其中一個Type II CMTSI。假使一個具鎖死狀態的可達圖，若其第二類型的鎖死標記同時存在兩個以上的最短路徑的CMTSI，從不同的標記分別到達不同的準鎖死標記，則兩個Type II CMTSI均需被控制住。



以CMTSI與區域理論為基礎的的鎖死預防策略演算法：

給定一個具鎖死狀態的裴氏圖：

Step 1. 產生可達圖 $R(N, M_0)$ 。

Step 2. 計算鎖死標記的數量。

Step 3. 驗證所有鎖死標記、準鎖死標記與合法標記，並確認系統最大可允許的限度數量。

Step 4. 找出所有的Type I CMTSI，假如所有鎖死標記都屬於第一類型的鎖死標記，則進入到Step 7，反之則進入到Step 6。

Step 5. 找出所有的Type II CMTSI。

Step 6. 取其中一個CMTSI來計算，直到所有的CMTSI都被計算。

Step 7. 產生事件狀態分離方程式：

$$M_o(P_c) + [N](P_c, \cdot) \vec{\Gamma}_M + [N](p, t) \leq -1$$

Step 8. 產生循環方程式：

$$\sum_{t \in T} [N](P_c, t) \vec{Y}(t) = 0, \forall Y \in C$$

Step 9. 產生所有合法可達狀態方程式：

$$M(P_c) = M_o(P_c) + [N](P_c, \cdot) \vec{\Gamma}_M \geq 0, \forall M \in M_L$$

Step 10. 獲得解答。

Step 11. 將所有的CMTSI計算出的解答去除重複解，極為控制器的組合。

潘彥良提出一個彈性製造系統模型(圖3-6)作為CMTSI演算法的說明：此彈性製造系統由兩個機具Machine1(以數學符號 M_1 表示)、Machine2(以數學符號 M_2 表示)和一個機械手臂(Robot)組成。該系統的製程共有兩個，分別為：

製程1： $I_1 \rightarrow M_1 \rightarrow Robot \rightarrow M_2 \rightarrow O_1$

製程2： $I_2 \rightarrow M_2 \rightarrow Robot \rightarrow M_1 \rightarrow O_2$

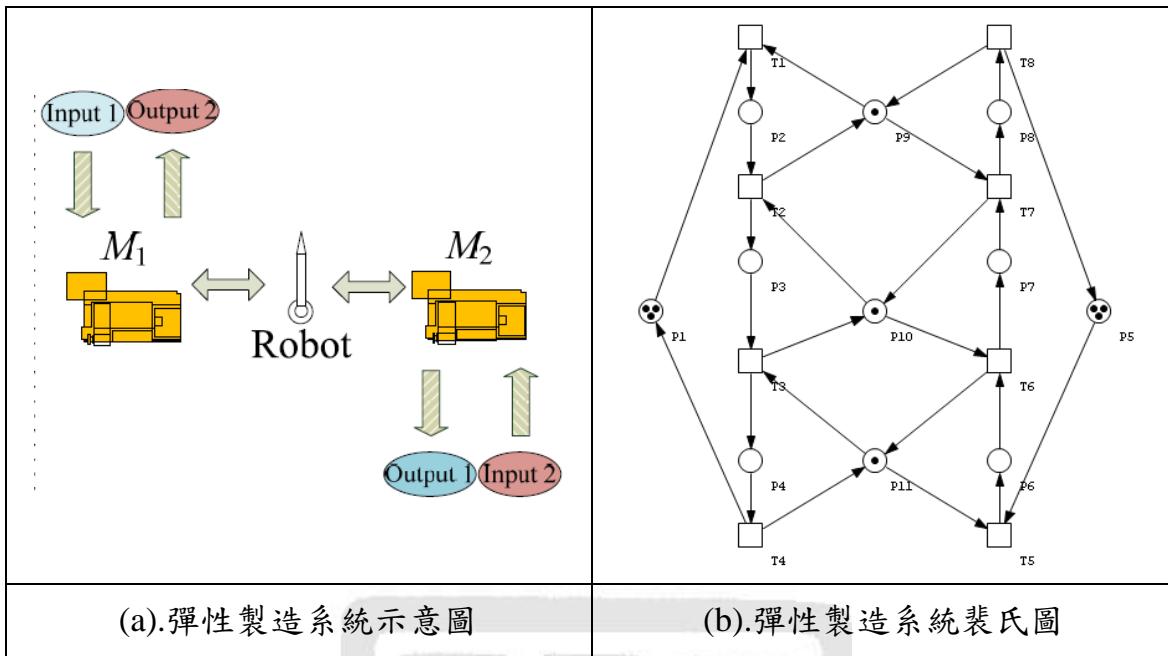


圖 3-6、彈性製造系統的示意圖與裴氏圖

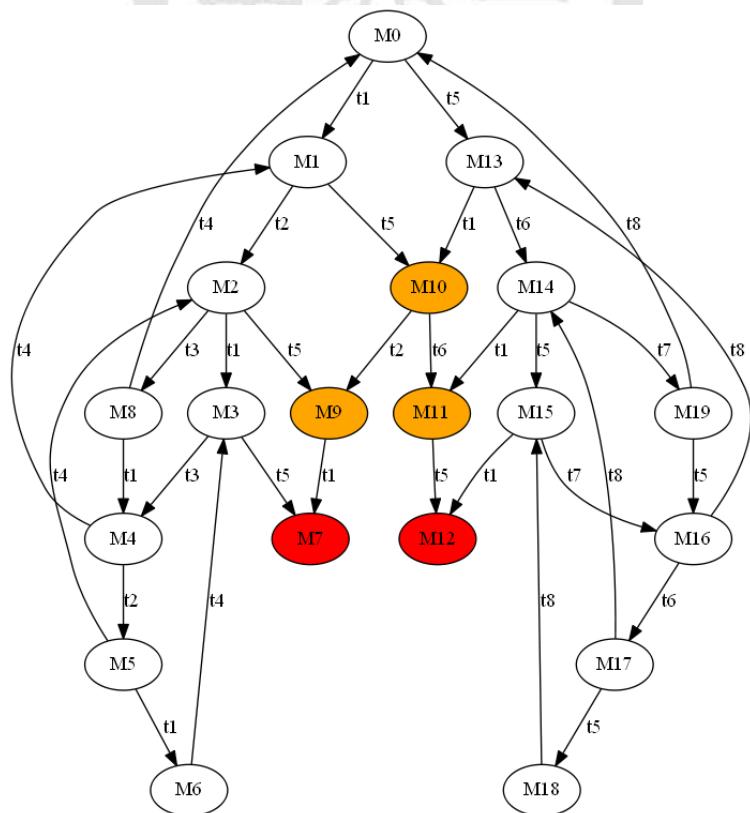


圖 3-7、彈性製造系統的可達圖

圖 3-7 為彈性製造系統的可達圖，從圖中可以看出準鎖死狀態為 M9、M10 和 M11；鎖死狀態為 M7 和 M12，其餘則為合法標記。潘彥良認為只要控制住進入最終的鎖死狀態 M7 和 M12 的路徑，即可解除系統的鎖死。因此共計需產生 2 個事件分離狀態方程式(Event separation equations)、15 個合法標記形成的可達狀態方程式(Reachability conidition equations)和 2 個循環方程式(Cycle equations)，進行求解。

事件分離狀態方程式

$$M_7(P_c) = M_0(P_c) + 2[N](P_c, t1) + [N](P_c, t2) + [N](P_c, t5) \leq -1 \quad (3.10)$$

$$M_{12}(P_c) = M_0(P_c) + [N](P_c, t1) + 2[N](P_c, t5) + [N](P_c, t6) \leq -1 \quad (3.11)$$

合法可達標記方程式

$$M_0(P_c) \geq 0 \quad (3.12)$$

$$M_1(P_c) = M_0(P_c) + [N](P_c, t1) \geq 0 \quad (3.13)$$

$$M_2(P_c) = M_0(P_c) + [N](P_c, t1) + [N](P_c, t2) \geq 0 \quad (3.14)$$

$$M_3(P_c) = M_0(P_c) + 2[N](P_c, t1) + [N](P_c, t2) \geq 0 \quad (3.15)$$

$$M_4(P_c) = M_0(P_c) + 2[N](P_c, t1) + [N](P_c, t2) + [N](P_c, t3) \geq 0 \quad (3.16)$$

$$M_5(P_c) = M_0(P_c) + 2[N](P_c, t1) + 2[N](P_c, t2) + [N](P_c, t3) \geq 0 \quad (3.17)$$

$$M_6(P_c) = M_0(P_c) + 3[N](P_c, t1) + 2[N](P_c, t2) + [N](P_c, t3) \geq 0 \quad (3.18)$$

$$M_8(P_c) = M_0(P_c) + [N](P_c, t1) + [N](P_c, t2) + [N](P_c, t3) \geq 0 \quad (3.19)$$

$$M_{13}(P_c) = M_0(P_c) + [N](P_c, t5) \geq 0 \quad (3.20)$$

$$M_{14}(P_c) = M_0(P_c) + [N](P_c, t5) + [N](P_c, t6) \geq 0 \quad (3.21)$$

$$M_{15}(P_c) = M_0(P_c) + 2[N](P_c, t5) + [N](P_c, t6) \geq 0 \quad (3.22)$$

$$M_{16}(P_c) = M_0(P_c) + 2[N](P_c, t5) + [N](P_c, t6) + [N](P_c, t7) \geq 0 \quad (3.23)$$

$$M_{17}(P_c) = M_0(P_c) + 2[N](P_c, t5) + 2[N](P_c, t6) + [N](P_c, t7) \geq 0 \quad (3.24)$$

$$M_{18}(P_c) = M_0(P_c) + 3[N](P_c, t5) + 2[N](P_c, t6) + [N](P_c, t7) \geq 0 \quad (3.25)$$

$$M_{19}(P_c) = M_0(P_c) + [N](P_c, t5) + [N](P_c, t6) + [N](P_c, t7) \geq 0 \quad (3.26)$$

循環方程式

$$t1+t2+t3+t4=0 \quad (3.27)$$

$$t5+t6+t7+t8=0 \quad (3.28)$$

當第一個 CMTSI{M₃, t₅} 所產生的第一個事件分離狀態方程組，亦即方程式(3.10)和方程式(3.12)~(3.28)，同時解聯立解，可以得到兩組最佳解：1 和 2(見表 3-3)；當第二個 CMTSI{M₁₅, t₁} 所產生的第一個事件分離狀態方程組，亦即方程式(3.11)和方程式(3.12)~(3.28)，同時解聯立解，可以得到兩組最佳解：3 和 4(見表 3-3)。從上述 4 組解可以得知，1 和 4 是重複的，因此可以移除第 4 組解。所以將這三組解加到原有的裴氏圖中，彈性製造系統的鎖死狀態就可以解除(圖 3-8)。

最佳解編號	新增暫存點的名稱	新增暫存點的初始浮標	從裴氏圖的轉移點連接到新增暫存點	從新增暫存點連接到裴氏圖中的轉移點
1	C _{P1}	1	t2、t6	t1、t5
2	C _{P2}	1	t3、t6	t2、t5
3	C _{P3}	1	t2、t7	t1、t6
4	C _{P4}	1	t2、t6	t1、t5

表 3-3、彈性製造系統鎖死狀態解

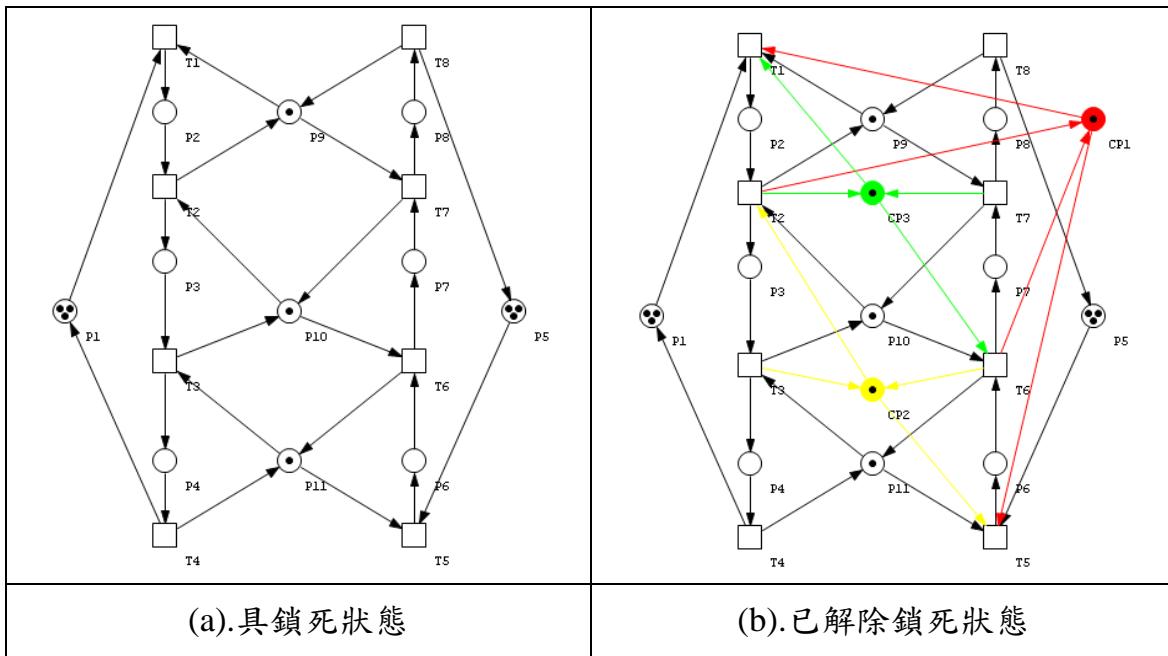


圖 3-8、彈性製造系統裴氏圖解除鎖死狀態比較

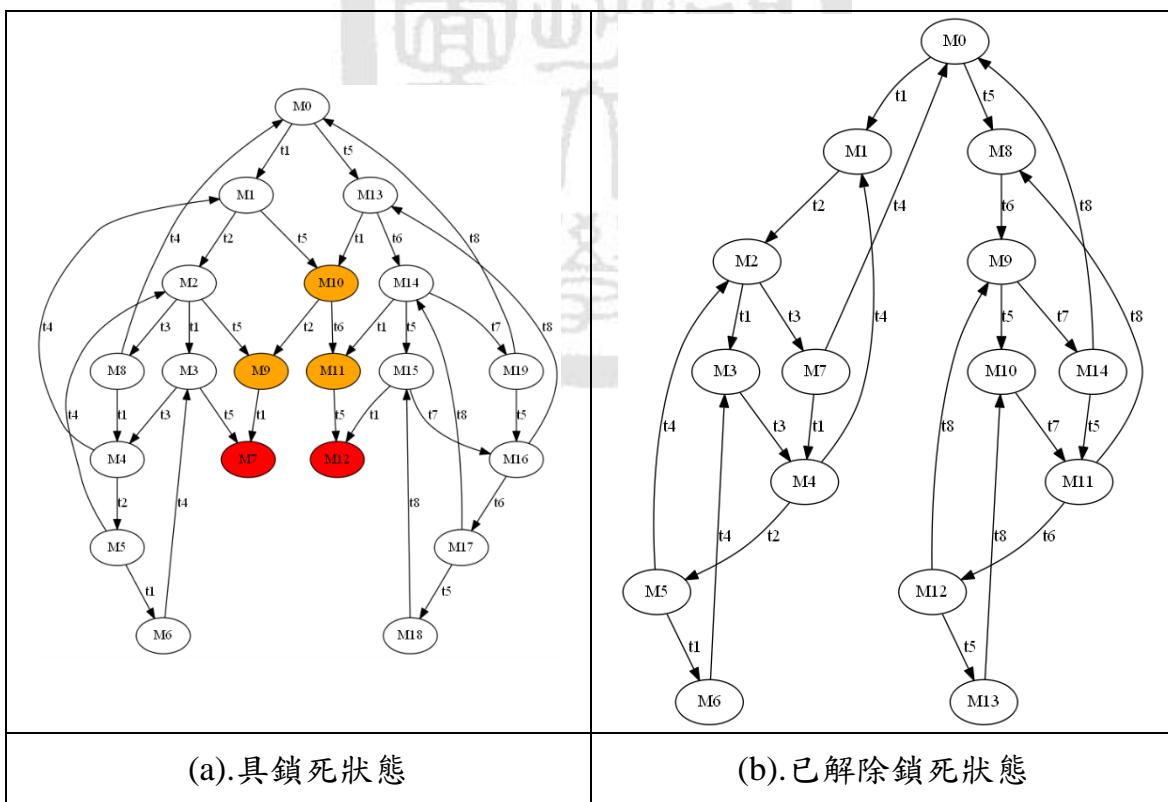


圖 3-9、彈性製造系統裴氏圖的可達圖解除鎖死狀態比較

從圖 3-9 可以看出，系統加入控制庫所後，其可達數仍可保持最大的合

法數量(15 個合法標記)。相較於 Uzam [2002]提出的方法，必須解 6 個 MTSI(亦即進行 6 次聯立方程式求解)，而潘彥良提出的 CMTSI 控制方法，只需要解 2 個 MTSI(亦即解 2 次聯立方程組)，效率確實提升不少。

值得注意的是，使用一般的數學規劃軟體，如 Lingo 和 Gurobi，進行規劃求解時，若遇到求解結果有多重最佳解時，軟體僅會提供一組最佳解而已。然而潘彥良提供的彈性製造系統的例子(本研究 3.2 節中的彈性製造系統例子)，針對每一組 CMTSI 方程組求解，卻能提供 2 組最佳解。並且經過測試後發現，確實需要將求解後的三組解代入原本的裴氏圖中，才能順利解除鎖死狀態(詳見附錄中的圖 A-3)。

3.3 裴氏圖化簡與可達圖的生成：使用裴氏圖軟體 INA

由於本研究藉由可達圖與區域理論、標記/移轉分離性質演算法進行裴氏圖鎖死狀態的處理，透過具有卓越分析能力的裴氏圖軟體 Integrated Net Analyzer (INA) 進行裴氏圖化簡與生成可達圖是最好的方法。INA 是由德國 Humboldt 大學發展，最新的版本為 v2.2(2003)。輸出 INA 的輸入檔(*.pnt)後，開啟裴氏圖分析軟體 INA (如圖 3-10 所示)，接下來的操作程序如下 [Roch & Starke, 1999]：

1. 讀取檔案：

在 INA 的介面中依序輸入「E(編輯)」後，輸入「RF(讀取檔案)」，表示開啟裴氏圖檔案，輸入裴氏圖檔案的檔名(*.pnt)後輸入「Q(離開)」離開讀取檔案介面。

2. 輸出可達圖檔案：

在 INA 的介面中輸入「A(分析)」後，輸入「R(計算可達圖)」，開始進行裴氏圖的可達圖分析。輸入「W(輸出計算後的可達圖)」，並儲存成可達圖輸出

檔(*.gra)。

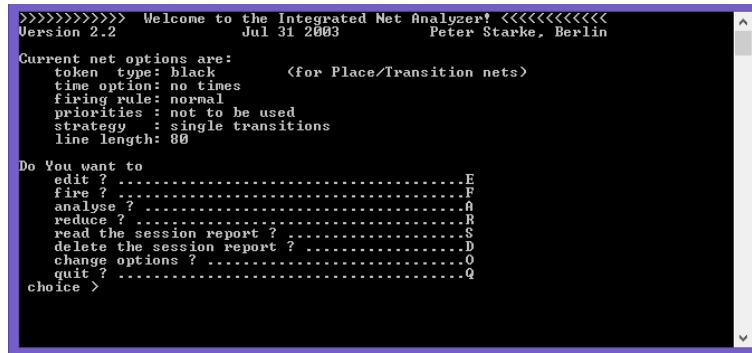


圖 3-10、裴氏圖分析軟體 INA

因為在進行裴氏圖的鎖死處理過程中，需使用可達圖中的迴圈資訊，因此在計算可達圖時，在裴氏圖軟體 INA 中輸入「K(生成可達圖中的迴圈)」，即可得到可達圖中迴圈的資訊，並可儲存成檔案(*.cir)，如圖 3-11。圖 3-11 中顯示，可達圖中存在兩個迴圈，並敘述迴圈中的暫存點與轉移點的關係。之後再以本研究撰寫的程式轉換為循環方程式。

```
1
2 Circuit nr. 1:
3 n1<-t0--n18<-t8--n17<-t7--n16<-t10--n15<-t9--n14<-t17--n13
4 <-t16--n12<-t5--n11<-t4--n10<-t6--n9<-t11--n8<-t13--n7
5 <-t12--n6<-t2--n5<-t1--n4<-t15--n3<-t14--n2<-t3--n1
6
7 Corresponding T-invariant:
8 TR: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
9   : 23
10  : 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0
11  | 0
12 Livelocked transitions:
13   1: 18, 19, 20, 21, 22, 23,
14 Circuit length = 18
15
16 Circuit nr. 2:
17 n1<-t0--n18<-t8--n17<-t7--n16<-t10--n15<-t9--n14<-t17--n13
18 <-t16--n12<-t19--n20<-t18--n19<-t22--n11<-t4--n10<-t6--n9
19 <-t11--n8<-t13--n7<-t12--n6<-t2--n5<-t1--n4<-t15--n3
20 <-t14--n2<-t3--n1
21 Corresponding T-invariant:
22 TR: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23   : 23
24   : 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1
25   | 0
26 Livelocked transitions:
27   2: 5, 20, 21, 23,
28 Circuit length = 20
29
30 Circuit nr. 1 has minimal length = 18.
31 Circuit nr. 2 has maximal length = 20.
32
```

圖 3-11、裴氏圖分析軟體 INA 生成的可達圖中的迴圈資訊

在計算可達圖的過程中，可能會因為裴氏圖過大，造成可達圖的狀態個數爆增的問題。藉由裴氏圖化簡的方法，可使原本的裴氏圖縮減成另一個較小的裴氏圖，並保留原本裴氏圖的特性。在 INA 中輸入「E(縮減)」開始裴氏圖的化簡程序，主要的化簡功能有[Roch & Starke, 1999]：

- 將相同的節點融合(Fusion of congruent nodes, F)
- 合併等價的暫存點(Merging of equivalent places, M)
- 消除 $F(P^F) = p\text{-places}$ (Elimination of $F(P^F) = p\text{-places}$, A)
- 消除 $(F^P)F = p\text{-places}$ (Elimination of $(F^P)F = p\text{-places}$, B)
- 消除具迴圈的暫存點 (Elimination of looping places, C)
- 刪除具迴圈的轉移點 (Deletion of looping transitions, U)
- 刪除單一輸出的轉移點 (Deletion of single output transitions, V)
- 刪除 PTP 序列 (Deletion of PTP-sequences, W)

首先輸入「P(編輯化簡程序)」後，輸入「FMABCUVW」，再輸入「Q(結束)」，最後輸入「N(儲存成縮減後的裴氏圖)」即完成裴氏圖的化簡。此外當進行可達圖的計算時，INA 還有出現兩個裴氏圖化簡選項，但這兩個選項預設是關閉的，若要進行裴氏圖化簡，可將其開啟：

- 結構化化簡(Symmetrical Reduction)
- 頑固結構化簡(Stubborn Reduction)

3.4 CMTSI 方程組求解：使用數學規劃軟體 Gurobi

建構好自動化系統的裴氏圖並取得相對應的可達圖後，如果可達圖中有鎖死狀態便可依據區域理論與標記/移轉分離性質演算法，得到一組或多組的 CMTSI 方程組(視鎖死狀態的數量而定)。本研究使用數學規劃軟體

Gurobi 進行求解。

Gurobi 為 CPLEX(IBM ILOG CPLEX Optimization Studio) 成員新開發的規劃求解工具，目前最新的版本為 5.6[2013]。可求解的問題類型有：

- 線性規劃(Linear Programming, LP)。
- 二次規劃(Quadratic Programming, QP)。
- 二次約束規劃(Quadratic Constrained Programming, QCP)。
- 整數規劃(Integer Programming, IP)。
- 混合整數規劃(Mixed-Integer Programming, MIP)。
- 混合整數二次規劃(Mixed-Integer Quadratic Programming, MIQP)。
- 混合整數二次約束規劃(Mixed-Integer Quadratic Constrained Programming, MIQCP)。

檔案格式規範：

- 變數格式：
 - 不可使用保留符號「+, -, *, /, ^, >, =, <, :, [,]」。
 - 可為任意英文、數字和符號，例如：x01、E(2)、7 等。
- 方程式格式：
 - 表示方式為係數乘以變數，乘號需省略，如： $2 x1 + 3 x2$ 。
 - 方程式命名方式以「:」表示，如：function1: $2 x1 + 3 x2$ 。
 - 運算符號為「+, -, *, /, ^」。
 - 括號為「[,]」。
 - 不等式符號為「<, ≤, ≥, >, =」。
 - 符號、係數、變數和常數都要以空白符號隔開。
 - 變數需放置在方程式的左手邊，常數放置於方程式的右手邊。

- 變數的範圍限制需表示為： $0 \leq x_1 \leq 10$ 。
- 解除變數的非負整數解限制，如： x_0 free。
- 變數的單邊無限制表示為： $-\infty \leq x_1 \leq 15$ 。
- 檔案儲存副檔名為*.lp，如 model.lp。
- 功能區塊格式
 - 目標式的表示：Maximize 最大化問題；Minimize 最小化問題。
 - 限制式的表示：Subject to。
 - 變數的範圍限制表示：Bounds，預設範圍限制為非負整數(≥ 0)。
 - 變數屬性：General 一般型態變數；Integer 整數型態變數；Binary 二元型態變數。
 - 結束標籤格式為 End。
- 範例：

```

Maximize
x +y +z

Subject to
C0: x +5 y +2 z ≤ 10
Qc0: x +y +[ x ^ 2 -2 x * y +3 y ^ 2] ≤ 5

Bounds
0 ≤ x ≤ 5
z ≥ 2

Genaral
x y z

end

```

- 手動求解步驟：

- 執行 gurobi.bat 檔案，啟動 Gurobi。
- 在 Gurobi 命令視窗中輸入指令。
- 讀取檔案：輸入「m=read('檔案位置')」，如：
「m=read('c:/model.lp')」。
- 開始求解：輸入「m.optimize()」。
- 等待求解完成或是使用 Ctrl + C 中斷。
- 目標值與相關求解資訊會顯示於畫面。
- 顯示決策變數，輸入「m.printAttr('x')」。
- 輸出求解結果，格式為「m.write('檔案名稱')」，如：
m.write('c:/output.sol')。
- 離開，輸入「quit()」。

本研究自行撰寫裴氏圖的鎖死狀態解除程式，並在程式中引用

Gurobi56.net.dll，加入 Gurobi 提供的範例程式碼(程式碼詳見附錄中的圖 A-1)後即可呼叫 Gurobi 進行數學規劃求解，Gurobi 的執行程序則如圖 3-12。求解完成後會在程式的目錄下產生一個 out.sol 檔案。進行 CMTSI 方程組求解時，需將表示裴氏圖中新增暫存點的變數 x_0 限定為非負整數解($0 \leq x_0 \leq n$)，因為暫存點中的浮標數不能為負。且表示新增暫存點與裴氏圖中各轉移點連接關係的變數 $x_1 \sim x_n$ ，限定解為 -1 、0、1，三種狀態，因為暫存點與轉移點的連接關係在關聯矩陣中的狀態，僅有-1 、0、1，三種狀態。

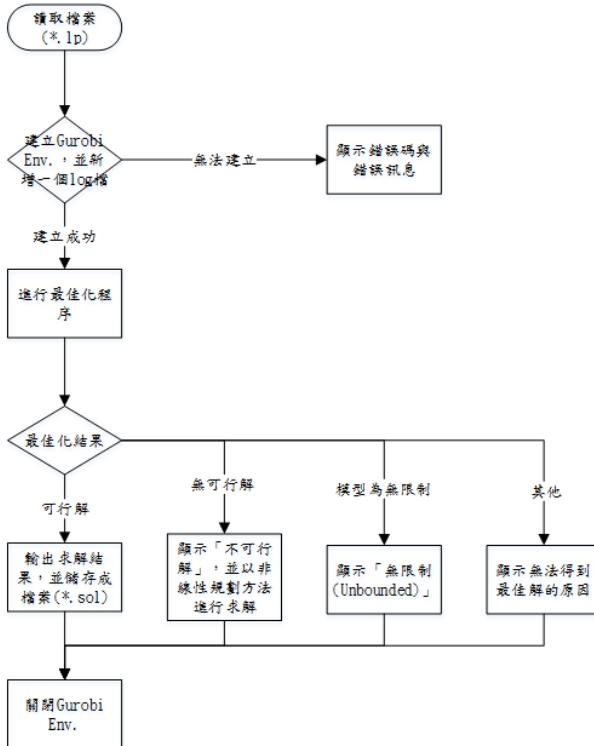


圖 3-12、數學規劃軟體 Gurobi 執行程序圖

首先以本研究自行撰寫的程式，如圖 3-13，開啟 INA 的可達圖輸出檔案後，依據標記/移轉分離性質(MTSI)演算法，列出 MTSI 的方程組，並開始進行具鎖死狀態裴氏圖的處理，如果鎖死狀態只有 1 個則僅需計算 1 次，2 個則須計算 2 次，依此類推。計算完成後。開啟原具有鎖死狀態的裴氏圖，將計算後的結果代入，得到無鎖死狀態的裴氏圖。

該程式首先分析 INA 的可達圖輸入檔，將可達圖的資訊節錄出來，以便能繪製成一個可達圖的圖形，並將可達圖的路徑表達為 CMTSI 方程式並儲存成數學規劃軟體 Gurobi 的輸入檔案(*.lp)，接著引用 Gurobi 的動態連結檔(Gurobi56.Net.dll)，進行整數規劃求解並儲存成檔案(*.sol)，程式碼(以 visual basic.net 進行撰寫)詳見附錄圖 A-1。最後將求解結果代入原本具有鎖死狀態的裴氏圖中，得到無鎖死狀態的裴氏圖。

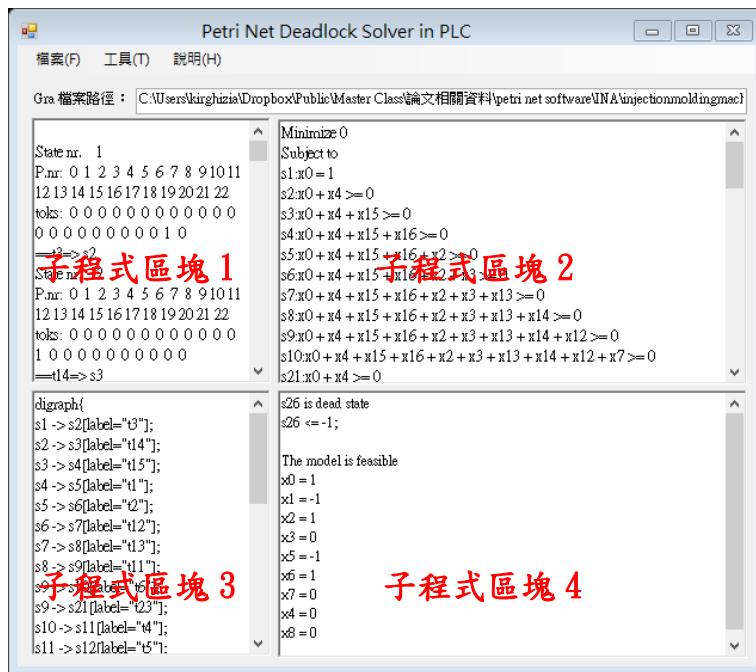


圖 3-13、本研究撰寫的裴氏圖鎖死求解軟體

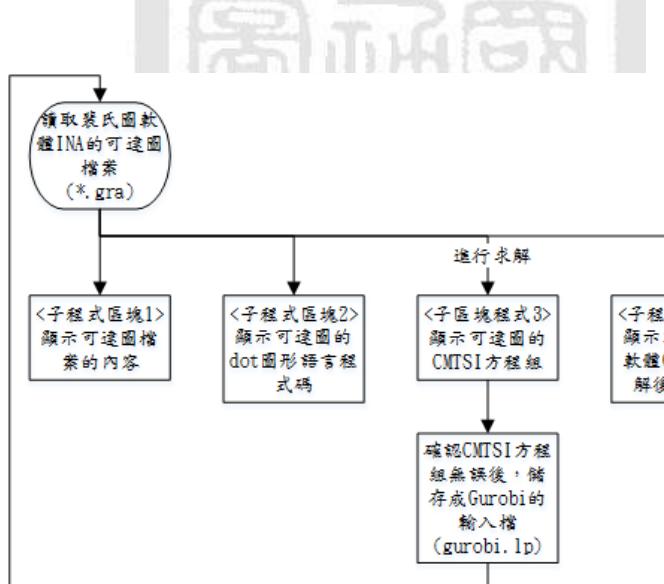


圖 3-14、本研究撰寫的裴氏圖鎖死求解軟體執行流程圖

此程式的操作程序如圖 3-14 所示，詳細內容見下文說明：

1. 開啟具有鎖死狀態的 INA 可達圖輸出檔(*.gra)，並顯示在子程式區塊 1。

如圖 3-15 所示。

```

State nr. 1
P.nr: 0 1 2 3 4 5 6 7 8 9 10 11
12 13 14 15 16 17 18 19 20 21 22
toks: 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0
==t3=>s2
State nr. 2
P.nr: 0 1 2 3 4 5 6 7 8 9 10 11
12 13 14 15 16 17 18 19 20 21 22
toks: 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0
==t4=>s3

```

圖 3-15、子程式區塊 1 中顯示的可達圖輸出檔(*.gra)

2. 輸出 Graphviz 程式碼，並顯示在子程式區塊 2，以便觀察可達圖，如圖 3-17 所示。Graphviz 為 AT&T 實驗室發展的開源軟體，用於繪製 DOT 等圖形語言指令碼描述的圖形。安裝 Graphviz 後開啟程式 gvedit，將 Graphviz 程式碼輸入 gvedit 視窗後，點擊「輸出(layout)」後會產生可達圖的圖形。若是要將該圖形儲存，則是點擊「設定(settings)」，設定輸出引擎(layout engine)、輸出檔案格式(output file type)和輸出檔案名稱(output file name)，建議將儲放路徑設定為磁碟槽根目錄下)完成後，點擊 ok 即完成。圖形語言 DOT 的格式為：

- (1). 首先定義圖為有向圖(Directed graph)或無向圖(Undirected graph)，若為有向圖，則格式為：digraph{ }
- (2). 敘述節點之間的關係的格式表示為：「節點名稱」->「節點名稱」。節點的形狀預設為圓形，可加入參數變更形狀。並可在節點關係中加入弧的名稱，格式為：[label="弧的名稱"]。

圖 3-16 為說明 Graphviz 的範例，若定義圖為有向圖時，輸入「digraph{}」，接著將圖中節點的關係輸入，並可加上節點的顏色與形狀的描述。將圖 3-16 左方的 dot 語言程式碼執行後，可得到圖 3-16 右方的有向圖。

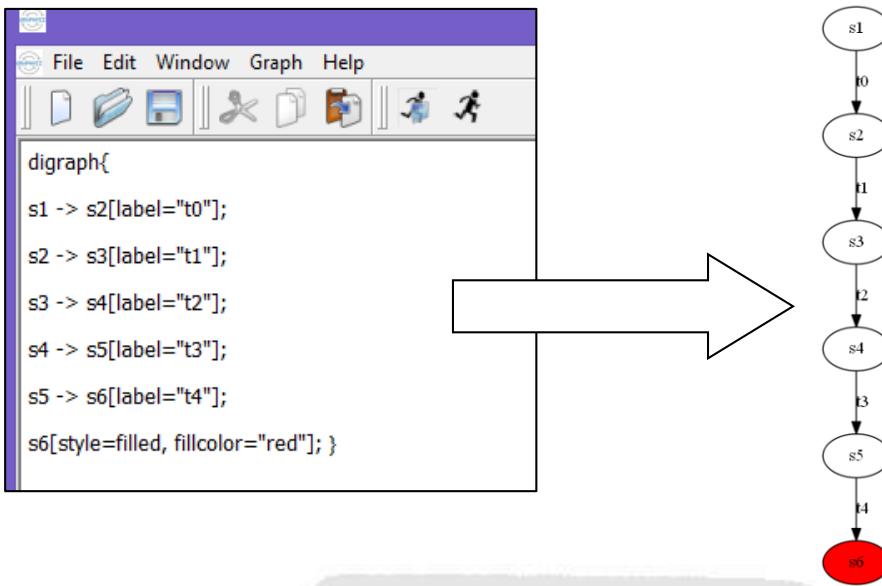


圖 3-16、圖形語言軟體 Graphviz 的 gvedit 操作介面與範例說明

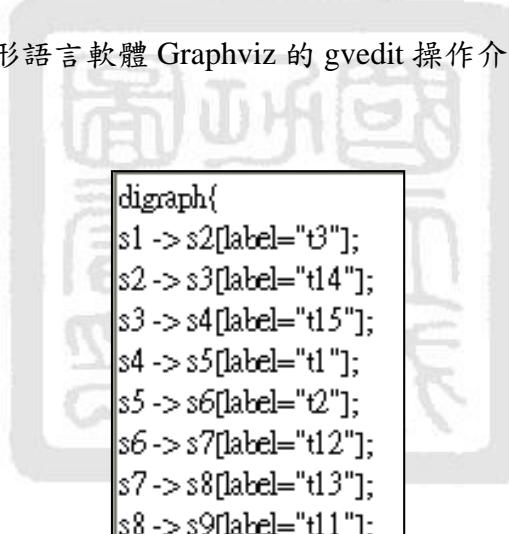


圖 3-17、子程式區塊 2 中顯示的可達圖的 Graphviz 程式碼

3. 將可達圖的路徑逐一列出形成一個 CMTSI 方程組，並顯示在子程式區塊 3，如圖 3-18 所示。方程式表示的意義為：假設在裴氏圖中新增一個暫存點，其浮標數定義為 x_0 ，新增暫存點與裴氏圖中的轉移點的連接關係則依序定義為 x_1, x_2, \dots, x_n 。當存在一個鎖死節點時，則該鎖死節點的方程式

表達為新增暫存點從初始狀態 s0 到鎖死節點之中行經的弧的加總後浮標數變化的過程，且在方程式的右手邊表示小於等於 -1，表達為數學式 3.29；若為非鎖死節點則表示為新增暫存點從初始狀態 s0 到鎖死節點之中行經的弧的加總後浮標數變化的過程，且在方程式的右手邊表示大於等於 0，表達為數學式 3.30。並將 INA 輸出的可達圖中迴圈資訊整理為循環方程式，如圖 3-19，選擇要開啟的檔案後，子視窗就會列印出可達圖中的循環方程式。

$$\sum_{i=0}^n xi \leq -1 , i = 0 \sim n \quad (3.29)$$

$$\sum_{i=0}^n xi \geq 0 , i = 0 \sim n \quad (3.30)$$

```

Minimize 0
Subject to
s1:x0=1
s2:x0+x4>=0
s3:x0+x4+x15>=0
s4:x0+x4+x15+x16>=0
s5:x0+x4+x15+x16+x2>=0
s6:x0+x4+x15+x16+x2+x3>=0
s7:x0+x4+x15+x16+x2+x3+x13>=0
s8:x0+x4+x15+x16+x2+x3+x13+x14>=0
s9:x0+x4+x15+x16+x2+x3+x13+x14+x12>=0
s10:x0+x4+x15+x16+x2+x3+x13+x14+x12+x7>=0
s21:x0+x4>=0

```

圖 3-18、子程式區塊 3 中顯示的 CMTSI 方程組

```

Form5
檔案位置: C:\Users\kirghizia\Dropbox\Public\Master Class\論文相關資料\petri net software\INA\injectionmoldingmachine_20140618.c

Circuit nr. 1:
n1<10-n18<18-n17<17-n16<10-n15<9-n14<17-n13
<16-n12<15-n11<14-n10<16-n9<11-n8<13-n7
<12-n6<2-n5<11-n4<15-n3<14-n2<3-n1

Corresponding T-invariant:
TR: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
22
: 23
: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0
| 0
Livelocked transitions:
1: 18, 19, 20, 21, 22, 23,
Circuit length = 18

Circuit nr. 2:
n1<10-n18<18-n17<17-n16<10-n15<9-n14<17-n13
<16-n12<19-n20<18-n19<22-n11<14-n10<16-n9
<11-n8<13-n7<12-n6<2-n5<11-n4<15-n3
<14-n2<3-n1

Corresponding T-invariant:
TR: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
22
: 23
: 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1
| 0
Livelocked transitions:
2: 5, 20, 21, 23,
Circuit length = 20

Circuit nr. 1 has minimal length = 18.
Circuit nr. 2 has maximal length = 20.

Cr1:
x1 + x9 + x8 + x11 + x10 + x18 +
x17 + x6 + x5 + x7 + x12 + x14 +
x13 + x3 + x2 + x15 + x16 + x4 = 0

Cr2:
x1 + x9 + x8 + x11 + x10 + x18 +
x17 + x20 - x19 + x23 + x5 + x7 +
x12 + x14 + x13 + x3 + x2 + x16 +
x15 + x4 = 0

```

圖 3-19、將 INA 輸出的可達圖中迴圈資訊整理為循環方程式

4. 呼叫數學規劃軟體 Gurobi 進行求解。

求解後的結果顯示在子程式區塊 4，如圖 3-20 所示。

- 如果結果顯示變數 x_0 為 1，表示新增暫存點的浮標數為 1；
- 若變數 x_1 到 x_n 為 1 時，表示需從該對應的轉移點新增一條輸出弧到新增暫存點；
- 若變數 x_1 到 x_n 為 -1 時，表示需從新增暫存點新增一條輸出弧到該對應的轉移點。

```

s26 is dead state
s26 <=-1;

The model is feasible
x0 = 1
x1 = -1
x2 = 1
x3 = 0
x5 = -1
x6 = 1
x7 = 0
x4 = 0
x8 = 0

```

圖 3-20、子程式區塊 4 中顯示的 MTSI 方程組求解結果

3.5 裴氏圖轉換為階梯圖：使用裴氏圖軟體 Petri LLD

得到無鎖死狀態的裴氏圖後，便能進行裴氏圖轉換為階梯圖的步驟。本研究使用裴氏圖軟體 Petri LLD 進行該步驟。Petri LLD 是由英國劍橋大學 (Cambridge University)學者 James Brusey 發展[Brusey, 2006]，可將裴氏圖轉換為可程式邏輯控制器的階梯圖(Ladder Diagram, LD)或是基本指令(Instruction List, IL)等。Petri LLD 和一般的裴氏圖軟體不同，該軟體是基於訊號詮釋裴氏圖 (Signal Interpreted Petri Net, SIPN)的研究建構。SIPN 是由德國 Saarland University 學者 Gerog Frey 發展[Frey & Wagner, 2006]，他將裴氏圖中的暫存點分為輸入型態的暫存點(Input place)、輸出型態的暫存點(External place)和一般的暫存點，以便能夠更貼近可程式邏輯控制器的架構，並且各個暫存點的浮標數限制最多為 1。最後必須建構可程式邏輯控制器的 I/O 位址表(Instance)，才能順利將裴氏圖轉換為階梯圖。

圖 3-21(a)、(b)和(c)為 Petri LLD 作者 James Brusey 說明的範例 - 鑽孔機，圖 3-21(a)為此系統的機械結構，圖 3-21(b)為此系統的訊號詮釋裴氏圖，圖 3-

21(c)為此系統的 I/O 位址表，圖 3-22 為經由訊號詮釋裴氏圖轉換後的階梯圖。

該系統的動作流程為：

1. 按下啟動按鈕(start cycle)後，Z 軸馬達(Head motor)開始向下移動，並且鑽孔馬達(Spindle motor)開始轉動。夾持機構(Clamp)開始夾持物件(Part)。
2. 當 Z 軸下位極限開關(LS2)被觸發，Z 軸馬達開始往上移動。
3. 當 Z 軸上位極限開關(LS1)被觸發，Z 軸馬達、鑽孔馬達與夾持機構皆停止。直到啟動按鈕再次被觸發後才進行下一次週期的動作。

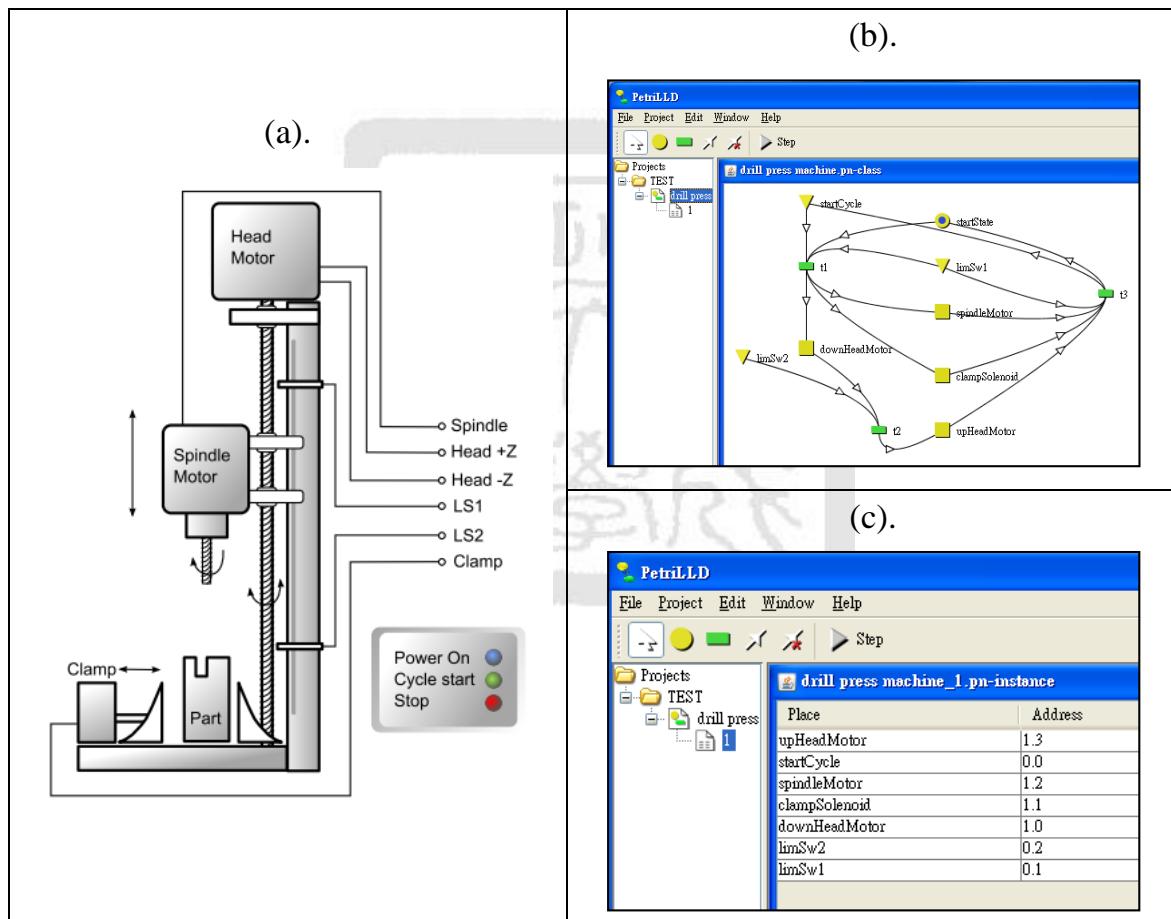


圖 3-21、Petri LLD 的範例：鑽孔機

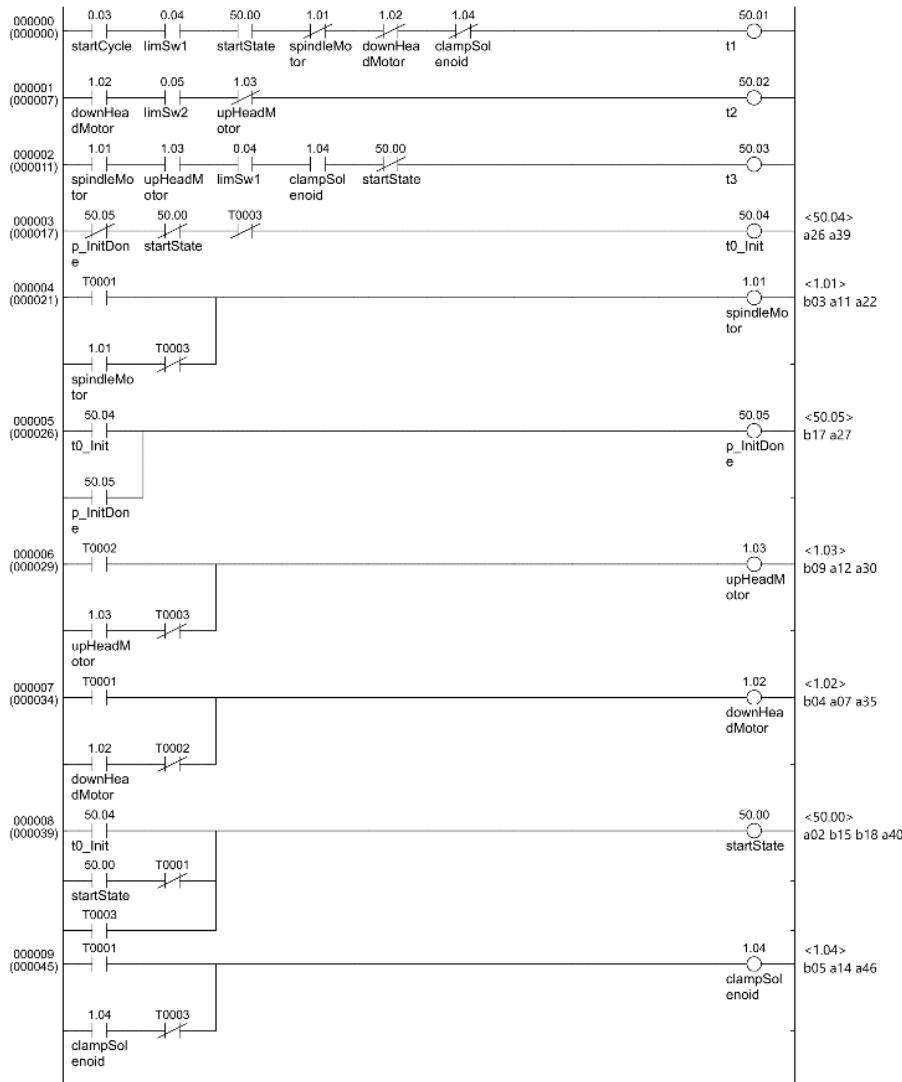


圖 3-22、經由訊號詮釋裴氏圖轉換後的階梯圖

從圖 3-22 可以發現，使用階梯圖撰寫順序動作流程時，在程式碼中「輸出線圈(OUT)」前的指令中，僅有執行該階段的動作時所需的觸發條件為常開接點，其餘常閉接點的指令則為下一個階段執行的動作。如階梯圖程式中的第一行程式碼，要使鑽孔機開始執行動作的觸發條件為：「按下啟動按鈕(startcycle)」、「Z 軸上位極限開關(limsw1)觸發」和「系統處於準備狀態(startstate)」。而「鑽孔馬達啟動(spindlemotor)」、「Z 軸馬達向下(downheadmotor)」和「夾持單元啟動(clampsolenid)」則為下一個階段要執行的動作。

4. 實例說明

本章將先以梁高榮 [民 98]提供的自動搬運車系統說明第三章研究方法中的鎖死處理的過程，接著介紹塑膠塑出成型機的基本構造與動作流程。並以塑膠射出成型機的動作流程說明本研究提出的自動化系統編程上的建構程序。本研究使用裴氏圖軟體 Snoopy 進行裴氏圖的繪製，Snoopy 是由德國布蘭登堡大學(Brandenburg University)學者 Monika Heiner 等人發展，因其繪製裴氏圖完成後，可輸出成裴氏圖軟體 Integrated Net Analyzer (INA)的輸入檔案。將繪製好的裴氏圖經過 INA 的分析後，得到一個可達圖的檔案，接著以本研究撰寫的程式產生 CMTSI 方程組，並以數學規劃軟體 Gurobi optimizer 進行鎖死狀態的求解，最後由裴氏圖軟體 Petri LLD 進行裴氏圖與階梯圖的轉換。本研究將以自動搬運車系統與塑膠射出成型機的動作流程為例，描述建構自動化系統時，如何有效率的進行動作流程程式的撰寫與驗證。

4.1 自動搬運車系統

首先以裴氏圖軟體 Snoopy 建構自動搬運車系統的模型，如圖 4-1，接著使用裴氏圖軟體 INA 進行裴氏圖化簡與輸出可達圖檔案，如圖 4-2。之後以本研究撰寫的程式讀取 INA 的可達圖檔案，產生 CMTSI 方程組，並以數學規劃軟體 Gurobi 進行求解。接著將得到的解(圖 4-3)代入原本具有鎖死狀態的裴氏圖(圖 4-4(a))中，得到無鎖死狀態的裴氏圖(圖 4-4(b))。

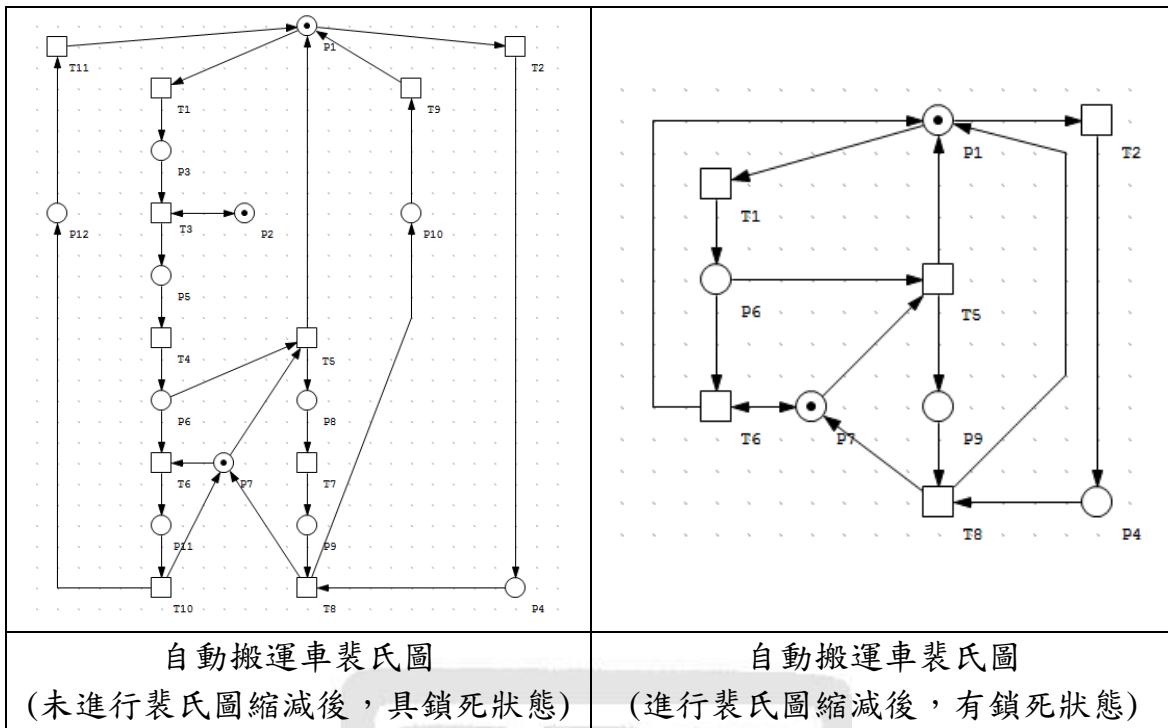


圖 4-1、自動搬運車系統的裴氏圖

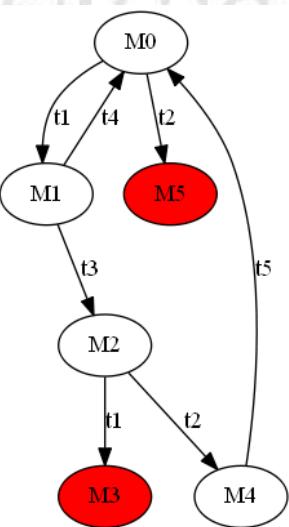
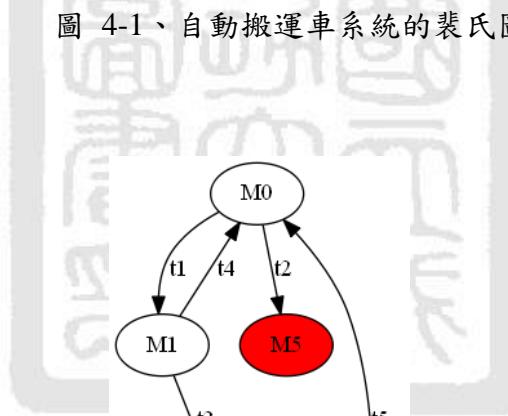


圖 4-2、自動搬運車系統的可達圖(有鎖死狀態)

自動搬運車系統的 CMTSI 方程組

可達狀態方程式

$$M0: x_0 = 1 \quad (4.1)$$

$$M1: x_0 + x_1 \geq 0 \quad (4.2)$$

$$M2: x_0 + x_1 + x_3 \geq 0 \quad (4.3)$$

$$M4: x_0 + x_1 + x_3 + x_2 \geq 0 \quad (4.4)$$

循環方程式

$$Cr1: x_5 + x_2 + x_3 + x_1 = 0 \quad (4.5)$$

$$Cr2: x_4 + x_1 = 0 \quad (4.6)$$

事件分離狀態方程式

$$M3: x_0 + x_1 + x_3 + x_1 \leq -1 \quad (4.7)$$

$$M5: x_0 + x_2 \leq -1 \quad (4.8)$$

<pre> 1 # Objective value = 0 2 0 0 3 x0 1 4 x1 -1 5 x3 0 6 x2 0 7 x5 1 8 x4 1 </pre>	<pre> 1 # Objective value = 0 2 0 0 3 x0 1 4 x1 -1 5 x3 2 6 x2 -2 7 x5 1 8 x4 1 9 </pre>
第一個 CMTSI 的求解結果 進行方程式(4.1)~(4.6)和方程式(4.7) 的聯立求解	第二個 CMTSI 的求解結果 進行方程式(4.1)~(4.6)和方程式(4.8) 的聯立求解

圖 4-3、自動搬運車系統的 CMTSI 方程組解

<pre> 1 p M PRE,POST NETZ 0:t 2 0 1 2 3 4 , 0 1 3 1 0 1 , 4 4 2 0 0 , 3 2 5 3 1 4 3 , 3 2 6 4 0 2 , 4 7 @ 8 place px. 9 0: P1 name capacity time 10 1: P4 65535 0 11 2: P6 65535 0 12 3: P7 65535 0 13 4: P9 65535 0 14 @ 15 trans px. 16 0: T1 0 0 17 1: T2 0 0 18 2: T5 0 0 19 3: T6 0 0 20 4: T8 0 0 21 @ </pre>	
(a).自動搬運車系統的 INA 輸入檔 (*.pnt)內容	(b).自動搬運車系統的裴氏圖 (已解除鎖死狀態)

圖 4-4、自動搬運車系統的裴氏圖比較

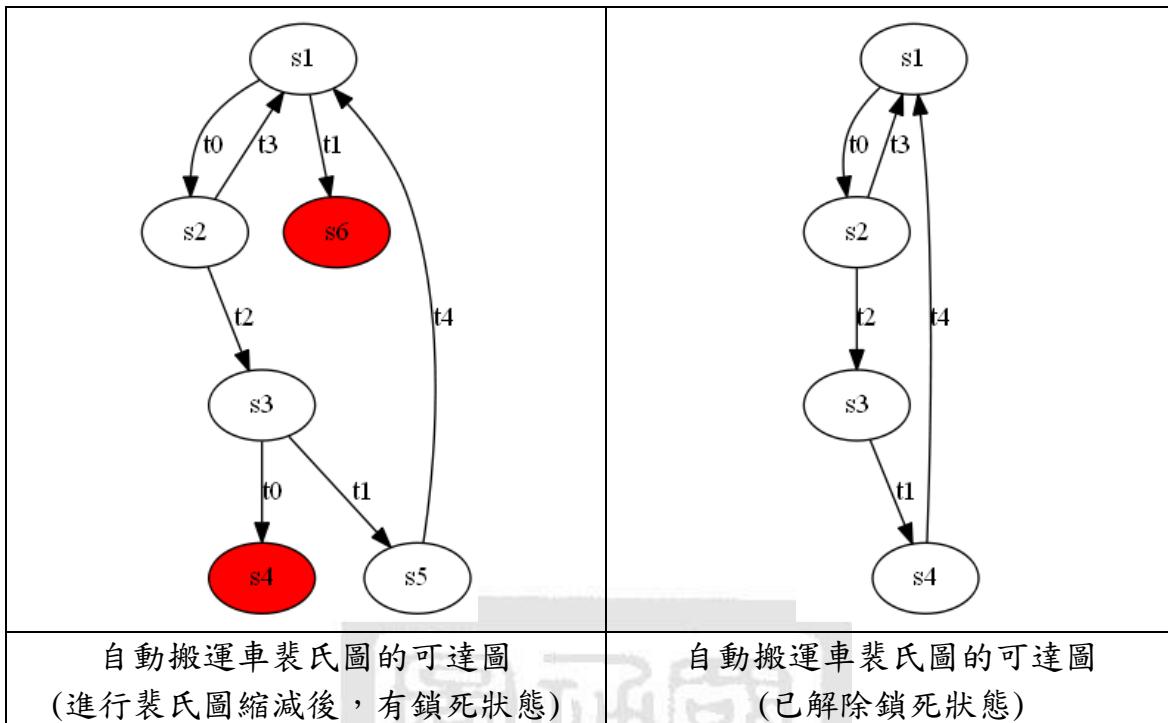


圖 4-5、自動搬運車系統的可達圖比較

從圖 4-5 可以發現，原本的狀態數共有 6 個(2 個鎖死狀態和 4 個合法狀態)，解除鎖死狀態後仍保有最大的可達狀態(4 個合法狀態)。

4.2 塑膠射出成型機簡介

塑膠射出成型機係將熔融的塑料注射到緊閉的模具中，整體的機械結構如圖 4-6，而塑膠射出成型機的動作流程大致上如下列文字敘述所示，整體的動作流程詳見圖 4-7。

● 開 / 關模(Mold open / Mold close)

射出成型是藉由將塑料注入緊閉的模具，經過冷卻成型的過程。因此射出成型機的動作流程中即包含成品冷卻後開啟模具與進行注塑前關閉模具之動作。

● 射座之前進 / 後退(Carriage fwd / Carriage bwd)

射座為射出成型機進行射出、保壓、儲料與鬆退之機構。射座之前進 / 後退的時機在於是否要進行射出動作。當進行射出動作時，必須將射座的射嘴抵住模具的澆口，產生一力量，避免在射出時因為射出的反作用力，將射嘴脫離模具之澆口，造成溢料。

● 封閉式射嘴、射出、保壓、儲料與前/後鬆退(Shutoff nozzle, Injection, Hold pressure, Dosing and Pre / Post decompression)

封閉試射嘴係指安裝在射嘴處一可開關射嘴的機構，可以氣、油壓缸致動。射出、保壓與鬆退皆是螺桿的直線運動，儲料則是螺桿的旋轉運動。射出成型機藉由將螺桿前推(射出)，把料管中熔融的液狀塑料注入模具。接著進行略低力量的螺桿前推動作，將塑料持續注入已經冷卻的模具，避免成品因冷卻產生的收縮現象(保壓)。鬆退動作則是將螺桿往後拉之動作，可分為(儲料)前鬆退與(儲料)後鬆退。其應用時機為：

1. 前鬆退(Pre-decompression)：

釋放熱澆道內的壓力。如果未釋放，存留的壓力會使熔化的塑料從熱澆道進入模穴，對下一次成型形成阻礙或影響成品品質。

2. 後鬆退(Post-decompression)：

- 利用後鬆退產生的吸力，防制溢料。
- 避免塑料因冷料堵住射嘴。
- 降低熱澆道模具的壓力。

儲料為螺桿的旋轉運動，藉由螺桿旋轉產生的剪切熱與料管加熱時的溫度，將塑料熔化與混煉，使塑料熔化得更均勻或是與色母混煉得更均勻。背壓(Back pressure)係指在進行儲料動作時，熔化之塑料會堆積在螺桿前端，前端的推力

使螺桿後退。藉由設定一壓力，使螺桿後退的速度變慢。背壓使用之時機：

- 增加塑料在料管中的混煉效果，尤其是有色色母與塑料的混合。
- 塑料在料管中的均質型不佳，尤其是塑料中有部分未熔化。
- 增加儲料量的精準度。

射出成型製程主要有充填、保壓、冷卻及頂出四個階段，射出單元主要為充填及保壓等功能，而在充填與保壓兩者之間壓縮轉換的程序稱為 V/P 切換。在充填階段，塑膠粒經過加熱及剪切變成融膠狀態，融膠經由螺桿轉動被推至螺桿射出端，再由停止轉動的螺桿向前擠壓至模具裡。填充過程，主要依照模穴大小、形狀與薄厚等多樣因素，執行速度輪廓控制，在充填期間的射出速度會依照射出時間或位置做多段性變化，此過程較著重於速度控制，而壓力控制只要做力量輸出的限制即可。在保壓階段時，融膠充滿了整個模穴，螺桿繼續加壓並注入融膠，以補償因為冷卻而造成的體積收縮，良好的保壓控制可以得到較佳的尺寸精度，在保壓階段，主要著重於壓力輪廓控制，此為在充填過程結束後的壓力調整。

● 托模(頂出)、風(壓)托模(Ejector, Air valve)

托模可分為油壓托模(Ejector)和風壓托模(Air valve)，視成品大小或其他因素使用。油壓托模係指藉由油壓缸移動托模板上的托模桿，觸碰模具內的頂針將成品頂出。風壓托模則是利用氣壓將模具內的成品脫模。

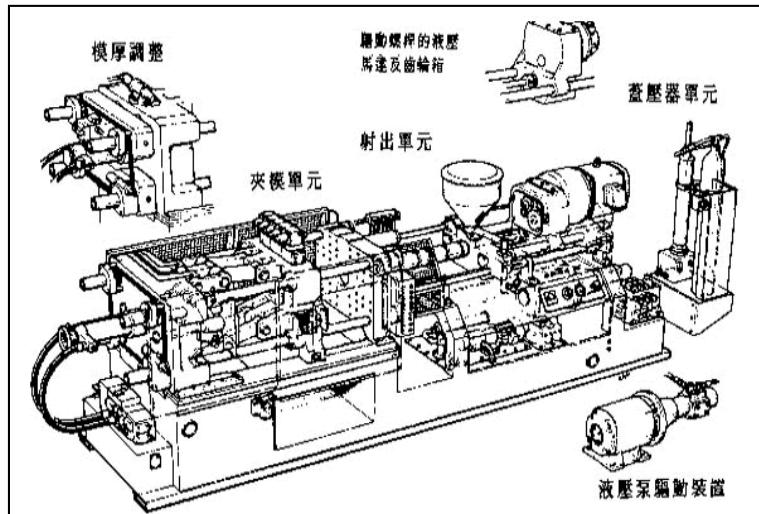


圖 4-6、塑膠射出成型機外觀圖

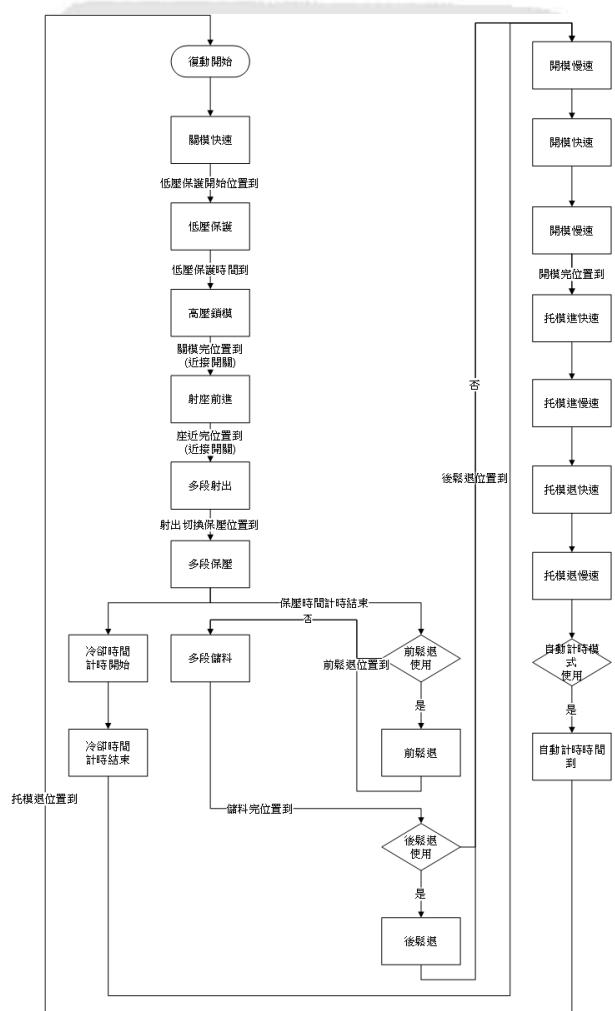


圖 4-7、塑膠射出成型機標準動作流程圖(自行整理)

4.3 塑膠射出成型機動作程式

本節將描述如何將塑膠射出成型機的動作流程，先繪製裴氏圖，進行分析與驗證，最後轉換為階梯圖。首先使用裴氏圖軟體 Snoopy 進行繪製。繪製完成後輸出為裴氏圖軟體 INA 的輸入檔(*.pnt)。接下來用裴氏圖軟體 INA 進行裴氏圖的可達圖分析，得到可達圖的輸出檔(*.gra)。之後再以本研究自行撰寫的程式進行具鎖死狀態裴氏圖的求解，並儲存成無鎖死狀態的裴氏圖。最後用裴氏圖軟體 Petri LLD 將裴氏圖轉換成階梯圖。

建構完成的裴氏圖詳見圖 4-8。將裴氏圖繪製好後，點擊檔案選項中「匯出(Export)」，選擇「Export to INA」選項，匯出成檔案(*.pnt)。INA 軟體的輸入檔案(*.pnt)主要由三個欄位組成：第一、二欄和暫存點有關，第三欄紀錄裴氏圖中各節點連結的關係。以圖 4-9 為例，第一欄表示共有 23 個暫存點，第二欄則是暫存點的初始浮標數。從圖中可得知僅第 1 個暫存點有浮標數 1。第三欄說明暫存點和轉移點的關係，逗號之前為進入暫存點的轉移點；逗號之後則為離開暫存點的轉移點。

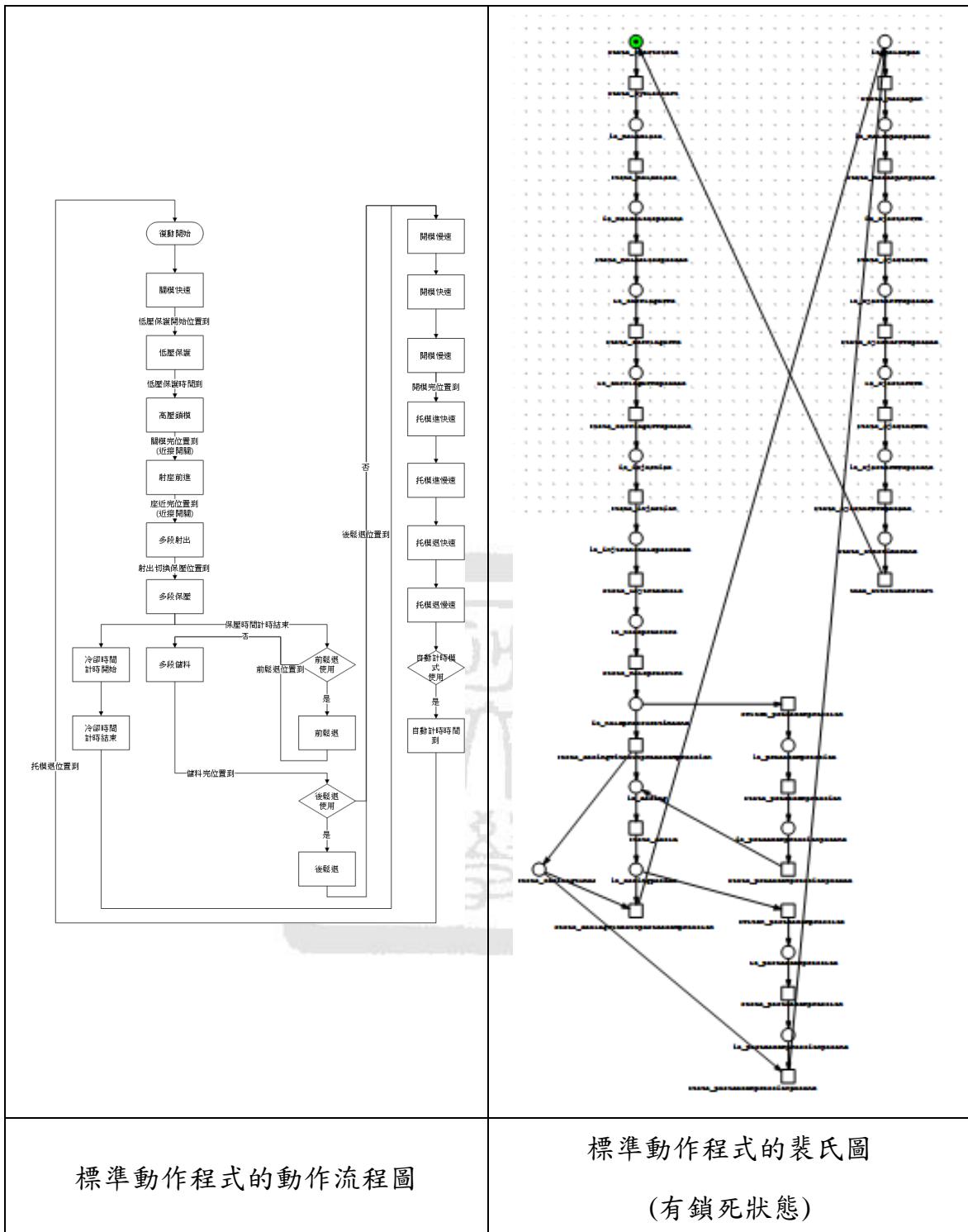


圖 4-8、標準動作程式的動作流程圖與具有鎖死狀態的裴氏圖

P	M	PRE, POST	NETZ	0:t
0	0	15 ,	1	
1	0	1 ,	2	
2	0	21 6 ,	4	
3	0	4 ,	22 5	
4	0	10 ,	7	
5	0	7 ,	8	
6	0	17 ,	9	
7	0	9 ,	10	
8	0	13 ,	11	
9	0	11 ,	23 6	
10	0	2 ,	12	
11	0	12 ,	13	
12	0	3 ,	14	
13	0	14 ,	15	
14	0	19 5 ,	16	
15	0	16 ,	17	
16	0	22 ,	18	
17	0	18 ,	19	
18	0	23 ,	20	
19	0	20 ,	21	
20	0	8 ,	0	
21	1	0 ,	3	
22	0	6 ,	5 19	
@				

圖 4-9、裴氏圖軟體 INA 的標準動作程式輸入檔

輸出 INA 的輸入檔(*.pnt)後，開啟裴氏圖分析軟體 INA，軟體的操作介面如圖 4-10，接下來的操作程序如下[Roch & Starke, 1999]：

1. 讀取檔案：

在 INA 的介面中依序輸入「E(編輯)」後，輸入「RF(讀取檔案)」，表示開啟裴氏圖檔案，輸入裴氏圖檔案的檔名(*.pnt)後輸入「Q(離開)」離開讀取檔案介面。

2. 輸出可達圖檔案：

在 INA 的介面中輸入「A(分析)」後，輸入「R(計算可達圖)」，開始進行裴氏圖的可達圖分析。輸入「W(輸出計算後的可達圖)」，並儲存成可達圖輸出檔(*.gra)。

因為在進行裴氏圖的鎖死處理過程中，需使用可達圖中的迴圈方程式，因此在計算可達圖時，在裴氏圖軟體 INA 中輸入「K(生成可達圖中的迴圈)」，即

可得到可達圖中迴圈的資訊，並可儲存成檔案(*.cir)，如圖 4-11。圖 4-11 中顯示，可達圖中存在兩個迴圈，並敘述迴圈中的暫存點與轉移點的關係。之後再以本研究撰寫的程式轉換為迴圈方程式。

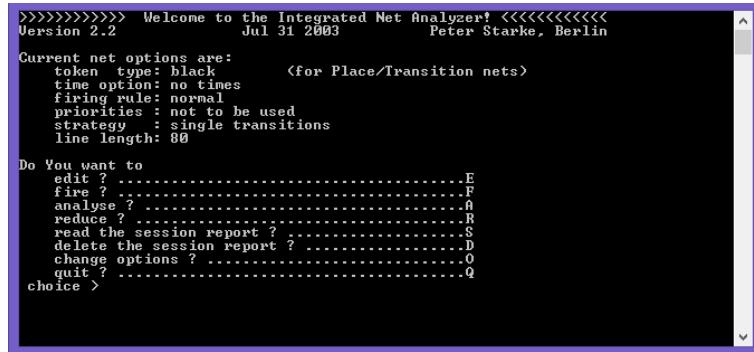


圖 4-10、裴氏圖分析軟體 INA

```

1
2 Circuit nr. 1:
3 n1<-t0--n18<-t8--n17<-t7--n16<-t10--n15<-t9--n14<-t17--n13
4 <-t16--n12<-t5--n11<-t4--n10<-t6--n9<-t11--n8<-t13--n7
5 <-t12--n6<-t2--n5<-t1--n4<-t15--n3<-t14--n2<-t3--n1
6
7 Corresponding T-invariant:
8 TR: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
9 : 23
10 : 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0
11 | 0
12 Livelooped transitions:
13 : 18, 19, 20, 21, 22, 23,
14 Circuit length = 18
15
16 Circuit nr. 2:
17 n1<-t0--n18<-t8--n17<-t7--n16<-t10--n15<-t9--n14<-t17--n13
18 <-t16--n12<-t19--n20<-t18--n19<-t22--n11<-t4--n10<-t6--n9
19 <-t11--n8<-t13--n7<-t12--n6<-t2--n5<-t1--n4<-t15--n3
20 <-t14--n2<-t3--n1
21 Corresponding T-invariant:
22 TR: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 : 23
24 : 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1
25 | 0
26 Livelooped transitions:
27 : 5, 20, 21, 23,
28 Circuit length = 20
29
30 Circuit nr. 1 has minimal length = 18.
31 Circuit nr. 2 has maximal length = 20.
32

```

圖 4-11、裴氏圖分析軟體 INA 生成的可達圖中的迴圈資訊

在計算可達圖的過程中，可能會因為裴氏圖過大，造成可達圖的狀態個數爆增的問題。藉由裴氏圖化簡的方法，可使原本的裴氏圖縮減成另一個較小的裴氏圖，並保留原本裴氏圖的特性。在 INA 中輸入「E(縮減)」開始裴氏圖的

化簡程序，主要的化簡功能有[Roch & Starke, 1999]：

- 將相同的節點融合(Fusion of congruent nodes, F)
- 合併等價的暫存點(Merging of equivalent places, M)
- 消除 $F(P^F) = p\text{-places}$ (Elimination of $F(P^F) = p\text{-places}$, A)
- 消除 $(F^P)F = p\text{-places}$ (Elimination of $(F^P)F = p\text{-places}$, B)
- 消除具迴圈的暫存點 (Elimination of looping places, C)
- 刪除具迴圈的轉移點 (Deletion of looping transitions, U)
- 刪除單一輸出的轉移點 (Deletion of single output transitions, V)
- 刪除 PTP 序列 (Deletion of PTP-sequences, W)

首先輸入「P(編輯化簡程序)」後，輸入「FMABCUVW」，再輸入「Q(結束)」，最後輸入「N(儲存成縮減後的裴氏圖)」即完成裴氏圖的化簡。此外當進行可達圖的計算時，INA 還有出現兩個裴氏圖化簡選項，但這兩個選項預設是關閉的，若要進行裴氏圖化簡，可將其開啟：

- 結構化化簡(Symmetrical Reduction)
- 頑固結構化簡(Stubborn Reduction)

接著以本研究自行撰寫的程式開啟 INA 的可達圖輸出檔案後，列出一組CMTSI 方程組，依據標記/移轉分離性質(MTSI)演算法法，開始進行具鎖死狀態裴氏圖的求解，如果鎖死狀態只有 1 個則僅需計算 1 次，2 個則須計算 2 次，依此類推。計算完成後。開啟原具有鎖死狀態的裴氏圖，將計算後的結果代入，得到無鎖死狀態的裴氏圖。由於標準動作程式 1 的裴氏圖中僅有 1 個鎖死狀態，因此僅需計算 1 次 CMTSI 方程組，包含 1 個事件分離狀態方程式、20 個可達狀態方程式和 2 個循環方程式，CMTSI 方程組詳見附錄中的圖 A-4。

從 Gurobi 得到的求解結果得知：需從新增暫存點連接到轉移點 t3(亦即 CMTSI 方程組中的變數 x4)、t23(亦即 CMTSI 方程組中的變數 x24)；需從 t0 (亦即 CMTSI 方程組中的變數 x1) 連接到新增戰存點。轉移點的編號與名稱的對照需參考裴氏圖軟體 INA 輸入檔(*.pnt)的描述。從圖 4-12 的轉移點編號與名稱對照得知，t3 (INA 輸入檔中的 tran 3)為轉移點「state_cyclestart」；t23 (INA 輸入檔中的 tran 23)為轉移點「switch_predecompression」；t0 (INA 輸入檔中的 tran 0)為「mode_autotimerstart」。對照求解後的處理：

- 如果結果顯示變數 x0 為 1，表示新增暫存點的浮標數為 1；
- 若變數 x1 到 xn 為 1 時，表示需從該對應的轉移點新增一條輸出弧到新增暫存點；
- 若變數 x1 到 xn 為 -1 時，表示需從新增暫存點新增一條輸出弧到該對應的轉移點。

<pre> 1 # Objective value = 0 2 0 0 3 x0 1 4 x4 -1 5 x15 0 6 x16 0 7 x2 0 8 x3 0 9 x13 0 10 x14 0 11 x12 0 12 x7 0 13 x5 0 14 x6 0 15 x17 0 16 x10 0 17 x18 0 18 x11 0 19 x8 0 20 x9 0 21 x23 0 22 x19 0 23 x24 -1 24 x21 0 25 x22 0 26 x1 1 27 x20 0 28 </pre>	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left;">tran_nr.</th><th style="text-align: left;">name</th><th style="text-align: left;">priority</th><th style="text-align: left;">time</th></tr> </thead> <tbody> <tr><td>51</td><td>0: mode_autotimerstart</td><td></td><td>0 0</td></tr> <tr><td>52</td><td>1: state_carriagefwd</td><td></td><td>0 0</td></tr> <tr><td>53</td><td>2: state_carriagefwdposend</td><td></td><td>0 0</td></tr> <tr><td>54</td><td>3: state_cyclestart</td><td></td><td>0 0</td></tr> <tr><td>55</td><td>4: state_dosing</td><td>0</td><td>0 0</td></tr> <tr><td>56</td><td>5: state_dosingwithoutpostdecompression</td><td></td><td>0 0</td></tr> <tr><td>57</td><td>6: state_dosingwithoutpredecompression</td><td></td><td>0 0</td></tr> <tr><td>58</td><td>7: state_ejectorbwrd</td><td></td><td>0 0</td></tr> <tr><td>59</td><td>8: state_ejectorbwrdposend</td><td></td><td>0 0</td></tr> <tr><td>60</td><td>9: state_ejectorfwd</td><td></td><td>0 0</td></tr> <tr><td>61</td><td>10: state_ejectorfwdposend</td><td></td><td>0 0</td></tr> <tr><td>62</td><td>11: state_holdpressure</td><td></td><td>0 0</td></tr> <tr><td>63</td><td>12: state_injection</td><td></td><td>0 0</td></tr> <tr><td>64</td><td>13: state_injtranshold</td><td></td><td>0 0</td></tr> <tr><td>65</td><td>14: state_moldclose</td><td></td><td>0 0</td></tr> <tr><td>66</td><td>15: state_moldcloseposend</td><td></td><td>0 0</td></tr> <tr><td>67</td><td>16: state_moldopen</td><td></td><td>0 0</td></tr> <tr><td>68</td><td>17: state_moldopenposend</td><td></td><td>0 0</td></tr> <tr><td>69</td><td>18: state_postdecompression</td><td></td><td>0 0</td></tr> <tr><td>70</td><td>19: state_postdecompressionposend</td><td></td><td>0 0</td></tr> <tr><td>71</td><td>20: state_predecompression</td><td></td><td>0 0</td></tr> <tr><td>72</td><td>21: state_predecompressionposend</td><td></td><td>0 0</td></tr> <tr><td>73</td><td>22: switch_postdecompression</td><td></td><td>0 0</td></tr> <tr><td>74</td><td>23: switch_predecompression</td><td></td><td>0 0</td></tr> <tr><td>75</td><td></td><td></td><td></td></tr> <tr><td>76</td><td>@</td><td></td><td></td></tr> <tr><td>77</td><td>--</td><td></td><td></td></tr> </tbody> </table>	tran_nr.	name	priority	time	51	0: mode_autotimerstart		0 0	52	1: state_carriagefwd		0 0	53	2: state_carriagefwdposend		0 0	54	3: state_cyclestart		0 0	55	4: state_dosing	0	0 0	56	5: state_dosingwithoutpostdecompression		0 0	57	6: state_dosingwithoutpredecompression		0 0	58	7: state_ejectorbwrd		0 0	59	8: state_ejectorbwrdposend		0 0	60	9: state_ejectorfwd		0 0	61	10: state_ejectorfwdposend		0 0	62	11: state_holdpressure		0 0	63	12: state_injection		0 0	64	13: state_injtranshold		0 0	65	14: state_moldclose		0 0	66	15: state_moldcloseposend		0 0	67	16: state_moldopen		0 0	68	17: state_moldopenposend		0 0	69	18: state_postdecompression		0 0	70	19: state_postdecompressionposend		0 0	71	20: state_predecompression		0 0	72	21: state_predecompressionposend		0 0	73	22: switch_postdecompression		0 0	74	23: switch_predecompression		0 0	75				76	@			77	--		
tran_nr.	name	priority	time																																																																																																														
51	0: mode_autotimerstart		0 0																																																																																																														
52	1: state_carriagefwd		0 0																																																																																																														
53	2: state_carriagefwdposend		0 0																																																																																																														
54	3: state_cyclestart		0 0																																																																																																														
55	4: state_dosing	0	0 0																																																																																																														
56	5: state_dosingwithoutpostdecompression		0 0																																																																																																														
57	6: state_dosingwithoutpredecompression		0 0																																																																																																														
58	7: state_ejectorbwrd		0 0																																																																																																														
59	8: state_ejectorbwrdposend		0 0																																																																																																														
60	9: state_ejectorfwd		0 0																																																																																																														
61	10: state_ejectorfwdposend		0 0																																																																																																														
62	11: state_holdpressure		0 0																																																																																																														
63	12: state_injection		0 0																																																																																																														
64	13: state_injtranshold		0 0																																																																																																														
65	14: state_moldclose		0 0																																																																																																														
66	15: state_moldcloseposend		0 0																																																																																																														
67	16: state_moldopen		0 0																																																																																																														
68	17: state_moldopenposend		0 0																																																																																																														
69	18: state_postdecompression		0 0																																																																																																														
70	19: state_postdecompressionposend		0 0																																																																																																														
71	20: state_predecompression		0 0																																																																																																														
72	21: state_predecompressionposend		0 0																																																																																																														
73	22: switch_postdecompression		0 0																																																																																																														
74	23: switch_predecompression		0 0																																																																																																														
75																																																																																																																	
76	@																																																																																																																
77	--																																																																																																																
標準動作程式裴氏圖 (有鎖死狀態)的求解結果	標準動作程式的 INA 輸入檔(*.pnt)內容																																																																																																																

圖 4-12、標準動作程式裴氏圖的求解結果與轉移點的編號和名稱對照

將求解結果代入原本具有鎖死狀態的裴氏圖中，得到無鎖死狀態的裴氏圖，如圖 4-13(b)。經過具有鎖死狀態的標準動作程式的裴氏圖與解除鎖死狀態的標準動作程式裴氏圖的可達圖（圖 4-14）比較後，發現鎖死的狀態已解除，且可達狀態數為刪除鎖死狀態（1 個狀態）與準鎖死狀態（5 個狀態）的最大可達狀態數（20 個狀態）。



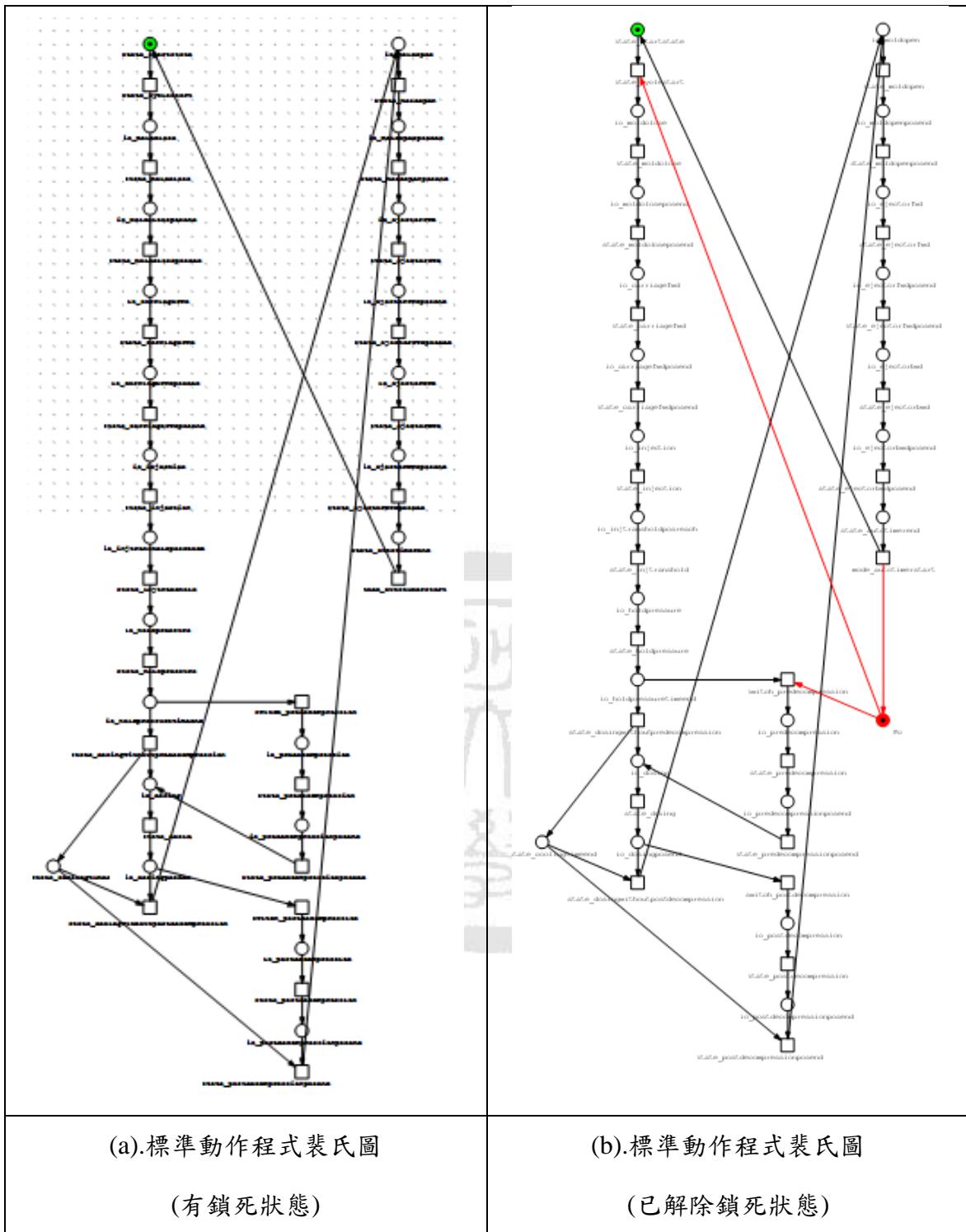


圖 4-13、標準動作程式「具鎖死狀態的裴氏圖」與「無鎖死狀態的裴氏圖」比較

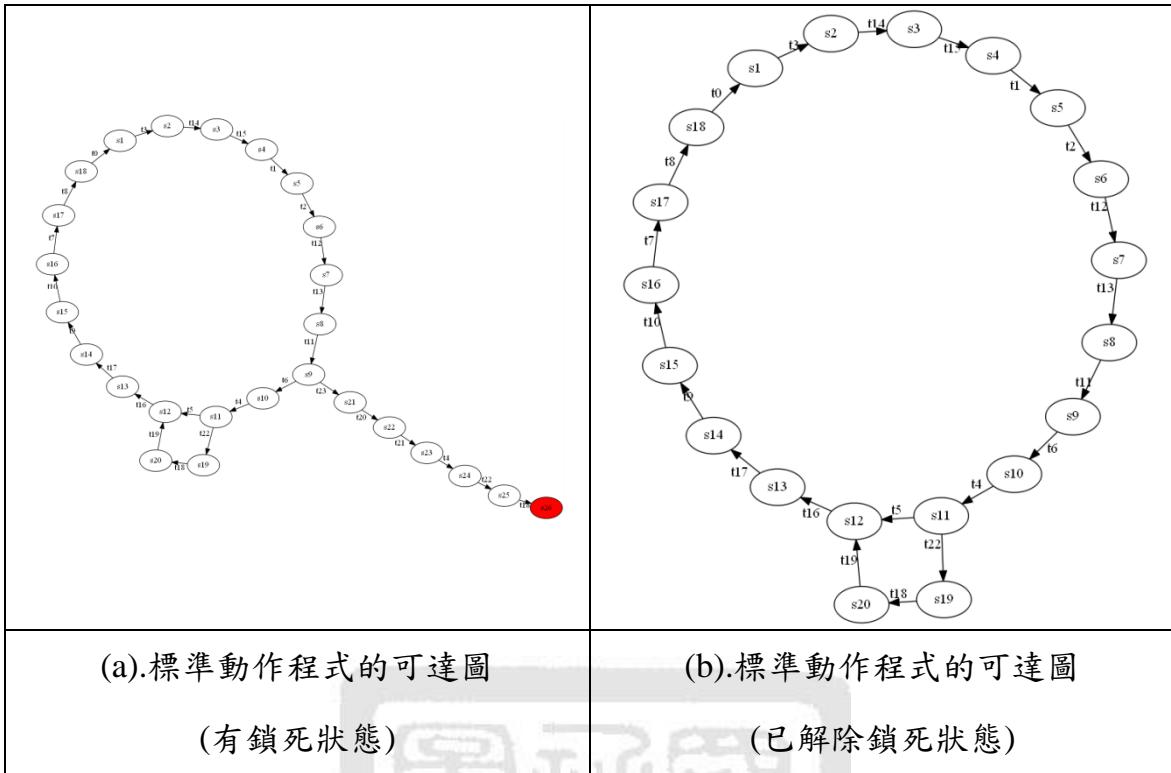
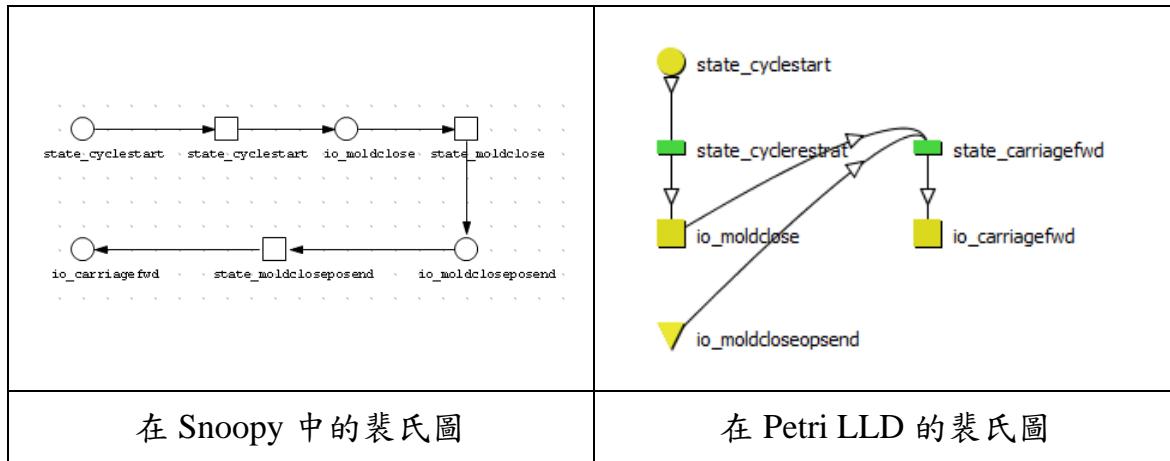


圖 4-14、標準動作程式「具鎖死狀態的可達圖」與「無鎖死狀態的可達圖」比較

取得無鎖死狀態的裴氏圖後，再以裴氏圖軟體 Petri LLD 進行裴氏圖與階梯圖的轉換。由於裴氏圖軟體 Snoopy 與 Petri LLD 的檔案格式不相容，因此需要進行格式上的轉換或是手動繪製。在進行檔案格式轉換或手動繪製時需注意，進行裴氏圖轉換階梯圖時，首先在 Petri LLD 新增一個專案(new project)和檔案(new net)。接著依照 INA 驗證後得到的無鎖死裴氏圖，在 Petri LLD 上重新呈現或是將無鎖死狀態的裴氏圖(*.pn-class)放在 Petri LLD 的專案資料夾中。過程中需將原裴氏圖中的輸入型態暫存點與其連結的輸出轉移點(Pre-Transition)修改為僅存在輸入型態暫存點。過程如表 4-1 所示：

表 4-1、裴氏圖在 Snoopy 和 Petri LLD 的對照



建構完裴氏圖後(圖 4-15(a))，進行裴氏圖與階梯圖的轉換。本研究選擇轉換為 Omron PLC 的格式。在程式樹狀目錄下，對專案進行右鍵點擊，會出現編譯(Compile project)的選項，選擇「Omron CX Programmer Ladder Diagram」後會輸出 Omron PLC 的輸入檔(*.cxt)。建議先開啟該檔案，將裴氏圖中暫存點和轉移點等的專案名稱和 net 的名稱清除，格式為(_Net 名稱_專案名稱)，再以 Omron PLC 程式編輯軟體 CX-ONE 開啟，會比較清晰明瞭。編譯完成後的階梯圖詳見附錄中的圖 4-16。經過圖 4-16：塑膠射出成型機標準動作程式的階梯圖與圖 4-15(a)：塑膠射出成型機動作程式的裴氏圖，裴氏圖的節點與弧的數量比較少。可見使用裴氏圖建構自動化系統後再轉換為階梯圖，確實能降低建構的時間，日後進行除錯或修改時也較容易。

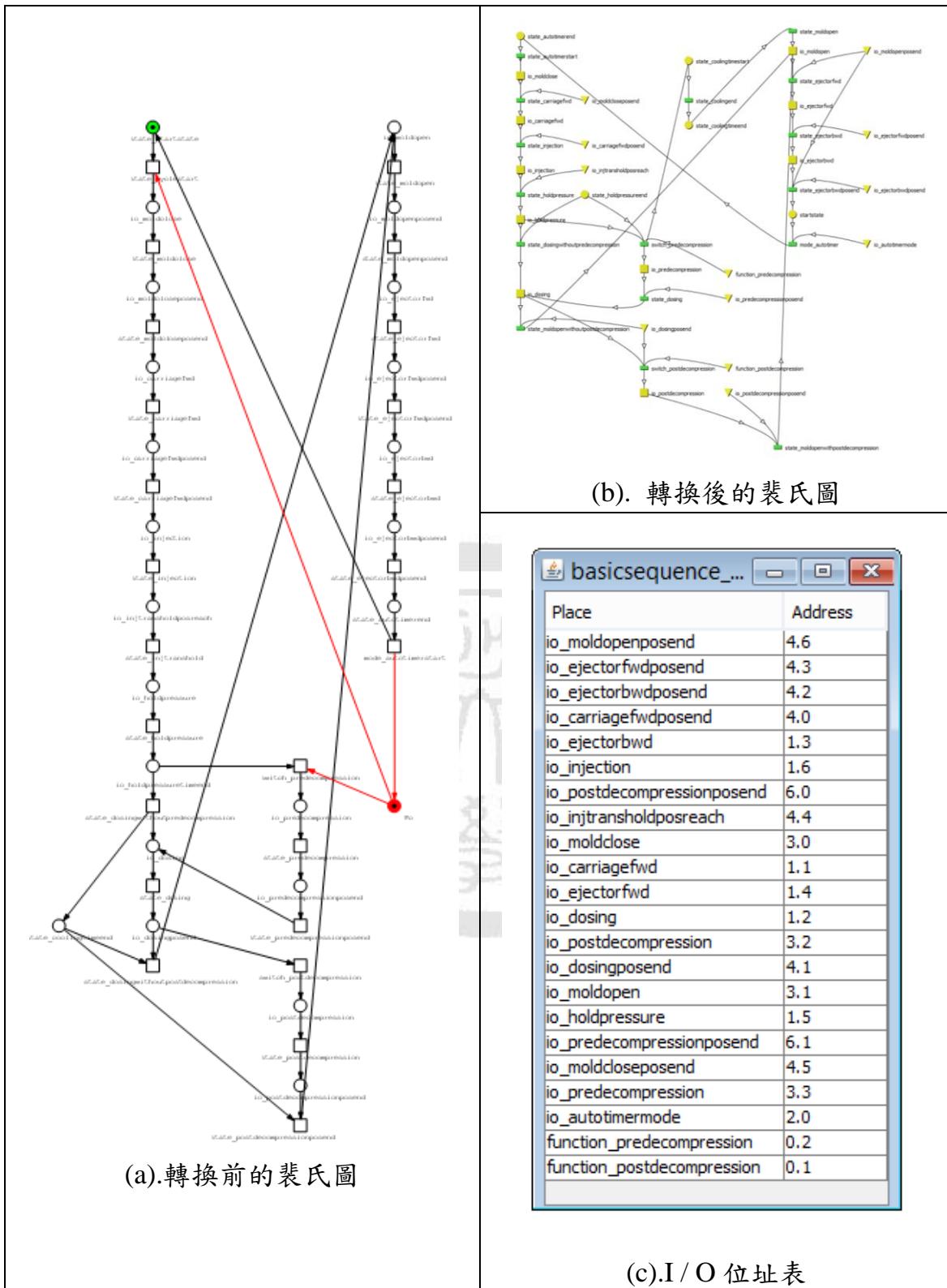


圖 4-15、轉換前與轉換後的裴氏圖比較

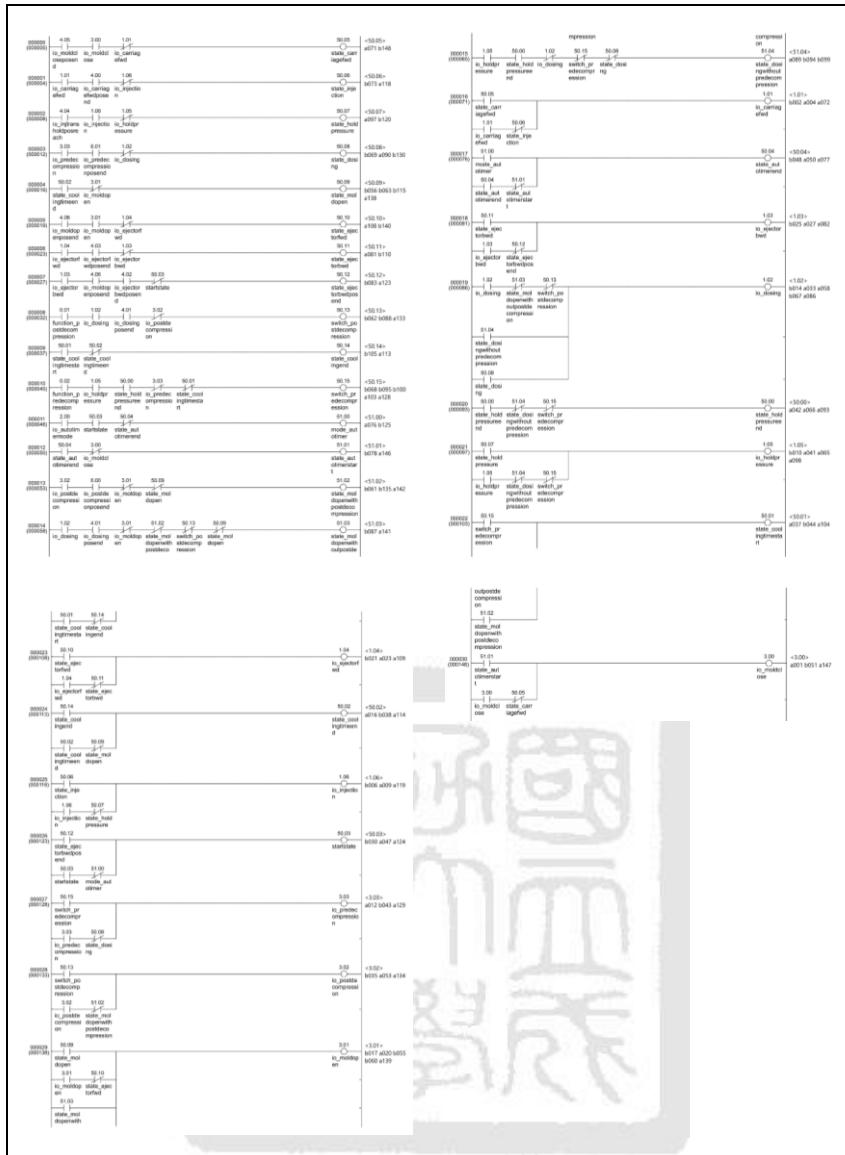


圖 4-16、經由裴氏圖軟體 Petri LLD 轉換後的標準動作程式階梯圖

5. 結論與後續研究建議

5.1 結論

因本研究主要為探討「基於裴氏圖方法的自動化系統編程方法階梯圖的鎖死處理」，因此將探討「裴氏圖鎖死處理」、「裴氏圖轉換為階梯圖」2個項目的相關文獻使用的方法，進行優劣比較後做為結論：

5.1.1 裴氏圖鎖死處理

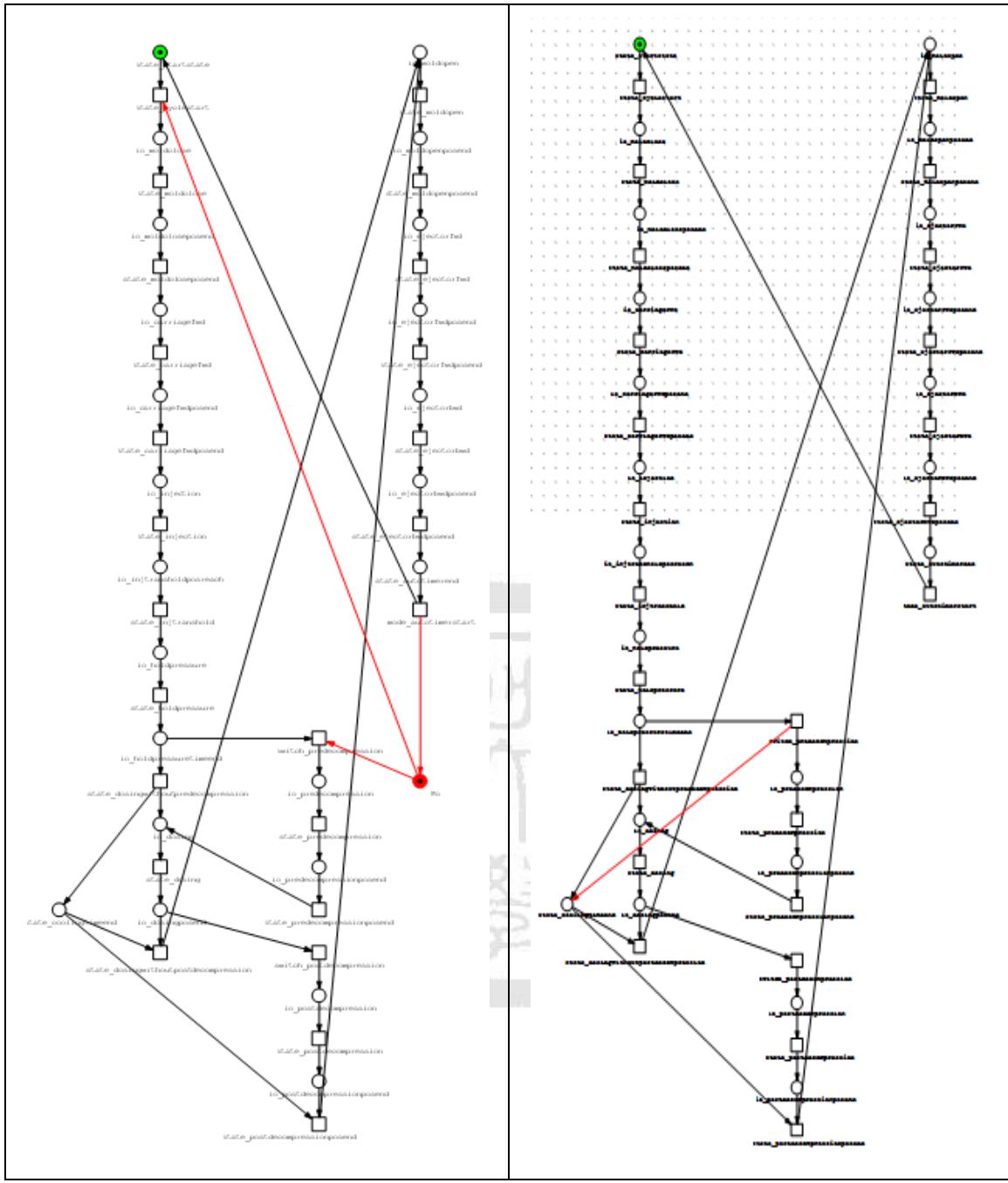
首先是「裴氏圖鎖死的處理」，根據以往的研究得知，裴氏圖鎖死問題處理方法的績效評估準則有：

- 最大的可達數
- 無冗餘的控制解數量
- 低計算成本(時間複雜度與空間複雜度)

本研究採用潘彥良 [民 100] 處理裴氏圖鎖死問題的方法：以區域理論與 CMTSI 的方法求出控制解，因此可得到最大的可達數，並以裴氏圖的化簡方法縮減裴氏圖的規模，能夠降低計算成本。依據潘彥良 [民 100] 的研究， Ezpeleta et al. [1995] 和 Li and Zhou [2004] 在處理裴氏圖鎖死問題的方法均無法達到保留最大可達數 (Maximally Permissive) 的目標，且均僅能使用在具有特殊結構裴氏圖的 S3PR 的彈性製造系統模式當中。Huang et al. [2006] 提出的方法突破了演算法 Ezpeleta et al. [1995] 及 Li and Zhou [2004] 被侷限在 S3PR 模型的缺點。但其採用迭代方式可能需要更多的時間及其方向弧是含有權重的則為其缺點。Uzam [2002]、Uzam and Zhou [2007]、Ghaffari et al. [2003] 與潘彥良 [民 100] 均藉由區域理論來獲得具鎖死彈性製造系統最大合法活性。

其中就演算法 Uzam [2002] 及 Uzam and Zhou [2007] 來看，後者採用迭代手段來獲得所有的控制庫所，而前者似乎又比後者來得有效益。而潘彥良 [民 100]採用結合簡化法、CMTSI 及限制理論為基底的預防鎖死策略來獲得最大活性的監控器，並宣稱其方法是目前裴氏圖鎖死處理的最佳方法。

然而就解除鎖死狀態後的可達圖而言，CMTSI 演算法係將可達圖中的鎖死狀態與準鎖死狀態剔除，僅保留可達圖中原有的合法可達狀態(圖 5-1(a))。以第四章實例說明中塑膠射出成型機動作流程為例，與原本設計的動作流程比較後，發現以 CMTSI 演算法解除鎖死狀態後的可達圖中，有一部分的動作流程被剔除。並且本研究經由試誤法發現另一個更佳的解，如圖 5-1(b)，僅在原本的裴氏圖中增加 1 條弧便可解除系統的鎖死狀態，並且可達圖的狀態與動作流程相符。經過具有鎖死狀態的裴氏圖與解除鎖死狀態後的裴氏圖的比較後，發現該系統的鎖死的狀態為「選擇開關_儲料前鬆退(switch_predecompression)到冷卻時間計時結束(coolingtimeend)」的動作沒有設計好而造成的鎖死。



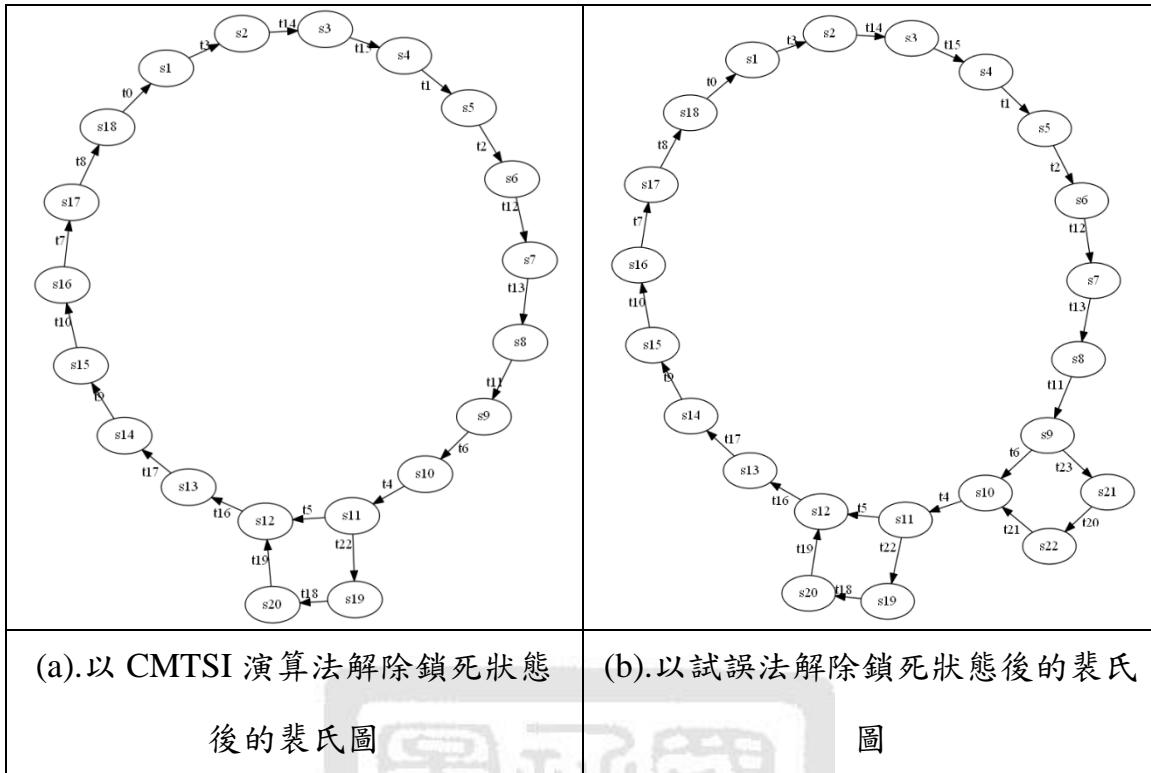


圖 5-1、CMTSI 演算法與試誤法處理裴氏圖鎖死狀態的比較

再者，潘彥良 [民 100]在其研究中提到 CMTSI 演算法有一些使用限制：無法處理具活鎖(Livelock)狀態的裴氏圖和初始狀態為鎖死狀態的裴氏圖。以 Brusey [2006]提供的實例 - 鑽孔機(drill press machine)，動作流程見本研究 3.5 節)來說明：圖 5-2(a)為具鎖死狀態的鑽孔機裴氏圖，使用 CMTSI 演算法進行鎖死狀態求解後(方程式 5.1 和 5.2)，得到圖 5-3 的解，將它代入圖 5-2(a)的裴氏圖中後得到圖 5-2(b)的裴氏圖。經觀察後發現鎖死狀態沒有解除，因為 CMTSI 演算法將圖 5-4(a)中的鎖死狀態和準鎖死狀態都剔除了，僅剩下為鎖死的初始狀態(圖 5-4(b))。相較於本研究經由試誤法的鎖死狀態求解，如圖 5-5(b)，可得到圖 5-6 的可達圖，不僅可以解除鎖死狀態，還保留了系統應有的動作流程可達狀態。

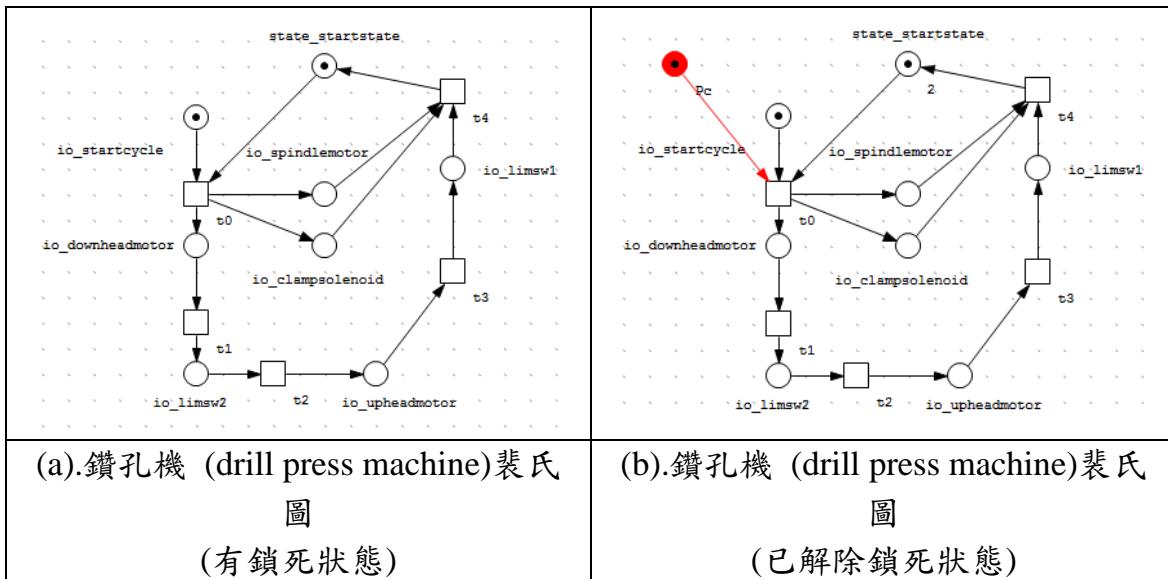


圖 5-2、鑽孔機裴氏圖解除鎖死狀態前後的比較

鑽孔機的CMTSI方程組：

事件分離狀態方程式

$$s6: x_0 + x_1 + x_2 + x_3 + x_5 + x_4 \leq -1 \quad (5.1)$$

可達狀態方程式

$$s1: x_0 = 1 \quad (5.2)$$

<pre>s6 is dead state s6 <= -1; The model is feasible x0 = 1 x1 = -2 x2 = 0 x3 = 0 x5 = 0 x4 = 0</pre>	<pre>21 trans ok- 22 0: _trans_0 23 1: _trans_1 24 2: _trans_2 25 3: _trans_3 26 4: _trans_4</pre>
<p>(a).鑽孔機鎖死狀態求解結果</p>	<p>(b).轉移點編號與名稱對照</p>

圖 5-3、鑽孔機鎖死狀態求解結果與裴氏圖中的轉移點編號與名稱對照

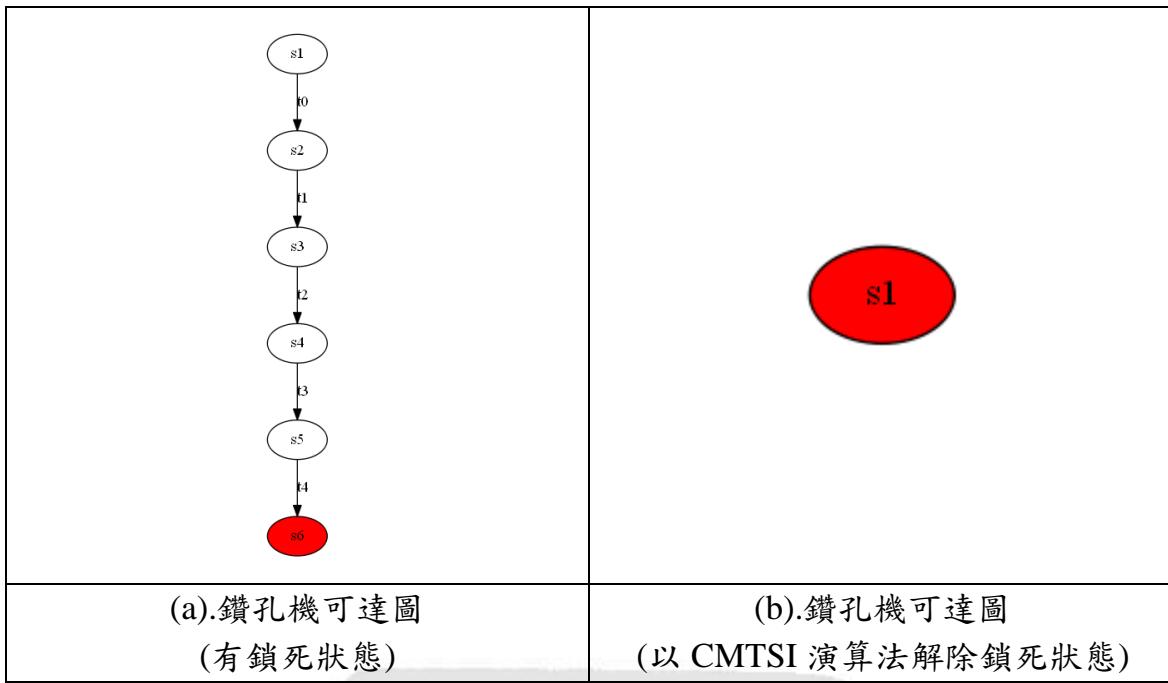


圖 5-4、鑽孔機解除鎖死狀態前後的可達圖比較

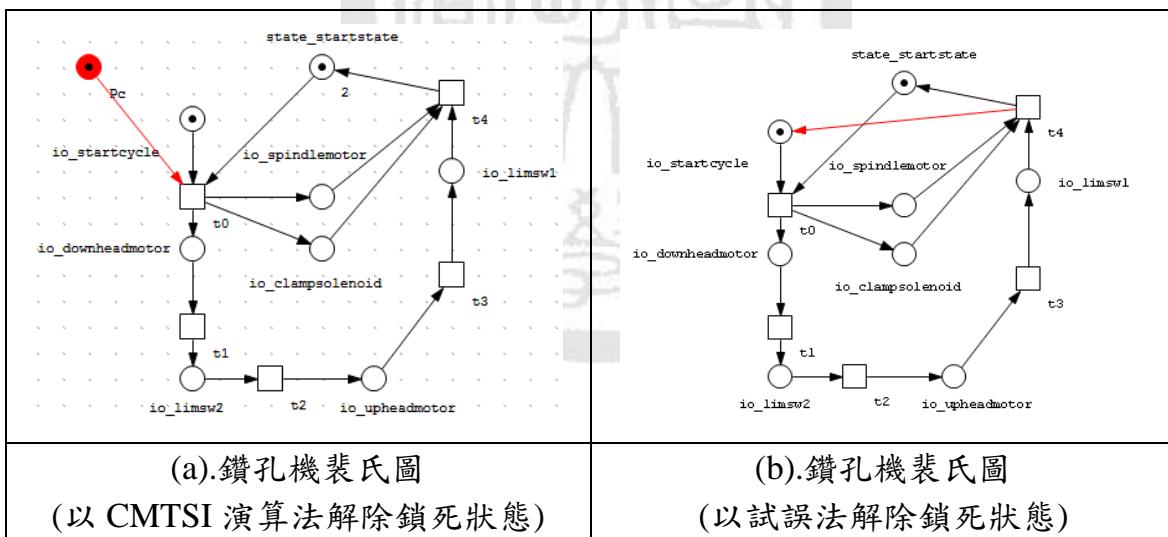


圖 5-5、鑽孔機裴氏圖解除鎖死狀態前後的裴氏圖比較

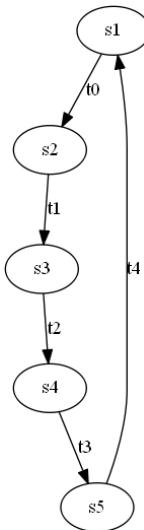


圖 5-6、鑽孔機裴氏圖解除鎖死狀態後的可達圖(以試誤法解除)

總結 5.1.1 節的內容，CMTSI 演算法雖可以解除系統的鎖死狀態，並可保留系統的最大可達狀態和保證系統不會出現鎖死狀態，但在「系統的初始狀態為鎖死狀態」和「系統具有活鎖狀態」的情況下，則無法作用。並且經由觀察發現，當系統的鎖死狀態包含初始狀態時，可以嘗試由可達圖中的最終鎖死狀態連接到初始狀態。對應到原本的裴氏圖時，則為從最終狀態的轉移點連接一條弧到初始的暫存點。若系統的鎖死狀態非上述的情況，則可嘗試將可達圖中的鎖死狀態連接弧到相鄰的最近可達狀態，以解除系統的鎖死狀態。

5.1.2 裴氏圖轉換為階梯圖

目前就「裴氏圖轉換為階梯圖」相關研究有：黃柏勳 [民 99]的三層式架構解裴氏圖/階梯圖轉換問題，但僅能將歐氏記號圖轉換為階梯圖。之後陳書皓 [民 102]提出五層式架構解歐氏記號圖/階梯圖轉換，可將浮標守恆裴氏圖轉換為階梯圖。Brusey [2006] 提出的裴氏圖軟體 Petri LLD 則是將訊號詮釋裴氏圖 (SIPN) 轉換為階梯圖，因訊號詮釋裴氏圖是專為可程式邏輯控制器的架構(將裴氏圖中的暫存點分為數位輸入(Digital Input, DI)、數位輸出(Digital Output, DO)

和暫存繼電器)發展，因此是較好的轉換方法。

5.2 後續研究建議

由於裴氏圖軟體的檔案格式不一，雖然有裴氏圖 XML (Petri net Markup Language, PNML)，仍須將檔案轉換為各種裴氏圖軟體的格式才能使用，此為本研究過程遭遇的難題之一。經過深入探討後，本研究認為裴氏圖軟體 INA 的輸入檔 (*pnt) 是最佳的儲存格式，因其結構簡潔，軟體本身又具備強大的分析能力。再者一般的可程式邏輯控制器編程人員大都沒有接觸過裴氏圖，建議未來能夠發展階梯圖(LD)或基本指令(IL)轉換為裴氏圖的軟體，方便未接觸過裴氏圖的使用者能夠加速建構裴氏圖的過程，也能提升階梯圖直接進行驗證的速度。最後裴氏圖軟體 Petri LLD 也提供 Visual Basic.net(vb.net)的輸出檔案，讓使用者能將裴氏圖編譯成 Visual Basic.net 檔案，便能夠製作遠端監控的介面或是搭配 Festo FluidSim 液壓模擬軟體，以 DDE / OPC 連線方式建構虛擬的模擬台，詳細的建構方法可參照李麗娜，柳洪義, and 孫一藍 [2007]的研究。

參考文獻

1. Aiken, A., Fahndrich, M., & Su, Z. 1998. Detecting Races in Relay Ladder Logic Programs. *Tools and Algorithms for the Construction and Analysis of Systems*, 1384.
2. Aybar, A., & Iftar, A. 2008. Deadlock Avoidance Controller Design for Timed Petri Nets Using Stretching. *IEEE Systems Journal*, 2.
3. Brusey, J. 2006. *PetriLLD Tutorial*.
4. Chen, S.-H., & Liang, G.-R. 2013. Five Layers Architecture Approach to Eulerian Marked Graph / Ladder Diagram Transformation Problem. *Journal of the Mechatronic Industry*, 363.
5. Cohen, Y., Wang, M.-E., & Bidanda, B. 2010. Automatic Translation of a Process Level Petri-Net to a Ladder Diagram. *Advanced Techniques in Computing Sciences and Software Engineering*.
6. Dohi, Y., Nomura, E., Shimoda, T., & Murakoshi, H. 1996. Petri Net Controller with Hardware to Avoid Deadlocks.
7. Ezpeleta, J., Colom, J. M., & Martinez, J. 1995. A Petri Net Based Dedlock Prevention Policy for Flexible Manufacturing Systems. *IEEE Transactions on Robotics and Automation*, 11: 173-184.
8. Frey, G., & Wagner, F. 2006. A Toolbox for the Development of Logic Controllers using Petri Nets. *Proceesings of the 8th International Workshop on Discrete Event System (WODES 2006)*.
9. Gang, X., & Wu, Z. 2004. Deadlock-Free Scheduling Strategy for Automated Production Cell. *IEEE Trans actions on Systems, Man, and Cybernetics*, 34.
10. Ghaffari, A., Rezg, N., & Xie, X. 2003. Design of a Live and Maximally Permissive Petri Net Controller Using the Theory of Regions. *IEEE Transactions on Robotics and Automation*, 19: 137-142.
11. Han, K.-H. 2010. *Object-Oriented Modeling, Simulation and Automatic Generation of PLC Ladder Logic*. intech.
12. Heiner, M., Herajy, M., Liu, F., Rohr, C., & Schwarick, M. 2012. Snoopy – A Unifying Petri Net Tool. In S. Haddad, & L. Pomello (Eds.), *Application and Theory of Petri Nets*, Vol. 7347: 398-407: Springer Berlin Heidelberg.
13. Huang, B.-X., & Liang, G.-R. 2010. Three Layers Architecture Approach to Petri Net / Ladder Diagram Transformation Problem. *Journal of the Mechatronic Industry*, 329.
14. Huang, Y.-S., Xie, X., & Chung, D.-H. 2006. Siphon-Based Deadlock Prevention Policy for Flexible Manufacturing Systems. *IEEE Trans actions on Systems, Man, and Cybernetics*, 36.
15. Lee, J.-S., & Hsu, P.-L. 2004. An Improved Evaluation of Ladder Logic Diagrams and

- Petri Nets for The Sequence Controller Design in Manufacturing Systems. *Intelligent Advanced Manufacture Technology*, 24: 279-287.
- 16. Lee, S., & Tilbury, D. M. 2007. Deadlock-Free Resource Allocation Control for a Reconfigurable Manufacturing System With Serial and Parallel Configuratio. *IEEE Transactions on Systems, Man, and Cybernetics*, 37.
 - 17. Li, L., Liu, H., & Sun, Y. 2007. A Method for the Simulation of Hydraulic System Controlled by PLC Based on LabView and FluidSIM-H. *Machine, Tool & Hydraulics*, 35.
 - 18. Li, Z.-W., & Zhou, M.-C. 2004. Elementary Siphons of Petri Nets and Their Application to Deadlock Prevention in Flexible Manufacturing Systems. *IEEE Transactions on Systems, Man, and Cybernetics*, 34.
 - 19. Li, Z., Zhou, M., & Wu, N. 2008. A Survey and Comparison of Petri Net-Based Deadlock Prevention Policies for Flexible Manufacturing Systems *IEEE Transactions on Robotics and Automation*: 173-188.
 - 20. Mohan, S., Yalcin, A., & Khator, S. 2004. Controller Design and Performance Evaluation for Deadlock Avoidance in Automated Flexible Manufacturing Cells. *Robotics and Computer-Integrated Manufacturing*.
 - 21. Moorthy, R. L., Wee, H.-G., Ng, W.-C., & Teo, C.-P. 2003. Cyclic Deadlock Prediction and Avoidance for Zone-Controlled AGV. *International Journal of Production Economics*.
 - 22. Optimizer, G. 2013. Gurobi Optimizer.
 - 23. Ramadge, P. J. G., & Wonham, W. M. 1989. The Control of Discrete Event Systems. *Proceedings of IEEE*, 77.
 - 24. Roch, S., & Starke, P. P. H. 1999. *INA - Integrated Net Analyzer_ Version 2.2*. Humboldt university.
 - 25. Silberschatz, A., Galvin, P. B., & Gagne, G. 2005. *Operating System Concepts* (7 ed.).
 - 26. Uzam, M. 2002. An Optimal Deadlock Prevention Policy for Flexible Manufacturing Systems Using Petri Net Models with Resources and the Theory of Regions. *Advanced manufacturing technology*, 19: 192-208.
 - 27. Uzam, M. 2004. The Use of the Petri Net Reduction Approach for an Optimal Deadlock Prevention Policy for Flexible Manufacturing Systems. *Advanced manufacturing technology*, 23: 204-219.
 - 28. Uzam, M., & Zhou, M.-C. 2007. An Iterative Synthesis Approach to Petri Net-Based Deadlock Prevention Policy for Flexible Manufacturing Systems. *IEEE Trans actions on Systems, Man, and Cybernetics*.
 - 29. Viswandham, N., Narahari, Y., & Johnson, T. L. 1990. Deadlock Prevention and Deadlock Avoidance in Flexible Manufacturing Systems Using Petri Net Models. *IEEE*

- Transactions on Robotics and Automation*, 6: 713-723.
30. Xing, K., Jin, X., & Feng, Y. 2005. Deadlock Avoidance Petri Net Controller for Manufacturing Systems with Multiple Resource Service, *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*.
 31. Yoon, H. J., & Lee, D. Y. 2004. Deadlock-Free Scheduling of Photolithography Equipment in Semiconductor Fabrication. *IEEE transactions on Semiconductor Manufacturing* 17.
 32. 三菱電機股份有限公司. 民 97. 三菱可程式控制器 PLC 入門篇.
 33. 方本欣, 張弘, & 梁高榮. 民 100. 用法則矩陣分解法從選項裴氏圖產生記號圖. *機械工業雜誌*, 341: 104-116.
 34. 王元昌. 民 98. 活用 Omron PLC 的技巧: 全華圖書股份有限公司.
 35. 甘清華. 2010. 基於信標選擇的死鎖控制算法研究. 西安電子科技大學.
 36. 吳添財. 民 91. 以裴氏網路建構彈性製造系統之動態模擬與線上診斷系統之研究. 國立高雄第一科技大學.
 37. 李志武, & 周孟初. 2009. 自動製造系統建模、分析與死鎖控制: 科學出版社.
 38. 李開南. 2010. 基於區域理論的 Petri 網活性控制器優化設計. 西安電子科技大學.
 39. 李麗娜, 柳洪義, & 孫一藍. 2007. 一種基於 LabView 及 FluidSim-H 的 PLC 控制液壓系統仿真方法. *機床與液壓*, 35.
 40. 宓哲民. 民 101. 機電整合 可程式應用原理與應用實務: 全華圖書股份有限公司.
 41. 林楊祥. 民 101. 可程式控制器程式設計技術的發展與驗證. 遠東科技大學.
 42. 林潔妤. 民 97. 裴氏圖軟體撰碼問題中自動規範測試系統的設計與實作. 國立交通大學.
 43. 梁高榮. 民 98. 裴氏圖與記號圖：可程式控制器的分析工具. *機械工業雜誌*, 319: 119-130.
 44. 梁高榮. 民 100. 利用動態擬陣理論設計台性製造系統偵查法則. *機械工業雜誌*, 340.
 45. 陳音帆. 民 97. 裴氏圖導向控制器開發系統的設計與實作. 國立交通大學.
 46. 陳書皓, 梁. 民 102. 五層式架構解歐氏記號圖 / 階梯圖轉換問題. *機械工業雜誌*, 361.
 47. 傅振棻. 民 97. 氣壓迴路設計經典: 全華圖書股份有限公司.
 48. 曾呂國. 民 94. 整合 IDEF 與派翠網路於物件導向至控制系統的分析與設計. 國立雲林科技大學.
 49. 黃柏勳, 梁. 民 99. 三層式架構解裴氏圖/階梯圖轉換問題. *機械工業雜誌*, 329: 114-126.
 50. 黃泰霖. 民 102. 應用派翠網路於資源限制改變之製造系統控制的動態重構. 國立雲林科技大學.
 51. 楊鎮宇, 何佳育, & 朱智煒. 民 99. CoDeSys 控制軟體操作說明. 逢甲大學.

52. 廖扶西. 民 89. **派翠網路的基本架構**. 國立政治大學.
53. 廖崧富. 民 93. **整合 IDEF/CTPN/SFC 於間斷事件控制系統 PLC 分析與設計之研究**. 國立雲林科技大學.
54. 潘彥良. 民 100. **以限制理論為基礎之彈性製造系統死鎖預防策略之研究**. 國防大學理工學院.
55. 鄭仲元. 民 99. **記號圖到階梯圖的自動物件轉換**. 國立交通大學.



附錄

```
Dim file As String
file = "gurobi.ilp"

Try
    Dim env As GRBEnv = New GRBEnv("outcome.log")
    Dim model As GRBModel = New GRBModel(env, file)
    model.Optimize()

    Dim optimstatus As Integer = model.Get(GRB.IntAttr.Status)

    If optimstatus = GRB.Status.INF_OR_UNBD Then
        model.GetEnv().Set(GRB.IntParam.Presolve, 0)
        model.Optimize()
        optimstatus = model.Get(GRB.IntAttr.Status)
    End If

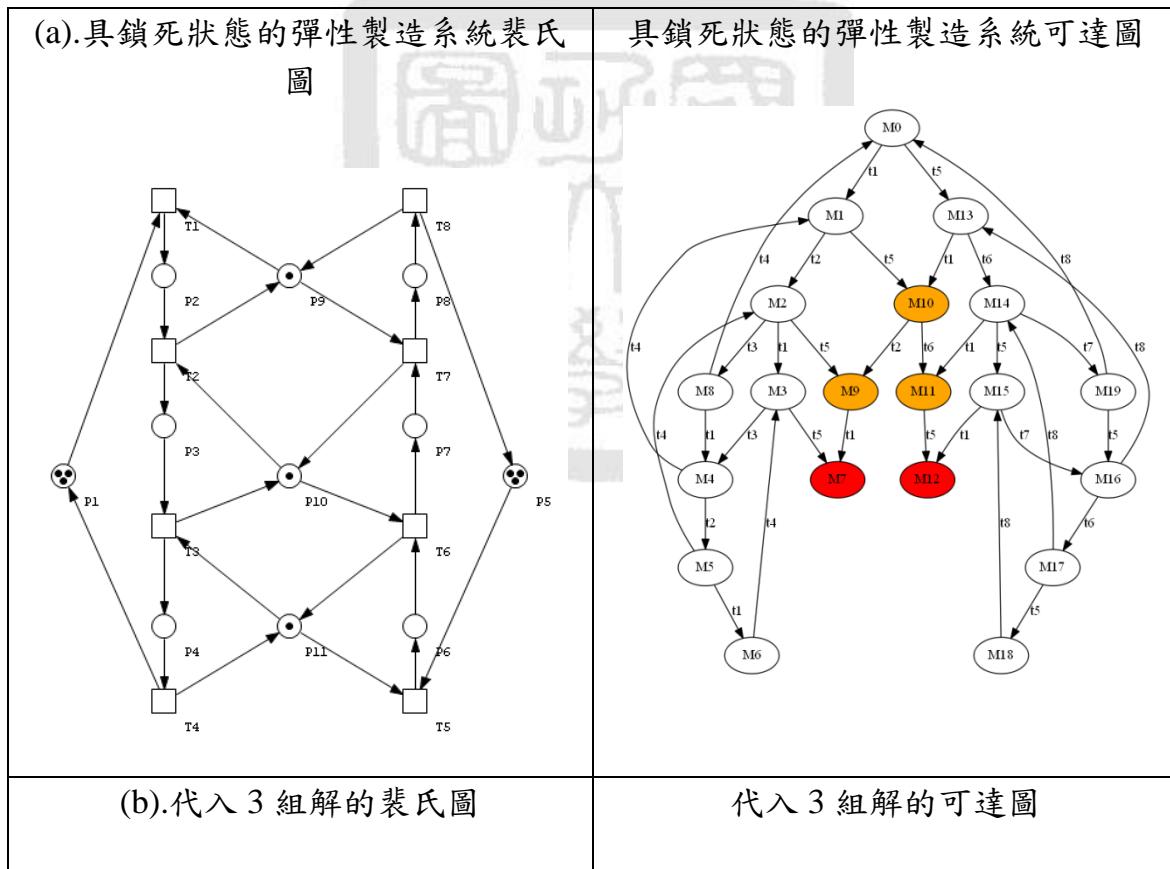
    If optimstatus = GRB.Status.OPTIMAL Then
        RichTextBox6.AppendText("The model is feasible" & vbCrLf)
        model.Write("out.sol")
    ElseIf optimstatus = GRB.Status.INFEASIBLE Then
        RichTextBox6.AppendText("Model is infeasible" & vbCrLf)
        model.ComputeIIS()
        model.Write("model.ilp")
    ElseIf optimstatus = GRB.Status.UNBOUNDED Then
        RichTextBox6.AppendText("Model is unbounded" & vbCrLf)
    Else
        RichTextBox6.AppendText("Optimization was stopped with status = " & optimstatus
        & vbCrLf)
    End If

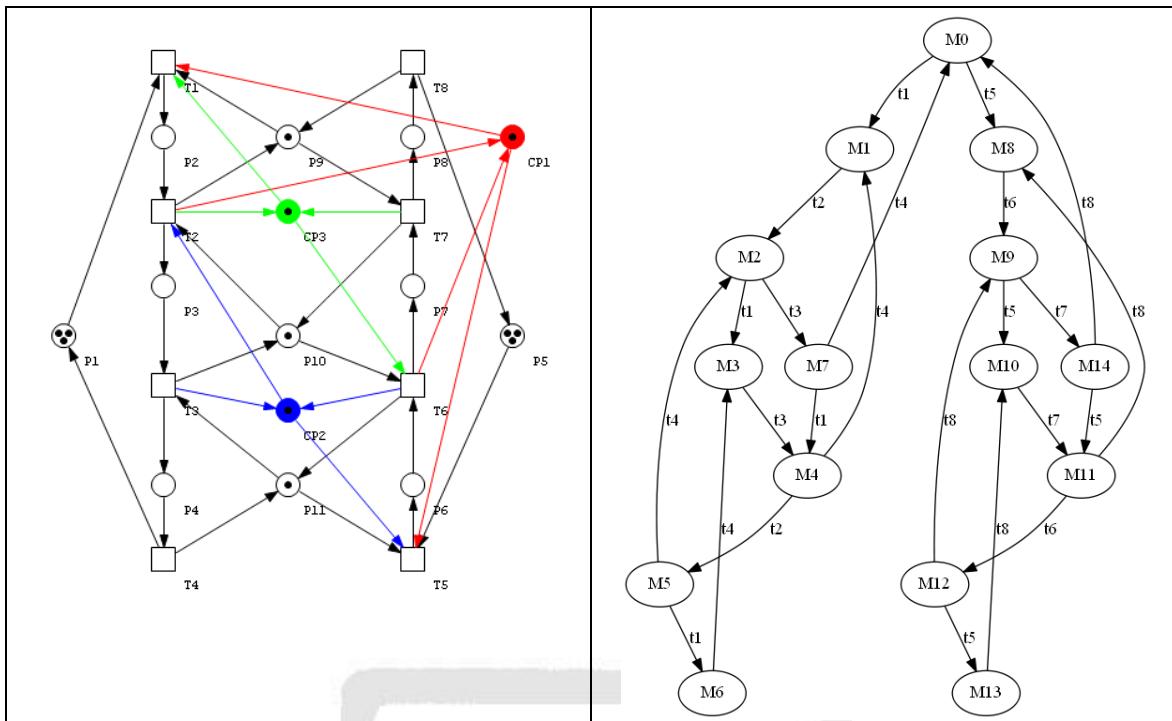
    ##### Dispose of model and env
    model.Dispose()
    env.Dispose()

    Catch ex As GRBException
        RichTextBox6.AppendText("Error code: " & ex.ErrorCode & ". " & ex.Message &
        vbCrLf & vbCrLf)
    End Try
```

圖 A-1、引用 Gurobi 的 DLL，進行數學規劃求解的程式碼

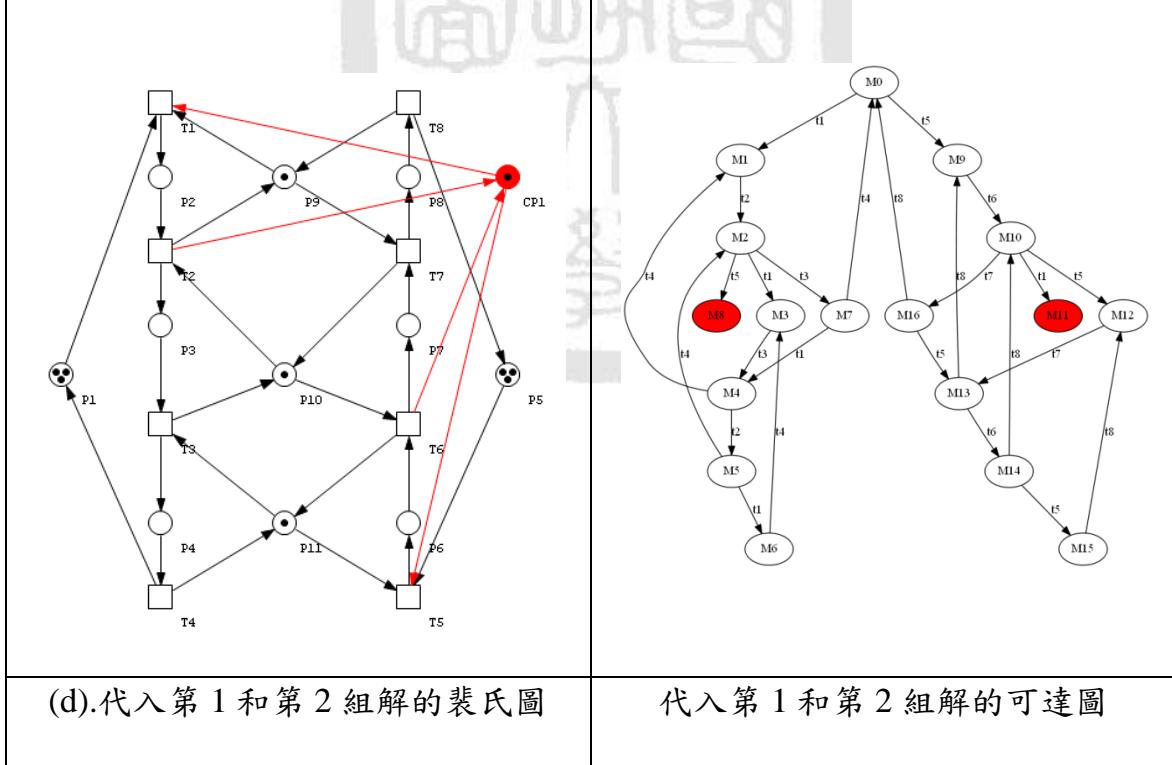
圖 A-2、裴氏圖軟體 INA 分析後的標準動作程式可達圖輸出檔(有鎖死狀態)





(c).代入第 1 組解的裴氏圖

代入第 1 組解的可達圖



(d).代入第 1 和第 2 組解的裴氏圖

代入第 1 和第 2 組解的可達圖

<p>(e).代入第 1 和第 3 組解的裴氏圖</p>	<p>代入第 1 和第 3 組解的可達圖</p>

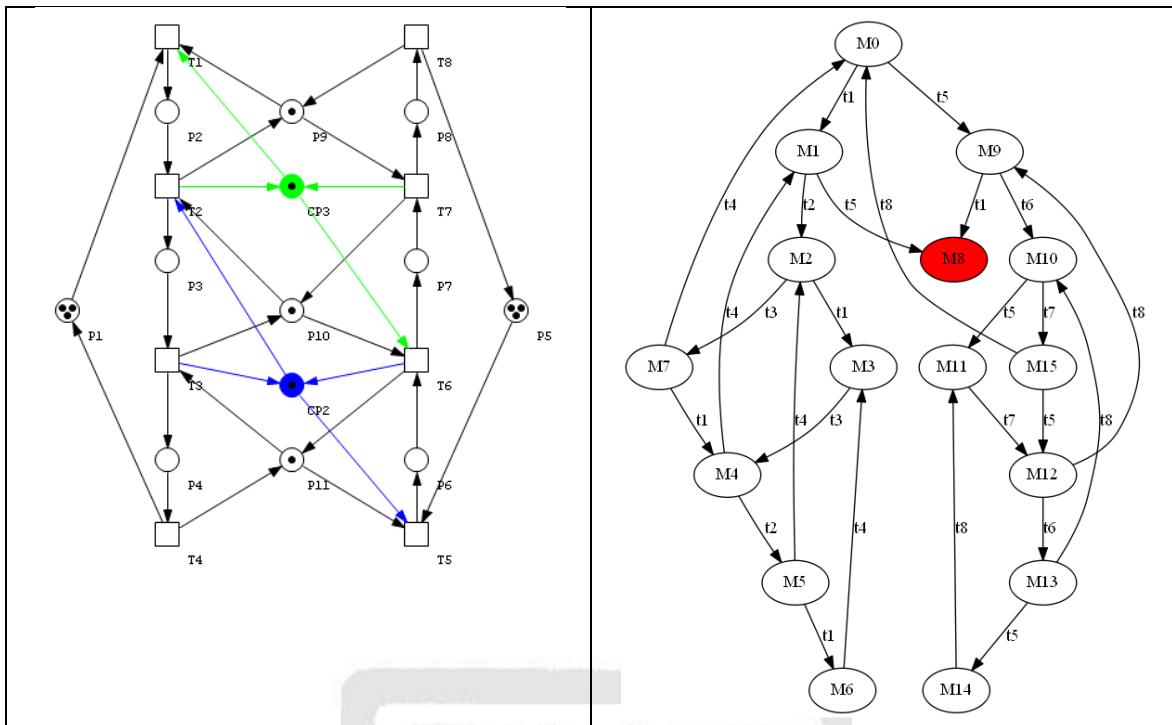


圖 A-3、塑膠射出成型機動作程式的可達圖生成的 CMSTI 方程組

```

Minimize 0
Subject to
s1:x0 = 1
s2:x0 + x4 >= 0
s3:x0 + x4 + x15 >= 0
s4:x0 + x4 + x15 + x16 >= 0
s5:x0 + x4 + x15 + x16 + x2 >= 0
s6:x0 + x4 + x15 + x16 + x2 + x3 >= 0
s7:x0 + x4 + x15 + x16 + x2 + x3 + x13 >= 0
s8:x0 + x4 + x15 + x16 + x2 + x3 + x13 + x14 >= 0
s9:x0 + x4 + x15 + x16 + x2 + x3 + x13 + x14 + x12 >= 0
s10:x0 + x4 + x15 + x16 + x2 + x3 + x13 + x14 + x12 + x7 >= 0
s11:x0 + x4 + x15 + x16 + x2 + x3 + x13 + x14 + x12 + x7 + x5 >= 0
s12:x0 + x4 + x15 + x16 + x2 + x3 + x13 + x14 + x12 + x7 + x5 + x6 >= 0
s13:x0 + x4 + x15 + x16 + x2 + x3 + x13 + x14 + x12 + x7 + x5 + x6 + x17 >= 0
s14:x0 + x4 + x15 + x16 + x2 + x3 + x13 + x14 + x12 + x7 + x5 + x6 + x17 + x10 >= 0
s15:x0 + x4 + x15 + x16 + x2 + x3 + x13 + x14 + x12 + x7 + x5 + x6 + x17 + x18 + x10 >= 0
s16:x0 + x4 + x15 + x16 + x2 + x3 + x13 + x14 + x12 + x7 + x5 + x6 + x17 + x18 + x10 + x11 >= 0
s17:x0 + x4 + x15 + x16 + x2 + x3 + x13 + x14 + x12 + x7 + x5 + x6 + x17 + x18 + x10 + x11 + x8 >= 0
s18:x0 + x4 + x15 + x16 + x2 + x3 + x13 + x14 + x12 + x7 + x5 + x6 + x17 + x18 + x10 + x11 + x8 + x9

```

```

>= 0
s19:x0 + x4 + x15 + x16 + x2 + x3 + x13 + x14 + x12 + x7 + x5 + x23 >= 0
s20:x0 + x4 + x15 + x16 + x2 + x3 + x13 + x14 + x12 + x7 + x5 + x23 + x19 >= 0
s26:x0 + x4 + x15 + x16 + x2 + x3 + x13 + x14 + x12 + x24 + x21 + x22 + x5 + x23 + x19 <= -1
Cr1:x1 + x9 + x8 + x11 + x10 + x18 + x17 + x6 + x5 + x7 + x12 + x14 + x13 + x3 + x2 + x16 + x15 + x4
= 0
Cr2:x1 + x9 + x8 + x11 + x10 + x18 + x17 + x20 + x19 + x23 + x5 + x7 + x12 + x14 + x13 + x3 + x2 + x16
+ x15 + x4 = 0
Bounds
x0 free
x1 free
x2 free
x3 free
x4 free
x5 free
x6 free
x7 free
x8 free
x9 free
x10 free
x11 free
x12 free
x13 free
x14 free
x15 free
x16 free
x17 free
x18 free
x19 free
x20 free
x21 free
x22 free
x23 free
x24 free
End

```



圖 A-4、塑膠射出成型機動作程式的可達圖生成的 CMSTI 方程組