引入時間中斷法之電子交易市場集中撮合機制

苑守慈
政治大學資訊管理學系
林孟彥
輔仁大學資訊管理學系

摘 要

網路市場的即時性與方便性，將逐漸取代顧客至傳統市場購物的方式，進而提供傳統市場無法提供的服務，因此本論文發展出一套新的未來購物模式，此套購物模式，可以藉由其它顧客的幫助，共同享受優惠方案的福利；或者廠商之間亦可以透過合作，提供顧客更完備的服務，本研究並考慮過去網路市場未考慮到的異業結盟問題；除此之外，此購物模式可以更貼切地反映顧客對產品的喜好，使撮合結果更符合使用者需求。本研究方法乃是一個新的時間中斷法，在時間有限的情況之下，提出一個可以在時間與顧客的整體利益間做取捨的演算方法，讓市場系統可以在使用者要求的時間之內將顧客的整體利益最大化。此方法並且具有「在時間充裕的前提之下必能求得最佳解」的特性，可知當硬體發展至一定程度時，本研究方法將成為一個能在最快時間之內求得最佳解的撮合方法，使本研究具前瞻性。

關鍵字：時間中斷法、智慧型代理人軟體、結盟自動化、電子市集

An Anytime Approach for Coalition Formation in E-Markets

Soe-Tsyr Yuan
MIS Dept., National Chengchi Univ.
Angus Lin
IM Department, Fu-Jen Univ.

Abstract

In agent-based electronic markets, agents may have partially/ fully overlapped goals in terms of sophisticated demands or supplies. Therefore, agents form coalitions on demand at runtime in order to obtain larger gains from transactions. This paper presents a centralized anytime-based coalition formation method that allows a marketer to flexibly control the tradeoff between coalition formation complexity and coalition quality in terms of satisfactory payoffs obtained at demanders and suppliers.

Keyword: anytime algorithm, intelligent agents, coalition formation, e-markets

壹、緒論

傳統市場的購買方式將因網際網路的興起而被打破，網路市場的即時性與方便性，將逐漸取代顧客至傳統市場購物的方式[Yahalom, et.al., 1998]，進而提供傳統市場無法提供的服務，發展出一套新的未來購物模式。

傳統市場中，礙於空間的限制，顧客要對商品進行比價，往往得費上大半天的時間與精神、體力，貨僅能比「三」家；網路市場的特色之一即是沒有空間與地域的限制，即使在地球的彼端，也只要彈指之間便能獲得需要的資訊，貨要比 300 家都不成問題。

傳統市場中，若想要享受廠商提供的優惠方案，往往得付出代價，例如購買多餘的商品。舉例來說，「買超值全餐再加 69 元，就送 Hello Kitty...」，然而消費者是否真的願意花錢購買超值全餐？但透過網路，你可以與想吃超值全餐的人進行合作，他出 79 元買一份原價 99 元的超值全餐，你出 89 元買一對 Hello Kitty，節省購買超值全餐的額外開銷，兩方都獲利，何樂而不為？這在傳統市場中看似不可能，對網路市場而言卻是輕而易舉。

另外，傳統市場中，「你要買手錶、鞋子和外套...」這樣的需求通常代表你必須跑三個地方，網路市場卻不必要。透過異業合作，廠商可以一次提供你所需要的任何東西，商品何必一定要店裡有？有網路便可以解決一切問題。

換言之，傳統市場中，顧客購買商品 a, b, c, d，可以先就各商店進行比價，再選擇單價最便宜的商店購買，例如商品 a, b 在 A 商店購買，商品 c, d 在 B 商店購買，假設這樣的購買方式共花費顧客 900 元。

現在有一家商店 C 推出優惠專案，聲明在店裡一次購足商品 a, b, c, d 即可享有 800 元低價優惠。雖然商品 a, b, c, d 在商店 C 的單買價格較商店 A 與商店 B 不划算，然而因為這項優惠，具有理性的顧客應該轉而向商店 C 購買全部商品。

如果商店 C 優惠專案僅包含商品 a, b, c，則顧客應該考量商店 C 的優惠專案加上商店 B 的商品 d 單賣價格是否低於 900 元再做打算；如果另外有商店 D 推出商品 b, c, d 的優惠專案，則顧客應該在 (a, b, c) 與 (b, c, d) 優惠專案之間做取捨；假設現在市場內共有 100 家商店推出 500 種優惠專案，則顧客應該如何選擇？

透過網路技術與智慧型代理人軟體 (Intelligent Agents) [Nwana, 1996]，網路市場中的類似行為在時間允許的情況之下，已經可以自動為顧客求算最佳的購買方式。但網路上登錄的廠商林林總總，提出的優惠方案可能上千或上萬，如何在時間與購買價格之間做取捨成為必須考慮的問題。Eric Steinmetz [Steinmetz, et.al., 1998] 在 1998 年提出一個時間中斷法(Anytime algorithm)，用以解決這類的問題，顧客可以在任何時刻要求系統停止運算，提供到目前為止系統求得的最佳購買方式，亦即在已知的條件之下，顧客欲購買的 n 個商品各應該到哪些商店購買，才能使顧客花費最少的金錢。

然而，Eric Steinmetz 的 algorithm 雖然可以解決上述問題，卻仍有許多可以改進的地方，本研究目的即是改進 Eric Steinmetz 的演算法，使其更符合現實生活中廠商與使用

者的需求，並利用網路環境達成現實生活中無法達到的目的。本研究與 Erik Steinmetz 的最大不同點在於：

（一）考量顧客之間的合作：

如同之前所提到的，如果一個顧客必須獨自滿足優惠方案中所列出的所有條件，則顧客可能必須購買自己不需要的商品，然而藉由其它顧客的幫助，則可以各取所需，共同享受優惠方案提供的福利[Klusch & Shehory, 1993; Shehory etl., 1993； Ketchpel, 1995; Sandholm, 1995]。Eric Steinmetz 提出的演算法僅只針對單一顧客考量多優惠方案的購買方式，一旦顧客購買的商品種類或數量不夠多，許多優惠方案即沒辦法利用，故本研究提出一個新的 Anytime algorithm，除了能利用網路環境達成顧客之間的合作之外，並能就每個顧客的立場作考量，使顧客不會為了達到優惠方案所提出的條件，買到自己不滿意的東西。

（二）考量廠商之間的合作：

除了顧客之外，廠商之間亦可以透過合作，提供顧客更完備的服務。Erik Steinmetz 的優惠方案僅只考慮單一廠商提出的優惠方案，卻忽略廠商與廠商之間同業或異業的合作行為。本研究改進 Erik Steinmetz 的演算法，使現實生活中不同廠商之間合作的情形亦能在網路市場中運作。

（三）更貼切表達顧客對商品的喜好：

Erik Steinmetz 中對商品的評價，僅著眼於價格的比較，現實生活中卻非如此。價格只是商品價值的一部份，還有廠牌、售後服務、經銷商、規格等屬性可以左右顧客的喜好。現實生活中顧客對商品進行評價，很少只考慮價格的因素，Bargain Finder (http://bf.cstar.ac.com/bf) 是一個線上比價的購物助理軟體系統，只要顧客對某一個商品提出需求，Bargain Finder 就會向其他網站取得商品價格。但許多網站卻關閉了 Bargain Finder 的詢價動作，原因即是他們不願消費者僅注意他們的價格，而希望能多注意他們商品的其它價值[Guttman, et.al., 1998]。本研究結合陳世峰[陳世峰，2000]提出的效用近似函數，改進 Erik Steinmetz 只對價格進行比較的方式，使顧客對商品的喜好能更貼切地表達出來。

本研究方法主要分為兩階段，第一階段為使用者形成 Group 的部份，一旦使用者進入市場系統並向系統提出買賣要求，系統便會依照使用者的時間需求與欲買賣商品的種類進行分群，分群完成後市場系統開始進行對各 Group 的後續處理動作，此為研究方法的第二階段，包含形成 Origin Node 與 Improving 兩步驟，市場系統會在兩種情況之下結束，一為撮合時間一旦到達該 Group 要求的交易截止時間，則停止撮合，並以目前找出的最佳配對方式為該 Group 的撮合結果；否則的話，系統會一直重覆執行 Improving 的動作，直到所有可以用來改良該 Group 配對方式的條件都用盡為止。各 Group 的撮合是獨立的，不受其它 Group 撮合的影響。

本文主要分為三部份。第一部份說明本研究方法之相關知識，如 Anytime Algorithm 概念、智慧型代理人技術與自動撮合機制等相關議。第二部份則呈現本研究之研究方法

與架構。第三部份列出了本研究之實驗設計，及藉由實驗數據驗証本研究之價值。最後為本文結論，並提出後續研究的建議。

貳、文獻探討

以下我們針對本文中牽涉的相關知識，包含 Anytime Algorithm、Intelligent Agent、與自動撮合機制等議題對過去有過的相關研究進行探討，學習過去的知識與經驗並找出過去研究不足之處。

一、時間中斷法(Anytime Algorithm)

在很多時候，由於時間的考量，工作沒有辦法做到盡善盡美，只能在有限的時間之內儘可能做到最好的程度，這就是 Anytime Algorithm 的基本概念。你可以在 Anytime Algorithm 執行的任何時候要求其提供一個解答，解答的品質必須隨時間的增加而提高。我們可以試著將時間想像成輸入，對於每一個輸入可以對應到一個解答的輸出，輸出的品質隨著輸入時間的增加而提高。

Anytime Algorithm 於 1980 年代由 T. L. Dean 與 Boddy 首先提出 [Dean, 1987; Dean, et.al, 1988]，利用 Performance Profiles (PPs) 為工具，協助決策者制定各種不同策略以因應不同的時間需求；E. J. Horvitz 亦於 1990 年與 1987 年提出與 T. L. Dean 和 Boddy 相類似的方法 Flexible Computation [Horvitz, 1987, 1990]，用來處理有限時間下的決策問題；這些研究的出發點皆源自於一個想法，亦即為了求得正確解或最佳解所花費的龐大時間，可能反而導致系統的整體效益下降；除此之外，許多問題並不一定需要求得最佳解，其可接受的解答可能只是在一個程度範圍之內，因此建立一個可以在解答品質與時間兩者之間做取捨的演算法有其必要性。

這幾年 Anytime Algorithm 的研究快速成長，主要歸功於 Anytime Algorithm 在許多領域應用的成功，例如關聯式資料庫的查詢 [Vrbsky, et.al., 1990]、model-based diagnosis [Pos, 1993]、constraint-satisfaction 問題 [Wallace & Freuder, 1995] 與 sensor interpretation and path planning [Zilberstein, et.al., 1993； Zilberstein, 1996] 等等。一個 Anytime Algorithm 的主要特性包含以下七種 [Zilberstein, 1996]：

1. Measurable Quality：結果的 quality 必須要能明確地被估計。
2. Recognizable Quality：在 Anytime Algorithm 執行的過程中，任何時點的 quality 都要能被輕易地計算出來。
3. Monotonicity：產生結果的 quality 必須要是時間的非遞減函數。事實上，只要任何時點的 quality 都能被輕易計算出來 (特性 2)，系統便可以藉由只傳回比上次更好的結果輕易達到 monotonicity 的目的。
4. Consistency：產生結果的品質應該要與計算時間的多寡密切相關。雖然一般來說，Anytime Algorithm 沒辦法保証使用多少時間做計算、就能得到多少品質的結果，但結果的品質還是應該與計算時間密切相關，使能達到品質預測的目的。

5. Diminishing Returns：結果改進的程度應該隨時間遞減。Algorithm 剛開始時的單位時間結果改進程度應該最大，其後隨著時間慢慢減小。
6. Interruptibility：意即 Anytime Algorithm 可以在任何時間停止，並給予使用者一個結果。
7. Preemptability：意即 Anytime Algorithm 執行的過程中可以被暫停，再於需要的時候重新執行。

二、智慧型代理人軟體

自 1990 年代智慧型代理人 (Intelligent Agents) 的概念被提出之後，越來越多的軟體開始引入智慧型自動判斷機制來協助或主動完成使用者的各種工作，例如糾正使用者在工作上的錯誤行為、或為其搜集整理過濾網路上的各式資訊等，各類型相關應用並已逐一提出，儼然已是軟體產業上一門不可或缺的新興技術，適時的引用智慧型代理人軟體，將可以節省許多過去必須由人來控制操作所浪費的人力與寶貴時間，並能完成許多人類所無法勝任的繁複工作，為各類型軟體應用提供更大的想像空間。例如應用智慧型代理人軟體進行買賣雙方之自動議約(Negotiation)進而達到自動化撮合機制。

由於議約的複雜度與花費的時間不一，現實中常被消費者列為是否繼續進行議約的重要考量因素，故本研究建構的系統將買賣雙方的議約過程集中處理，藉以控制議約產生的時間成本與複雜程度，並能從中提高消費者整體的平均獲利，此即為自動化撮合。然而無論是要做到系統自動化撮合、或是由 Agent 代理消費者向賣方進行議約，系統都必須要能充份表達消費者對各種具有不同屬性 (Attribute) 的商品的喜好程度，目前在這方面已經有一些研究探討自動化的協商機制，最早期的是 Kasbah (http://kasbah.media.mit.edu) [Anthony, 1996]，可以依據消費者提出的可接受最高價與時間需求等資訊，為消費者選擇最適合的商家；另外還有美國密西根大學發展出來的拍賣伺服器 AuctionBot (http://Negotiate.eecs.umich.edu)、與線上比價購物代理人軟體 BargainFinder (http://bf.cstar.ac.com/bf)，只要消費者選擇一項特定商品，則 BargainFinder 會主動向其它網站搜尋此一產品的價格進行比價，然而許多網站卻拒絕 BargainFinder 的詢價動作，因為他們希望消費者除了在比較價格之外，也能注意他們賦與商品的其它價值，例如提供快捷的維修服務，或比別人更長的保固期限等等。這反應了現實生活中的消費者行為模式，現實世界消費者評估一項產品，通常不只有對價格進行比較，而會考慮它的多重屬性。

為了反應消費者的這項行為，Martin 等人在 1999 年提出一套多重屬性拍賣 (Multi-attribute Auction) 機制 [Martin, et al, 1999]，以效用函數 (Utility Function) 決定提案 (Proposal) 的價值，藉以達到多重屬性評估的目的，然而在此機制要消費者賦與各屬性權重的時候，由於消費者本身不容易對自己的喜好做量化的評估，以致沒有辦法確實反應出使用者的需求；此外，此機制採用的效用函數僅考慮各屬性間的線性關係，卻忽略可能存在的非線性關係，有實際應用上的問題存在 [陳世峰, 2000]。

為了解決以上問題，陳世峰利用類神經網路技術 (Neural Network) 評估商品的多重屬性價值、亦即量化消費者對產品的喜好程度。由於類神經網路技術中各節點的權重是

由樣本資料學習而來，免除了由消費者直接指定權重所產生的問題；類神經網路可處理輸入節點間任何線性、非線性關係的特性，亦能充份表達各屬性間的交互影響，故本研究將引用陳世峰提出的 Evaluator 機制，評估消費者對各 supplier 販賣的商品的喜好程度。

三、本研究方法之概念

本文在處理賣方（supplier）與買方（demander）的撮合上採用乃是站在 巨觀（meta-Level）的角度為使用者進行撮合，可以輕易知道哪些賣方與哪些買方湊合在一起交易能使整體利益提昇至最高，比起傳統的議約撮合方式，巨觀的角度可以在更短的時間之內得到相同的利益，並且在時間充裕的情況之下能保証為使用者求得最佳撮合結果。然而由於建構於網路上的市場太過龐大，要在短時間之內求得最佳的撮合結果並不容易，此時便需要借用 Anytime Algorithm 的觀念讓使用者可以在時間與撮合結果的品質兩者間做取捨。利用 Eric Steinmet 的方法可以為單一買方提供 Anytime 的購物結果，然而對於買方之間與賣方之間的合作，以及表達買方喜好的方式仍有相當大的改善空間。此外，為了能達到系統自動撮合的目的，在評估消費者對產品的喜好程度方面，我們引入陳世峰的 Evaluator 機制求算商品的多重屬性價值，藉以充份表達消費者對各項商品的喜好程度。

參、研究方法

本研究方法主要由兩個階段構成，第一階段自市場中選擇賣方與買方形成 Group，目的在結合具有相同時間需求的使用者。(現實生活中，賣方與買方會希望交易在一定時間內完成，然而同一時間內，市場中存在許多具有不同需求的賣方與買方，等待 marketer 為其撮合，並且每個賣方與買方要求停止撮合的時間各異，因此第一階段目的在於決定哪幾個賣方與買方適合放在一起進行供需撮合。)；第二階段針對具有相同時間需求的使用者所形成的 Group 進行供給與需求之撮合。撮合過程包含兩階段，第一階段完成第一次供需撮合；第二階段針對第一次的供需撮合結果以 Anytime 精神進行改善，目的在有限的時間之內將所得的 coalitions 中買方之整體效益最大化。在說明本研究方法前，我們先介紹相關名詞其定義與假設。

一、相關名詞其定義與假設

以下定義本文中提到的名詞及其假設：

（一）市場(Market)

市場為建構在 Internet 上的虛擬市場，類型本研究定位在以電腦零組件與週邊設備為商品的二手市場。市場上的使用者分為買方與賣方兩種，各自提出需求後由系統於使用者要求的有限時間之內為其進行撮合。為方便系統進行，市場中無論買方或賣方只要系統為其形成 Group 之後，即不得擅自結束交易。由於系統設計為根據使用者提出的要求進行撮合，撮合結果必定滿足使用者對商品的最低要求，因此對於撮合完成的結果，

使用者不得拒絕且必須接受。

（二）商品屬性(attribute)

過去的研究對於商品與顧客之間的撮合，往往只著重於價格上的比較，卻忽略價格只是商品的眾多屬性之一而已。除了價格屬性之外，商品的可能屬性還包括廠牌、經銷商、保固年限等等，商品藉由這些屬性組合與其它商品有所區隔，廠商則根據銷售策略改變商品屬性吸引顧客消費，例如延長保固年限，或降低售價等等。

（三）買方(demander)

買方為市場使用者之一，支付金錢予賣方以交換商品。買方對市場提出的要求分為兩種：對時間的要求與對商品的要求。對時間的要求為買方希望撮合完成的時間，對商品的需求則包括兩種，一種是對商品效用指數的要求 (utility)，一種是對商品數量的要求 (顧客希望購買的商品數量)。買方可以同時對一個以上同種或多種商品產生需求，每種不同屬性組合的商品需求稱為一個 demand。

（四）買方需求商品(demand)

買方需求商品為買方在市場上提出的需求，每一種具有不同屬性值組合的商品需求視為一個買方需求商品，例如買方 1 在市場中提出購買 IBM 硬碟 45G 與 30G 兩種規格各一個，則代表買方 1 在市場中產生 2 個需求(買方需求商品)。

（五）賣方(supplier)

賣方為市場使用者之一，提供商品給買方以換取金錢，並對市場提出需求，要求交易在一定的時間內完成；一個賣方可以提供一個以上同種 (屬性值組合) 或多種商品給買方；除了商品本身以外，賣方可以依據策略提出一至數個優惠方案(promotion) 予買方，藉以提高商品被採用的機會。

（六）賣方販賣商品(supply)

具有不同屬性值組合的商品視為一個賣方販賣商品，例如賣方 1 提供的 5000 元 Quantum 硬碟、賣方 1 提供的 5200 元 Seagate 硬碟、賣方 2 提供的 5200 元 Seagate 硬碟與賣方 2 提供的 9000 元 HP 印表機分別被視作賣方販賣商品 1, 賣方販賣商品 2, 賣方販賣商品 3 與賣方販賣商品 4。

在本研究方法中，實際撮合的並非買方與賣方本身，而是買方產生的買方需求商品與賣方產生的賣方販賣商品 。我們可以將每個買方需求商品與賣方販賣商品視為一個單位買方與單位賣方，每個單位賣方被限制僅能提供單一種屬性值組合的商品給單位買方，而單位買方也僅能對單一種屬性組合的商品提出需求。

（七）效用指數(Utility)

效用指數為買方對商品的主觀評價，由多重屬性構成，包含價格、品牌、規格、經銷商、售後服務等，視商品的種類而定；賣方決定各屬性值，由買方給予評價；買方對商品的主觀評價係透過類神經網路 (Artificial Neural Network) 學習得來，利用不同的屬性值組合 (商品) 做為輸入節點，並將買方對該屬性值組合的喜好程度做為輸出，據此

建立一個類神經網路(陳世峰，2000)，亦即買方對商品主觀評價 (效用指數) 的近似函數 (approximator)。

為了讓效用指數之間能做比較，商品 (屬性值組合) 透過此函數所產生的數值必須加以正規化，正規化後的數值我們便稱之為效用指數。

（八）優惠方案(promotion)

在現實生活的市場交易行為裡，優惠方案是各家廠商常見的宣傳促銷手法之一，普遍存在於各類型買賣交易，例如近幾年常見的手機搭配門號出售低價優惠方案，以及各大唱片行「紅標配綠標」的促銷手段。賣方依據本身行銷經營策略，將商品配套販賣或與其它賣方所提供的商品搭配出售，並以較個別單買更優惠的條件吸引顧客上門消費，此即優惠方案的基本定義。藉由優惠方案的提出，賣方可以更有效率達到出清存貨、刺激買氣等商業目的，買方也可從中受惠，是一種雙贏的行銷策略。假設賣方 S1 產生賣方販賣商品 s1 與 s2，賣方 S2 產生賣方販賣商品 s3，則優惠方案可概分為以下類型：

1. 單純刺激買氣的優惠：

   這是一般常見的促銷方式，例如以低價鼓勵消費者同時購買多個商品。以式子 Promotion = 5s1 而言即表示同時購買 5 單位的 s1 可以享有廠商所提供的優惠。

2. 商品搭配販售的優惠：

   這種促銷方式通常出現在功能相關性產品，例如手機搭配門號販售，或者買電腦送印表機等，以式子表示可以為 Promotion = s1 + s2，亦即同時購買賣方 S1 的 s1 與 s2 產品各 1 單位，可以享有 S1 所提供的商品優惠。由於在系統設計上優惠是由不同消費者共同合作享有，廠商在提出優惠方案的同時應該為各項產品制定優惠標準。

3. 異業結盟：

   以上的優惠方案可以由同一個廠商提出，亦可由多個廠商共同合作。以式子表示可以為 Promotion = s1 + s3，亦即消費者同時購買 S1 的產品 s1 與 S2 的產品 s3 各 1 單位，可以享有廠商 S1 與 S2 所提供的商品優惠。

傳統市場中，為了達到賣方提出的優惠方案，買方通常必須購買自己不需要的商品；網路市場中，買方可以輕易與具有不同需要的買方合作，共同享有賣方提供的優惠，例如一個對 s1 具有需求的買方與一個對 s3 具有需求的買方即可透過合作享有優惠方案所提供的優惠，不需要另外購買自己不需要的商品，像這樣的合作關係即是系統據以改良撮合結果的基本概念。

（九）撮合

撮合主要有兩種，一種是配對關係，即賣方與買方之間的撮合；另一種是合作關係，亦即買方與買方之間的撮合，或賣方與賣方之間的撮合。就賣方與買方之間的撮合而言，主要分為兩階段：第一階段將具有相同時間需求的賣方與買方形成一個 Group，第二階段針對形成的 Group 進行配對工作；每個買方需求商品對自己需求的商品具有一定要

求，意即該賣方販賣商品 必須至少達到一定程度的效用指數，才會被買方需求商品所接受，此亦系統對買方與賣方進行撮合的基本原則：賣方販賣商品 必須至少達到買方需求商品對商品的基本要求，系統才會將該賣方販賣商品 分配予買方需求商品。

為了達到賣方提出的優惠方案，買方與買方之間時常需要合作，此即買方之間的撮合 (合作)。系統會在求取買方整體效益最大化的前提之下，決定哪些買方應該合作滿足優惠方案 1，而哪些買方應該合作滿足優惠方案 2。除了買方之間的撮合之外，系統亦對賣方進行撮合，目的在利用不同的賣方販賣商品 形成的組合，提升消費者的整體效益。

（十）撮合完成時間

現實生活中，賣方與買方會希望交易在一定時間內完成，網路市場亦是如此。系統在形成第一次的撮合之後，會利用賣方提出的優惠方案對之前的撮合結果以 Anytime 精神進行一次又一次改良，藉以提升買方的整體效益。然而由於每次的改良必須透過龐大的計算完成，花費時間頗鉅，如何在時間與買方整體效益之間做取捨，是本研究所要解決的問題之一。本研究對 Eric Steinmetz 的演算法做改良，提出一個 Anytime Algorithm，用以解決這個問題。

從上面的討論可以知道本研究所指的「撮合完成時間」，並不是指撮合結果達到最佳狀態所花費的時間，而是指使用者願意花在撮合上面的時間，一旦使用者要求的時間結束，撮合也就結束，所以「撮合完成時間」亦可說成「撮合結束時間」。事實上前面提過本系統架構由兩階段構成，即 Group 的形成與撮合兩階段，然而就時間而言，形成 Group 的時間遠比進行撮合花費的時間來得要少，所以本論文所提到的「撮合完成時間」亦即「完成交易時間」，也就是使用者提出要求後一直到要求被滿足所花費的時間

（十一）Anytime Algorithm

所謂 Anytime Algorithm，即無論處於任何時間點，一旦使用者提出要求，即可給予使用者一個結果，此結果必須隨著使用時間的增加而改善。本文中的研究方法另外具備一項特性，即只要時間允許，系統必可求得最佳解 (即在目前的條件下，使買方的整體效益最大化的配對結果) [Hansen & Zilberstein, 1996; Larson & Sandholm, 1999; Ziberstein, 1996]。

二、Group 的形成

同一時間內，市場中存在許多具有不同需求的賣方與買方，等待 marketer 為其撮合。由於每個賣方與買方要求停止撮合的時間各異，究竟哪幾個賣方與買方適合放在一起進行撮合，必須透過一些方法選擇：

（一）選擇賣方與買方：

本研究所採取之選擇的方法是將同一時間內提出相同時間要求的使用者放在同一個 Group。系統提供數個「撮合完成時間」選項供使用者作選擇，並以各提供選項的 1/6 時間為週期 (cycle) 分別對整個市場進行掃描監控 (Monitor)。我們稱每一種週期 (或者每一種撮合完成時間) 形成的掃瞄監控為一個 Monitor，一旦 Monitor 查覺同一個週期裡有

適當的使用者向其提出交易需求，便將之併入同一個 Group。以撮合完成時間為 30 分鐘的選項為例，系統會為其形成一個以 30 / 6 = 5 分鐘為週期的 Monitor 對市場進行監控，在系統時間 0-5 分 (不包含 5 分) 之間對這個 Monitor 提出交易要求 (亦即要求 30 分鐘後完成交易) 的所有使用者會被形成一個 Group，並於系統時間 30 分 的時候停止系統對這個 Group 的撮合以完成交易；系統時間 5-10 分 (不包含 10 分) 之間對 Monitor 提出交易要求的使用者會形成另一個 Group，並於系統時間 35 分的時候停止系統對這個 Group 的撮合以完成交易，依此類推。

圖 1 是一個市場中使用者形成 Group 的例子，圖中使用者所附加的數字代表其進入市場的先後次序，例如 D2 比 D7 早進入市場。系統一共提供 30 分鐘、60 分鐘、90 分鐘與 120 分鐘 4 種不同的撮合完成時間給使用者做選擇。系統會依據這 4 個撮合完成時間選項形成 Monitor A, Monitor B, Monitor C 與 Monitor D，分別以 5 分鐘、10 分鐘、15 分鐘與 20 分鐘對整個市場進行掃瞄監控。以 Monitor A 為例，其在

系統時間 0-5 分監控到有買方 D1 與賣方 S2 向其提出進行交易的要求，故將這兩個使用者形成一個 Group，同理 S8 與 D7 亦形成一個 Group (在系統時間 40-45 分之間向 Monitor A 提出進行交易的要求)，S5 與 S7 雖然也在同一個週期內向 Monitor A 提出交易要求，但由於兩者都是賣方，無法交易，故不能形成 Group。按照以上的方式 4 個 Monitor 一共產生 Group1 (D1, S2), Group2 (S1, D2, S3, D5), Group3 (S4, S6, D6)與 Group4 (S8, D7)，未形成 Group 的則有買方 D3, D4 與賣方 S5, S7。

（二）未能形成 Group 的使用者之處理：

對於未能形成 Group 的使用者，我們在不影響其交易完成時間的前提之下，將其併入一個最近且使用撮合時間最長的 Group。

以圖 1 為例，D3 為一個未能形成 Group 的使用者，系統預定在系統時間 90 分的時候為買方 D3 完成交易。為了在系統時間 90 分之前為 D3 完成交易，我們沒有辦法將 D3 併入 Monitor C 的任何週期形成的 Group，必須從次於原撮合時間長度的 Monitor B 中尋找可能被併入的 Group。從圖 1 我們可以知道 Monitor B 最近的一個週期為 10-20 分，在這個週期內向 Monitor B 提出交易要求的為買方 D4，由於 D3 與 D4 都是買方，彼此並無法交易，故必須繼續等待 Monitor B 的下一個 Group 產生。由圖中知道接下來的 20-30 分、30-40 分兩週期並沒有任何使用者向 Monitor B 提出需求，此時我們無法再繼續等待 Monitor B 的下一個 Group 產生，因為 Monitor B 在 30-40 分提出需求的使用者其交易完成系統時間已經是 90 分，再下一個週期的交易完成時間就是 100 分，超過 D3 要求交易完成的時間，故我們必須轉而在 Monitor A 中尋找可以併入的 Group。

在確定 Monitor B 中沒有可以併入的 Group 的時候，系統時間已經到了 40 分。Monitor A 中與 40 分最接近的週期是 40-45 分，有賣方 S8 與買方 D7 向 Monitor A 提出需求，即先前提到的 Group4 (S8, D7)，故我們可以將 D3 併入 Group4 中，形成 Group4'(D3, S8, D7)。

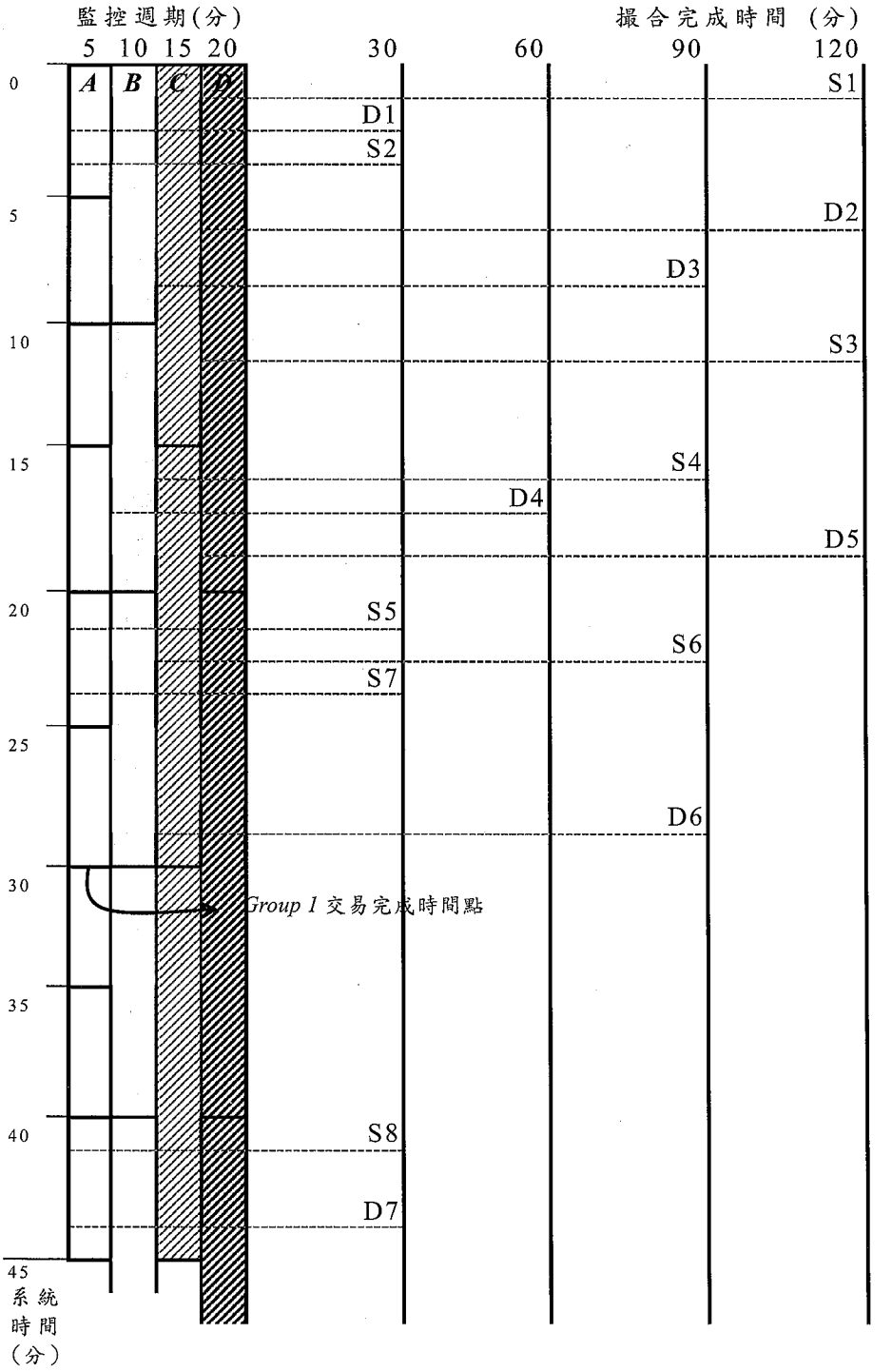

圖 1：group 的形成示意

值得注意的是，Monitor 中原本無法形成 Group 的使用者，也有可能因為別的使用者的併入而形成 Group。以 S5 與 S7 為例，因為 20-25 分只有他們向 Monitor A 提出需求，賣方與賣方之間無法交易，原本不能形成 Group，但由於之前在 Monitor B 中有一個 D4 不能形成 Group，按照先前的規則，D4 要在 Monitor A 中等待可以被併入 Group 的週期，因為最近的一個週期是 20-25 分，故 D4 可以和 S5 與 S7 形成一個 Group，即 Group5 (D4, S5, S7)。

三、撮合

假設 Group 裡面共有 5 位買方，其對商品的需求分別如表 1 所示。

表 1：Group 中買方對商品之需求

商品	demand (Utility ×Quantity)				
	D1	D2	D3	D4	D5
G1	0.9 ×1	0.6 ×1	Null	0.5 ×2	0.6 ×1
G2	Null	0.7 ×1	0.7 ×4	0.6 ×1	Null
G3	Null	0.6 ×1	Null	0.6 ×2	0.5 ×5

買方對商品的需求分為兩種，一種是對商品品質的要求，這裡所指的品質泛指買方對一切商品屬性 (attribute) 的主觀評價，包括價格、品牌、規格、售後服務等等，本研究統一以效用指數稱之；另一種是對該產品數量的要求 (欲購買的商品個數)。從表 1 我們可以知道：買方 D1 的需求為效用指數在 0.5 以上 (≧0.5) 的商品 G1 一個；買方 D4 的需求為效用指數在 0.5 以上的商品 G1 兩個、效用指數在 0.6 以上的商品 G2 一個、以及效用指數在 0.6 以上的商品 G3 兩個。

表 2：賣方販賣商品提供之商品數量與各買方對商品之主觀評價

商品	supply	數量	Utility				
			D1	D2	D3	D4	D5
G1	s1	3	0.6	0.7	Null	0.5	0.7
	s2	4	0.4	0.5	Null	0.6	0.5
	s3	1	0.5	0.6	Null	0.7	0.9
	s4	2	0.1	0.1	Null	0.1	0.1
G2	s5	5	Null	0.8	0.8	0.6	Null
	s6	1	Null	0.5	0.8	0.5	Null
	s7	3	Null	0.1	0.1	0.1	Null
G3	s8	2	Null	0.1	Null	0.2	0.3
	s9	4	Null	0.5	Null	0.8	0.6
	s10	1	Null	0.7	Null	0.4	0.5
	s11	2	Null	0.1	Null	0.1	0.1

假設 Group 中的賣方共產生 11 個賣方販賣商品 ，提供買方需要的商品 G1, G2 與 G3。每個賣方販賣商品 可能來自相同的賣方，也可能來自不同的賣方。每個賣方販賣商品 只允許提供單一種屬性值組合的商品，買方對每個賣方販賣商品 則各自有其主觀的評價 (效用指數)，例如同樣就商品 G1 而言，買方 D1 對賣方販賣商品 s1, s2, s3, s4

所提供商品的評價分別是 0.6, 0.4, 0.5 和 0.1。各賣方販賣商品提供的商品數量(Quantity)與各買方對賣方販賣商品提供商品的主觀評價（效用指數）詳列於上表 2。

本研究方法的第二階段，包含形成 Origin Node 與以 Anytime 精神進行 Improving 的兩個步驟。Origin Node 為初步供需撮合結果。Improving 程序則進行尋找更佳的供需撮合；尋找過程中產生 Origin Node 之子孫節點（descendant nodes），行成一 Improving 搜尋樹（例如圖 2 所示）。任一子孫節點代表一個更佳的供需撮合。而 Improving 搜尋樹深度之成長乃依時間之允許情況來決定。

（一）Origin Node 之選擇

確定 Group 中各買方與賣方販賣商品 之後，我們必須為這些使用者進行第一次供需撮合配對。第一次完成的配對組合稱為 Origin Node，並以之作為日後改進 (improve) 的依據。所謂改進是以站在買方的觀點，謀求買方整體效益之最大化。Origin Node 之選擇步驟如下：

1. 選出所有賣方販賣商品 滿足買方之配對：

這裡所說的滿足，意指買方對賣方販賣商品 所提供的商品之主觀評價 (效用指數) 大於或等於買方對這項商品所要求的效用指數。根據此原則所選擇出來的配對稱為 Satisfactory Pair (表 3 中以方框 ☐ 表示)。由於買方 D1, D2, D4, D5 對商品 G1 所要求的效用指數分別是 0.9, 0.6, 0.5, 0.6，對賣方販賣商品 s1 的主觀評價 (效用指數) 則是 0.6, 0.7, 0.5, 0.7，可知 s1 滿足 D2, D4, D5 對商品 G1 的要求，(s1, D2), (s1, D4), (s1, D5) 為 3 個 satisfactory Pair。

考慮買方對商品的需求量，D1, D2, D4, D5 對商品 G1 的需求量分別是 1 個, 1 個, 2 個, 1 個；賣方販賣商品 s1 只滿足了 D2, D4, D5 對效用指數的要求，故其所滿足的商品需求量為 D2: 1 個, D4: 2 個, D5: 1 個共 4 個，像這樣各賣方販賣商品 所能滿足的商品需求量稱為 **Satisfactory No.**，列示於表 3 最右欄。

2. 選出各買方評價最高之 Satisfactory Pair：

對買方而言，滿足需求的賣方販賣商品 可能不只一個，與買方構成 satisfactory Pair 之賣方販賣商品中評價最高的我們稱之為 **Ideal Pair** (表 3 以加底線的方框 ▭ 表示)，意謂假使賣方販賣商品 提供充足的商品予買方，則 Group 中的所有 Ideal Pair 便構成 Origin node。這裡所說的「充足的商品」意指：無論買方要求的商品數量為何，賣方販賣商品皆能滿足需求。然而現實生活中，賣方提供的商品數量通常有限，必須做進一步的處理以求得 Origin node。值得注意的是，(s2, D4) 與 (s10, D5) 並不是在此步驟被標示為 Ideal Pair，下一步驟會做說明。

3. 依據賣方販賣商品 提供之商品數量分配適當的 Satisfactory Pair：

說明商品分配原則之前，為了對 Origin node 之選擇過程能有全盤認識，先就以下問題「誰比較喜歡 s1？」進行討論。

s1 能滿足 D2, D4 與 D5 對商品 G1 的要求，但對 s1 提供的商品 G1 而言，哪位買方獲得的滿足程度較高？

衡量買方滿足程度的指標我們以 ***PsiDj = UssiDj / UdsiDj*** 表示。其中 *UdsiDj* 為買方 Dj 對賣方販賣商品 si 所提供商品之效用指數的要求，而 *UssiDj* 代表買方 Dj 對賣方販賣商品 si 所提供商品之主觀評價 (效用指數)；*PsiDj* 與 *PsxDy* 只有當 i=x 的時候才能比較。以 s1 而言，*Ps1D2* = 0.7 / 0.6 = 1.17，*Ps1D4* = 0.5 / 0.5 = 1，*Ps1D5* = 0.7 / 0.6 = 1.17，所以 D4 對 s1 的滿足程度最低，D2 和 D5 的滿足程度一樣。

在決定賣方販賣商品 的商品應該分配予哪個 Satisfactory Pair 的時候（亦即分配給哪個要求被滿足的買方的時候），如果有衝突，我們將賣方販賣商品 的商品分配給滿足程度最高的買方；如果 2 個買方的滿足程度相同，則分配給先進入市場的買方，以鼓勵買方及早進入市場。

以下說明如何將賣方販賣商品 分配予適當的 Satisfactory Pair（參見表 3）：

1. 以商品種類為單位進行商品分配。
2. 逐列尋 Ideal Pair，按 PsiDj 值高低將賣方販賣商品 依次分配給該列 PsiDj 值最高、次高...之 Ideal Pair，直到滿足 Ideal Pair 需求或該賣方販賣商品 商品全數分配予買方為止。
3. Ideal Pair 的需求未被滿足，首先向之前分配過商品的 Ideal Pair 做查詢，若其買方對其它賣方販賣商品 亦給予相同評價（效用指數），則由其它賣方販賣商品 取代原先提供商品的賣方販賣商品 ，取代掉的賣方販賣商品 則轉為提供商品給該買方。以商品 G2 而言，賣方販賣商品 s5 將商品依次分配給 D2, D3 之後已經沒有商品可以分配給 D4，但由於買方 D3 對 s5 與 s6 之評價相同，故將 1 個原來由 s5 提供的商品轉由 s6 提供，則取代掉的 s5 便可滿足 D4 的要求。倘若之前分配過的買方沒有其他評價相同的賣方販賣商品 ，則將該買方評價次高的 satisfactory pair 標示 Ideal Pair，例如 (s3, D4) 與 (s10, D5)，依此類推。
4. 搜尋該商品中其它尚能被賣方販賣商品 滿足但未滿足的 Ideal Pair，按照第二步驟加以分配。

當 Group 中各種商品 (G1, G2, G3)皆完成分配，則得出來之 pair set 即為 Origin node。表 3 標示選擇 Origin node 的過程，表 4 為 Origin node。

表 3：選擇 Origin node 的過程與 Satisfactory No.

商品	supply	Quantity	Utility					Satisfactory No.
			D1	D2	D3	D4	D5	
G1	s1	3	0.6	[0.7]	Null	[0.5]	[0.7]	4
	s2	4	0.4	0.5	Null	[0.6]	0.5	2
	s3	1	0.5	[0.6]	Null	[0.7]	[0.9]	4
	s4	2	0.1	0.1	Null	0.1	0.1	0
G2	s5	5	Null	[0.8]	[0.8]	[0.6]	Null	6
	s6	1	Null	0.5	[0.8]	0.5	Null	4
	s7	3	Null	0.1	0.1	0.1	Null	0
G3	s8	2	Null	0.1	Null	0.2	0.3	0
	s9	4	Null	0.5	Null	[0.8]	[0.6]	7
	s10	1	Null	[0.7]	Null	0.4	[0.5]	6
	s11	2	Null	0.1	Null	0.1	0.1	0

: Satisfactory Pair [ ] :Ideal Pair
: The pair in the Origin node, in which demander is entirely satisfied.
: The pair in the Origin node, in which demander is partly satisfied.

表 4：Origin Node

Origin	Parent Node: None			
promotions	None			
商品	demander	supply	Utility	Improved?
G1	D2	s1	0.7	F
	D4	s2	0.6	F
	D4	s2	0.6	F
	D5	s3	0.9	F
G2	D2	s5	0.8	F
	D3	s5	0.8	F
	D3	s5	0.8	F
	D3	s5	0.8	F
	D3	s6	0.8	F
	D4	s5	0.6	F
G3	D2	s10	0.7	F
	D4	s9	0.8	F
	D4	s9	0.8	F
	D5	s9	0.6	F
	D5	s9	0.6	F
Total Utility			10.8	

（二）Anytime 方式之 Improving

Origin node 產生之後，開始進行 improving 的動作。在進行 improving 之前，我們先簡單闡述 improving 的意義。*所謂 Anytime improving 是指根據賣方所提出的商品配套優惠（優惠方案），在所要求之時間內，來對買方與賣方販賣商品 做較佳的配對組合之尋找，以求提昇買方的總體利益*。優惠方案 可以由 1 位賣方提出，亦可由多位賣方合作共同提出 (即各賣方提供自己的部份賣方販賣商品 組成 1 個優惠方案)。假設 Group 中賣方提出的優惠方案共有以下六個：

Prem1 = 4G1 (s2') + 5G2 (s5')
Prem2 = 3G1 (s1) + 2G3 (s8')
Prem3 = + 2G2 (s5'')
Prem4 = 2G1 (s4') + 3G2 (s7') + 2G3 (s11')
Prem5 = 2G1 (s2') + 2G3 (s8'')
Prem6 = 1G1 (s4) + 2G2 (s5')

其中 si' 代表原本由賣方販賣商品 si 提供的商品，因為使用配套優惠之後其部份屬性值改變 (例如售價降低)，造成效用指數改變，故以 si'表示。以 Prom1 為例，假使買方能夠同時購買 4 個 s2 提供的商品 G1，以及 5 個 s5 所提供的商品 G2，則提出這個優惠方案的賣方將改進 s2 與 s5 提供的商品的屬性 (例如降低售價) 以提高商品的效用指數，嘉惠買方，所以用 s2'與 s5'表示商品效用指數改變後的 s2 與 s5 以資區別；至於 Prom3 中的 s5''則代表 s5 在另一個配套優惠中改變後的屬性組合，既不同於原來 s5 所提供的，也與 Prom1 中的 s5'不一樣，故另外以 s5''表示。然而效用指數是很主觀的東西，同樣的

屬性值對不同的買方來說可能產生不同的效用指數，表 5 詳列屬性值改變之後買方對各賣方販賣商品 的新評價 (效用指數)，其中 Unit 欄位表示該賣方販賣商品 是以幾個商品為單位構成優惠方案，例如 Prom1 = 4G1 (s2') + 5G2 (s5')，則 s2'的 Unit 為 4，s5'的 Unit 為 5。值得注意的是，s1 與 s4 雖然有加入優惠方案中，但其屬性值並未改變，故不需讓買方重新評價。

表 5：改變屬性值後買方與賣方販賣商品的相關資料

商品	supply	Unit	Utility				Satisfactory No.
			D2	D3	D4	D5	
G1	s2 '	2 / 4	0.7	Null	0.7	0.9	4
	s4 '	2	0.7	Null	0.8	0.3	3
G2	s5'	5	0.9	0.9	0.8	Null	6
	s5''	2	0.8	0.9	0.8	Null	6
	s7 '	3	0.8	0.7	0.9	Null	6
G3	s8 '	2	0.3	Null	0.8	0.4	2
	s8''	2	0.6	Null	0.5	0.4	1
	s11 '	2	0.9	Null	0.8	0.2	3

<u>優惠方案的篩選</u>

在利用每個優惠方案提升買方的總體利益時，都必須搜尋整個 Group 中的使用者，耗費時間非常驚人，故先對優惠方案進行簡單的篩選，除去無效的優惠方案。由於購買的商品必須滿足一定的數量才能使用優惠方案，故 Satisfactory No. 必須大於或等於該賣方販賣商品 的 Unit，由於 Prom5 中的 s8''與 Prom6 中的 s4 與此原則相違背，故可以篩去 Prom5 與 Prom6，留下 Prom1, Prom2, Prom3 與 Prom4：

Prom1 = 4G1 (s2') + 5G2 (s5')
Prom2 = 3G1 (s1) + 2G3 (s8')
Prom3 = + 2G2 (s5'')
Prom4 = 2G1 (s4') + 3G2 (s7') + 2G3 (s11')

<u>Improving 的 Search Space:</u>

我們按照以下的順序進行對買方與賣方販賣商品 做較佳的配對組合之尋找(換言之，**依時間之允許情況來決定在以下 Improving 搜尋樹之成長深度**)：

1. 利用 Proem1~Prom4 對 Origin node 進行 Improving，產生 Node1~Node4，其改進的 node (Origin) 為其 parent node，並計算 Node1~Node4 改進後的總體效用指數。
2. 比較 Node1~Node4 的總體效用指數與其 parent node (Origin)之總體效用指數，若比 parent node 大則留下進行下一次 Improving，等於或小於 parent node 之總體效用指數 y 的 node 不再進行 Improving。
3. 利用 Prom1~Prom4 進行第 2 次 Improving，留下獲得改進的 node (改進後的總體效用指數大於其 parent node)進行再下次的 Improving，直到所有的 node 都不能再被改進為止。
4. 比較所有產生的 node，選取其總體效用指數最大的為本 Group 之最佳解。

必須注意的是，由於每次計算皆須搜尋整個 Group，耗費的時間很大，通常沒辦法計算至第4步驟則時間終了，*這時候目前所產生的node 中總體效用指數最大的為最佳解*。

使用優惠方案進行 Improving 的方法

1. 以商品種類為單位進行 improving。
2. 利用組成優惠方案的賣方販賣商品取代 parent node 中的賣方販賣商品，在取代的時候必須注意，取代後買方對該賣方販賣商品的評價(效用指數)仍然不能小於買方對該商品效用指數之要求，在這個前提之下，從取代之後能增加的效用指數最高的先取代，依此類推。若有兩個以上的 pair 取代之後增加的效用指數相同，則按照買方進入市場順序進行取代。以 Node 1 的商品 G1 為例，由於 D2, D4, D5 對 s2'的評價分別為 0.7, 0.7, 0.9，對其 parent node 的賣方販賣商品的評價為 0.7, 0.6, 0.9，效用指數分別增加 0.0, 0.1, 0.0，所以由 D4 的賣方販賣商品 s2 先行取代，其次是 D2 的賣方販賣商品 s1，最後是 D5 的賣方販賣商品 s3。
3. 取代過的 pair 標示為 T，表示已經過 Improving，不可更動。故在對 parent node 進行 improving 的時候，只能對未標示為 T (即標示為 F) 的 pair 進行更動。
4. 各種商品都進行過 improving 之後，即產生新的 node。

Improving 的例子

Level n 的定義為「利用 n 個優惠方案來改善 Origin node」，以本節的例子而言，Improving Search Space 共有 10 個 node，在時間充裕的前提下，系統撮合停止於 Level 3 的 Node 3-3-2，其流程圖如圖 2 所示。

從圖 2 的例子中可以看出，在時間充裕的前提之下，最佳解出現在第 2 次改善的 Node 4-3，其使用優惠方案的方式為 Prom4 → Prom3，總體效用指數為 11.9。值得注意的是，Node 3-4 同樣使用了 Prom3 與 Prom4 做 Improving，然而由於其使用順序不同，造成效用指數有 0.3 的差距。

在時間有限的情況之下，此流程可以在任何時間停止，比較目前產生的所有 node 之後，以效用指數最高的做為最後供需撮合結果。例如假設系統計算至 Node 3 時已達使用者要求的交易結束時間，則效用指數最高為 11.7，Node 1 即為撮合結果。以下是系統對 Origin node 進行 improving 的詳細過程與數據。

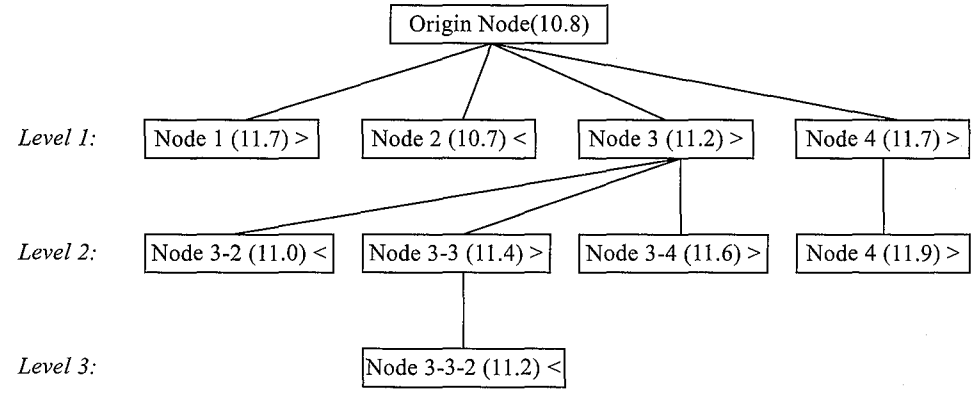

圖 2：Improving 流程圖

Level 1：

Node 1, Node 2, Node 3 與 Node 4 之中，由於 Node 2 的效用指數值 10.7 小於 Origin node 的效用指數值 10.8，繼續 improving 不具意義，故以 Node 2 為 Level 1 的 improving 到此為止。

表 6-1：Node 1

Node 1	Parent Node: Origin			
promotions	Prom1 = 4G1 (s2') + 5G2 (s5')			
商品	demander	supply	Utility	Improved?
G1	D2	s2'	0.7	T
	D4	s2'	0.7	T
	D4	s2'	0.7	T
	D5	s2'	0.9	T
G2	D2	s5'	0.9	T
	D3	s5'	0.9	T
	D3	s5'	0.9	T
	D3	s5'	0.9	T
	D3	s6	0.8	F
	D4	s5'	0.8	T
G3	D2	s10	0.7	F
	D4	s9	0.8	F
	D4	s9	0.8	F
	D5	s9	0.6	F
	D5	s9	0.6	F
Total Utility			11.7	

表 6-2：Node 2

Node 2	Parent Node: Origin			
promotions	Prom2 = 3G1 (s1) + 2G3 (s8')			
商品	demander	supply	Utility	Improved?
G1	D2	s1	0.7	T
	D4	s1	0.5	T
	D4	s1	0.5	T
	D5	s3	0.9	F
G2	D2	s5	0.8	F
	D3	s5	0.8	F
	D3	s5	0.8	F
	D3	s5	0.8	F
	D3	s6	0.8	F
	D4	s5	0.6	F
G3	D2	s10	0.7	F
	D4	s8'	0.8	T
	D4	s8'	0.8	T
	D5	s9	0.6	F
	D5	s9	0.6	F
Total Utility			10.7	

表 6-3：Node 3

Node 3	Parent Node: Origin			
promotions	Prom3 =+2G2 (s5")			
商品	demander	supply	Utility	Improved?
G1	D2	s1	0.7	F
	D4	s2	0.6	F
	D4	s2	0.6	F
	D5	s3	0.9	F
G2	D2	s5	0.8	F
	D3	s5"	0.9	T
	D3	s5	0.8	F
	D3	s5	0.8	F
	D3	s6	0.8	F
	D4	s5"	0.8	T
G3	D2	s10	0.7	F
	D4	s9	0.8	F
	D4	s9	0.8	F
	D5	s9	0.6	F
	D5	s9	0.6	F
Total Utility			11.2	

表 6-4：Node 4

Node 4	Parent Node: Origin			
promotions	Prom4 = 2G1 (s4') + 3G2 (s7') + 2G3 (s11')			
商品	demander	supply	Utility	Improved?
G1	D2	s1	0.7	F
	D4	s4'	0.8	T
	D4	s4'	0.8	T
	D5	s3	0.9	F
G2	D2	s7'	0.8	T
	D3	s7'	0.7	T
	D3	s5	0.8	F
	D3	s5	0.8	F
	D3	s6	0.8	F
	D4	s7'	0.9	T
G3	D2	s11'	0.9	T
	D4	s11'	0.8	T
	D4	s9	0.8	F
	D5	s9	0.6	F
	D5	s9	0.6	F
Total Utility			11.7	

Level 2：

表 6-5：Node 3-2

Node 3-2	**Parent Node: Node 3**			
promotions	**Prom3 = 2G2 (s5") Prom2 = 3G1 (s1) + 2G3 (s8')**			
商品	demander	supply	Utility	Improved?
G1	D2	s1	0.7	T
	D4	s1	0.5	T
	D4	s1	0.5	T
	D5	s3	0.9	F
G2	D2	s5	0.8	F
	D3	s5"	0.9	T
	D3	s5	0.8	F
	D3	s5	0.8	F
	D3	s6	0.8	F
	D4	s5"	0.8	T
G3	D2	s10	0.7	F
	D4	s8'	0.8	T
	D4	s8'	0.8	T
	D5	s9	0.6	F
	D5	s9	0.6	F
Total Utility			**11.0**	

表 6-6：Node 3-3

Node 3-3	**Parent Node: Node 3**			
promotions	**Prom3 = 2G2 (s5") Prom3 = 2G2 (s5")**			
商品	demander	supply	Utility	Improved?
G1	D2	s1	0.7	F
	D4	s2	0.6	F
	D4	s2	0.6	F
	D5	s3	0.9	F
G2	D2	s5	0.8	F
	D3	s5"	0.9	T
	D3	s5"	0.9	T
	D3	s5"	0.9	T
	D3	s6	0.8	F
	D4	s5"	0.8	T
G3	D2	s10	0.7	F
	D4	s9	0.8	F
	D4	s9	0.8	F
	D5	s9	0.6	F
	D5	s9	0.6	F
Total Utility			**11.4**	

表 6-7：Node 3-4

Node 3-4	Parent Node: Node 3			
promotions	Prom3 = 2G2 (s5") Prom4 = 2G1 (s4') + 3G2 (s7') + 2G3 (s11')			
商品	demander	supply	Utility	Improved?
G1	D2	s1	0.7	F
	D4	s4'	0.8	T
	D4	s4'	0.8	T
	D5	s3	0.9	F
G2	D2	s7'	0.8	T
	D3	s5"	0.9	T
	D3	s7'	0.7	T
	D3	s7'	0.7	T
	D3	s6	0.8	F
	D4	s5"	0.8	T
G3	D2	s11'	0.9	T
	D4	s11'	0.8	T
	D4	s9	0.8	F
	D5	s9	0.6	F
	D5	s9	0.6	F
Total Utility			**11.6**	

表 6-8：Node 4-3

Node 4-3	Parent Node: Node 4			
promotions	Prom4 = 2G1 (s4') + 3G2 (s7') + 2G3 (s11') Prom3 = 2G2 (s5")			
商品	demander	supply	Utility	Improved?
G1	D2	s1	0.7	F
	D4	s4'	0.8	T
	D4	s4'	0.8	T
	D5	s3	0.9	F
G2	D2	s7'	0.8	T
	D3	s7'	0.7	T
	D3	s5"	0.9	T
	D3	s5"	0.9	T
	D3	s6	0.8	F
	D4	s7'	0.9	T
G3	D2	s11'	0.9	T
	D4	s11'	0.8	T
	D4	s9	0.8	F
	D5	s9	0.6	F
	D5	s9	0.6	F
Total Utility			**11.9**	

Level 2 中僅產生 5 個 node，另外 7 個 node 因為要用來 improving 的優惠方案與之前使用過的優惠方案相抵觸（即 improving 後的結果會影響前面已經完成的 improving），所以無法 improving 來形成新的 node。

Level 3：

Level 3 僅產生一個新的 node，並且無法利用任何已知的優惠方案對 Node 3-3-2 進行 improving，所以系統運算至此即告結束，並在比較所有產生的 node 之後，得出 Node 4-3 為最佳解。

表 6-9：Node 3-3-2

.Node 3-3-2	Parent Node: Node 3-3			
promotions	Prom3 = 2G2 (s5") Prom3 = 2G2 (s5") Prom2 = 3G1 (s1) + 2G3 (s8')			
商品	demander	supply	Utility	Improved?
G1	D2	s1	0.7	T
	D4	s1	0.5	T
	D4	s1	0.5	T
	D5	s3	0.9	F
G2	D2	s5	0.8	F
	D3	s5"	0.9	T
	D3	s5"	0.9	T
	D3	s5"	0.9	T
	D3	s6	0.8	F
	D4	s5"	0.8	T
G3	D2	s10	0.7	F
	D4	s8'	0.8	T
	D4	s8'	0.8	T
	D5	s9	0.6	F
	D5	s9	0.6	F
Total Utility			**11.2**	

肆、實驗評估

網路為一開放性環境，為了減少其不確定性與方便實驗進行，必須對實驗環境制定限制。本系統實驗環境將市場上流動的商品種類定為三種，可以想像成現實生活中市場販賣的商品種類通常僅侷限於某些產品；其次，我們假設市場買方需求商品的數量與賣方販賣商品 的數量接近，這可以想像為二手市場的交易情形並且不會影響系統的實驗結果。因為系統自動根據買方的喜好為買方挑選合適的賣方販賣商品 進行配對，不合適的賣方販賣商品必遭淘汰，不因市場中使用者多寡有所影響。

由於形成 Group 之後，每個 Group 的撮合皆為獨立運作，不受其它 Group 的影響，故以下各節我們針對 Group 內部的狀態進行實驗，不考慮 Group 外部的因素。

我們的實驗平臺是 Pentium III 550 MHz CUP，256 MB RAM 的六臺 PC，如果以這樣的平臺執行系統，則實驗數據之時間單位可視為分鐘。

以下定義幾個實驗中提到的名詞：

1. **消費者整體平均獲利**：消費者整體平均獲利為本系統一項很重要的指標，定義為：(Group 中所有形成 Pair 的買方需求商品之獲利總和) / (形成該 Group 的買方需求

商品總數)，亦即將未形成 Pair 的買方需求商品獲利視為 0，求算 Group 的買方需求商品平均獲利。

2. **可行解與最佳解**：在 Improving 的過程中，所有產生的 Node 都是可行解，因為它們都可以滿足現存的 Pair 中買方需求商品與賣方販賣商品 的需求，差別只在於該可行解產生的消費者整體獲利大小而已。一旦使用者要求撮合的時間終止或者撮合完畢，則目前能使消費者整體獲利最高的可行解為最佳解，可知最佳解可能不只一個。
3. **可行解 (或最佳解) 的優惠方案組合**：亦即該可行解 (或最佳解) 是經由哪些優惠方案改進(Improving) 的，此優惠方案組合具有順序性，因為不同的 Improving 順序可能導致消費者的整體獲利不同，但也有可能不會影響。
4. **系統撮合時間**：系統撮合時間一般包含兩部份，即 Origin Node 形成所耗費的時間與進行 Improving 的時間，但由於 Origin Node 花費的時間與 Improving 花費的時間相較之下微乎其微 (稍後我們會以實驗証明)，故為了方便起見，本實驗所提到的系統撮合時間實際上忽略 Origin Node 所花費的時間，而以 Improving 所費的時間代替。
5. **系統撮合結束與系統撮合中斷**：系統停止撮合的情況有兩種，一種是沒有加入時間中斷機制，讓系統不斷嘗試以可用的優惠方案對 Group 進行 Improving，一直到沒有可用的優惠方案為止；另一種是加入了時間中斷機制，只要到了使用者要求中斷的時間就停止 Improving，無論系統結束與否，並以目前最佳的可行解為最佳解。為了分辨這兩種情形，我們把未加入時間中斷機制，而是由系統自行結束的情形稱為「系統撮合結束」，而加入時間中斷機制，一旦時間一到就停止撮合的情形稱為「系統撮合中斷」。

一、實驗系統變數產生的影響

實驗系統中的主要變數有兩個，分別是(1) Group 中的買方或賣方個數與(2) Group 中由賣方提出的優惠方案個數。這兩個變數是可以由系統控制的，也就是說，系統可以透過內部所預設的條件控制形成的 Group 中使用者個數，或限制賣方可以提出的優惠方案個數為多少，藉以決定系統所要產生的結果，因此我們針對這兩個變數做實驗。至於其它的變數，例如買方所要求的最低效用指數等，由於這是屬於使用者的主觀意識，非能經由系統控制的，所以不在我們實驗的範圍之內。

本研究引入 Anytime Algorithm 的概念，所以撮合時間是我們實驗中的重要指標之一，以下兩個實驗都將探討不同的變數對撮合時間造成的影響，並以此証明 Anytime Algorithm 的重要性；另外，消費者的整體平均獲利亦為本系統的重點之一，故我們也會對不同的變數造成整體平均獲利的影響做深入探討，了解變數與消費者整體平均獲利間的關係。

在變數實驗階段，對於撮合時間我們指的是系統進行 Improving 至無任何優惠方案可以再被用來改進目前這個 Group 為止所花費的時間，也就是「系統撮合結束」所花費的時間，用以探究不同變數與完全執行完畢所需要花費的時間之間的關係。

（一）Group 中的使用者個數

首先我們假設 Group 中的買方個數等於賣方個數，因而產生出來的買方需求商品個數與賣方販賣商品 個數會相近，此外，我們控制 Group 中被賣方提出的優惠方案總數為 3，以這樣的環境做為我們的實驗環境。

Group 中的使用者個數對撮合時間造成的影響

一個 Group 從形成到完成撮合的時間主要有兩個階段，一個是形成 Origin Node 所花費的時間，一個是 Improving 所花費的時間。從圖 3 中我們可以看到，形成 Origin Node 所花費的時間與 Improving 階段花費的時間相較之下微乎其微，由其當 Group 中的使用者數量增加之後，Improving 的時間呈現明顯的增加，形成 Origin Node 所花費的時間卻沒有太大的改變，仍然在幾單位的時間之內即可以完成，所以我們可以把 Improving 的時間直接視為 Group 的撮合時間(Coalition Time)，忽略 Origin Node 造成的影響。之後的實驗我們將直接將 Improving Time 視為 Coalition Time，不再多做說明。

在 Improving 的部份，從圖 3 中我們可以明顯看出，一旦使用者數增加到 40 人之後，撮合時間即呈現大幅度的增加，如果是以 Pentium III 550 MHz CPU、256 MB RAM 的 PC 為實驗平臺，則我們可以將時間單位視為分鐘，如此一旦使用者人數到達 50 人，撮合時間將接近兩天，更不用說組成人數在 50 人以上的 Group。在現實生活中，每個人願意花費在購物上的時間不一，有的人為了比價可以跑數十個店家，花費一個禮拜的時間來買到最便宜的東西，也有人認為時間就是金錢，只願意花極少的時間在購物上，只要東西不要差太多就好了，但無論如何，其願意花費的時間都有一個上限，不可能無止盡的比價下去。由實驗我們可以知道一旦使用者超過 50 個人，則撮合時間就要以日來計，這已經超過某些人能接受的購物時間上限，故的確有引入 Anytime Algorithm 的必要，以在使用者要求的時間範圍內為使用者達成撮合的目的。

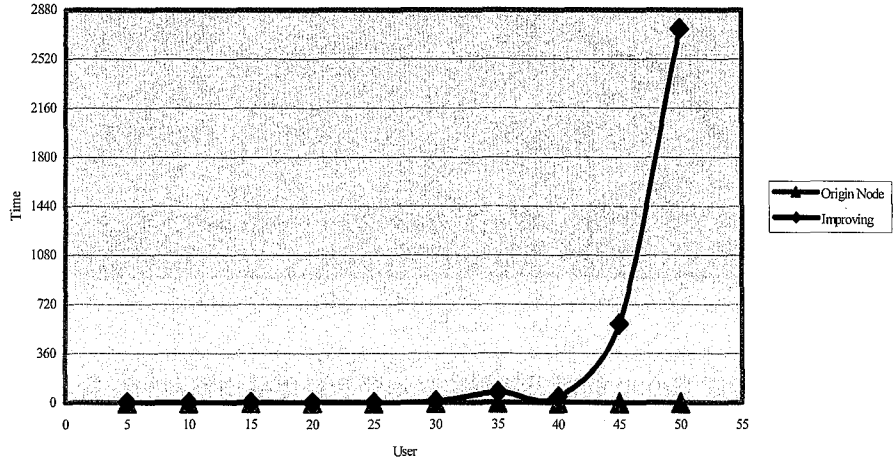

圖 3：形成 Group 的使用者個數對系統不同階段花費時間的影響

Group 中的使用者個數對消費者整體獲利造成的影響

從圖 4 中我們可以看得出，隨著 Group 中使用者人數的增加，其消費者整體平均獲利亦會隨之增加，並漸趨平緩，因此我們可以知道，以目前的實驗環境而言，系統應該盡量避免 Group 內的人數低於 25，而最好能控制在 35 人以上，因為在使用者數小於 25 之前，消費者整體獲利隨使用者人數增加而做大幅度提高，之後才漸趨平緩，若 Group 內的人數小於 25 則易導致消費者的整體獲利下降。

為了進一步探討消費者整體平均獲利隨 Group 中使用者數增加而提高的原因，我們將 Group 中賣方販賣商品 與買方需求商品被系統形成 Pair 的比率與使用者的總人數做一比較，從圖 5 中我們可以看出，無論是賣方販賣商品 或買方需求商品，形成 Group 的人數愈多，則賣方販賣商品 與買方需求商品形成 Pair 的比率也愈高，這解釋了為何隨著 Group 中使用者人數的增加，消費者整體平均獲利會愈高的原因。由於 Group 中使用者的人數愈多，則配對的可選擇性也會隨之提高，因而造成形成 Pair 的比率上升，這與實驗結果相符合。

心得

由於 Group 中的使用者數增加會產生兩種利多 1. 消費者整體獲利的提高；2. 消費者與賣方形成 Pair 的比率提高 (賣方形成 Pair 的比率提高可視為賣方整體獲利提高)，所以應該增加 Group 中的使用者數以增加買賣雙方的利益，但由於增加 Group 的使用者數會造成撮合時間的上升，提高時間成本，故必須引入 Anytime Algorithm，在買賣雙方都能接受的時間範圍內，求取最利於買賣雙方的配對組合。

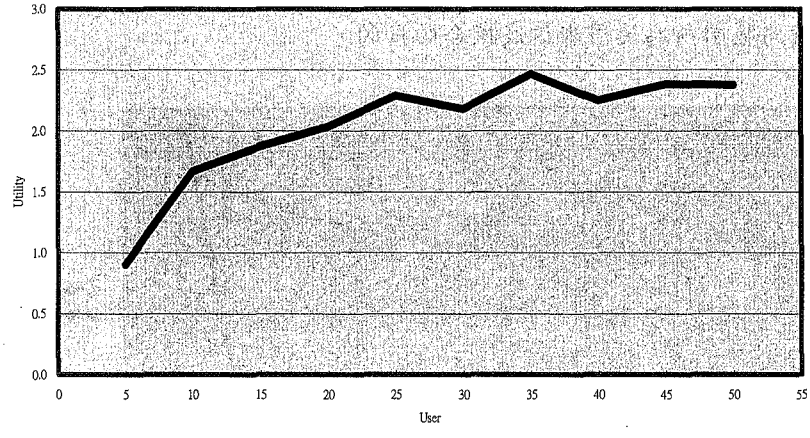

圖 4：形成 Group 的使用者個數對消費者整體平均獲利的影響

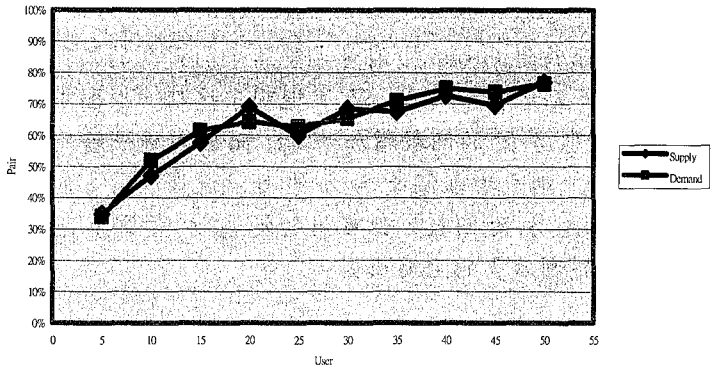

圖 5：形成 Group 的使用者 個數與形成 Pair 比例間的關係

（二）Group 中的優惠方案個數

系統中另一個重要變數是 Group 中的優惠方案個數，在系統設計時我們曾經提到賣方提出的優惠方案個數是影響系統撮合時間的重要因素之一，由於系統將每一個 Improving 過程中形成的 Node 都再以所有的優惠方案進行遞迴式的 Improving，勢必在事前加以篩選以減少增加的龐大運算時間，現在我們要用實驗証明這點。

Group 中的優惠方案個數對撮合時間造成的影響

在這個實驗裡我們將形成 Group 的使用者個數控制為 30 人。從圖 6 我們可以明顯看出，隨著優惠方案個數的增加，撮合時間呈現明顯的上升，這與 Group 中使用者個數造成撮合時間增加有些許不同。雖然大致上來說 Group 中的使用者個數增加會導致撮合時間的上升，但卻不是穩定增加，例如圖 3 使用者人數等於 40 的時候，其撮合時間反而下降；但在這個實驗裡，Group 中的撮合時間不但隨著優惠方案個數的增加絕對遞增，而且一旦 Group 中被提出的優惠方案 總數到達 5 以上，則撮合時間呈現大幅度的上升，若以我們的實驗環境而言，在 Promotion = 4 的時候撮合時間為 10 分鐘，Promotion = 5 的時候就必須花費 4 個小時，到了 Promotion = 6 的時候其撮合時間將近 5 天，就現實生活而言已經超過某些人可以接受的購物時間成本，故有引入 Anytime Algorithm 的必要。

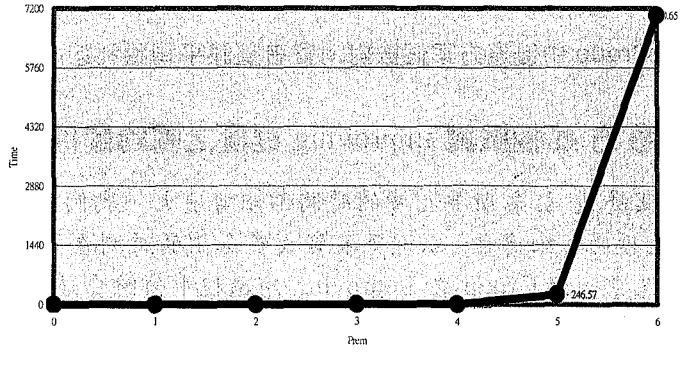

圖 6：Group 中賣方提出的優惠方案個數對撮合時間的影響

Group 中優惠方案個數對消費者整體平均獲利造成的影響

優惠方案是系統用來改進 Group 中消費者整體平均獲利的條件，所以預期效用指數應該隨優惠方案的增加呈現絕對遞增，圖 7 證明了這點。從圖中我們可以看出消費者整體平均獲利明顯隨優惠方案的個數增加遞增，而後漸趨平緩。漸趨平緩的主要原因是當優惠方案個數愈多，則平均每個優惠方案可以選擇的賣方販賣商品個數就相對減少，因此雖然增加了優惠方案個數，但可用的優惠方案卻不見得呈現等倍數的增加，這是消費者整體平均獲利減少的主要原因。

心得

由於消費者整體平均獲利會隨 Group 中的優惠方案個數增加遞增，所以應該增加 Group 中的優惠方案個數以增加消費者的整體獲利，但由於增加 Group 中的優惠方案個數會造成撮合時間劇增，故除了引入 Anytime Algorithm 的機制以期在消費者能接受的時間範圍內求取消費者的整體獲利最高之外，由於無論優惠方案對 Group 能否產生實際效用，一旦進入 Improving 階段，每個優惠方案所要花費的時間都相當可觀，所以系統更應該在進行 Improving 之前先對優惠方案進行篩選，剔除無效或重覆的優惠方案，如此可節省大量因為優惠方案數量增加產生的龐大系統撮合時間。

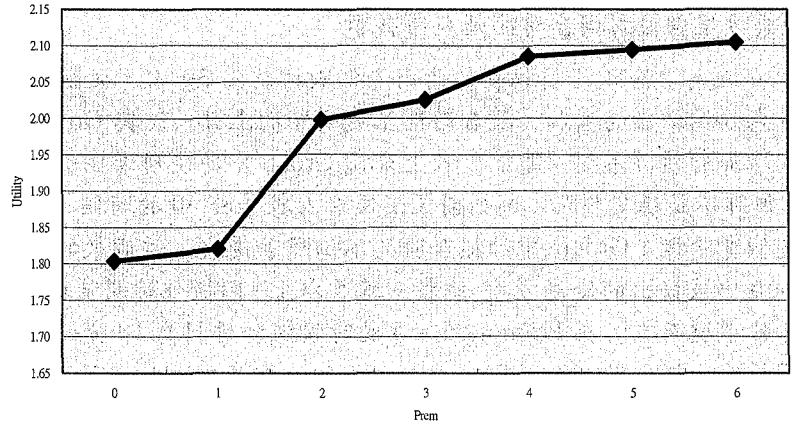

圖 7：Group 中優惠方案個數對消費者整體平均獲利造成的影響

二、Group 中優惠方案個數與 Improving 程度之間的關係

既然每增加一個優惠方案所要花費的時間都相當龐大，我們就必須了解 Group 透過優惠方案進行改進的詳細過程，以避免浪費太多無謂的時間在 Improving 上。所以以下的實驗我們首先探究一個 Group 在 Improving 的過程中何時會達到最佳解，如此可以節省達到最佳解之後再繼續搜尋的時間；其次，我們記錄一個 Group 中 Improving 的詳細過程，藉以了解消費者應該如何選擇時間中斷點會最恰當。

優惠方案個數與系統求得最佳解的時間

這個實驗中我們設定形成 Group 的使用者總數為 30 人。圖 8 中，縱軸部份是系統達到最佳解的時間佔整個撮合完成時間的百分比，橫軸為 Group 中優惠方案的個數。從圖 8 中我們可以看到，隨著優惠方案 個數的增加，達到最佳解的時間相對而言會愈短，也就是說，當 Group 中的優惠方案個數愈多，則達到最佳解的時間佔系統完成撮合的時間比例會愈低，並且隨著優惠方案個數的增加會漸趨平緩。由於可行解的優惠方案組合雖然有其順序性，但通常不會互相影響，亦即 Node 1-2-3 與 Node 3-2-1 通常消費者整體平均獲利相等，所以以這個例子而言，在 Node 1-2-3 的時候系統就已經找到最佳解，故優惠方案 的個數愈多，相對地系統會愈早找到最佳解。這樣的實驗結果對我們來說是好的，雖然優惠方案 的個數愈多需要的撮合時間會劇增，但相對的找到最佳解的時間佔全部撮合時間的比例也會減少，如此我們不需要等待系統撮合結束即可找到最佳的解決方案。

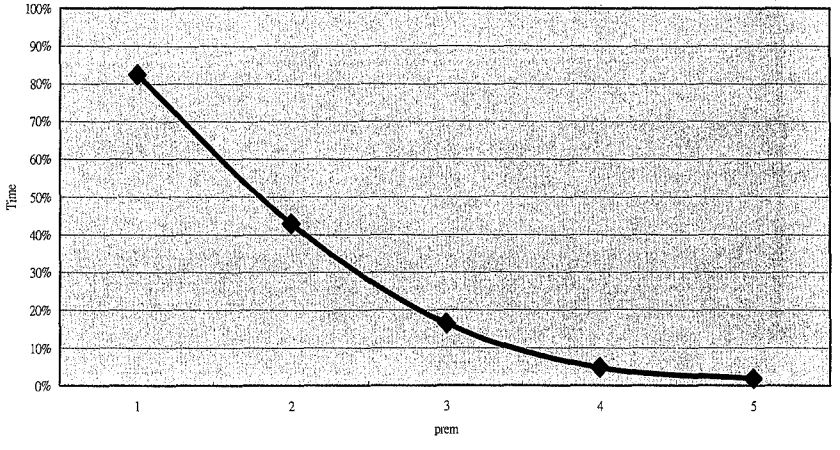

圖 8：Group 中優惠方案個數與系統求得最佳解的時間

Improving 的過程

既然 Improving 不需要等到撮合結束即可求得最佳解，那究竟何時中斷撮合才是恰當的時機呢？為了探究這個問題，我們假設 Group 中的使用者總數為 30 人，並以 4 個優惠方案做為這次實驗的環境變數，圖 9 顯示了時間與改進程度之間的關係。從圖 9 中我們可以看到在 Improving 進行之初，其改進的程度最為明顯，幾乎呈直線上升。(在最初的 10 秒內系統就已經為 Group 的效用指數提升了一成多，而在第 11 分鐘與第 12 分鐘的時候，又提升了約 0.6%的效用指數，此後一小段持平。約過了一個小時左右，又有另一波較明顯的 Improving 發生，從第 50 分鐘一直到第 72 分鐘，消費者的整體平均獲利一共提升了 1%左右，到達 99.7642%，此時圖 9 所顯示的 Improving 趨勢已逐漸平緩，當系統時間到了全程的 1.0%，也就是大約經過 6 個小時的撮合之後，已經相當接近最佳解的狀態，改進的程度達到 99.9476%。) 所以在同樣的實驗環境之下，我們會建議

使用者撮合時間維持在一個小時左右會最恰當，如此既可以達到接近最佳解的狀態，又能節省龐大的撮合時間。

心得

在「Group 中優惠方案個數與系統求得最佳解的時間」這個實驗中我們發現，在其它條件都固定的情況之下，優惠方案 的個數愈多，則其到達最佳解的時間佔全部撮合完成時間的比例愈低，由於我們在上一節的實驗中發現每增加一個優惠方案都會造成系統撮合時間的劇增，故可以利用本節實驗發現的這項特性，引入 Anytime Algorithm 在適當的時機將撮合中斷，以節省浪費在無意義 Improving 上的大量時間。為了找出最適當的中斷點，我們把 Improving 的情形做了詳細記錄，發現在本研究的實驗環境之下，其明顯 Improving 的部份都在系統撮合時間的前 0.5%，所以當本系統於現實情況下運作時，可以將過去的經驗告知使用者，使其能在花費最少的時間之下得到整體最高的利益。

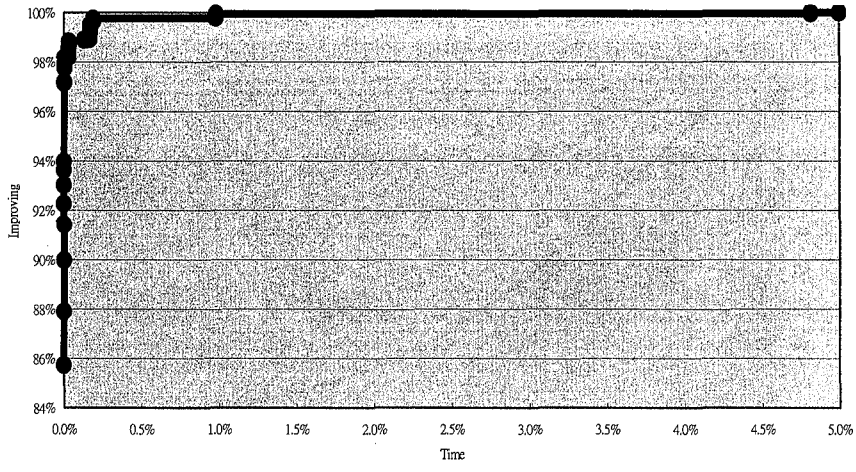

圖 9：Improving 的過程

三、時間充裕下系統 Improving 的結果

在前面的章節裡我們曾經提到：「只要時間充裕 (即允許系統執行至撮合結束)，必能求出使消費者整體獲利最高之最佳解」，現在我們來驗証這件事。

我們的實驗環境設定為：Group 中的使用者個數為 30 人，其中賣方總共提出 3 個優惠方案供系統進行 Improving。

為了能証明系統在時間充裕的前提之下能使消費者的整體獲利最高，我們為本實驗設立了一個實驗組與一個對照組，其中我們使實驗組的賣方販賣商品 分配上並不完全依照系統設計的規則進行（如變化 Ideal Pairs 之行成）；而對照組則完全按照本系統的規則執行（即對照組的撮合機制即是未對本研究提出的機制做任何改變），藉以比較兩者之間的差異。我們的實驗方法如下：

1. 實驗組：選取一組實驗數據，然後在 Improving 進行之前從 Origin Node 中任意選

出一個買方需求商品 dx，並強制將其最喜愛的賣方販賣商品 sx 分配給它 (亦即 UTILITY (dx, sx)[註1] ≧ UTILITY (dx, sn))。如果 Origin Node 中分配給 dx 的已經是它最喜愛的賣方販賣商品 ，則另外找一個 sx'分配給它，此 sx'必須要能滿足 UTILITY (dx, sx') = UTILITY (dx, sx)，如果這一組實驗數據中找不到一個這樣的賣方販賣商品 ，則捨棄這組實驗數據另外嘗試一組新的數據，直到該組數據能滿足為止。

2. **對照組**：以和實驗組相同的數據「正常」進行。所謂正常即依照原先的系統設計去運作，不做任何改變。

因為我們要証明在時間充裕的情況之下，系統必能找出最佳解，所以在實驗中我們不加入任何時間中斷的機制，讓系統能擁有「充裕」的時間進行撮合。以這樣的方法取得十組數據之後，我們將結果製成表 7，並且為了方便比較，將實驗組與對照組產生的消費者整體獲利繪成圖 10。

雖然在實驗組的 10 組資料當中買方需求商品 dx 自身所得到的利益都會是最高，但從圖 10 中我們可以發現，一旦我們提高 dx 的自身獲利，則這個 Group 中消費者的整體獲利將會下降，亦即 dx 自私地提高本身獲利將會犧牲掉其它買方需求商品的利益。為了突顯這樣的現象，我們把兩者之間的差繪製成圖 11，其中縱軸為對照組撮合結果的總體效用指數(total utility)減掉實驗組撮合結果的總體效用指數(亦即未置換時得到的總體效用指數減掉將 dx 的賣方販賣商品 置換成 sx 之後所得到的總體效用指數)，橫軸為各組實驗數據的編號。

將圖 11 對照表 7 我們可以發現，10 組實驗數據撮合出來的結果，實驗組所獲得的總體效用指數皆沒有大於對照組所獲得的總體效用指數 (因為圖 11 沒有產生負值)，並且除了第 9 組數據的實驗組與對照組獲利相等之外，其餘 9 組的實驗組獲利都明顯小於對照組的整體獲利，可以知道按照對照組的撮合機制我們可以使消費者的整體利益最大化。由於對照組的撮合機制即是未對本研究提出的系統做任何改變，所以從實驗中我們驗証了本系統在時間充裕的情況之下，能求得使消費者整體獲利最大的最佳解。

就消費者的角度而言，當然會希望自己的獲利愈高愈好；然而就一個市場永續經營的角度來看，根據我們的實驗結果，一個消費者的獲利提高將會導致其它消費者的獲利降低，並且犧牲的效用指數總和大於等於該消費者增加的效用指數，如此則提高一個人的獲利可能因為犧牲更多人的利益而導致部份消費者離開市場，如此則市場中的消費人數將逐漸減少。而消費者減少，賣方所能賣出的東西也會受到影響，進而影響到賣方進入這個市場的意願，難保這個市場的永續經營。所以我們在所有買方需求商品都滿意的前提之下，再對每個買方需求商品進行 Improving 的動作，並透過中央撮合的方式求取整體利益最大化，以求得市場的永續經營。而這個實驗証明了本系統的撮合方式在時間充裕的前提之下，的確能使消費者的整體獲利最大化。

[註1] ***dx*** 對 ***sx*** 的喜好程度

表 7：求証「時間充裕之下系統得到的結果為最好的結果」之實驗數據

編號	實驗組		對照組		(實驗組 - 對照組)	
	Total Utility	*dx* Utility	Total Utility	*dx* Utility	Total Utility	*dx* Utility
1	378.73957	1.75	379.41441	1.25	-0.67484	0.50
2	379.90589	2.67	380.77626	1.67	-0.87037	1.00
3	372.00479	1.40	372.34368	1.00	-0.33889	0.40
4	436.40238	1.40	436.80238	1.40	-0.40000	0.00
5	326.20952	1.29	330.27769	1.29	-4.06817	0.00
6	274.14602	4.00	274.29184	2.50	-0.14582	1.50
7	293.06786	4.50	293.35357	4.50	-0.28571	0.00
8	294.77143	2.67	301.83117	1.67	-7.05974	1.00
9	285.69762	3.00	285.69762	3.00	0.00000	0.00
10	281.47143	2.25	287.42602	1.25	-5.95459	1.00

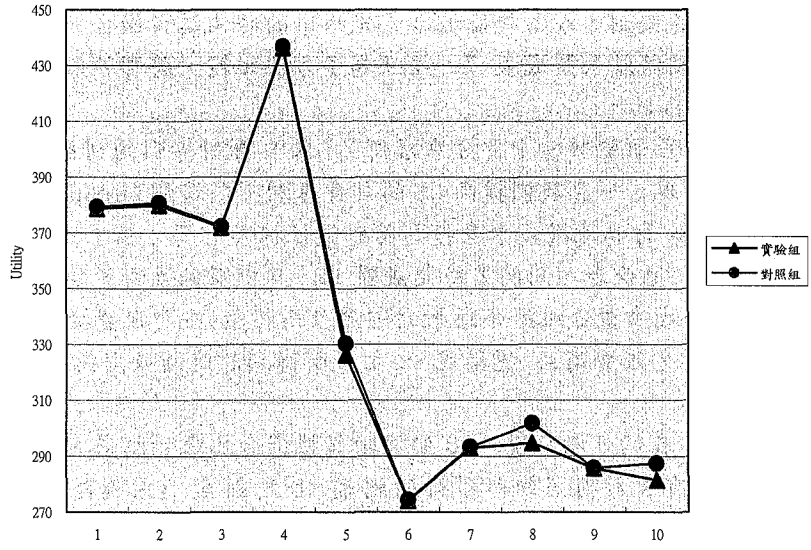

圖 10：實驗組與對照組撮合結果之消費者整體獲利示意圖

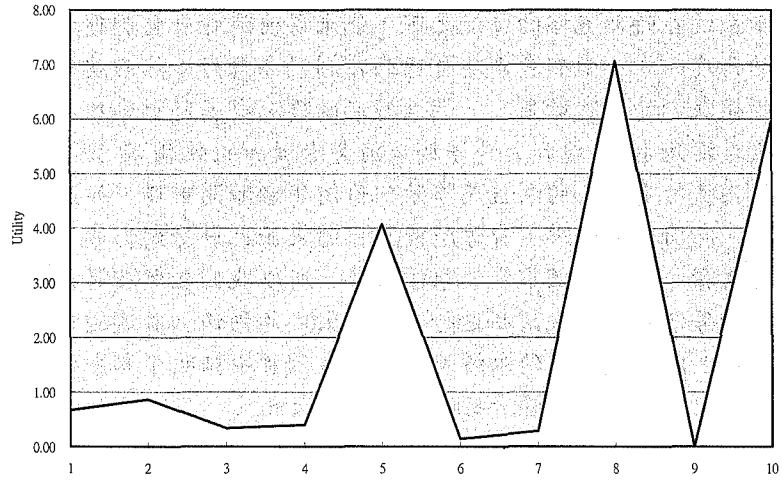

圖 11：實驗組與對照組之間總體效用指數的差距

伍、結論

網路市場的即時性與方便性，將逐漸取代顧客至傳統市場購物的方式，進而提供傳統市場無法提供的服務，因此本文提出一套新的未來購物模式，此套購物模式，可以藉由其它顧客的幫助，共同享受優惠方案的福利；廠商之間亦可以透過合作異業結盟，提供顧客更完備的服務。

另外本研究方法尚有以下特點：

（一）更貼切反映顧客對產品的喜好：

本研究評估消費者對多重屬性商品的喜好程度，摒棄過去僅對價格做比較的方式，並可以避免消費者必須主觀給定效用函數權值的缺點。另外本研究根據此機制衍生出評估不同消費者對不同商品喜好程度的絕對指標，使不同消費者對產品的喜好程度亦可以拿來做比較，使撮合結果能更貼切反映出顧客的喜好。

（二）以中央撮合方式取代傳統的個別議約：

中央集中撮合的方式比傳統的個別議約更能掌握整個市場的狀況。傳統的個別議約方式雖然是以求個人的自身利益最大化為目的，然而在每個人都是自私的情況之下，反而沒有辦法得到一個整體利益最佳化的狀態，長期下來其平均獲利反而會比中央集中撮合的方式要來得低，此推論已在第四章的最後一個實驗中獲得証明。因此為了市場的永續經營與個人的長遠利益著想，以中央撮合方式求得整體平均獲利的最大值會是比較好的辦法。

必須強調的一點是，雖然中央撮合的方式在每一次的撮合過程中不一定會為消費者個人取得最大的獲利，但因為系統本身的設計，分配賣方販賣商品 給買方需求商品的原則是該賣方販賣商品 至少必須滿足買方需求商品的基本要求，以此前提再進行 Improving 的動作，所以最後得出來的結果必定會大於等於買方需求商品的基本要求，就買方需求商品本身而言其獲利亦獲得改善，並能在系統集中控制之下使得整體利益最大化，而達到個體整體兩者兼顧的目的。

（三）在有限時間之內謀求賣方與買方的最大利益：

由於 Anytime Algorithm 概念的引用，本研究可以在有限的時間之內求算出該條件之下使消費者整體獲利最大的最佳解。由於實驗結果顯示增加 Group 中的使用者數與優惠方案數皆能提高 Group 的整體利益，然而無論是增加使用者或優惠方案其增加的撮合時間都相當可觀，此時引入 Anytime Algorithm 是相當合適並且正確的。

換言之，本研究提出一個可以在時間與使用者的整體利益之間做取捨的演算方法，讓系統可以在使用者要求的時間之內將整體利益最大化。現今硬體產業發展十分快速，軟體是否能配合硬體發展並有效利用硬體效能，一直是軟體業界所重視的主題之一。由於電腦的計算能力將隨科技的發展突飛猛進，大大縮短系統撮合所需要的計算時間，Anytime 演算法便成為能隨硬體演進而提高效能的一種有效演算法。由於本演算法採 Anytime 的設計，並且具有「在時間充裕的前提之下必能求得最佳解」的特性，可知當

硬體發展至一定程度時，本演算法將成為一個能在最快時間之內求得最佳解的撮合演算法，使本研究極具前瞻性。

目前我們的研究著重在中央集中控管的撮合機制與 Anytime 的 Improving 演算法，未來希望能朝以下幾個方向努力，使 Anytime Coalition Formation 的效能更臻完美：

（一）Origin Node 的選擇：

由於我們的重點放在 Improving，目前的 Origin Node 部份僅只按照喜好程度與進入市場的先後順序做為分配的依據，然而現實生活中可能還有許多因素可以納入，例如買方或賣方的信譽等。Origin Node 為一切 Improving 之始，選擇一個好的 Origin Node 勢必能提升整個 Group 的效益，這是我們未來必須努力的方向。

（二）優惠方案的篩選：

從實驗中我們可以知道，每增加一個優惠方案所增加的時間量相當龐大，因而如何在進行 Improving 之前先對優惠方案進行篩選過濾整理就變得非常重要。由於優惠方案是由賣方提出的，使用者本身可能沒有很好的組織能力，可能會在 Group 中提出重覆的、錯誤的、不合乎邏輯的或無法滿足現有條件的優惠方案，這些都應該先經過妥善的處理再進行 Improving，否則只會大大降低 Improving 的效能，讓大部份的時間花在處理不合用的優惠方案上。在本文中我們曾提出篩揀優惠方案的概念，然而只是透過簡單的篩選做處理而已，未來希望能在這方向加強，使被用來當做 Improving 條件的優惠方案都是最精簡的。

（三）電子市集型態之探索：

本研究實驗環境假設市場上流動的商品種類通常僅侷限於若干種類之產品；市場中使用者多寡不對本研究所提之機制有所影響；其次，我們假設市場買方需求商品的數量與賣方販賣商品的數量接近 (如二手市場的交易情)。未來可改進本研究機制以期將之推廣至其他電子市集型態。

參考文獻

1. 李慶發、民 89，架構於智慧型助理軟體的分散式數位商品市場，私立天主教輔仁大學資訊管理學系碩士論文。
2. 陳世峰、民 89，自動議約助理軟體，私立天主教輔仁大學資訊管理學系碩士論文。
3. Bichler, Martin, Kaukal, Marion, and Segev, Arie, "Multiattribute auctions for electronic procurement," Proceedings of the First IBM IAC Workshop on Internet Based Negotiation Technologies, Yorktown Heights, USA, March, 1999.
4. Caglayan, Alper and Harrison, Colin，"Agent Sourcebook - A Complete Guide to Desktop, Internet, and Intranet Agents," John Wiley & Sons, Inc.，1997.
5. Chavez, Anthony, and Maes, Pattie, "Kasbah: An Agent Marketplace for Buying and Selling Goods," Proceedings of the First International Conference on the Practical

Application of Intelligent Agents and Multi-Agent Technology (PAAM'96), London, UK, April 1996.

6. Dean, T. L., "Intractability and Time-Dependent Planning." In Proceedings of the 1986 Workshop on Reasoning about Actions and Plans, eds. M. P. Georgeff and A. L. Lansky. San Francisco, Calif.: Morgan Kaufmann, 1987.
7. Dean, T. L., and Boddy, M. "An Analysis of Time0Dependent Planning." In Proceedings of the Seventh National Conference on Artificial Intelligence, Menlo Park, Calif.: American Association for Artificial Intelligence, 1988.
8. Guttman, Robert H., Moukas, Alexandros G, and Maes, Pattie, "Agent-mediated Electronic Commerce: A Survey," Knowledge Engineering Review, June 1998.
9. Hansen, Eric A. and Zilberstein, Shlomo, "Monitoring Anytime Algorithms," SIGART Bulletin, Vol. 7, No. 2, 1996.
10. Horvitz, E. J., "Computation and Action under Bounded Resources." Ph.D. diss., Departments of Computer Science and Medicine, Stanford University, 1990.
11. Horvitz, E. J., "Reasoning about Beliefs and Actions under Computational Resource Constraints." Paper presented at the 1987 Workshop on Uncertainty in Artificial Intelligence, Seattle, Washington, 1987.
12. Jennings, N. R. and Wooldridge, M., "Applying Agent Technology," In Journal of Applied Artificial Intelligence special issue on Intelligent Agents and Multi-Agent Systems, 1995.
13. Ketchpel, Steven P., "Coalition Formation Among Autonomous Agents. "Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI 95), Montréal, Québec, Canada, 1995.
14. Klusch, Matthias and Shehory, Onn, "A Polynomial Kernel-Oriented Coalition Algorithm for Rational Information Agents," supported in part bye the NSF, Grant IRI-9423967, 1994.
15. Larson, Kate S. and Sandholm, Tuomas W., "Anytime Coalition Structure Generation: An Average Case Study," Department of Computer Science, Autonomous Agents '99, Seattle WA, USA, 1999.
16. Nwana, Hyacinth S., "Software Agents: An Overview," Intelligent Systems Research Advanced Applications & Technology Department BT Laboratoris, Martlesham Heath Ipswich, Suffolk, IP5 7RE, U.K., Cambridge University Press, 1996.
17. Pos, A., "Time-Constrained Model-Based Diagnosis," Master's thesis, Department of Computer Science, University of Twente, The Netherlands, 1993.
18. Sandholm, Tuomas W., Lesser, Victor R., "Coalition Formation among Bounded Rational Agents," Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI 95), Montréal, Québec, Canada, 1995.
19. Shehory, Onn, Sycara, Katia and Jha, Somesh, "Multi-agent Coordination through Coalition Formation, "supported in part by ARPA Grant #F33615-93-1-1330, 1993.

20. Steinmetz, Erik, Collins, John, Gini, Maria, and Mobasher, Bamshad, "An Efficient Algorithm for Multiple-Component Bid Selection in Automated Contracting," Workshop of the 2nd International Conference on Autonomous Agents (Agents '98), Minneapolis/St Paul, USA, May 10, 1998.
21. Vrbsky, S. V., Liu, J. W. S. and Smith, K.P., 1990. "An Object-Oriented Query Processor That Returns Monotonically Improving Approximate Answers." Technical Report, UIUCDCS-R-90-1568, University of Illinois at Urbana-Champaign.
22. Wallace, R., and Freuder, E., "Anytime Algorithms for Constraint Satisfaction and SAT Problems." IJCAI-95 Workshop on Anytime Algorithms and Deliberation Scheduling, 20 August, Montreal, Canada, 1995.
23. Yahalom, Raphael, Madnick, Stuart E., "e-Commerce Bargain-Hunting with an unBun Model," MIT-Sloan School of Management Cambridge, MA, USA, 1998.
24. Zilberstein, S., "Operational Rationality through Compilation of Anytime Algorithms." PH.D. diss., Computer Science Division, University of California at Berkeley, 1993.
25. Zilberstein,S., "Resource-Bounded Sensing and Planning in Autonomous Systems." Autonomous Robots, Vol. 3, 1996.
26. Ziberstein, S., "Using Anytime Algorithms in Intelligent Systems," American Association for Artificial Intelligence, 1996.