

กลไกสร้างแรงจูงใจที่เข้ากันได้สำหรับการประเมินที่จัดบุุค



นาย ภูษิตย์ สฤษดิชัยนันทา

ศูนย์วิทยทรัพยากร  
จุฬาลงกรณ์มหาวิทยาลัย

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต

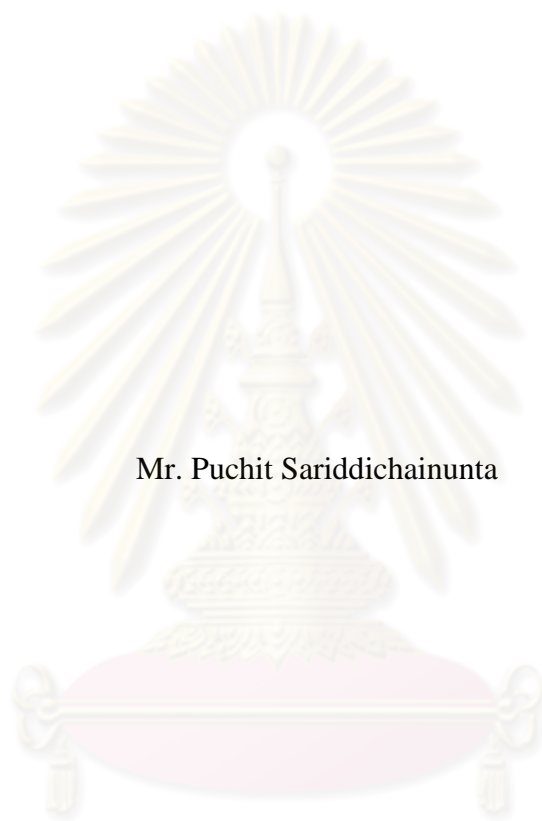
สาขาวิชาคณิตศาสตร์ประยุกต์และวิทยาการคณนา ภาควิชาคณิตศาสตร์

คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2553

ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

AN INCENTIVE COMPATIBLE MECHANISM FOR BOOTH AUCTION



Mr. Puchit Sariddichainunta

ศูนย์วิทยทรัพยากร  
จุฬาลงกรณ์มหาวิทยาลัย

A Thesis Submitted in Partial Fulfillment of the Requirements  
for the Degree of Master of Science Program in Applied Mathematics and Computational Science

Department of Mathematics

Faculty of Science

Chulalongkorn University

Academic Year 2010

Copyright of Chulalongkorn University

Thesis Title	AN INCENTIVE COMPATIBLE MECHANISM FOR BOOTH AUCTION
By	Mr. Puchit Sariddichainunta
Field of Study	Applied Mathematics and Computational Science
Thesis Advisor	Assistant Professor Krung Sinapiromsaran, Ph.D.

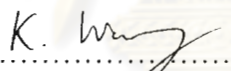
---

Accepted by the Faculty of Science, Chulalongkorn University in Partial Fulfillment of the Requirements for the Master's Degree



..... Dean of the Faculty of Science  
(Professor Supot Hannongbua, Dr.rer.nat.)

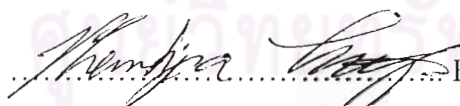
#### THESIS COMMITTEE



..... Chairman  
(Kittipat Wong, Ph.D.)



..... Thesis Advisor  
(Assistant Professor Krung Sinapiromsaran, Ph.D.)



..... Examiner  
(Phantipa Thipwiwatpotjana, Ph.D.)



..... External Examiner  
(Pattara Leelaprute, Ph.D.)

ภูษิต์ สฤทธิชัยนันตา: กลไกสร้างแรงจูงใจที่เข้ากันได้สำหรับการประมูลพื้นที่จัดบูธ.  
(AN INCENTIVE COMPATIBLE MECHANISM FOR BOOTH AUCTION)  
อ. ที่ปรึกษาวิทยานิพนธ์หลัก : ผู้ช่วยศาสตราจารย์ ดร.กรง สินอภิรมย์สรราชู, 53 หน้า.

ตัวแบบปัญหาการตัดสินใจผู้ชนะประมูลสำหรับสิ่งของหนึ่งชิ้นสามารถหาคำตอบโดยง่ายด้วยขั้นตอนวิธีละโมบ (Greedy Algorithm) นอกจากนี้ปัญหาดังกล่าวสามารถแปลงเป็นปัญหากำหนดการเชิงเส้นจำนวนเต็มผสม และหาผลเฉลยโดยใช้โปรแกรมแก้ปัญหากำหนดการเชิงเส้นจำนวนเต็มผสมได้ แต่พบว่าปัญหาการตัดสินใจผู้ชนะประมูลสำหรับสิ่งของหลายชิ้นพร้อมกันซึ่งเป็นปัญหา NP-hard งานวิจัยนี้ปรับปรุงขั้นตอนวิธีการแก้ปัญหของ Rothkopf et al (1998) และวิเคราะห์ตัวแบบกำหนดการเชิงเส้นจำนวนเต็มสำหรับการประมูลสิ่งของหลายสิ่งที่มีการวางตัวในเชิงเส้น นอกจากนี้ผู้วิจัยได้สร้างขั้นตอนวิธีโพลีโนเมียลในการแก้ปัญหการประมูลดังกล่าวสุดท้ายได้วิเคราะห์ผลการเปรียบเทียบเวลาทำงานและอภิปรายการแก้ปัญหด้วยตัวแบบกำหนดการจำนวนเต็มและระเบียบวิธีของผู้วิจัย

ศูนย์วิทยทรัพยากร  
จุฬาลงกรณ์มหาวิทยาลัย

ภาควิชา.....คณิตศาสตร์.....ลายมือชื่อนิสิต.....  
สาขาวิชา.....คณิตศาสตร์ประยุกต์ และ.....ลายมือชื่อ อ.ที่ปรึกษาวิทยานิพนธ์หลัก.....  
.....วิทยาการคณนา.....  
ปีการศึกษา 2553.....

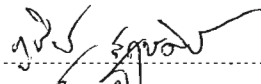
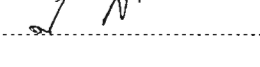
## 5272481623 MAJOR APPLIED MATHEMATICS AND COMPUTATIONAL SCIENCE

KEYWORDS : BOOTH AUCTION / WINNER DETERMINATION PROBLEM /  
INTEGER PROGRAMMING/ DYNAMIC PROGRAMMING/ POLYNOMIAL  
TIME ALGORITHMS

PUCHIT SARIDDICHAINUNTA : AN INCENTIVE COMPATIBLE  
MECHANISM FOR BOOTH AUCTION. ADVISOR : ASSISTANT  
PROFESSOR KRUNG SINAPIROMSARAN, Ph.D., 53 pp.

The winner determination problem (WDP) for a single object auction is a relatively easy problem to solve using the greedy algorithm. It can be formulated and solved using the MIP optimization solver. In this thesis, we applied WDP to solve a booth auction which is one of the nonidentical multiple-object auctions known to be NP-hard. Formulation of the winner determination model for a linear arrangement of a multiple-object auction is explained in this study. Moreover, this research improves the algorithm from the study of Rothkopf et al (1998) having polynomial time complexity. Finally, the comparison of a running time exhibits the advantage of our proposed algorithm. The simulation results are discussed.

ศูนย์วิทยทรัพยากร  
จุฬาลงกรณ์มหาวิทยาลัย

Department : Mathematics Student's Signature   
Field of Study : Applied Mathematics and Advisor's Signature   
Computational Science  
Academic Year : 2010

## Acknowledgements

First of all, I am deeply indebted to my advisor, Assistant Professor Dr. Krung Sinapiromsaran. He is not only a generous couch, but a role model which show that industrious work always repays. As well, he actively encouraged me to overcome all difficulties during course work and thesis completion. Meanwhile, I could not accomplish this thesis without his inspiration and persuasive instruction. The word thank you might not be enough.

Sincere thanks and deep appreciation are also extended to thesis committees and conference anonymous reviewers of my submission at ICMSBE 2011 and IC<sup>2</sup>IT 2011. They suggested me constructively to solidify my research. Besides, I would like to thank all professors in Master of Science Program in Applied Mathematics and Computational Science, Department of Mathematics, Faculty of Science, Chulalongkorn University, granted me scholarship and teaching assistantship. Also, they provided me financial support to participate the international conference.

Furthermore, I wish to thank my friends: Suchit Pongnumkul and Charn Pruksapa who never ignore any of my words. Suchit initialized my knowledge endeavor and Charn examined feasibility of my ideas. I always obtained thoughtful comments in the dialogues with them. Moreover, all companions in AMCS computer lab are thankful to respond cordially to my inquiries. All of them constantly encourage me to complete this master thesis within two years.

Importantly, I am most grateful to my family which is the most important thing in my life. Their unconditional love has brought me up. I really owe what I am to them. Especially, my mother and father always trust me to pursue my passion of higher education. My sisters look after me in many details to intensify my effort in this graduation and also my brother supports me from the heaven. They are very important part of my success.

All honors of my graduation should go to people listed above and all errors should remain to me only.

# Contents

Abstract (Thai) .....	iv
Abstract (English) .....	v
Acknowledgements .....	vi
Contents .....	vii
List of Tables .....	ix
List of Figures .....	x
Chapter I Introduction .....	1
Chapter II Problem background .....	5
2.1 Problem specification .....	5
2.1.1 Linear alignment of a booth auction .....	5
2.1.2 Failure of a simple greedy algorithm .....	8
2.2 Auction scenario .....	9
2.2.1 Type of auctions .....	9
2.2.2 Setup and process .....	11
2.3 Theoretical concepts .....	14
2.3.1 Preference and valuation .....	14
2.3.2 Mechanism design .....	15
Chapter III Literature review .....	17
3.1 Tractable structures of WDP .....	17
3.2 Mechanism design for WDP .....	19
Chapter IV Methodology .....	21
4.1 Simplification of layout problem .....	21
4.1.1 Integer programming .....	21
4.1.2 Dynamic programming .....	23
4.2 Extensions .....	27
4.2.1 Integer programming .....	27
4.2.2 Dynamic programming .....	28
Chapter V Experiment and analysis .....	30
5.1 Simulation environments .....	30
5.2 Random sample simulation process .....	30
5.3 Comparisons of algorithms running time .....	31

5.3.1 Single line case .....	32
5.3.2 Double line case .....	35
5.4 Discussion of simulation results .....	38
Chapter VI Conclusion .....	39
References .....	41
Appendices .....	44
Appendix A Sketch proofs of counting .....	45
Appendix B Total unimodularity .....	49
Biography .....	53



ศูนย์วิทยทรัพยากร  
จุฬาลงกรณ์มหาวิทยาลัย



## List of Tables

Table 2.1: Combinatorial bidding case which a greedy algorithm cannot achieve global optimal .....	9
Table 2.2: The matrix displays categories of some well-known auctions.....	10
Table 5.1: The average running time of each algorithm for the single line case .....	32
Table 5.2: The average running time for the single line case when the number of bidders increases .....	34
Table 5.3: The average running time of each algorithm for the double line case.....	35
Table 5.4: The average running time for the double line case when the number of bidders increases .....	37



ศูนย์วิทยทรัพยากร  
จุฬาลงกรณ์มหาวิทยาลัย

## List of Figures

Figure 2.1: A layout in the exhibition hall.....	5
Figure 2.2: Single line and double line layout .....	6
Figure 2.3: Notation for a bundle of blocks in both single and double line case.....	6
Figure 4.1: The two-dimensional array data structure corresponding to the bundle notation.....	23
Figure 4.2: Layout with multiple zones. ....	27
Figure 4.3: Layout with multiple zones and physical obstructions .....	27
Figure 4.4: The illustrations of data structure corresponding to the layout in Fig. 4.2, the multiple zones.....	28
Figure 4.5: The illustrations of data structure corresponding to the layout in Fig. 4.3, the multiple zones with obstructions .....	29
Figure 5.1: The comparison of average time among three methods (single line).....	32
Figure 5.2: The distribution of running time among three methods (single line).....	33
Figure 5.3: The distribution of running time for each algorithm for the single line case when the number of bidders increases.....	34
Figure 5.4: The comparison of average time between two methods (double line).....	35
Figure 5.5: The distribution of running time among three methods (double line) .....	36
Figure 5.6: The distribution of running time for each algorithm for the double line case when the number of bidders increases.....	37
Figure A.1: The induction flow of the selectable options, the single line case. ....	45
Figure A.2: The induction flow of the selectable options, the double line case.....	46

# Chapter I

## Introduction

Auction is a well-known process to determine allocation of some scarce objects which are highly demanded by many agents . In a general auction<sup>1</sup>, there is only one owner, so-called auctioneer, and several agents, so-called bidders to participate in the event . For the purpose of maximizing revenue, the auctioneer imposes justifiable competition rule such as price submission method and payment method for bidders. On the other hand, bidders submit their competitive price according to their willingness to pay in order to maximize their utilities. At the end of an auction period, the auctioneer evaluates those offers and determines the most advantageous bidder to be the winner of the auction. This kind of revenue maximization problem is known as the winner determination problem (WDP) .

### **Combinatorial auction**

Combinatorial auction is the advance study in auction theory. To illustrate difference, the general auction usually refers to a single object auction, while the combinatorial auction refers to multiple objects auction as well as the characteristics of objects such as divisibility of goods, identical or nonidentical objects, and continuous or discrete quantities. Moreover, bidding options and expression are another concern. Bidders offer price for the bundle of objects following the rule set by the auctioneer such as permitted bidding options, bidding language and payment method. Consequently, it is a complex process not only for the auctioneer to decide his optimal allocation, but also for the bidders to determine their optimal strategies for individual best pay-off. Therefore, the characteristics of a combinatorial auction, which relax bidding options and additional rules, incur tremendous calculation cost to both the auctioneer and the bidders.

It is quite reasonable to think that the utilization of two or more objects yield better benefit, so the bidders who wish to consume those objects can offer higher price to the auctioneer. For example, in 1994, the spectrum auction of Federal

---

<sup>1</sup> In the open procurement, it is called inverse auction to search for lowest price offers from suppliers who try to bid for purchase order acquisition.

Communication Commission (FCC)<sup>2</sup> sets up rules of bidding multiple licenses over regions. This is because geographical synergy of telecommunication industry exists and companies have motivations to gain profit from such situation. It claimed for \$60 billion USD at that time. Some similar spectrum auctions also occur in Germany, UK and Sweden. As well, a combinatorial auction still appears to wider areas; e.g., U.S. Treasury note; electricity grid auctions in UK.

A plenty evidences of combinatorial auction are available in industry practice, too. For instance, airport arrival-departure time slot approach for New York's LaGuardia airport; bus route market in London; sponsored search for ad-slot service in Google and Yahoo. Example from the procurement in the combinatorial auction styles are such as milk procurement for public school in Chile; the procurement of freight transportation services; industrial procurement and so on. Thus, these evidences represent the usefulness and expectation of a combinatorial auction in practice.

Those real world applications attract many scholars from economics, operations research, and computer science to progressively conduct researches in combinatorial auction in both theory and applications. First, theoretical economists explore the possibility of the auction theory to enhance the efficiency of market-like mechanism for a combinatorial auction in aspect of game theory and mechanism design. As well, package bidding in operations research emphasizes the necessary techniques to solve the combinatorial optimization via the state-of-art optimization solver. Finally, computer scientists extensively make effort on an algorithmic design to improve speed and inspect the expressiveness for bidding languages. Indeed, the crossing of these three academic collaborations describes the development and the importance of combinatorial auctions.

### **Booth auction**

Booth auction is one of the combinatorial auctions that bidding auction must be the consecutive objects. The bidders are not allowed to bid booth and skip or make a hole, but they can propose every bidding option at the same time. This type of auction is classified as geometry-based structure , linear alignment , consecutive

---

<sup>2</sup> <http://www.fcc.gov/wtb/auctions/>

objects, etc. Those auction structures can be proved to be tractable that it is possible to solve in polynomial time. After the work of Rothkopf et al introduced the dynamic programming style for this problem, there is no further investigation of its extension in a double line case. This thesis contribution is the extension of previous work and the acceleration of algorithms relevant to a booth auction.

Note the characteristics of booth auction are widely observable. Basically the objects in the booth auction refer to the blocks in the hall space which located back-to-back. This correlates to the auction which depends on geographic adjacency such as pieces of land, paddles or space; the spectrum auction for radio licenses, the territory for milk delivery and advertisement space in the classified. Other applications could be the problem to determine time schedule or time slot; priority queue for server time slot, meeting appointment allocation and so on. The booth auction thus has application for allocation of multiple objects which has synergy effect among those consecutive bundles.

As the perspective of a rational auctioneer, this thesis would be able to provide a tool for solving WDP as well as a suggestion to those auction practitioners who will set up an auction alike the problems in this thesis – the linear alignment of multiple objects.

### **Research objectives, questions and scope**

Under the research interest of the combinatorial auction, the objectives of this thesis are to develop an approach to obtain the optimal solution for WDP and to study an incentive compatible mechanism for a certain structure of auction scenario. The former means that this thesis develops an algorithm for WDP, and the latter refer to the review of study to analyze the property of the algorithm in a mechanism design aspect. This thesis scope is limited to the specific domain of the problem, booth auction, which the characteristic of linear alignment depends on the geographic layout. The problem specification is described in the next chapter.

Specifically, the research question is to solve an allocation problem of booth layout in an exhibition space. The work is an extension of Rothkopf et al who studied the computationally feasible case of a combinatorial auction in the geographic problem domain. They illustrated polynomial time algorithm for linear objects allocation. Consequently, this thesis contributes to a general case not only for a single

line booth auction but also for a double line booth auction and some additional obstructions.

The structure of this thesis is as the followings. Chapter 2, problem background, introduces the problem specification, some necessary theorems and notations of this thesis. Thereafter, the literature reviews of related work will be described in chapter 3. The methodology of this research work is explained in chapter 4, and the experimental results of the simulation are presented in chapter 5. Finally, chapter 6 concludes the works of this thesis and discusses future works.



## Chapter II

### Problem background

According to numerous details of a combinatorial auction, this chapter provides preliminary information, such as a problem specification, a bidding scenario, and a theoretical notation, which are used in this thesis.

#### 2.1 Problem specification

Rather study for a general case of a combinatorial auction, this thesis narrow its scope to a specific problem domain, linear alignment of a booth auction. It is a geographic allocation problem of consecutive blocks Fig. 2.1. Rothkopf et al first introduce this structure of auction objects regarding to the merit of obtaining telecommunication licenses of the northern to the southern region consecutively in spectrum auction. This case is also known as a tractable combinatorial auction .

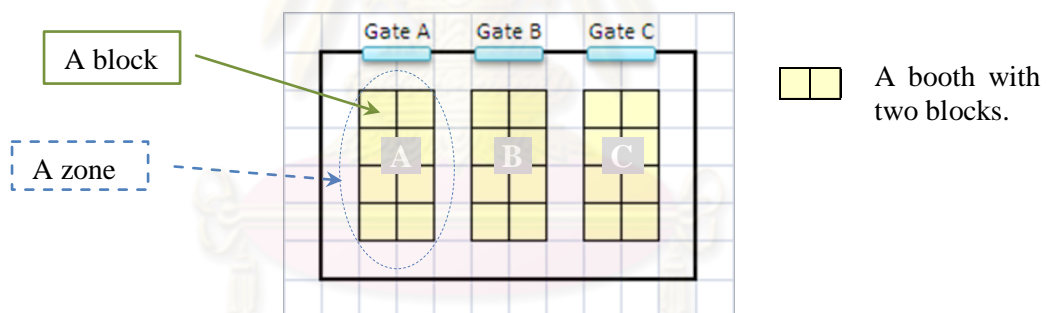


Figure 2.1: A layout in the exhibition hall

##### 2.1.1 Linear alignment of a booth auction

Linear alignment of a booth auction refers to the allocation of blocks as a singleton of its geographic adjacency. Given an empty space in a hall, the organizer plans an event and divides the hall space into a certain size of blocks for a booth layout as in Fig. 2.1. A booth could contain more than one adjacent block. The layout of booth can be one line of blocks or double line of blocks with a pathway in between. Definitely, there could be more than one block in the booths, but the blocks must be located consecutively back-to-back in a rectangular form. For the argument

simplicity, the case of one row of a single line and a double line is given, as in Fig. 2.2. Hence, the bidding options are set up by the layout of booths.

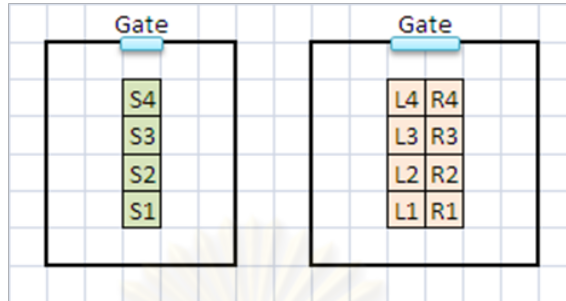


Figure 2.2: Single line and double line layout

Bidders can offer one price for a bundle of blocks in a rectangular form, which lie consecutively as a filled rectangular block. A bundle is denoted by starting and ending blocks, [start,end]. Based on the booth layout in Fig. 2.2, the starting blocks is the largest index in a lexicographic order, and the ending block is the smallest index. In the case of crossing blocks for a double line configuration, the starting block is the block from the right hand side, and the ending block is the block from the left hand side. The following examples are valid bundling options. The bundle only S1 is indicated by [S1,S1]. Bundle S2 and S3 appears as [S3,S2]. Bundle L1, L2 and L3 appears as [L3,L1], and bundle L2 to L4 and R2 to R4 appears as [R4,L2]. Unconnected or non-rectangular blocks are not permitted; e.g., a bundle S4 and S2 or a bundle R3, R2 and L2 are invalid bundles.

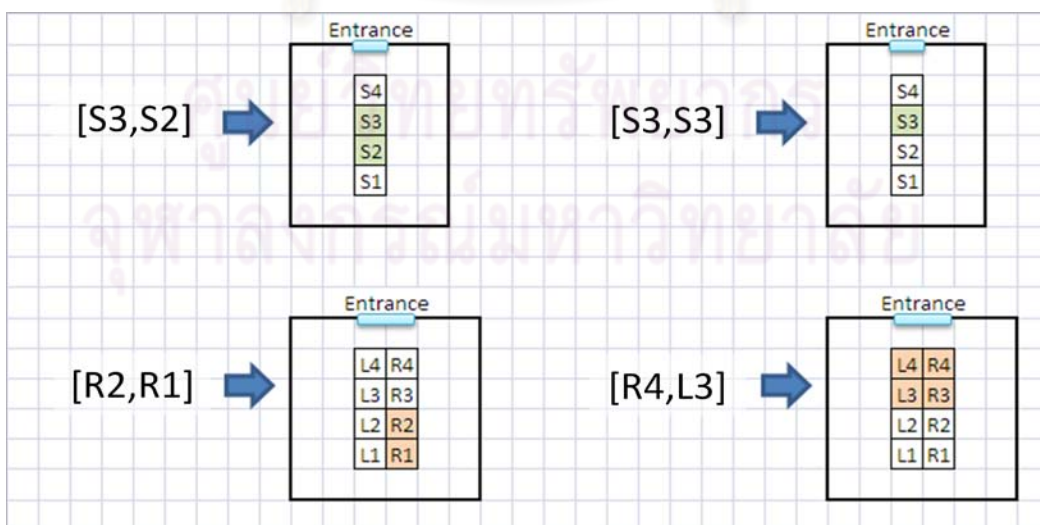


Figure 2.3: Notation for a bundle of blocks in both single and double line case.



The method to count the number of options is straightforward using the mathematical induction<sup>3</sup>. In the case of a single line, if we have  $n$  blocks, the number of possible bundle is  $\frac{n(n+1)}{2}$ . An example of the booth of three blocks is displayed in Table 2.1 that there are 6 possible bidding options. If  $n = 4$ , there are 10 options: [S1,S1], [S2,S1], [S3,S1], [S4,S1], [S2,S2], [S2,S3], [S2,S4], [S3,S3], [S3,S4], and [S4,S4]. In the case of a double line, the number of possible bundles is  $\frac{3n(n+1)}{2}$  providing rectangular blocks. These combinations are from two times of single line, left and right case, plus crossing line case. Therefore, the option in this combinatorial auction does not grow exponentially. This is a special structure of the linear alignment in our problem domain.

In the actual booth layout, there are some obstacles that may prohibit a location of some booths such as multiple zones and physical obstructions. These conditions restrict bidding options, and would probably increase the complexity of the problem. First, the multiple zones are the distinct area for booths which locate sporadically from each other. For example, in Fig. 2.1, three distinct zones are displayed. The number of zones expands the bidding options for the discontinuous selection of booths. Bidders can choose a bundle of blocks in the same zone which does not cross over another zone. Similarly, the physical obstructions also cause additional restrictions in bidding options. The auctioneer must explicitly identify the obstruction positions, so that bidders can avoid including those positions. Hence, both sources of restriction increase the complexity to obtain optimal solutions.

In order to handle a combination of restriction for both the multiple zones and the physical obstructions, we add some constraints to the winner determination problem, using the integer programming solver and dynamic programming application, to obtain the solution. Their specific details are explained in the chapter 4 methodology.

---

<sup>3</sup> See the sketch proof in appendix A.

### 2.1.2 Failure of a simple greedy algorithm

It is known that a combinatorial auction is related to a knapsack problem. To explain shortly, auctioneer has to choose various bids to maximize his revenue instead of selecting items to maximize utility as in the knapsack problem. The greedy algorithm is a common method to solve a knapsack problem. Since the greedy algorithm chooses the local optimal solution without considering every feasible substructures, it could not guarantee the global optimal which lead to an inferior solution. Moreover, this may lead to the complaint of unfair allocation which deprives bidders' incentive to offer a high-valued submission price.

Generally the rational auctioneer wants to select the best offers from many bidders to maximize his revenue. The greedy algorithm is useful for a single-object case, but it would not be suitable for a multi-objects case. The following instance demonstrates a failure of the simple greedy algorithm<sup>4</sup>:

Considering the situation of three-block-long single line, it has the price offer from each option – S1, S2, S3, S1US2, S2US3, S1US2US3 – as in Table 2.1. There are bidders A and B compete in the auction. If the auctioneer begins with the biggest bundle S1US2US3, he obtains bid value 7. Next step, the auctioneer compares result with its local partition [S1, S2US3]; the revenue becomes  $1+4=5$ . This new partition returns the total price less than the former; therefore, the greedy algorithm terminates<sup>5</sup> with the result [S1US2US3]. However, the partition [S1, S2, S3] provides a better solution which the bid result is  $1+5+2=8$ . Hence, the greedy algorithm might not give the optimal solution for the auctioneer.

Indeed, this research employs other methods to find the optimal solution of a determination problem in a linear alignment of a booth auction. One common method to determine the solution is to use the integer programming which is one of the mathematical programming technique. The other is the dynamic programming which determines the optimal substructure via recursive technique. Their specific details are explained in chapter IV.

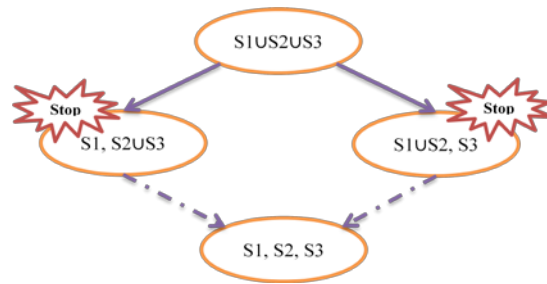
---

<sup>4</sup> In this context, we set up the greedy algorithm that will terminate if it discovers a result which is not better off than the former.

<sup>5</sup> We assume that the greedy algorithm terminates when it cannot obtain strictly better-off solution in the next comparison.

Table 2.1: Combinatorial bidding case which a greedy algorithm cannot achieve global optimal

Bundle	A	B	Max
S1	1	1	1
S2	3	5	5
S3	2	1	2
S1US2	4	5	5
S2US3	4	3	4
S1US2US3	7	6	7



## 2.2 Auction scenario

This section explains the setup for a combinatorial auction scenario with notation remarked for the rest of this thesis. As well, bidding process, winner determination and payment rules are discussed here.

### 2.2.1 Type of auctions

There are many ways to categorize auction, price submission methods and the payment rules. The sealed-bid auction is a static type which allows bidders to submit price only one time. The bidders cannot a priori know the other bidders' submitted price. Usually the highest offer price obtains the right to acquire that target; however, the payment would probably be the first-price or the second-price depending on the payment rules. In the first-price sealed bid auction, the winner is obliged to pay the winning price. On the other hand, the winner is allowed to pay the next winning price for the second-price sealed bid auction. This second-price sealed bid auction (Vickrey auction) is famous for its incentive compatible property which motivate bidder to submit their estimated value as bidding price truthfully.

Next, the dynamic aspect of auction is used in an ascending-bid auction (English auction) and a descending-bid auction (Dutch auction). These mechanisms devise price discovery system which motivate bidders tend to offer price closed to their estimated valuation to win the auction. In an ascending-bid auction, bidders will raise their price gradually in order to win his target. However, in a descending-price auction, the auctioneer will decrease the price until a bidder agrees with it. As a result, both mechanisms lead to discover the highest price bidders who will be the winner of the auction.

Table 2.1: The matrix displays categories of some well-known auctions

		Payment Rules	
		First-price	Second-price
Price Submission Method	Static	First-price sealed-bid auctions	Vickrey auctions
	Dynamic	English auctions Dutch auctions	

In fact, many bidding methods and payment rules are invented to enhance some preferable outcomes in a mechanism design such as the truthfulness of bid prices. The combinatorial auctions are especially the motivation for the invention. For instance, Ausubel and Milgrom propose a multiple unit auction in [1]. It is an alternative ascending-price format of which outcome equivalent to the Vickrey auction. Iterative combinatorial auction method by Parkes and Ungar [2] invents another method for auction agents that assign non-additive values to resources, such as distributed scheduling and task assignment problems. Both are a few examples for a special auction which is made for specific circumstances.

Finally, bidding language is a necessary tool to understand category of auctions [3]. It considers how the expression in a combinatorial auction creates complexity for WDP. The major ideas is initiated in OR and XOR type. OR means each bidder is willing to obtain any bundle options for which they submit price, while the individual desire at most one bundle in XOR. Further that, many complicated expressiveness are the combination of OR and XOR for some specific options; theoretically, it refers to analysis of a Boolean logic in the various expression.

In summary, basic categories for auctions are the price submission method and payment rules. Since the potential of the combinatorial auction is studied, scholars invent new auctions rule to achieve the equivalent property once inherit in the Vickery's auction. As well, bidding expressiveness emphasizes the critical issue of bidding language. Indeed, these categories are important to understand both operation and outcome of auctions.

## 2.2.2 Setup and process

As stated in the introduction, auction is a market-like process to determine allocation decision. The auction process is simple or not depends on type of auction and payment rules. It usually starts with calling for bidders to compete by submitting the price offer according to their financial ability. Bidders know their objects values, but they might not be able to determine the best submitting price accurately. After bidders submit price in the permitted method, which may end in one round or several rounds, the auctioneer determines the final winner of the auction. In a combinatorial auction, there would be several winners. Finally, the auctioneer requests for the payment of allocation result.

### A. Definition and notations

The followings are notations used in this thesis. They are explained according to the auction process based on game theoretical notations.

- **Players:** Let  $\mathcal{I} = \{0, 1, \dots, i\}$  be a finite set of players; the number 0 is an auctioneer and the others represent bidders. An auctioneer and  $i$  bidders are the players in this auction game.
- **Goods:** Let  $\mathcal{G} = \{1, \dots, g\}$  be a finite set of goods. The auctioneer sells  $m$  goods to  $n$  potential bidders. The bundle of goods is denoted by  $\mathcal{S} \subseteq \mathcal{G}, \mathcal{S} \neq \emptyset$ . Hence, the possible bidding options set  $\mathcal{H}$  is the power set  $2^{|\mathcal{G}|}$  excluded empty set,  $\mathcal{H} = 2^{|\mathcal{G}|} \setminus \{\emptyset\}$  for every  $\mathcal{S} \in \mathcal{H}$ .  $\mathcal{H}$  has  $t = 2^g - 1$  bidding options<sup>6</sup> since empty set is excluded.
- **Bidding:** Bidders have a nonnegative valuation for each bundle,  $v^i(\mathcal{S}_i) \geq 0$  and  $v^i(\emptyset) = 0$ , and submit bidding price  $b^i(\mathcal{S}_i) \geq 0$  such that  $v^i(\mathcal{S}_i) - b^i(\mathcal{S}_i) \geq 0$  in their bidding profile.
- **Payment:** Since auctioneer can arrange the payment to motivate bidders submit truthful information, the payment  $p(\mathcal{S}_i)$ , could not be greater than the

---

<sup>6</sup> However, in our problem specification the number of options is fewer due to the alignment restriction.

bidding price corresponding to the bundle,  $p(\mathcal{S}_i) \leq b^i(\mathcal{S}_i)$ . For simplicity, let  $p(\mathcal{S}_i) = b^i(\mathcal{S}_i)$  be the payment rule.

- **Profile:** Let  $\mathcal{X} = \{\mathcal{X}^0, \mathcal{X}^1, \dots, \mathcal{X}^i\}$  be the total profile. To allocate all goods, auctioneer intakes profile,  $\mathcal{X}^i$ , from each bidder, then he will determine the winner and announce allocation profile,  $\mathcal{X}^0$ .

$\mathcal{X}^i = \{x_1^i = (\mathcal{S}_1, b(\mathcal{S}_1), i), \dots, x_t^i = (\mathcal{S}_t, b(\mathcal{S}_t), i)\}$  is the bidding profile of bidder  $i$ .

$\mathcal{X}^0 = \{x_1^0 = (\mathcal{S}_1, p(\mathcal{S}_1), w_1), \dots, x_t^0 = (\mathcal{S}_t, p(\mathcal{S}_t), w_t)\}$  is the allocation profile of the auctioneer. The third element of tuple  $x_t^0, w_t \in \mathcal{I}$  represents allocation decision – a bidder who obtains that bundle or nobody does.

- **Payoff:** utility (profit) functions are defined for the auctioneer and bidders.

$$\text{Bidder's utility} \quad u^i(\mathcal{X}^i) = v^i(\mathcal{X}^i) - p(\mathcal{X}^i) = \sum_{k=1}^t (v^i(x_k^i) - p(x_k^i))$$

$$\text{Auctioneer's utility} \quad u^0(\mathcal{X}^0) = \sum_{k=1}^t p(x_k)$$

- **Optimization problem:**

Bidders are to maximize their utilities by offering a bidding price vector,

$$\mathbf{b}^i = \operatorname{argmax} u^i(\mathcal{X}^i), \text{ for } \mathbf{b}^i = \langle b^i(\mathcal{S}_1), \dots, b^i(\mathcal{S}_t) \rangle \text{ under the budget constraint,}$$

if any.

Auctioneer is to maximize his revenue by searching the decision vector,

$$\mathbf{x}^* = \operatorname{argmax} \sum_{k=1}^t p(x_k), \text{ for } \mathbf{x} = \langle x_1^0, \dots, x_t^0 \rangle, \text{ under constraint of no duplicated}$$

goods allocation.

## B. Auction process: an example

The behavior of bidders and auctioneer in an auction are to optimize each individual. Their scope of actions is reviewed in this section along with the timeline. To illustrate, the FCC spectrum auction, Simultaneous Multiple Round (SMR) Auction, is an instance for this study. SMR is conducted in Web-Based Bidding System. The auction rule contains that each individual submission is offered at the same time, and it will be repeated for multiple bidding rounds. Bidding is confidential

during a round. When a round ends, results are processed and made public. The auction will be terminated when nothing happened at that round.

- **Pre-auction**

First of all, the auctioneer announces the auction event for public notice. He will disclose all necessary information: objects information, price submission method, decision method, prohibited action, payment rules. Consequently, the participants in auction, bidders, are measured for their bidding eligibility. It is to prevent for collusions among bidders, to clarify financial status, and to underwrite their provisional price submission.

- **Bidding**

Bidders must bid throughout the auction. According to the bidding rules, bidders offer the price which could maximize their expected utility. In this case simultaneous multiple round, Minimum opening bids are established prior to the auction. All submission prices are disclosed at the end of a round to see provisionally winning bids (PWBs), all bids, bid withdrawals, and proactive waivers.

- **Winner determination**

The Auctioneer is responsible for finding the combination of the best offer in order to maximize his benefit from those proposed bidding at the end of bidding. The comparison for the best offer is the core of price discovery. In the final stage, usually the combinations of highest submission that do not duplicate objects allocation are the optimal solutions.

- **Payment request**

Finally, the auctioneer issues the bill of payment<sup>7</sup> to the winners to pay for the amounts in their price submission.

Actually the above is an auction design of combinatorial auctions. There are four basic steps to understand a certain auction. It is possible to generate other mechanisms to enhance price discovery for auction objects. However, those mechanisms should hold some preferable properties in mechanism design theory to

---

<sup>7</sup> The payments usually do not exceed their submission as well as their anticipated valuation.

correspond to bidders' strategic behaviors. Those concepts are reviewed in section 2.3.2.

## 2.3 Theoretical concepts

This section describes some theoretical concepts, preference and valuation and mechanism design, which are necessary to all following chapters. These concepts can be found in the standard text book for the auction theory and multiagent system analysis .

### 2.3.1 Preference and valuation

Specific relationship among set of objects is considered. When the auction goods have nonidentical characteristics and they are being sold altogether, bidders would evaluate the additional goods to take in option in two different ways, *substitute* or *complement*. I will follow the formal definition in Krishna .

- **Substitute:** bidder  $i$  considers the goods in  $\mathcal{G}$  to be substituted if for all  $a \in \mathcal{G}$  and bundle  $\mathcal{S}$  and  $\mathcal{T}$  not containing  $a$ , such that  $\mathcal{S} \subset \mathcal{T}$  ,

$$v^i(\mathcal{S} \cup \{a\}) - v^i(\mathcal{S}) \geq v^i(\mathcal{T} \cup \{a\}) - v^i(\mathcal{T}) \quad (2.1)$$

The inequality (2.1) is equivalent to requiring that for all bundles  $\mathcal{S}$  and  $\mathcal{T}$  ,

$$v^i(\mathcal{S}) - v^i(\mathcal{T}) \geq v^i(\mathcal{S} \cup \mathcal{T}) + v^i(\mathcal{S} \cap \mathcal{T}) \quad (2.2)$$

Functions satisfying (2.2) are called **submodular**. In particular, if  $\mathcal{S} \cap \mathcal{T} = \emptyset$ , then, since  $v^i(\emptyset) = 0$ , the inequality in (2.2) reduces to

$$v^i(\mathcal{S}) - v^i(\mathcal{T}) \geq v^i(\mathcal{S} \cup \mathcal{T})$$

Hence, the substitute property implies that  $v^i(\cdot)$  is a **subadditive** function over the bundle of goods.

- **Complement:** It is similar to the substitute case, but the opposite side of inequality.

$$v^i(\mathcal{S} \cup \{a\}) - v^i(\mathcal{S}) \leq v^i(\mathcal{T} \cup \{a\}) - v^i(\mathcal{T}) \quad (2.3)$$

The inequality (2.3) is equivalent to requiring that for all bundles  $\mathcal{S}$  and  $\mathcal{T}$  ,

$$v^i(\mathcal{S}) - v^i(\mathcal{T}) \leq v^i(\mathcal{S} \cup \mathcal{T}) + v^i(\mathcal{S} \cap \mathcal{T}) \quad (2.4)$$



Functions satisfying (2.4) are called *supermodular*. In particular, if  $\mathcal{S} \cap \mathcal{T} = \emptyset$ , then, since  $v^i(\emptyset) = 0$ , the inequality in (2.4) reduces to

$$v^i(\mathcal{S}) - v^i(\mathcal{T}) \leq v^i(\mathcal{S} \cup \mathcal{T})$$

This implies complement property of  $v^i(\cdot)$  is a *superadditive* function over the bundle of goods.

- **Additive:** If both (2.1) and (2.3) are hold, then the value are *additive*. That is, the value of any bundle  $\mathcal{S}$  is simply the sum of the values of the individual objects in that bundle. In this case, it is useful to think of the different objects as being completely unrelated since the value derived from a particular object  $a$  does not depend on whether another object  $b$  is obtained.

### 2.3.2 Mechanism design

An auctioneer has a particular objective - maximizing revenue. Although he has an algorithm to determine the best solution from several bid options, the solution could be improved when the bids are set to the highest price. Because the participant bidders have incentive to maximize their payoff, they would not tell their valuation and lower submission price. Therefore, in order to maximize the auctioneer expected revenue, it is necessary to design a mechanism that enhance bidders participate and offer bidding prices as high as possible.

Mechanism design is a very important topic in Microeconomic and gain much attentions in Multiagent Systems . It studies for a property in the mechanism to induce some preferable outcome, which is a reverse engineering of the game theory. It is implemented under the Bayesian game setting. Hereafter, the mechanism refers to the auction and the agent as the bidder. Those properties to be shortly introduced are participation constraint (PC), incentive compatibility (IC), and Efficiency (EF). See for the rigorous definition.

- **Participation constraint**

Participation constraint or Individual rationality is that the agent has incentive to participate in the mechanism because of the non-negative utility,  $u^i(\mathcal{X}^i) \geq 0$ .

- **Incentive compatibility**

Incentive compatibility or Truthfulness refers to the situation that agent  $i$  is to adopt the strategy  $b^{*i}(\mathcal{S}_i) = v^i(\mathcal{S}_i)$  for the Bayesian-Nash equilibrium. The revelation principle claims that the truthful mechanism always exists; however, it is very difficult to find for a computational view point .

- **Efficiency**

The mechanism is efficient if it is Pareto efficient,  $u^0(\mathcal{X}^{*0}) \geq u^0(\mathcal{X}^0)$ . The most famous class of a mechanism for the auction is Vickrey–Clarke–Groves mechanisms (VCG). It motivates agents to choose the socially efficient allocation even if agents have privately known valuations and even though it is very difficult to implement in reality .

In fact, the mechanism design theory suggests more properties which are also preferable. This thesis survey just a few that directly relates to strengthen the maximization of auctioneer's revenue. Moreover, the formal analysis of a mechanism to prove for the existence of those properties has not yet been conducted in this research. Therefore, this thesis does not cover an analysis of auction as a mechanism which considers all parties react strategically. This aspect will be discussed later in future.

# Chapter III

## Literature review

Combinatorial auction has been studied in both theoretical and algorithmic aspects. The former focuses on an auction design and the latter on an efficient algorithm. Auction design constructs and analyzes the auction rules whether it achieves some preferable properties. However, the auction rules would be used inefficiently if the computation is costly and impractical. Therefore, many scholars make some attempts on the research of specific domains observed in reality, propose manageable solving method, and evaluate efficiency.

This chapter reviews some literatures on WDP for some specific domains, such as, internet ad-slot auction and geometric allocation.

### 3.1 Tractable structures of WDP

The structure of combinatorial auctions imposes complexity to search for the optimal solution in WDP . First, the number of objects in auction is the major source of complexity to determine the winner in the auction. The bidding strategy for a single object is simple, but WDP becomes unmanageable when bidders have more options to bid their targets. In addition, if the object is indivisible, it is necessary to identify only the integer solution. As well, the case of non-identical multiple objects imposes computational burden on the auctioneer in searching for the optimal solution. Indeed, WDP is an NP-hard problem because its decision variable sparsely growing in number of bidders and combination of options.

#### A. Rothkopf, Pekeč and Harstad (1998)

Tractable structure of WDP has been firstly introduced by Rothkopf et al . In this famous paper, they discuss the importance of a computable combinatorial auction that the algorithm should be in a class of polynomial time complexity. Along with the structure of a combinatorial auction in practice, they prove and introduce algorithms for the problem structures which are solvable in polynomial time; i.e., nested structures, cardinality-based structures and geometry-based structures. Nested structures are that only one type of combination is able to bid together, while the

cardinality-based structures illustrate more than one type of groups. The geometry-based structure is the most relevant to our problem which binding bidding options depending on their adjacency as explained in the problem background.

In this paper, the geometry-based allocation is our main concern. Based on their proof of the optimality for the dynamic programming substructure, they begin with the single line and then followed by the circular case and propose the polynomial time algorithm. Furthermore, they show that the generalization of  $m \times n$  blocks allocation is NP-complete. The linear alignment of consecutive blocks in this thesis is influenced by Rothkopf et al. In addition, this thesis has introduced a faster algorithm to solve for a single line case, and then a double line case which does not exist in their work. The approach to the general layout that has multiple zones and physical obstruction are the extensions to Rothkopf et al work.

### **B. Tennenholtz (2000)**

Tennenholtz further investigates the tractable case for a combinatorial auction. He proves polynomial running time for a combinatorial network auctions, various sub-additive combinatorial auctions, and some restricted forms of multiple-objects auctions. The allocation of objects in geometry-based structure could be one of the restricted forms. In his proofs for polynomial complexities solutions, he elaborates b-matching techniques in graph algorithms to identify those tractable combinatorial auctions. Even though there are no implementation results in this work, the computationally tractable in a polynomial complexity is guaranteed for combinatorial auction in the geometry-based structure.

### **C. Sandholm (2002)**

This paper initiates an analysis in some bidding languages with full expressiveness called XOR-bids and OR-of-XORs as well as tractable algorithms. His bidding language enhances expression of the general preference, both of complementarity and substitutability. He also proposes the optimal search algorithm and preprocessors to cope with the problem of new bidding languages. The optimal search algorithm is constructed by four approaches: allow bidding on combinations, find the optimal solution, completely avoid loop and redundant generation of vertices and capitalize heavily on the sparseness of bids. For preprocessing, he also suggests three jobs: keep only the highest bid for a combination, remove provable

noncompetitive bids, and decompose the set of bids into connected components. In sum, this paper extensively demonstrates a constructive method to solve WDP in a general case of a combinatorial auction.

### 3.2 Mechanism design for WDP

The analysis of strategic behaviors among bidders is an important issue for a mechanism design for an auctioneer to reap the maximum revenue. Especially, in a combinatorial auction, the auctioneer would gain more or lose some benefit from bidders' strategic behavior. They always have a motivation to submit price below their valuation. Therefore, the auctioneer has to design an incentive compatible mechanism<sup>8</sup> to generate sufficient incentive for bidders to submit price as high as the valuation. These following researches have proposed the mechanism which equip with the preferable properties of mechanism design, and also relate to this thesis problem specification.

#### A. Parke and Shneidman (2004)

From an iterative method algorithm iBundle in , Parke and Shneidman suggest the partition principle to prove equivalent of a mechanism to Vickrey–Clarke–Groves mechanism (VCG) in . Unlike the centralize mechanism design, the partition principle is implemented by computation of each self-interest agent. They also recognize the weakness of methods based on the principle and propose several principles to conduct the distribution of this computation focusing in particular on VCG mechanisms for implementing outcomes that maximize the total utility. However, many problems still remain such as costly computation, restricted communication networks, self-enforcing outcome and specific instantiations.

#### B. Petcu, Faltings and Parke (2006)

The decentralize mechanism for an efficient allocation has been introduced by Petcu et al . Their research highlights a combinatorial auction as an instance of a social choice problem which can be implemented by M-DPOP, their algorithm. The special characteristics of the algorithm will redistribute a problem to each agent to perform computation, report information, and send messages that is in its own best

---

<sup>8</sup> In mechanism design, this term is said as truthful mechanism or incentive compatible mechanism.

interest. At the same time, it provides a faithful distributed implementation for an efficient social choice to self-interest agents. The proof of truthfulness in their proposed algorithm is based on the partition principle of Parke and Shneidman and can be applied efficiently to social choice problems, not limited to just a combinatorial auction.

### **C. Feldman, Muthukrishnan, Nikolova and Pál (2006)**

Internet ad slot auction is another important research target actively conducted by researchers not only in academics but in Yahoo and Google. Internet ad slot auction is similar to a booth auction that bidders want the best position with the high rate of visits. Regarding to bidders' pricing for different positions and budget constraints, Feldman et al set up an allocation rule for an advertisement slot and algorithms to solve WDP. Since the unit of a marginal benefit to acquire from ad slot in this auction is the number of clicks, the nature of auction object is divisible when consider the allocation as proportion occupied each slot. Moreover, bidder is not able to request for a specific ad slot position, while he could obtain just clicks which redistribute after each winners being determined. Thus, the bidders' specific option in position arrangement has not yet been studied for the internet ad slot auction.

In sum, this thesis will concentrate on a special case - the linear alignment of a multiple-object auction, and explain the success of a dynamic programming approach to maximize the auctioneer's revenue. The new algorithms are developed based on Rothkopf et al with the explanation of Tennenoltz and Sandholm . Unfortunately, it is beyond the scope of this thesis to exhibit a comparison analysis of the search algorithm approach by Sandholm . Besides, it is also important to know the limitation of the solution under a certain bidding environments which bidders probably act differently,. Therefore, this thesis provides discussion about the properties in a mechanism design to the solution method of WDP.

# Chapter IV

## Methodology

This chapter is the main contribution of this thesis. It explains the methodology in this research. First, it begins with the solution method for winner determination problem using the integer programming approach and the dynamic programming approach. After introducing the simplified case, it describes the method for extensional cases: multiple zones and physical obstructions.

### 4.1 Simplification of layout problem

In this section, the solution method for WDP is explained: integer and dynamic programming. The integer programming is a flexible methodology to solve WDP ; the computation complexity relies on the optimization solver. Note that the bidding option grows rapidly as the number of decision variables increase which incurs computation cost. On the other hand, a dynamic programming approach can alternatively be used to avoid an expensive calculation. While the efficient calculation is guaranteed, the disadvantage is the limitation to adopt various problem constraints in the solution step, unlike integer programming for a general WDP. The explanations of both methodologies are followed.

#### 4.1.1 Integer programming

Winner determination problem (WDP) is based on the assignment problem . The problem objective is to maximize the auctioneer's revenue. In the problem formulation, binary decision variables are used to indicate the optimal selection and allocation constraints.

There is a finite set of bidders,  $\mathcal{N}$ , with  $n$  bidders, and a finite set of indivisible objects,  $\mathcal{G}$ , with  $m$  distinct blocks. Each bidder  $i \in \mathcal{N}$  has a non-negative, integer valuation for each bundle of objects  $\mathcal{S} \subseteq \mathcal{G}$  denoted by  $b_i(\mathcal{S}) \in \mathbb{Z}_0$ . The binary decision variables are defined by  $x_i(\mathcal{S}) \in \{0,1\}$ ;  $x_i(\mathcal{S}) = 1$  means that the bundle  $\mathcal{S}$  is allocate to bidder  $i$  and otherwise; it will not allocate to this bidder  $i$ .

The possible bundle is illustrated as  $\mathcal{S}=[a,b]$ :  $a$  is the begin block position and  $b$  is the end block position. For example, [S2,S2] means only one single block at S2 in the single line case. For the double line case, [R3,L1] is a rectangular six blocks from L1 to R3, and [L4,L2] is a bundle of three blocks on the left side from L2 to L4 consecutively. Thus, we can represent all possible options as index of column in a coefficient matrix – the decision variables.

$$\begin{aligned}
 \text{(IP1)} \quad & \max \sum_{i=1}^n \sum_{\mathcal{S} \in \mathcal{G}} b_i(\mathcal{S}) x_i(\mathcal{S}) \\
 & \sum_{i=1}^n \sum_{\mathcal{S} \in \mathcal{G}, \mathcal{S} \ni j} x_i(\mathcal{S}) \leq 1 \quad \text{for all } j \in \mathcal{G} \\
 & x_i(\mathcal{S}) \in \{0,1\}.
 \end{aligned}$$

WDP is formulated as in (IP1). The interpretation of each line is straightforward. The objective function is to maximize the revenue for all possible bidding options, and the constraints are to prevent duplicated allocation of each object. One bidder is allowed to have more than one object. Specifically, this WDP is in the *OR* bidding language defined in . The matrix  $\mathbf{A}_i$  is the coefficient matrix in the constraint for the bidder  $i$ , and the sparse structure of coefficients is observed. For example, considering the size of  $\mathcal{G}$  to be 4 is used to explain the structure of a coefficient matrix.

### Single line case

Let  $m$  be the number of row blocks,  $m=|\mathcal{G}|$ .

$$\begin{array}{cccccccccc}
 \text{S1S1} & \text{S2S1} & \text{S3S1} & \text{S4S1} & \text{S2S2} & \text{S3S2} & \text{S4S2} & \text{S3S3} & \text{S4S3} & \text{S4S4} & \leftarrow \mathcal{S}=[a,b] \\
 \mathbf{A}_i = & \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} & \begin{array}{l} \text{for all } i \text{ bidder} \\ \mathbf{A}_i \sim m \times 0.5m(m+1) \end{array} \\
 \mathbf{A} = & [\mathbf{A}_1 \quad \cdots \quad \mathbf{A}_i \quad \cdots \quad \mathbf{A}_n]
 \end{array}$$

### Double line case

$$\begin{aligned}
 \tilde{\mathbf{A}}_i &= \begin{bmatrix} \mathbf{A}_i & \mathbf{0} & \mathbf{A}_i \\ \mathbf{0} & \mathbf{A}_i & \mathbf{A}_i \end{bmatrix} \quad \text{for each } i \text{ bidder, } \tilde{\mathbf{A}}_i \sim 2m \times 1.5m(m+1) \\
 \tilde{\mathbf{A}} &= [\tilde{\mathbf{A}}_1 \quad \cdots \quad \tilde{\mathbf{A}}_i \quad \cdots \quad \tilde{\mathbf{A}}_n]
 \end{aligned}$$



According to the special structure of the layout booths, the coefficient matrices have the total unimodularity property<sup>9</sup>. For the sake of integral values in the right hand side and the total unimodularity, the integer formulation (IP1) is reducible to (LP1) which is equivalent to a linear programming model. This yields an integer outcome, in this case just 1 or 0. Hence, it is eligible to employ just linear programming methods, (LP1) and (LP2), instead of the integer programming model.

$$\begin{aligned}
 \text{(LP1)} \quad & \max \sum_{i=1}^n \mathbf{b}_i \mathbf{x}_i \\
 & \sum_{i=1}^n \mathbf{A}_i \mathbf{x}_i \leq \mathbf{1} \\
 & \mathbf{x}_i \geq 0
 \end{aligned}
 \qquad
 \begin{aligned}
 \text{(LP2)} \quad & \max \sum_{i=1}^n \tilde{\mathbf{b}}_i \tilde{\mathbf{x}}_i \\
 & \sum_{i=1}^n \tilde{\mathbf{A}}_i \tilde{\mathbf{x}}_i \leq \mathbf{1} \\
 & \tilde{\mathbf{x}}_i \geq 0
 \end{aligned}$$

#### 4.1.2 Dynamic programming

A dynamic programming can be used to solve the optimization problem . Optimality in dynamic programming can be proved by the mathematical induction as regarded in . We exert a two-dimensional array as in Fig. 4.1 and its index to construct a data structure. It is suitable to solve for the dynamic programming approach since the bundle notation corresponds to the index and easy to call and update the memory in this structure.

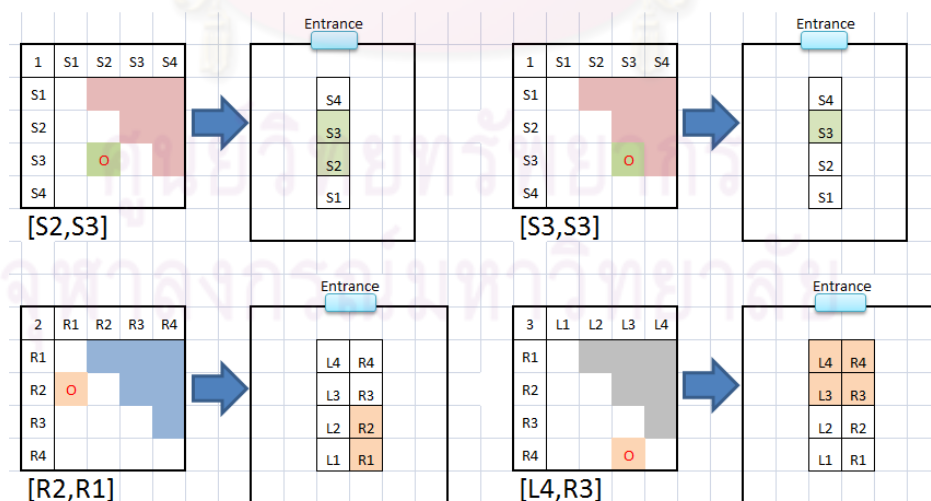


Figure 4.1: The two-dimensional array data structure corresponding to the bundle notation.

<sup>9</sup> More explanation of total unimodularity in Appendix B.

Furthermore, the method of Rothkopf et al which is related to the setup for the single line<sup>10</sup> is the third algorithm in . We shortly called RPH's method. Consequently, the new method proposed in this thesis is called PK method by the initial letter of the authors of. Although both methods are different in the step to update memory, the optimal value is obtained by the final comparison.

### A. RPH's method

RPH method, their algorithm is rewritten in pseudo code 1 and 2. RPH's complexity is  $\mathcal{O}(n^2)$  for the single line with a fixed start, say  $[1, n]$ . Pseudo code 1

demonstrates this method; specifically, it has  $\sum_{k=1}^n (k-1) = \frac{n(n-1)}{2}$  comparison works.

However, the method which is suitable to our problem is the intervals on the line. They interpreted as the intervals on the circle of which computational complexity is  $\mathcal{O}(n^3)$ . The reason is that it repeats each block to start again by renumbering the objects. The first round is  $[1, n]$ , then the second round is  $[2, n+1]$  which  $n+1$  refer to the first block, and so on. The first block connects to the end and the second block becomes a new fixed start. After that, we bring those  $n$  outcomes to contest for the most valuable solution. This method is displayed in pseudo code 2. Precisely, the count<sup>11</sup> is  $\frac{n^2(n-1)}{2} + n$ . The former part is to repeat the previous algorithm  $n$  time, and the latter is to compare the results of each round for the global maximum value.

**Pseudo code 1: Fixed start**

```

step 0  Input   $p(i, j)$  for all  $i, j$ 
step 1  Set    $w(1) = p(1, 1)$ . Set  $r = 2$ .
step 2  Set    $w(r) = p(1, r)$ .
step 3  For    $i = 2$  to  $r$ 
         If    $w(i-1) + p(1, r) > w(r)$ 
         Then  $w(r) = w(i-1) + p(1, r)$ 
step 4  If    $r < n$ , then set  $r = r + 1$  and go to step 2
         Otherwise, terminate with optimal revenue  $w(n)$ .

```

<sup>10</sup> In their manuscript, they call this case the consecutive asset in geometry-based structures.

<sup>11</sup> See the sketch proof in Appendix A.

**Pseudo code 2: Intervals on the circle**

```

step 0 Input   $p(i,j)$  for all  $i,j$ 
step 1 For  $k = 1$  to  $n$ 
    1.1 Set  $w(k) = p(k, k)$ . Set  $r = k + 1$ .
    1.2 Set  $w(r) = p(k,r)$ .
    1.3 For  $i = k + 1$  to  $r$ 
        If  $w(i - k) + p(k,r) > w(r)$ 
        Then  $w(r) = w(i - k) + p(k,r)$ 
    1.4 If  $r < n+1 - k$ , then set  $r = r+1$  and go to step 1.2
        Otherwise, get the  $k^{\text{th}}$  round optimality  $w(n+1-k)$ 
step 2  $w(\text{opt}) = \text{Max}\{ w(n), \dots, w(2n-1) \}$ .

```

To implement each algorithm, the bidding values must be sorted into a descendent order. The most valuable bid in each option becomes the first input in a square matrix of the algorithm. For a single line case, only one matrix is sufficient to keep the highest bid for every option. Then we can begin our algorithm to search for the optimal value which will be at the final element of the calculated matrix,  $[1,n]$ .

**B. PK's methods**

On the other hand, our modified methods for a single line case and a double line case are in pseudo code 3 and 4 respectively. Firstly, the single line case has computational complexity equivalent to  $\mathcal{O}(n^3)$ . Precisely, the work is counted as

$$\sum_{k=1}^n (k-1)(n-k+1) = \frac{n(n-1)(n+1)}{6}$$

which is less than the case of algorithm in pseudo

code 2. PK's method, algorithm in pseudo code 3, works more efficiently since we reap the benefit of the data structure more effectively than the RPH's method in pseudo code 2 which has to reorder the block position to move. Further that, our approach can easily apply to a double line case which has complexity in  $\mathcal{O}(n^3)$  as well, see pseudo code 4. The calculation burden is obviously three times of the single line case, since it works repetitively for the left, right and crossing line. Thus, we improve RPH's dynamic programming for both configurations.

To implement pseudo code 3 and 4, WDP is characterized and recursively defined by the two-dimensional array. Step 1 informs the stage of computation. Next, step 2 characterizes the maximum value referred to a related value from the previous stage. Subsequently, it leads to the maximum value in the final stage. In another word, the final result depends on comparisons of their substitutable pair that each

component, and also relies on the relevant pair backwardly. Those previous comparison results are put conveniently in the callable memory in our data structure. Certainly, the optimal value is obtained from the final calculation of the final element.

**Pseudo code 3: Single line algorithm**

```

step 0 Given square matrix array size  $n \times n$ , say  $\mathbf{B} \sim n \times n$ 
      Let  $k = 0$  be the diagonal line in the matrix;  $k+1$  is the consecutive lower
      diagonal line.
step 1 For  $k \geq 1$ 
      For  $j = 1$  to  $n$ 
         $w = j + k$ 
        If  $w \leq n$ 
          For  $k = j$  to  $(w-1)$ 
            If  $B(w,j) < B(k,j) + B(w,k+1)$ 
              Then  $B(w,j) = B(k,j) + B(w,k+1)$ 
step 2 Terminate when  $k = n$ .

```

**Pseudo code 4: Double line algorithm**

```

step 0 Run single column algorithm for the left and right column and keep result in the
      square matrix L and R respectively.
      Given a square matrix C for cross-side options value
step 1 Let  $k = 0$  be the diagonal line in the matrix;  $k+1$  is the consecutive lower
      diagonal line.
step 2 For  $k \geq 0$ 
      For  $j = 1$  to  $n$ 
         $w = j + k$ 
        If  $w = j$ 
          If  $C(j,j) < L(j,j) + R(j,j)$ 
            Then  $C(j,j) = L(j,j) + R(j,j)$ 
        If  $w \leq k$ 
          For  $k = j$  to  $(w-1)$ 
            If  $C(w,j) < C(k,j) + C(w,k+1)$ 
              Then  $C(w,j) = C(k,j) + C(w,k+1)$ 
step 3 Terminate when  $k = n$ .

```

In pseudo code 4, a double line case, the sequence is more complex. It is necessary to utilize three matrices for valuation inputs; i.e., left line, right line and crossing line. These three matrices represent the best price of each option. Next, we optimize for the single left line and the single right line. Moreover, there are more calls to compare the option crossing between left and right; however, the main idea, to compare bidding price from the lowest single level first and consecutively move to the biggest bundle later, is unchanged. Finally, the optimal value is located in the final

element of the crossing line matrix. Thus, with some adjustments in pseudo code 3 we can precede a double line booth auction as in pseudo code 4.

## 4.2 Extensions

In this section, two methods to solve WDP for the multi-block booth allocation are explained. It is straightforward to extend the integer programming method for the layout with multiple zones see Fig. 4.2, even the layout with obstruction in Fig. 4.3. The method is to nullify the coefficient of decision variables of which options must be excluded. Contrastingly, the extension of the problems in the dynamic programming method is required some additional techniques. In this section, we extend method for dynamic programming to cope with the realistic conditions, multiple zones and obstructions, which still maintain its computational advantage.

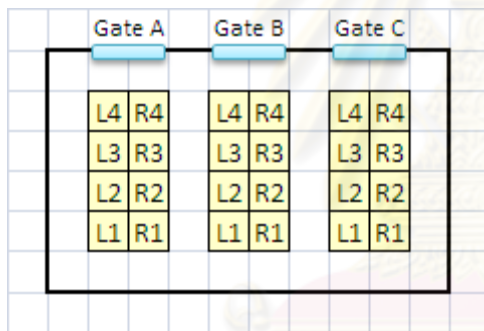


Figure 4.2: Layout with multiple zones.

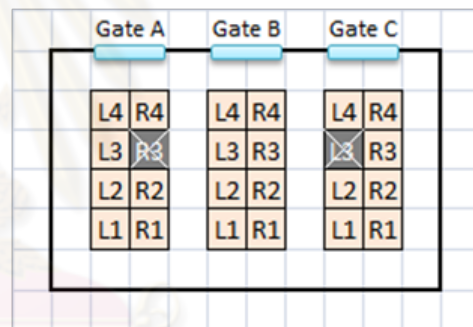


Figure 4.3: Layout with multiple zones and physical obstructions

### 4.2.1 Integer programming

To add some constraints in a mathematical model is straightforward. First, we determine allowable combinations and mark prohibited blocks. Next, we convert those logical statements into variables and inequality. Considering situation in Fig. 4.2 the layout with three multiple zones, we just construct decision variable of a selectable bundle. For example, if the bundle of block crossing over the multiple zones is not allowed, we can construct a decision variable as the following coefficient matrix  $\tilde{\mathbf{A}}_i = [\tilde{\mathbf{A}}_i^a \quad \tilde{\mathbf{A}}_i^b \quad \tilde{\mathbf{A}}_i^c]$ .  $\tilde{\mathbf{A}}_i^a$  represents 'zone  $a$ ', which hold the same structure as  $\tilde{\mathbf{A}}_i$  in the previous section. In the case of physical obstructions, we just eliminate

the decision variables of a bundle that contains such blocks. This treatment thus reduces the number of decision variables, which would be an advantage for mathematical programming, unlike the method in dynamic programming that still maintains full connection structure.

### 4.2.2 Dynamic programming

We consider multiple zones for a double line case, Fig. 4.2 and Fig. 4.3. There are three zones with the same amount of blocks. To apply the dynamic programming algorithm explained in pseudo code 4, we connect all three zones from left to the right to maintain the same structure as the case of a double line. Now it looks as if the bidder bids for the triple-long size of one zone; our proposed algorithm is enabling to solve this optimally. In fact, it is necessary to avoid the combination of options that crossing over the zone. Therefore, we ignore the final round calculation result in our algorithm but pick up the solution from our feasible options; just the substructures are adequate to yield the optimal outcome. In the case of Fig. 4.2, we have four rows, double columns and three zones. The solution comes from three elements of the final two-dimensional array,  $C(A4,A1)$ ,  $C(B4,B1)$  and  $C(C4,C1)$ , the left-low-corner elements. These are the maximum revenue from zone A, B and C respectively and the summation is the optimal value for the auctioneer. The data structure of this case is represented in Fig. 4.4.

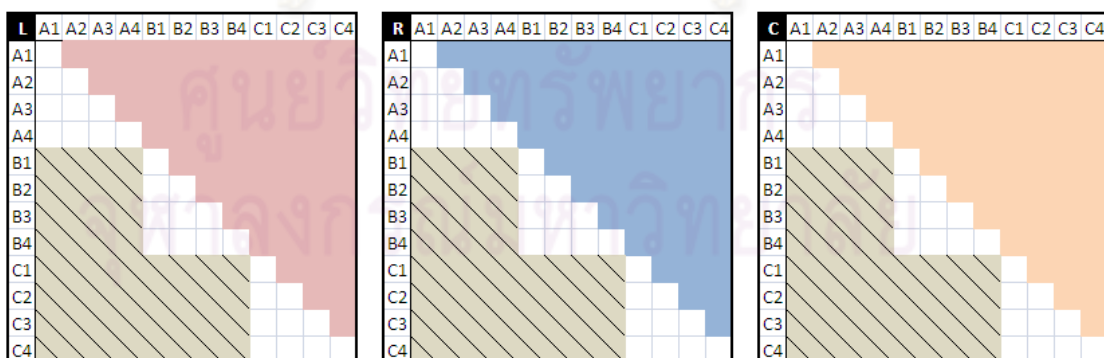


Figure 4.4: The illustrations of data structure corresponding to the layout in Fig. 4.2, the multiple zones

Next, it is possible to apply both multiple zones and obstructions together in our dynamic programming. The similar tactic from the previous case is still applicable to overcome these difficulties. The layout of Fig. 4.3 is an example that block R3 of

zone A and block L3 of zone C become the obstructions. First, we combine three zones into one and mark invalid element in the data structure which are prohibited options due to cross-over zone restriction. Moreover, the positions of obstruction blocks are marked in the data structure to avoid those impossible options. Fig. 4.5 illustrates both marking case. Subsequently, the options related to prohibited selection, the elements with diagonal line marker, and the obstruction-related blocks, the element with crossing marker, are set to zero. Especially in the matrix for the crossing-line options, it must reflect those influences from the left-line and right-line matrix to avoid prohibited blocks and impossible options, see Fig. 4.5. Finally, the answer for optimality are obtained from the final element which are the low-left corner,  $C(A2,A1)$ ,  $C(A4,A4)$ ,  $C(B4,B1)$ ,  $C(C2,C1)$ , and  $C(C4,C4)$  to be combined.

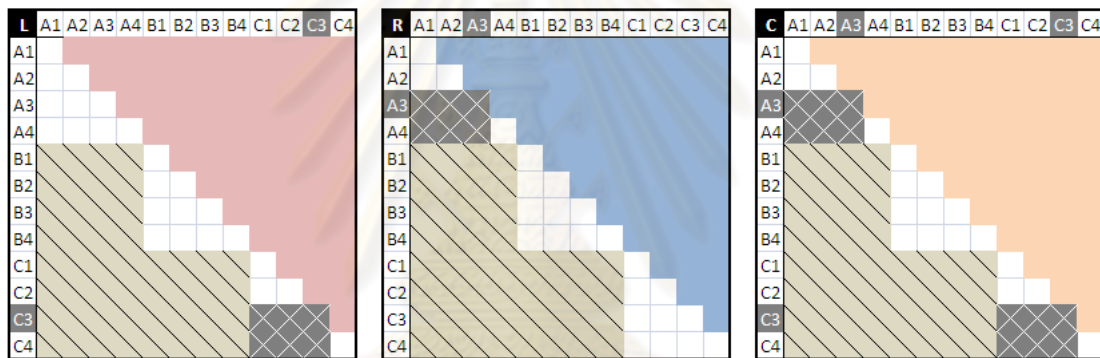


Figure 4.5: The illustrations of data structure corresponding to the layout in Fig. 4.3, the multiple zones with obstructions

# Chapter V

## Experiment and analysis

Simulation experiments and analysis of our results are summarized in this chapter. To demonstrate the advantage of our method, we report the experimental results comparing with other algorithms. For the short notation, the Rothkopf et al method is noted as RPH method and our method is noted by PK method. The main result is that the dynamic programming approach consumes less time than the integer programming approach, and PK method is faster than RPH method.

### 5.1 Simulation environments

The experimental environments, parameter setting and simulation results are described in this section. First, the simulation experiments are performed on Intel<sup>R</sup> Core<sup>TM</sup> i3 Processor 3.07 GHz CPU with RAM 2 GB. The operating system is Windows 7. Linear programming solver and the other algorithmic codes are implemented by Matlab. As well, bidding values vector for each option are generated by the method explain in section 5.2, in Matlab internal pseudorandom environment.

The time measurement is recorded from their actual jobs. The solver for the linear programming model is the simplex method, and the time is measured soon after the optimal solution revealed. For the dynamic programming, the running time in the preprocess, to select the best offer for each option, is included and add with the computation time of the comparison process.

### 5.2 Random sample simulation process

To avoid the selection bias and the unrealistic sample, the simulation must maintain two assumptions. One is a bundling assumption: the more combining blocks, the greater valuation – supermodular property. The other is a position advantage: the nearer the gate, the more expected benefit. See Fig. 4.1 and Fig. 4.2 to conceptualize the image. As well, the random number of bidding price vector is converted to the ceiling integer number to represent each individual evaluation.

First, we generate the random variables for an individual block from uniform distribution between zero and one. According to the assumption that the nearer the



gate the more expected benefit, it is necessary to construct ordered statistics distribution from the uniform distribution. The standard beta distribution,  $Beta(\alpha, \beta)$ , can bring about the random number generation in shorter time;  $n$  is total number of block and  $j$  the position, then  $X_{(j)} \sim Beta(j, n+1-j)$ . For example, if we have 4 blocks, the value for the block label 1 is drawn from the beta distribution with the following parameter,  $X_{(1)} \sim Beta(1, 4)$ .

In addition to generate a random value for a consecutive-blocks bundle, we add generate the random number from the summation of random number of the corresponding block. For example, the value for the bundle [S3,S1] is obtained by generating  $X_{(1)} + X_{(2)} + X_{(3)}$ . Also we impose the condition on the value for the bundles that the more combining blocks, the greater valuation. Moreover, the sub-additivity and the super-additivity condition could be applied into effect. These condition leads to tremendous time consumption in generating random valuation for  $n > 5$ .

In the case of a double line auction, the process to generate the random number for bidding price is similar. First, we generate for a single left row and right row. Next we combine both values for the crossing options and add a small value to increase the bid price to preserve the super modularity price structure in the simulated bids. Each case of a generated price is stored in a different matrix data structure according to pseudo code 6.

Finally, the generated value is multiply by a thousand and rounded up to obtain integer valuation. The generated values are stored in a ready-to-use data structure and will be randomly drawn again to use as a sample in the simulation.

### 5.3 Comparisons of algorithms running time

There are two cases of simulation as the following: a single line case and a double line case. The single line case is the comparison among three methods: linear programming relaxation (LP), RPH and PK. On the other hand, the double line case is the comparison between two methods: LP and PK. To grasp the basic performance of each algorithm, the case of 10 bidders and corresponding with the number of blocks, the average running time from the 1,000 simulation experiments are shown in Table 5.1 and Table 5.3. Next, we also investigate when the number of blocks and bidders

increase progressively, Table 5.2 and Table 5.4. In such situations, the problem size is enlarged in the direction of variables and memory usages in calculation for each method. The experiments are carried out under the limited computing resource as mentioned in section 5.1.

### 5.3.1 Single line case

The experimental results of a single line case is obvious that the dynamic programming approach, RPH's and PK's method, are further effective than linear programming approach.

#### A. Basic simulation results

The line plot in Fig. 5.1 also shows the advantage of PK's method over the linear programming and RPH's method. The average running time in our method is less than RPH's method, statistically significant at the 0.01 by the two-tailed t-test. This simulation result consequently supported our theoretical computation in the methodology.

Table 5.1: The average running time of each algorithm for the single line case

unit: micro seconds

No. of blocks:	1	2	3	4	5	6	7	8	9	10
Lin Prog	6,917	6,117	6,608	7,205	7,483	8,208	8,905	9,733	10,597	11,772
RPH	120	110	113	116	167	135	155	164	186	211
PK	60	59	59	61	79	66	72	74	80	88
t-statistics	28.90	18.55	15.65	20.67	14.23	39.22	27.27	28.90	52.35	26.72
p-value	0	0	0	0	0	0	0	0	0	0

#### Single line case

Average time in log scale

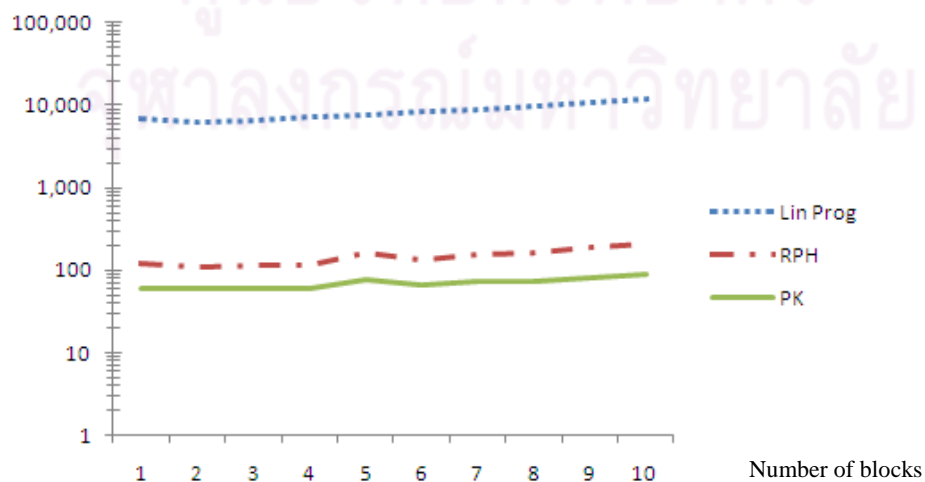


Figure 5.1: The comparison of average time among three methods (single line)

The box plot in Fig. 5.2 displays the scatter of the running time results from the experiments. The trend is that the more blocks, the more running time. It is noticeable in the linear programming, but it is not very obvious in the dynamic programming in both RPH's and PK's method. It seems that the running time results of RPH's and PK's method do not change much in these controlled experiments, 10 bidders.

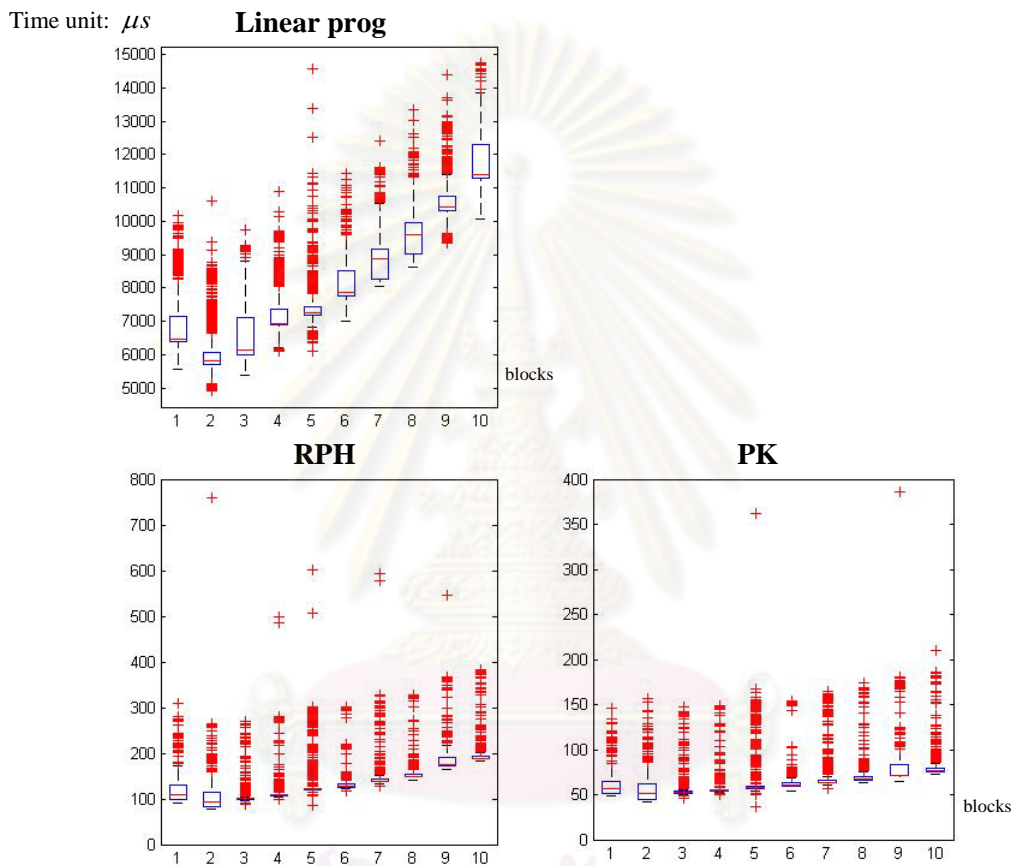


Figure 5.2: The distribution of running time among three methods (single line)

## B. Incremental simulation results

The results from the experiments clearly show that LP suffers the most when the number of bidders and blocks increase. In Table 5.2, it displays that LP cannot operate at all when the number of bidders reach 10 for 70 blocks single line auction. Consequently, it gets worse in the situation of 50 bidders and 100 bidders that it cannot work for 40 blocks and for 20 blocks, respectively.

On the other hand, the dynamic programming approach performs well on each situation control. When considering for the average running time, PK's method performs faster than RPH's method. As well, the box plots in Fig. 5.3 illustrate that

Table 5.2: The average running time for the single line case when the number of bidders increases

**Single line case** unit: micro seconds

		Number of bidders								
		10			50			100		
Number of blocks		Linear Prog	RPH	PK	Linear Prog	RPH	PK	Linear Prog	RPH	PK
	10	10,898	216	86	31,297	282	165	56,425	378	259
	20	38,397	463	170	189,533	691	313	N.A.	713	440
	30	120,184	1,021	328	708,492	2,308	1,588	N.A.	2,096	1,434
	40	320,774	2,201	592	N.A.	3,797	2,498	N.A.	3,506	2,188
	50	693,506	4,129	1,789	N.A.	4,943	2,635	N.A.	5,334	3,013
	60	1,271,899	5,478	1,697	N.A.	7,030	3,303	N.A.	7,875	4,131
	70	N.A.	8,353	2,745	N.A.	9,484	3,815	N.A.	11,009	5,339
	80	N.A.	12,161	4,109	N.A.	13,278	5,118	N.A.	14,998	6,835
	90	N.A.	16,464	5,286	N.A.	18,027	6,681	N.A.	19,959	8,520
100	N.A.	21,099	5,987	N.A.	23,378	8,023	N.A.	25,301	9,948	

Remarks: N.A. means the data is not available due to the unsuccessful performance of the algorithm.

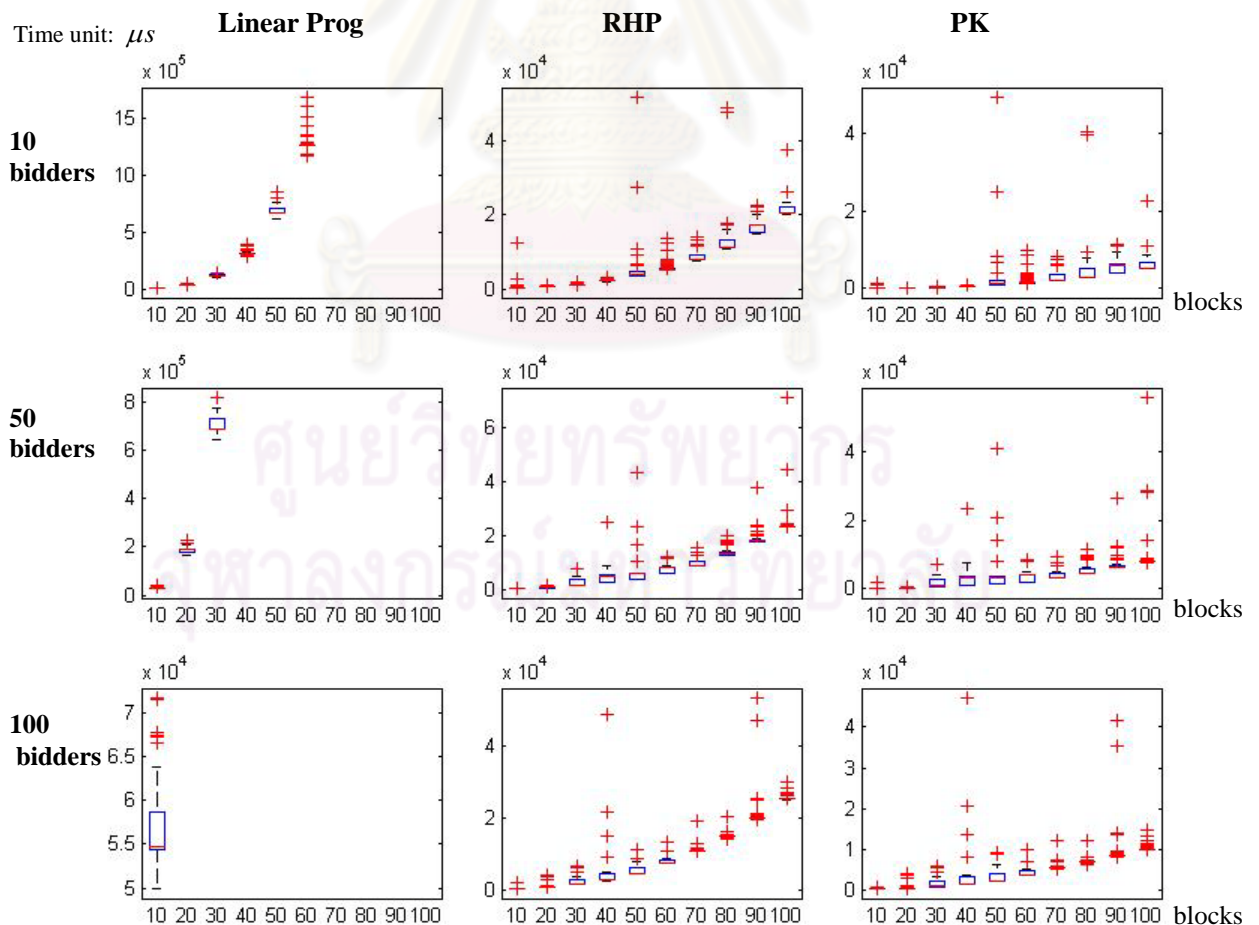


Figure 5.3: The distribution of running time for each algorithm for the single line case when the number of bidders increases

the maximum running time of PK's method likely lie below the minimum running time of RPH's method. Therefore, it could be concluded that PK's method is the most efficient among three methods in the experiment for the single line case.

### 5.3.2 Double line case

In the double line case, the result is consistent to the single line case. The dynamic programming approach hereafter refers to the authors' method only since RPH's method is not adaptable to the double line case.

#### A. Basic simulation results

From the result in Table 5.3 and box plots, the average time operated in the dynamic programming approach is far superior to the linear programming approach. The first reason is the structure of coefficient matrix in the linear programming bears too many feasible solutions. In addition, iteratively solving for the linear programming approach is very costly because it has to update sparsely matrix .

Table 5.3: The average running time of each algorithm for the double line case  
unit: micro seconds

No. of blocks	1	2	3	4	5	6	7	8	9	10
Linear Prog	7,006	7,329	8,297	9,641	11,196	13,237	15,690	18,896	22,882	27,784
PK	115	120	128	131	138	154	169	193	207	256

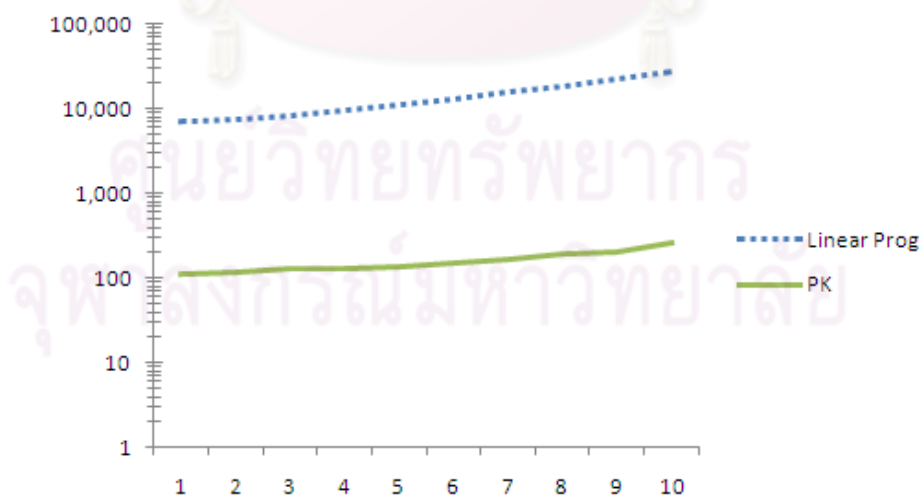


Figure 5.4: The comparison of average time between two methods (double line)

The box plots displayed in Fig. 5.5 show more details to our simulation results. The vertical axis is the running time in micro seconds and the horizontal axis is the total number of bidders in the auction. It is obvious that the running time

monotonically increase when the number of bidders grows up for both algorithms. The dynamic programming shows some outliers that may need to be investigated more.

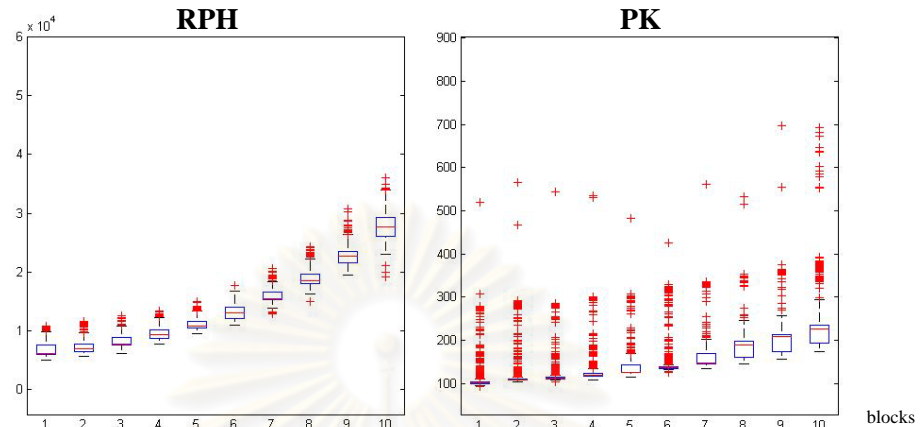


Figure 5.5: The distribution of running time among three methods (double line)

## B. Incremental simulation results

The experiment results of the double line case are similar to the single line case. The LP performance is the most incompetent for the greater number of bidders and blocks. Fig. 5.5 demonstrates that LP cannot work from the case of 10 bidders and 40 blocks in the single line auction. Further that, the performances in the case of 50 bidder and 100 bidders suggest the handicap of LP method: The operations are incomplete from 20 blocks and from 10 blocks respectively.

Although PK's method performance is acceptable in most cases, there is a signal of its weakness in the case of 100 bidders and 80 blocks. The reason is similar to LP's weakness that the requirement of memory allocation prior to the main algorithm is not sufficient. From Fig. 5.6, it suggests that the trend of PK's method is still efficient unlike the LP's method which consumes much more computing resources and yet slower.

Table 5.4: The average running time for the double line case when the number of bidders increases

**Double line case** unit: micro seconds

		Number of bidders					
		10		50		100	
		Linear Prog	PK	Linear Prog	PK	Linear Prog	PK
<b>Number of blocks</b>	<b>10</b>	26,866	256	117,726	425	257,550	2,032
	<b>20</b>	166,863	484	N.A.	1,744	N.A.	2,337
	<b>30</b>	675,006	1,783	N.A.	3,009	N.A.	3,855
	<b>40</b>	1,865,431	2,267	N.A.	4,307	N.A.	5,745
	<b>50</b>	N.A.	4,356	N.A.	6,031	N.A.	8,664
	<b>60</b>	N.A.	6,016	N.A.	8,506	N.A.	11,607
	<b>70</b>	N.A.	8,187	N.A.	12,123	N.A.	16,027
	<b>80</b>	N.A.	10,422	N.A.	15,448	N.A.	21,748
	<b>90</b>	N.A.	14,229	N.A.	19,597	N.A.	N.A.
	<b>100</b>	N.A.	17,736	N.A.	25,454	N.A.	N.A.

Remarks: N.A. means the data is not available due to the unsuccessful performance of the algorithm.

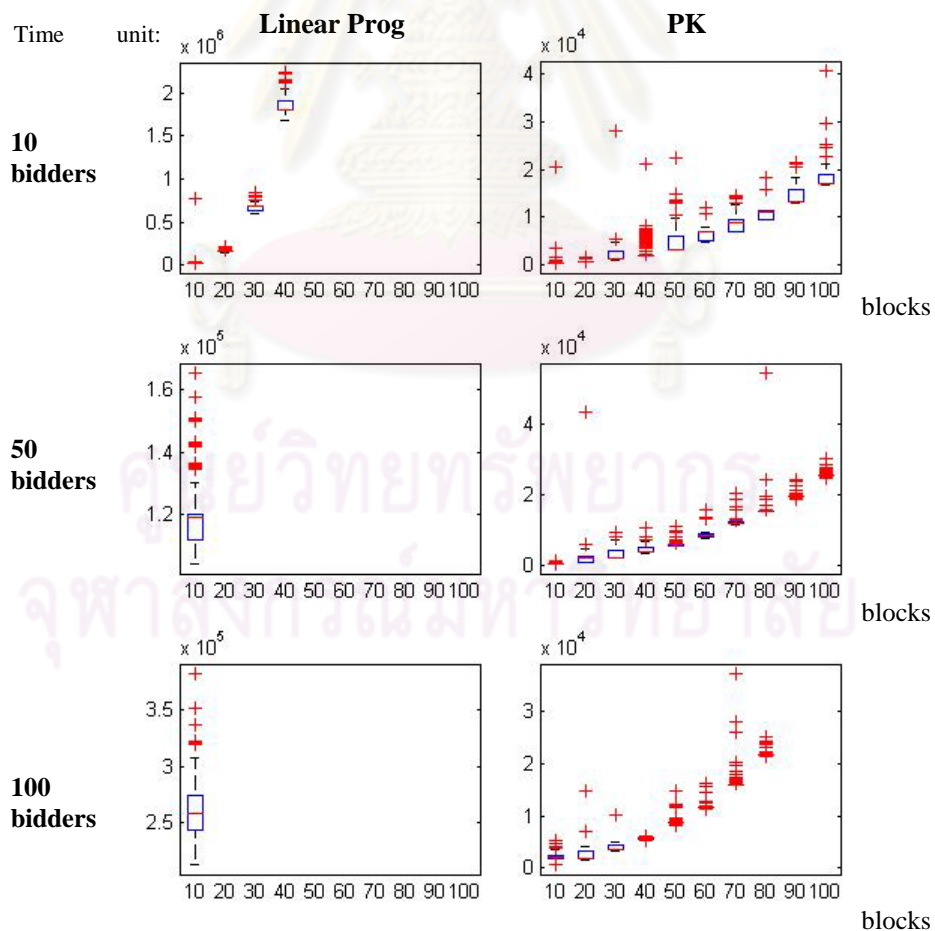


Figure 5.5: The distribution of running time for each algorithm for the double line case when the number of bidders increases.

## 5.4 Discussion of simulation results

The evidence that the integer programming solved by the LP relaxation is slower than the dynamic programming approach has been obtained from the experimental results of both the single line case and the double line case. The major reason is that the simplex itself work slowly in our large coefficient matrix of WDP for a booth auction, because the size of matrix in the simplex boost the more computation time, . Moreover, the number of bidders multiplies the size of decision variables in the integer programming also leads to more feasible solutions which require a heavy load for the simplex in LP.

For the comparisons of the dynamic programming algorithm, it is shown that the average running time of PK's method is significantly different and smaller than RPH's method, by two-sample t-tests for a difference in mean. In the box plot, there could be some cases that RPH's running time is equivalent to PK's when the number of blocks is smaller than 10. However, the comparison of running time in Table 5.2 and the box plots of Fig. 5.3 exhibit the robust tendency that PK's method is superior to RPH's method. Thus, the theoretical estimation of the number of work, demonstrated in chapter 4, is supported by the experiments.

The running time resulted in the experiments is measured at micro seconds level for every controls. It is not a great burden in practice to use any of the methods. However, it was found that the integer programming consumes the great number of memory resources and may halt during the coefficient matrix preparation process. On the other hand, the dynamic programming approach is still workable to optimize the solution. Although the economical memory usage of the dynamic programming algorithm is another advantage shown in the experiments, it is necessary to improve the memory usage to operate PK's method in a limited computing environment.



# Chapter VI

## Conclusion

This thesis has developed methods to solve the winner determination problem (WDP) in a combinatorial auction for a multi-block booth allocation. It is much influenced by Rothkopf et al who proposed the integer programming and the dynamic programming algorithm. This thesis also explains the detailed structure in the integer programming related to the model of a booth auction and linear programming relaxation for integer programming.

By the worst case analysis of the computation complexity, it is shown that the new method in this thesis, PK method, is faster than the method of Rothkopf et al , RPH method. As well, the experimental results suggest that the dynamic programming performance is more superior to the integer programming performance in our setting environment. When changing the quantity of blocks or bidders, the results reinforce this tendency.

In addition, another contribution of this thesis is to model a double line booth auction and to develop an algorithm for WDP of a double line booth auction. Thank to the merit of the matrix data structure for a dynamic programming algorithm, the algorithms for a double line booth auction maintains the same complexity as the case of a single line auction.

Furthermore, this thesis extends geometry-based structures of the booth allocation with restriction of multiple zones and obstructions. Not only the integer programming model is well-known for its flexibility, but our proposed algorithm by the dynamic programming is also extensible to overcome these restrictions due to the data structure, two-dimensional array. This data structure can control state and renew memory for each stage effectively.

The future work of this thesis is discussed on the ground of the limitation of this work. This work is developed just algorithms to solve WDP by the auctioneer stand point, not yet completed the full auction design. From the game theoretical aspect and the mechanism design, it is necessary to consider strategic behavior bidders which might report untruthfully bid price. This strategic behavior will reduce

the auctioneer revenue. Therefore, the complete design for a booth auction should consider this mechanism.

In the aspect of algorithms design, PK's method is very useful for superadditive valuation of each bidder and guarantees optimal solution in WDP for the auctioneer. However, the solution might be unjustified in the case: the same bidder submits the bids that value in subadditive and become the maximum value for a bundle and all substructure options. Moreover, this algorithm is weak in a more complex bidding language that proposed by Nissan . For example, OR-of-XOR offer bidding condition to select the exact preference for the bidders.

Last but not least, the implication of our proposed algorithm would not be limited to the allocation of space. The other similar bundles bidding which have a specific directional relation would be solved by this algorithm.



ศูนย์วิทยทรัพยากร  
จุฬาลงกรณ์มหาวิทยาลัย

## References

- [1] V. Krishna, Auction Theory, 2nd ed. Amsterdam: Elsevier, 2010.
- [2] Shoham, Y. and Leyton-Brown, K., Multiagent systems : algorithmic, game-theoretic, and logical foundations. Cambridge: Cambridge University Press, 2008.
- [3] Lehmann, D., Müller, R. and Sandholm, T., The Winner Determination Problem. In Y. Shoham and R. Steinberg P. Cramton (ed.), Combinatorial Auctions, pp. 297-317, Cambridge, Massachusetts: The MIT Press, 2006.
- [4] Rothkopf, M.H., Pekeč, A. and Harstad, R.M. Computationally Manageable Combinational Auction. Management Science 44 (August 1998): 1131-1147.
- [5] Nisan, N. and Ronen, A. Computationally Feasible VCG Mechanisms. Journal of Artificial Intelligence Research 29 (May 2007): 19-47.
- [6] Tennenholtz M. Some Tractable Combinatorial Auctions. In the Seventeenth National Conference on Artificial Intelligence. Austin, 2000, pp. 98-103.
- [7] Müller, R. Tractable Cases of the Winner Determination Problem. In Y. Shoham and R. Steinberg P. Cramton (ed.), Combinatorial Auctions, pp 319-336, Cambridge, Massachusetts: The MIT Press, 2006.
- [8] Ausubel, L.M. and Milgrom, P.R. Ascending Auctions with Package Bidding. Frontiers of Theoretical Economics 1 (August 2002): 1-44.
- [9] Holte, R.C. Combinatorial Auction, Knapsack Problems and Hill Climbing Search. In Canadian Conference on AI, 2001, pp. 57-66.
- [10] Parkes, D.C. and Ungar, L.H. Iterative Combinatorial Auctions: Theory and Practice. In Seventeenth National Conference on Artificial Intelligence and Twelfth Innovative Applications of Artificial Intelligence Conference, Menlo Park, CA, 2000, pp. 74-81.
- [11] Nisan, N. Bidding Languages for Combinatorial Auctions. In Y. Shoham and R. Steinberg P. Cramton (ed.), Combinatorial Auctions, pp. 215-231, Cambridge, Massachusetts: The MIT Press, 2006.
- [12] Federal Communications Commisison. About Auctions [Online]. 2011. Available from: [http://wireless.fcc.gov/auctions/default.htm?job=about\\_auctions](http://wireless.fcc.gov/auctions/default.htm?job=about_auctions) [2011, April]
- [13] Mas-Colell, A., Whinston, M.D. and Green, J.R. Microeconomic theory. New York: Oxford Academic Press, 1995.

- [14] Ausubel, L.M. and Milgrom, P., The Lovely but Lonely Vickrey Auction In Y. Shoham and R. Steinberg P. Cramton (ed.), Combinatorial Auctions, pp. 18-40, Cambridge, Massachusetts: The MIT Press, 2006.
- [15] Ahuja, R.K., Magnanti, T.L., and Orlin, J.B. Network flows : theory, algorithms, and applications. New Jersey: Prentice-Hall International, 1993.
- [16] Sandholm, T. Algorithm for Optimal Winner Determination in Combinatorial Auctions. Artificial Intelligence 135 (February 2002): 1-54.
- [17] Sandholm, T. Optimal Winner Determination Algorithms. In Y. Shoham and R. Steinberg P. Cramton (ed.), Combinatorial Auctions, pp. 337-368, Cambridge, Massachusetts: The MIT Press, 2006.
- [18] Parkes, D.C. Iterative Combinatorial Auctions: Achieving Economic and Computational Efficiency. Ph.D. Desertation, Department of Computer and Information Science, University of Pennsylvania, 2001.
- [19] Parkes, D.C. and Shneidman, J. Distributed implementations of Vickrey-Clarke-grooves Mechanisms. In 3rd International Joint Conferenceon Autonomous Agents and Multi Agent Systems, New York, NY, 2004, pp. 261-268.
- [20] Petcu, A., Faltings, B. and Parkes, D.C. M-DPOP: Faithful Distributed Implementation of Efficient Social Choice Problems. Journal of Artificial Intelligence Research 32 (July 2008): 706-755.
- [21] Feldman, J., Muthukrishnan, S., Nikolova, E. and Pál, M. A Truthful Mechanism for Offline Ad Slot Scheduling. In B. Monien and U. Schroeder (ed.), Lecture Note in Computer Science, pp. 182-193. Heidelberg: Springer-Verlag, 2008.
- [22] Edelman, B., Ostrovsky, M. and Schwarz, M., Internet Advertising and the Generalized Second-Price Auction: Selling Billions of Dollars Worth of Keywords. American Economic Review 97 (March 2007): 242-259.
- [23] Andersson, A., Tenhunen, M. and Ygge, F. Integer Programming for Combinatorial Auction Winnner Determination. In 4th International Conference on Multi-Agent Systems, Boston, MA, 2000, pp. 39-46.
- [24] Bikhchandani, S. and Ostroy, J., From the Assignment Model to Combinatorial Auctions. In Y. Shoham and R. Steinberg P. Cramton (ed.), Combinatorial Auctions, pp. 189-214, Cambridge, MA: The MIT Press, 2006.
- [25] Schrijver, A. Theory of Linear and Integer Programming. Chichester: John Willey & Sons, 1986.
- [26] Sandholm, T. Approaches to winner determination in combinatorial auctions. Decision Support Systems 28 (March 2000): 165-176.

- [27] Manber, U. Introduction to Algorithms: A Creative Approach. Reading, Massachusetts: Addison-Wesley, 1989.
- [28] Sariddichainunta, P. and Sinapiromsaran, K. The Winner Determination Model and Computation for Linear Arrangement of Booth Auction. In 7th International Conference on Computing and Information Technology, Bangkok, Thailand, 2011, pp. 74-78.
- [29] Sariddichainunta, P. and Sinapiromsaran, K. The Winner Determination Problem in Combinatorial Auction for Booth Allocation: Algorithms and Extensions in Booth Layout Problem. In 2011 International Conference on computer Control and Automation, Jeju, Korea, 2011, pp. 739-743.
- [30] Bazaraa, M.S., Jarvis, J.J. and Sherali, H.D., Linear Programming and Network Flows, 3rd ed. New York: Willey, 2004.
- [31] Seymour, P.D. Decomposition of regular matroids. Journal of Combinatorial Theory, Series B 28 (June 1980): 305-359.



ศูนย์วิทยทรัพยากร  
จุฬาลงกรณ์มหาวิทยาลัย



Appendices

ศูนย์วิทยทรัพยากร  
จุฬาลงกรณ์มหาวิทยาลัย

## Appendix A

### Sketch proofs of counting

This appendix will provide proof of the enumeration of bidding options in chapter 2 and the comparison works of the algorithms in chapter 4.

**Claim 1:** The number of possible bundles in a line booth auction is  $\frac{n(n+1)}{2}$  bundles for  $n$  is the number of blocks.

**Proof**

Let  $P(n)$  be the number of possible options,  $P(n) = \frac{n(n+1)}{2}$ .

*Basic step:* Let  $n=1$ . When we have one block, it is obvious that we have one option. So  $P(1)=1$  is true.

*Induction step:* Assume that  $P(k)$  is true, then  $P(k) = \frac{k(k+1)}{2}$ .

When  $n = k + 1$ , there are more  $k + 1$  possible options additionally.

$$\frac{k(k+1)}{2} + (k+1) = (k+1)\left(\frac{k}{2} + 1\right) = \frac{(k+1)(k+2)}{2}.$$

As a result,  $P(k+1)$  is true by the mathematical induction. □

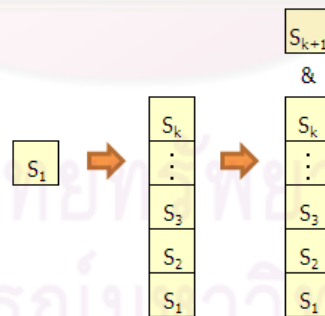


Figure A.1: The induction flow of the selectable options, the single line case.

**Claim 2:** The number of possible bundles in a double line booth auction is  $\frac{3n(n+1)}{2}$

bundles for  $n$  is the number of row blocks.

**Proof**

Let  $P(n)$  be the number of possible options,  $P(n) = \frac{3n(n+1)}{2}$ .

*Basic step:* Let  $n=1$ . When we have one row block, there are three possible bidding options, L1, R1 and L1R1. So  $P(1) = 3$  is true.

*Induction step:* Assume that  $P(k)$  is true, then  $P(k) = \frac{3k(k+1)}{2}$ .

When  $n = k + 1$ , we consider the additional possible options from left column, right column and both columns. Each case contributes more  $k + 1$  possible options.

$$\frac{3k(k+1)}{2} + 3(k+1) = 3(k+1)\left(\frac{k}{2} + 1\right) = \frac{3(k+1)(k+2)}{2}.$$

As a result,  $P(k + 1)$  is true by the mathematical induction. □

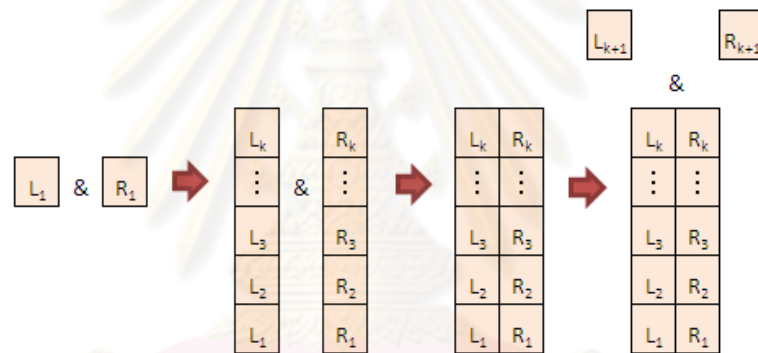


Figure A.2: The induction flow of the selectable options, the double line case.

**Claim 3:** The number of jobs in the pseudo code 1 (fixed start) is  $\sum_{k=1}^n (k-1) = \frac{n(n-1)}{2}$ ,

for  $k$  is the number of stages and  $n$  is the number of blocks.

**Proof**

In the pseudo code 1, there is no comparison in stage 1 since it is the initial step.

Next, in stage 2 there are one comparison; so that, the total number of jobs becomes 3.

Assume that in stage  $t$  there are  $t - 1$  comparisons.

The total jobs is calculated as the following summation,  $\sum_{k=1}^t (k-1) = \frac{t(t-1)}{2}$ .

When there is an additional block the comparison increase  $t$  jobs, because the possible options increase  $t$  options.



Therefore, the total jobs become  $\sum_{k=1}^t (k-1) + t = \frac{t(t+1)}{2}$

By the mathematical induction, we conclude that the number of job is  $\frac{n(n-1)}{2}$ .  $\square$

**Claim 4:** The number of jobs in the pseudo code 2 (interval in the circle) is  $\frac{n^2(n-1)}{2}$ ,

for  $n$  is the number of block.

*Proof*

It is obvious that the number of job in pseudo code 2 is the  $n$  times repetition of pseudo code 1.  $\square$

**Claim 5:** The number of jobs in pseudo code 3 is  $\sum_{k=1}^n (k-1)(n-k+1) = \frac{n(n-1)(n+1)}{6}$ ,

a single line case, for  $k$  is the number of stage and  $n$  is the number of block.

*Proof*

In stage 1, there is no comparison job.

Next, in stage 2 there are one comparison job for each two-consecutive bundle,  $(n-2+1)$  possible options. Then, the work load  $(2-1) \times (n-2+1)$ .

Thus, the basis step is true.

Consequently, the number of jobs increases for each stage is  $(k-1)(n-k+1)$ , and the total comparison jobs is the summation of that increment; i.e.,

$\sum_{k=1}^n (k-1)(n-k+1) = \frac{n(n-1)(n+1)}{6}$ . This is induction step.

When there is an additional blocks, the incremental of work load from stage 1 to stage  $n+1$  is  $0, 1, \dots, n$  respectively.

Then, the additional comparisons work load is  $\sum_{k=1}^{n+1} (k-1) = \frac{n(n+1)}{2}$ .

Hence, the total comparison for  $n+1$  block is  $\frac{n(n-1)(n+1)}{6} + \frac{n(n+1)}{2} = \frac{n(n+1)(n+2)}{6}$ .

By the mathematical induction, we conclude that the number of job is  $\frac{n(n-1)(n+1)}{6}$ .  $\square$

**Claim 6:** The number of jobs in pseudo code 4 (double line case) is  $\frac{n(n^2+1)}{2}$ , for  $n$  is the number of row block.

**Proof**

It is obvious that the number of jobs in the pseudo code 4 is the 3 times repetition of single line case for left column, right column and both columns.

Furthermore in stage one of the crossing block, it incurs  $n$  time comparison jobs.

Hence, the summation of actual jobs become  $\frac{3n(n-1)(n+1)}{6} + n = \frac{n(n^2+1)}{2}$ . □



## Appendix B

### Total unimodularity

This appendix is written summaries of other related works, based on the book of Schrijver (1990), “Theory of linear and integer programming”, Chapter 19.

**Definition:** Unimodularity matrix (UM)

A square integer matrix  $\mathbf{B}$  is unimodular (UM) if its determinant is 1 or 0 or  $-1$ .

**Definition:** Total unimodularity property (TUM)

An integer matrix  $\mathbf{A}$  is called totally unimodular (TUM) if every square, nonsingular submatrix of  $\mathbf{A}$  is UM.

**Hoffman and Kruskal’s Theorem**

Let  $\mathbf{A}$  be an integral matrix. Then  $\mathbf{A}$  is total unimodular, if and only if for each integral vector  $\mathbf{b}$  the polyhedron  $\{\mathbf{x} \mid \mathbf{x} \geq \mathbf{0}; \mathbf{Ax} \leq \mathbf{b}\}$  is integral.

**The preservation of TUM**

Seymour's decomposition theorem<sup>12</sup> for totally unimodular matrices.

Total unimodularity is preserved under the following operation

- 1) Permuting rows or column;
- 2) Taking the transpose;
- 3) Multiplying a row or column by  $-1$ ;
- 4) Pivoting;
- 5) Adding all-zero row or column, or adding a row or column with one nonzero, being  $\pm 1$ ;
- 6) Repeating a row or column.

Moreover, by Seymour’s characterization, total unimodularity is preserved under the following conditions:

---

<sup>12</sup> P.D. Seymour, “Decomposition of regular matroids,” *Journal of Combinatorial Theory Series B*, vol. 28, issue 3, pp. 305–359, June 1980.

$$\begin{aligned}
(1\text{-sum}) \quad \mathbf{A} \oplus_1 \mathbf{B} &:= \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} \end{bmatrix} \\
(2\text{-sum}) \quad [\mathbf{A} \ a] \oplus_2 \begin{bmatrix} b \\ \mathbf{B} \end{bmatrix} &:= \begin{bmatrix} \mathbf{A} & ab \\ \mathbf{0} & \mathbf{B} \end{bmatrix} \\
(3\text{-sum}) \quad \begin{bmatrix} \mathbf{A} & a & a \\ c & 0 & 1 \end{bmatrix} \oplus_3 \begin{bmatrix} 1 & 0 & b \\ d & d & \mathbf{B} \end{bmatrix} &:= \begin{bmatrix} \mathbf{A} & ab \\ dc & \mathbf{B} \end{bmatrix}
\end{aligned}$$

**Proposition 1:** The constraint coefficient matrix structure in the integer programming for a single line booth auction has total unimodularity property.

*Proof*

Let  $\mathbf{A}$  be a constraint coefficient matrix for the single line case with  $n$  blocks. From Hoffman and Kruskal's Theorem, it is sufficient to show that the basic feasible solutions (BFS) of  $\mathbf{A}$  have integral solution. This is also equivalent to prove that the square matrix of matrix  $\mathbf{B} = [\mathbf{A}_i | \mathbf{S}]$  which is basic feasible solutions have determinant value 1, 0, -1. Therefore, TUM property holds.

Let  $\mathbf{B} = [\mathbf{A}_i | \mathbf{S}]$ , which  $\mathbf{S}$  contains coefficients of additional slack variables. Then we consider all basic feasible solutions that maintain consecutive property. It is the square matrix obtained by the combination of column in of  $\mathbf{B}$  as the following  $n$  cases:

Case 1 the feasible solutions are the columns represent the bundle containing all blocks and slack variables column; e.g.,

$$\begin{bmatrix} 1 & 0 & 1 & \dots & 0 \\ 1 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 1 & 0 & 0 & \dots & 1 \\ 1 & 0 & 0 & \dots & 0 \end{bmatrix}$$

Obviously, this matrix can be converted to triangular matrix by row operation of which the diagonal element contains 1. Therefore, the determinant of this matrix is 1.

Case 2 the feasible solution are the set of two bundles and slacks variable column

$$\begin{bmatrix} 1 & 0 & 1 & \dots & 0 \\ 1 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 1 & 0 & 0 & \dots & 1 \\ 0 & 1 & 0 & \dots & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 1 & \dots & 0 \\ 1 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 1 & 0 & \dots & 1 \\ 0 & 1 & 0 & \dots & 0 \end{bmatrix}, \dots, \begin{bmatrix} 1 & 0 & 1 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 1 & 0 & \dots & 1 \\ 0 & 1 & 0 & \dots & 0 \end{bmatrix}, \text{ and so on}$$

By determinant operation, we can make those matrices to be a matrix which can have determinant 1, 0, or -1.

Analogously, it can be shown for each feasible solution by strong induction.

Case  $n-1$  the feasible solutions have  $n-1$  bundles and one slack variable column.

$$\begin{bmatrix} 1 & 0 & 1 & \dots & 0 \\ 1 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ 0 & 1 & 0 & \dots & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \end{bmatrix}, \dots, \begin{bmatrix} 1 & 0 & 1 & \dots & 0 \\ 1 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 1 & 0 & 0 & \dots & 1 \\ 0 & 1 & 0 & \dots & 0 \end{bmatrix}, \text{ and so on.}$$

By the determinant operation, those matrices have the determinant value 1, 0, or -1.

Case  $n$  the feasible solution is the bundle with a single block, so the matrix is a identity matrix. It is obvious that its determinant is 1.

As the result of each case, it concludes that all feasible solutions that make the square matrices have the determinant of 1, 0, or -1. Indeed, this yield TUM for  $\mathbf{A}$ .  $\square$

**Proposition 2:** The constraint coefficient matrix structure in the integer programming for a double line booth auction has total unimodularity property.

**Proof**

Let  $\tilde{\mathbf{A}}$  be the constraint coefficient matrix structure for the single line case with  $n$  row blocks, and  $\tilde{\mathbf{A}}_i$  is of the individual bidder. Those are defined as the following:

$$\tilde{\mathbf{A}} = [\tilde{\mathbf{A}}_1 \quad \dots \quad \tilde{\mathbf{A}}_i \quad \dots \quad \tilde{\mathbf{A}}_n], \text{ for } \tilde{\mathbf{A}}_i = \begin{bmatrix} \mathbf{A}_i & \mathbf{0} & \mathbf{A}_i \\ \mathbf{0} & \mathbf{A}_i & \mathbf{A}_i \end{bmatrix}.$$

It is analogous to proposition 1 that it is sufficient to prove TUM of  $\tilde{\mathbf{A}}_i$ .

Since  $\mathbf{A}_i$  is TUM,  $\mathbf{A}_i \oplus_1 \mathbf{A}_i := \begin{bmatrix} \mathbf{A}_i & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_i \end{bmatrix}$  is also TUM by Seymour's characterization (1-sum).

Further that  $\begin{bmatrix} \mathbf{A}_i \\ \mathbf{A}_i \end{bmatrix}$  is also TUM by Seymour's decomposition – the repeating of row.

For the reason that  $\tilde{\mathbf{A}}_i$  is the composition of  $\mathbf{A}_i$ , we conclude that the TUM property is preserved.  $\square$



## Biography

Puchit Sariddichainunta was born in 1980 in Bangkok. His primary education commenced at Rachavinit Primary School. After that, he entered his secondary education at Suankularb Wittayalai School, where he earned his High School Diploma. In 1999, he was successfully granted the Japanese Government Scholarship from Ministry of Education, Sports, Science and Technology Research, to study at university level in Japan for seven years. He graduates two degrees, Bachelor of Economics and Master of Economics, from Hitotsubashi University.

For a couple of years, he worked in a multinational trading company and Thailand Development Research Institute. He was accepted as the Master candidate in Applied Mathematics and Computational Science in 2009. He conducted research under Asst. Prof. Krung Sinapiromsaran and belonged to OR group – operations research. His research interests focused on optimization problem, mathematical model formulation, Game Theory as well as Queuing Theory.

A part of this thesis is published in the proceedings of the Seventh International Conference on Computing and Information Technology, IC<sup>2</sup>IT 2011, at King Mongkut's University of Technology North Bangkok. The title is "The Winner Determination Model and Computation for Linear Arrangement of Booth Auction". Moreover, he has presented the extensions of this work in IEEE conference, International Conference of Management Science and Business Engineering, ICMSBE 2011, at Jeju University, Korea. The title is "The Winner Determination Problem in Combinatorial Auction for Booth Allocation: Algorithms and Extensions in Booth Layout Problem". It is published in the ICCCA2011 conference proceeding by IEEE, and submitted to be indexed by EI Compendex, INSPEC, Thomson ISI, IEEE Xplore.

After this graduation, he plans to pursue for Ph.D. study in Operations Research.