EVENT AND CLOCK BASED REPRESENTATIONS OF TIME IN MATHEMATICAL OPTIMIZATION

by

Özge Sürer

B.S., Industrial Engineering, İstanbul Technical University, 2011

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Industrial Engineering

Boğaziçi University

2014

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisor, Prof. İ. Kuban Altınel. I am indebted to him what I have learned during this study. He has not only provided me his professional expertise, but also shared with me his invaluable world-view. I feel motivated and encouraged every time I attend his meeting. I consider it is an honor to work with him.

I am grateful to my thesis committee members Assoc. Prof. Z. Caner Taşkın and Assist. Prof. Yavuz Boğaç Türkoğulları for taking time to examine my thesis and taking part in my thesis jury.

I am also very thankful to all my friends and work fellows in İstanbul Technical University for their endless support and encouragement. My special thanks go to Dr. Murat Engin Ünal for his useful suggestions and help.

I wish to express my thankfulness to my lovely friends Yasemin, Özlem, Kadir, Gizem and Pelin with whom I share many unforgettable memories.

Words are inadequate in offering my thanks to my family for their understanding, encouragement and unconditional love. My mother and father sacrifice their life for our happiness and success. İrem, you are the best sister ever. You make me laugh even in trouble times.

I would also like to thank TÜBİTAK for providing scholarship during my master program.

ABSTRACT

EVENT AND CLOCK BASED REPRESENTATIONS OF TIME IN MATHEMATICAL OPTIMIZATION

Time representation is one of the most important issues in mathematical optimization in terms of computational time and efficiency. The discrete time formulation requires large number of binary variables which affects the solution time of the problems. Instead of relying on the traditional uniform discretization of the time horizon, the event-based formulations are proposed to deal with the instances that have a long horizon. In the event-based approach, events are the situations that cause a change in the system and mathematical model is constructed based on the events. In this study, first, we briefly review the main characteristics of the two approaches and outline their advantages and disadvantages. Then, we focus on the event-based formulations of the two problems: location and scheduling problem (LASP) in wireless sensor networks (WSN) and berth allocation problem (BAP). We try to generalize the event-based formulations for different types of problems. In order to enhance the event-based formulations, we strengthen the constraints. Finally, we propose a branch and price algorithm by decomposing the models based on events. Although it provides bounds, which are better than the linear programming relaxation of original formulation, the branch and price algorithm is not very efficient. We also propose simple heuristics to find good feasible solutions. In our experiments, we compare two modeling approaches by solving the formulations using a state-of-the-art solver on the generated test bed. Then, we assess the performance of the branch and price algorithm. According to our experiments, generally the event-based formulations require less computational time. However, finding an optimum event number can require considerable computational effort. The decision maker should choose the best representation of time based on the advantages and disadvantages of the two approaches.

ÖZET

MATEMATİKSEL ENİYİLEMEDE ZAMANIN OLAY VE SAAT TABANLI GÖSTERİMİ

Zamanın gösterimi işlem süresi ve verim açısından matematiksel programlamada en önemli konulardan biridir. Kesikli zaman gösterimi problemlerin çözüm süresini etkileyen pek çok sayıda ikili değişken içermektedir. Zamanın geleneksel eş uzunluklu aralıklar kullanılarak kesiklenmesi yerine, olay tabanlı gösterim uzun zaman planına sahip örnekler için önerilmektedir. Olay tabanlı yaklaşımda, olaylar dizgede değişiklik yaratan durumlardır ve matematiksel model bu olay noktalarına göre kurulur. Bu çalışmada ilk olarak iki yaklaşımın temel özelliklerini gözden geçirdik ve her iki yaklaşımın artı ve eksilerini özetledik. Daha sonra iki problemin olay tabanlı gösterimlenmesine odaklandık: kablosuz duygaç ağlarında yer seçimi-çizelgeleme ve rıhtım paylaştırma problemlemleri. Olay tabanlı gösterimi değişik problemlerde uygulamak üzere genelleştirmeye çalıştık ve geliştirmek için güçlendirici kısıtlar önerdik. Son olarak, modelleri olaylara göre ayrıştıran yeni bir dal-eder algoritması geliştirdik. Olağan gösterimin doğrusal program gevşetmesinden daha iyi olan bir sınır bulsa da, dal-eder algoritmasının başarımı sınırlıdır. Ayrıca iyi olurlu sonuçlar elde etmek için basit sezgisel yöntemler de önerdik. Deneylerimizde, her iki yaklaşımı bir karışık tam sayılı programlama çözücüsü ile çözerek karşılaştırdık ve dal-eder algoritmasının doğruluğunu sınadık. Deneylerimize göre, genel olarak olay tabanlı gösterim daha az çözüm süresine gereksinmektedir. Ancak, en iyi çözümü veren olay sayısını bulmak bilgisayısal anlamda çok emek isteyen bir iştir. Karar verici en iyi gösterimi her iki yaklaşımın artı ve eksilerini çözmek istediği problem özelinde karşılaştırarak seçmelidir.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS . . . iii
ABSTRACT . . . iv
ÖZET . . . v
LIST OF FIGURES . . . viii
LIST OF TABLES . . . ix
LIST OF SYMBOLS . . . xii
LIST OF ACRONYMS/ABBREVIATIONS . . . xiv
1. INTRODUCTION . . . 1
2. LITERATURE SURVEY . . . 4
2.1. Continuous-Time Approaches . . . 4
2.1.1. Event-Based Approaches . . . 5
2.1.1.1. Global Event-Based Models . . . 6
2.1.1.2. Unit-Specific Event Based Models . . . 8
2.1.2. Other Continuous Time Models . . . 9
2.2. Application Areas of Event-Based Approaches . . . 9
2.3. Optimum Number of Events . . . 14
2.4. Solution Methods . . . 15
3. EVENT-BASED FORMULATIONS OF INTEGER PROGRAMS . . . 18
3.1. General Properties of the Event-Based Formulations . . . 18
3.2. Location and Scheduling Problem . . . 19
3.2.1. Event-Based Mathematical Formulation . . . 20
3.3. Two-Dimensional Problems . . . 22
3.3.1. Strip Packing Problems . . . 23
3.3.1.1. Formulation of Level Strip Packing Problems . . . 23
3.3.1.2. Formulation of Strip Packing Problems . . . 25
3.3.2. Berth Allocation Problems . . . 28
3.3.2.1. Formulation of Level Berth Allocation Problems . . . 28
3.3.2.2. Formulation of Berth Allocation Problem . . . 31

4. ENHANCEMENTS OF THE FORMULATIONS 37
4.1. Location and Scheduling Problem 37
4.1.1. Symmetry-Breaking Constraints 37
4.1.2. Tightening the Constraints 38
4.2. Desired Berth Allocation Problem 40
5. A BRANCH AND PRICE ALGORITHM 43
5.1. Reformulation of the Location and Scheduling Problem 44
5.2. Reformulation of the Desired Berth Allocation Problem 48
5.3. Implementation Issues 52
5.3.1. Initialization 52
5.3.2. Column Selection Strategy 52
5.3.3. Branching Strategy 53
5.3.4. Node Selection 54
5.3.5. Finding a Feasible Solution 54
6. COMPUTATIONAL EXPERIMENTS 60
6.1. Instance Generation 60
6.2. Experimental Results 61
6.2.1. Comparison of the Model Size 62
6.2.2. The Strength of the New Valid Inequalities 63
6.2.3. Comparison of the Models 65
6.2.4. Testing the Accuracy of the Branch and Price Algorithm 73
7. CONCLUSION 89
APPENDIX A: CLOCK-BASED FORMULATIONS OF APPLICATIONS . . 91
A.1. Clock-based Formulation of Location and Scheduling Problem 91
A.2. Clock-based Formulation of Desired Berth Allocation Problem 92
APPENDIX B: COMPARISON OF MODEL SIZE 94
APPENDIX C: RESULTS OF CLOCK-BASED FORMULATIONS 97
REFERENCES 103

LIST OF FIGURES

Figure 2.1. Different time representations. 8

Figure 3.1. Representation of LSP solution. 25

Figure 3.2. Representation of SP solution. 27

Figure 3.3. Representation of level BAP solution. 30

Figure 3.4. Representation of DBAP solution (USEB). 32

Figure 3.5. Representation of DBAP solution (GEB). 36

Figure 5.1. Branch and Price algorithm for DBAP. 57

Figure 5.2. Algorithm for finding a feasible solution for LASP. 58

Figure 5.3. Algorithm for finding a feasible solution for DBAP. 59

Figure 6.1. Clock-based vs. event-based: model sizes for LASP. 62

Figure 6.2. Clock-based vs. event-based: model sizes for DBAP. 63

Figure 6.3. Model sizes for event-based formulations: DBAP. 64

LIST OF TABLES

Table 2.1. Comparison of different time representations. 5

Table 3.1. Parameters and decision variables for LASP. 20

Table 3.2. Parameters and decision variables for LSP. 24

Table 3.3. Decision variables for SP. 26

Table 3.4. Parameters and decision variables for level BAP. 29

Table 3.5. Decision variables for DBAP. 31

Table 6.1. Sensor specifications. 61

Table 6.2. Strength of the new valid inequalities: LASP. 65

Table 6.3. Strength of the new valid inequalities: DBAP. 66

Table 6.4. LASP results. 68

Table 6.5. Results for GEB formulation (tight). 70

Table 6.6. Results for GEB formulation (loose). 71

Table 6.7. Results for small and medium size instances with L=12 (tight). . . 76

Table 6.8. Results for small size instances with L=24 (tight). 77

Table 6.9. Results for medium size instances with L=24 (tight). 78

Table 6.10. Results for small and medium size instances with L=12 (loose). . . 79

Table 6.11. Results for small size instances with L=24 (loose). 80

Table 6.12. Results for medium size instances with L=24 (loose). 81

Table 6.13. B&P results of LASP. 82

Table 6.14. Small size instances of DBAP with L=24 (tight). 83

Table 6.15. Medium size instances of DBAP with L=24 (tight). 84

Table 6.16. Small and medium size instances of DBAP with L=12 (tight). . . 85

Table 6.17. Small and medium size instances of DBAP with L=12 (loose). . . 86

Table 6.18. Small size instances of DBAP with L=24 (loose). 87

Table 6.19. Medium size instances of DBAP with L=24 (loose). 88

Table A.1. Parameters and decision variables for the clock-based formulation of LASP. 92

Table A.2. Parameters and decision variables for the clock-based formulation of DBAP. 92

Table B.1. Number of variables and constraints in LASP. 94

Table B.2. Number of variables and constraints in DBAP (tight). 95

Table B.3. Number of variables and constraints in DBAP (loose). 96

Table C.1. Clock-based results for small size instances with L=24 (tight). . . . 97

Table C.2. Clock-based results for medium size instances with L=24 (tight). . 98

Table C.3. Clock-based results for small and medium size instances with L=12 (tight). 99

Table C.4. Clock-based results for small size instances with L=24 (loose). . . 100

Table C.5. Clock-based results for medium size instances with L=24 (loose). . 101

Table C.6. Clock-based results for small and medium size instances with L=12 (loose). 102

LIST OF SYMBOLS

a_k	Arrival time of vessel k
a_{ijn}	Indicates whether a type n sensor placed on point i can cover point j or not
B	Total available budget
c_n	Unit cost for a type n sensor
c_k	Departure time of vessel k (completion time)
d_j	Minimum number of sensors that should cover point j
d_k	Due time of vessel k
des_k	Desired berthing section of vessel k
E	Total number of events
e_n	Energy consumption rate for a type n sensor at active mode
E_n	Initial battery of a type n sensor
f_k	Lateness penalty for vessel k
H	Strip height
H_e	Height of event e
h_k	Height of item k
I	Total number of points to place the sensors
J	Total number of points to be covered
K	Total number of vessels
L	Width of berth
LT	Lifetime of the network
M	Large number
N	Total number of types of sensors
p_k	Processing time of vessel k
ST_{le}	Time of event e on section l
s_n	Sensing range of a type n sensor
s_{ine}	Active time length of a type n sensor placed on point i at event e

T	Planning horizon
T_e	Time of event e
W_e	Duration of event e
w_k	Width of item or vessel k
X	Width of strip
x_{in}	Indicates whether a type n sensor is placed on point i or not
Y_{le}	Y coordinate of event e on section l
y_t	Indicates whether a period t is within lifetime LT or not
z_k	Deviation from due time of vessel k
z_{ine}	Indicates whether a type n sensor placed on point i is active at event e or not
z_{int}	Indicates whether a type n sensor placed on point i is active in period t or not
δ_{ke}	Indicates whether item or vessel k is assigned to event e or not
δ_{kle}	Indicates whether vessel k's bottom left corner is assigned to event e on berth section l or not
δ_{kxe}	Indicates whether item k's bottom left corner is assigned to event e on grid x or not
δ_{klt}	Indicates whether vessel k arrives at berth section l at time t or not
σ_{kxy}	Indicates whether vessel k covers block (x, y)

LIST OF ACRONYMS/ABBREVIATIONS

2D	Two Dimensional
BAP	Berth Allocation Problem
CPU	Central Processing Unit
DBAP	Desired Berth Allocation Problem
GEB	Global Event-Based
LP	Linear Programming
LASP	Location and Scheduling Problem
LSP	Level Strip Packing
MILP	Mixed Integer Linear Programming
NP	Non-Deterministic Polynomial Time
RMP	Restricted Master Problem
SP	Strip Packing
USEB	Unit-Specific Event-Based
WSN	Wireless Sensor Network

1. INTRODUCTION

There can be several mathematical formulations of a problem having the same optimal solution. However, in terms of computational time and efficiency, some of the formulations are better than the others. Given the complexity of the problems, the most efficient formulation has a crucial importance in terms of computational time savings. The problem representation and assumptions have to be done before modeling, since sometimes the mathematical model cannot fix the problem description and there can be some inconsistencies between mathematical model and problem description [1]. According to Méndez *et al.* [2], time representation is the most important issue in terms of the problem formulation. By means of time-indexed variables, the state of resources can be monitored and challenging constraints can be modeled easily.

Based on the time domain, the mathematical formulations of the optimization problems are divided into two groups: discrete and continuous time formulations. In the discrete time formulations, time is divided into equal length intervals each of which representing a "tick" of the clock and the events are allowed to occur at the beginning or ending of a task [2]. The formulations obtained using discrete time approach, which we will refer as *clock -based* formulations, are structured and well-defined, since there is a reference point for activities to occur [3]. Especially, based on these reference points it is easy to model the constraints related to activities which use the same resources [4]. However, the changes in the system occur at specific times such as the beginning of an interval. Therefore, discretization creates an approximation of time [2, 3, 5]. Uniform discretization of time requires a large number of binary variables for each of the small, uniform time slots. Hence, the problem size increases and it becomes hard to solve [3, 4]. The greatest common divisor of processing times can be calculated to achieve a suitable approximation and be used as an equal time interval [4,6]. Especially, in terms of real life applications, clock-based formulations are hard to solve and time-consuming [4]. In contrast to the clock-based models, continuous time models include variable length time intervals. Since activities can occur at any time, this approach

leads to more accurate time approximations. However, due to variable time length, the continuous time modeling becomes more challenging than uniform discretization of time [3], since it requires the usage of nonlinear terms during the formulations. In order to prevent the non-linearity of the constraints, the additional constraints are added to the mathematical model.

Although clock-based models have many advantages, the continuous time models seem to be superior in the field of solution quality and computational time. This motivated us to examine the existing mathematical optimization models based on different time representations and compare the performances of both approaches. We are going to implement different time representations for the same problem in different areas and find that whether there exist other application areas in which all types of time representations cannot give a feasible solution. With these new aspects, we are going to search a general framework based on the time domain for mathematical optimization problems.

In this study, we apply event-based time formulation on location and scheduling problem (LASP) in wireless sensor networks (WSN) and desired berth allocation problem (DBAP). We concentrate on the event-based formulations of the two problems and compare the results with clock-based formulation by directly solving with a mixed integer linear programming (MILP) solver. Although the numerical results confirm the effectiveness of the event-based formulations, solving large problem instances and getting tight bounds are research challenges. Hence, an alternative exact solution method is proposed. This is a branch and price algorithm based on the decomposition of the problems with respect to the events. Our motivation is to develop an exact approach that can be applied to the most of the event-based formulations.

The rest of this thesis is organized as follows: Chapter 2 contains literature survey about different time representations and their application areas. Then, in Chapter 3 we give the event based formulations of the chosen problems. In order to improve the mathematical models of the event-based representations, tightening and symmetry

breaking constraints are proposed in Chapter 4. Next, we give the details of the branch and price algorithm in Chapter 5. Finally, Chapter 6 includes the computational results of our study.

2. LITERATURE SURVEY

There are several alternative ways of representing the same problem based on the time domain. Generally, the mathematical optimization models are divided into discrete and continuous time models according to the time representation. In the discrete time models, time horizon is divided into equal length intervals. These small intervals correspond to one time unit in the time horizon. Hence, the formulations based on discretization of time are called clock-based formulations in this study. In the clock-based representation, events occur at the predefined time points. On the other hand, in the continuous time representation, events can take place at any time along the time horizon [2]. We can easily model the problems from various aspects of the problem environment using clock-based representation, since it is suitable for different constraint types and easily adaptable. Although modeling with clock-based representation is easy, generally problem size and computational time increase with an increasing time horizon. Especially, problems have large and unmanageable size in real life cases. In order to overcome the inefficiency of the clock-based modeling, the variable length approach is proposed in the literature. Mostly, these types of formulations are applied in the field of process scheduling. In the papers [2, 4, 7], the readers can find a detailed reviews of different models based on time representations for chemical industry process scheduling. Floudas and Lin [4] state that chemical processes include a lot of discrete decisions. In order to overcome the complexity of these types of problems, continuous time approach is suggested. To sum up, we can outline the differences between the continuous time and clock-based modeling in Table 2.1 based on the study [8] and the reviews about time representations.

2.1. Continuous-Time Approaches

Continuous time models are offered to overcome two main disadvantages of the clock-based modeling: very large problem size and suboptimal solutions. In a variable length approach, the binary variables representing the assignment of activities for

Table 2.1. Comparison of different time representations.

Clock-based Time Modeling	Continuous Time Modeling
Equal length between time intervals	Variable length between time intervals
More time points	Fewer time points
Suboptimal solutions	Exact time approximation
More binary variables	Fewer binary variables
Ease of modeling	Hard to consider different type constraints
Less big-M constraints	Big-M constraints for linearization
Tighter relaxation	Poor linear programming (LP) relaxation
Less degeneracy	Highly degenerate LP relaxation

each of the uniform time lengths are eliminated. In order to determine the occurrence time of events, continuous variables are required. Hence, we have more accurate time approximation in continuous time representation. For example, if a processing time changes between 0.738 and 11.250, without rounding procedure we cannot obtain the exact solution in a clock-based model [5]. However, the use of continuous variables requires some big-M constraints. Although we have better solutions for the continuous time models, proving the optimality and reducing the optimality gap increase the computational time. In general, the continuous time formulation is divided into two classes: event-based models and sequential process models. Event-based models have two main groups: global time points and unit-specific time points. In sequential processes, slot-based and precedence-based time formulations can be used. In this thesis, we mainly focus on the event-based representation of the time in mathematical models.

2.1.1. Event-Based Approaches

Event-based approach in continuous time domain emerges from batch processes in chemical manufacturing systems [6, 9, 10]. In clock-based formulations, we assign

a binary variable for each of the equal length small intervals. In the event-based approach, time horizon is partitioned according to event intervals. Due to the variable event time, the inactive periods' assignment variables are eliminated from the event-based formulations, hence the problem size decreases significantly. A single period corresponds to a new event. In contrast to clock-based models, we can group the uniform time intervals if a state of the system does not change. When the system state changes, we accept that an event occurs. Therefore, instead of using variables indexed by time, we use variables indexed by event in the event-based approaches.

In batch processes, there are tasks, which are performed in units, and there are constraints related to capacity, storage, material balance and demand [9]. In such a system, events are defined as the start or end of a task or change in resource level and beginning of unit utilization. In the study by Mockus and Reklaitis [6], non-uniform discretization of time is used and the variables are defined based on the exact time when the events take place. Events are defined as the start or end of a task and time changes when these events occur. As an output of the mathematical program, the starting times of the tasks are found. Therefore, when there is no change in the system, extra binary variables are eliminated due to the variables indexed by event [6]. However, the main problem in the event-based approach is to find the optimum number of events. Small number of events results in suboptimal solutions. However, large number of events increases the computational time. There are two main approaches in the event-based representation: global event-based (GEB) and unit-specific event-based (USEB) approaches [3].

2.1.1.1. Global Event-Based Models. In GEB approach, event time is shared across the sub-systems. When an event occurs in one sub-system, the number of global events increases by one and the time of this event remains the same for all sub-systems. Then, all operations starting in the same event share the same event timing. GEB models are the early applications of the event-based formulations.

GEB approach is applied in papers by [6,10–13]. Mockus and Reklaitis [6] formu-

late the variable batch size scheduling problem as a mixed integer non-linear program to eliminate the binary variables. Then, it is linearized to MILP. The start of a task or change in resource level is accepted as an event. In their following study [11], they extend their formulation in order to apply in both batch and continuous processes. They use separate binary variables in order to represent the start and end of an event and timing decision is considered as a continuous variable. Zhang and Sargent [10] use the key variable representing whether task i starts in unit j at event time e which is similar in clock-based model of [14]. In clock-based representation, this variable is used to represent whether task i starts in given unit j at the beginning of time interval t. They use two types of continuous variables to show the occurrence of an event. One of the main continuous variables represents the global event time across all units and the other represents the starting time of a task if it starts at a given event. Maravelias and Grossmann [12] use binary variables indicating for only the assignment of tasks to events. They assert the importance of eliminating big-M constraints in continuous time formulations in order to improve the LP relaxations. In the GEB approach, the times of events are not determined, so we need time synchronizing constraints to synchronize a time point with the starting/finishing time of tasks. Hazaras *et al.* [13] extend the formulation given in [12] by adding maintenance events to their model.

Figure 2.1 shows how the same schedule is constructed with different time representations. We can understand that an optimal solution of the problem requires different number of events for each of the modeling approaches. Although we need seven equal time intervals in clock-based approach, in GEB approach we need four global events.

The main difference in two models is that in the clock-based model, a variable to represent time is not necessary. The time index expresses the exact time in a clock-based model. However, the continuous variable showing the event time is required in the GEB approach. It is clear that extra constraint should indicate the increasing order of this time variable indexed by the event number. Generally, in the GEB formulation, products of integer and continuous variables take place in the constraints and the

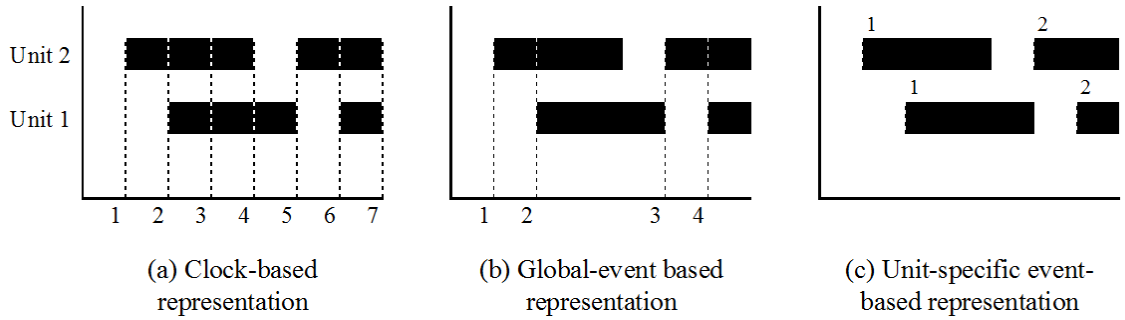

Figure 2.1. Different time representations.

linearization of these constraints creates additional constraints [4].

2.1.1.2. Unit-Specific Event Based Models. It is the relaxation of global time point representation in which the events can occur at different time in different units [2]. Although the location of events is different across the units, the number of events is the same for all units. At Figure 2.1, we can see that the unit-specific event based (USEB) approach requires two events in each unit. Comparing to the GEB model, this one requires less binary variables. Hence, if we use an iterative increment of the number of events approach, the improvement in the objective function can be larger in the USEB approach for an increase in the number of events [4].

USEB approach is first used in [9] in the field of batch processing schedule. The initiation of a task or the beginning of unit utilization is defined as an event. Instead of using a binary variable expressing the start of task i in unit j at event e, two separate variables, defining the assignment of task i at event e and the utilization of unit j at event e, are used. This decoupling approach reduces the number of binary variables. Janak *et al.* [15] use the same modeling approach with the study [9] by considering additional resource constraints. Giannelos and Georgiadis [16] model the similar problem and define the events as the termination of a task.

2.1.2. Other Continuous Time Models

Slot-based and precedence-based continuous time models are generally developed for sequential processes in scheduling problems [4]. In sequential processes, batches should follow a sequence based on the product recipe [2].

In slot-based models, time horizon is divided into a predefined number of slots with variable lengths and batches are allocated to these slots for production. Sundaramoorthy and Karimi [17] use a slot-based approach for short-term scheduling in multipurpose batch plants and offer an improved model without using big-M constraints. Slot-based models are divided into two groups as in event-based models: synchronous (or common) and asynchronous (or uncommon). Synchronous slot models are similar to GEB models and asynchronous slot models are similar to the USEB models.

In precedence-based models, binary variables are used for representing the jobs' predecessors and successors. Méndez *et al.* [18] give the model for the scheduling of a multistage multiproduct batch plant problem based on the precedence-based time domain. In this model, the variables' representation relies on the ordering of the jobs. They use separate binary variables in order to involve the assignment and sequencing decisions. Assignment decisions refer to the allocation of orders into units and sequencing decisions indicate the orders' and stages' sequences. The model includes a continuous variable denoting the completion time of the order. Stefansson *et al.* [19] compare clock-based and continuous time scheduling for multipurpose plants. They use the same representation as Méndez *et al.* [18] in continuous time models.

2.2. Application Areas of Event-Based Approaches

Generally, event-based time representation is applied in the field of chemical industry and batch processes. In this part, we give different examples from various application areas of event-based time representation.

Saharidis *et al.* [20] offer a mixed integer linear programming (MILP) model for the scheduling of loading and unloading of crude oil in a refinery. They present models based on clock-based and event-driven representations. In a given real life example, there are 720 time periods in 30 days in clock-based modeling approach. On the other hand, if a period changes based on the events such as boat arrivals and the change in the blend composition, the optimum solution includes 10 intervals. A decrease in the number of intervals provides a decrease in problem size and thus in problem complexity. Although the first model requires 16h processing time, second model requires only 1h processing time.

Wang and Guignard [21] give both clock-based and activity-based continuous time formulations for batch sizing and scheduling problems. In the clock-based modeling, while a binary variable is determining whether batch j is scheduled on machine m beginning from time t; similar binary variable is used in the activity-based continuous time model to determine whether job j is scheduled for the eth activity of machine m. In modeling using these approaches, the main differences appear in the index of the variables and time constraints. In the continuous time approach, event number e is used instead of time index t; and the remaining indices are similar. Hence, if we ignore the difference between the indices t and e, we get nearly the same model. In the continuous time representation, the additional continuous variables, which represent the starting and finishing times of the activities on the machines and the time indicating a change in the inventory status of the product, are used. In order to show their sequential relationships, extra constraints are added to the model. The rest of the constraints concerning batch sizing, inventory balance and demand are nearly the same as the clock-based model. We emphasize that if the representation of time period is necessary in clock-based model constraints, the duration of time is added to or subtracted from time index t and summation over the binary variables is done between these time intervals. However, such an index does not exist in the activity-based model. The continuous variables are used to represent the duration of time. The binary variables are used to determine the activities in an event's period. Using only the variable indexed by event number e is enough to represent the same thing in the clock-based model. The

constraints below show this situation; (2.1) and (2.2) refer to the assignment constraints in the event based approach and clock-based model. Constraints (2.1) guarantee that no two jobs can be scheduled on the same machine at the same event, constraints (2.2) say at any given time period t, at most one task can be executed on each machine.

$$\sum_{j=1}^{J} W_{jme} \leq 1 \qquad m = 1, \ldots, M; e = 1, \ldots, E \qquad (2.1)$$

$$\sum_{j=1}^{J} \sum_{t_1=t-l(j,m)}^{t} W_{jm(t_1)} \leq 1 \qquad m = 1, \ldots, M; t = 1, \ldots, T \qquad (2.2)$$

Koné *et al.* [5] give two different event-based continuous time formulations for the resource constrained project scheduling problems. Since it is an NP-hard problem and the number of binary variables increases with increasing time horizon, event-based formulations are more advantageous than clock-based models. In the clock-based model, the key binary variable x_{it} represents whether an activity i starts at time t. In the continuous time model with sequencing approach, the key variable x_{ij} equals 1 when activity i is processed before activity j. In one of the event-based approaches, the start or end of activities is accepted as an event. There are two types of key decision variables and x_{ie}/ y_{ie} represents if activity i starts/finishes at event e. In order to demonstrate the occurrence time of events, the continuous variable t_e is used to show the date of event e. The new constraints related to the continuous variable t_e are implemented in the event-based formulation. Second event-based formulation utilizes the binary variable defining whether activity i starts at event e or processing after event e or not.

Zapata *et al.* [22] give three formulations for the multi-mode resource-constrained multi-project scheduling problem. In the clock-based formulation, two key binary variables defining whether task i of project p starts/finishes at the beginning of time t with resource combination j are used. In the precedence-based formulation, the binary variable $Y_{pip'i'}$ represents if task i' of project p' has been completed after the starting of

task i of project p. By using these variables, in the precedence formulation, the time is eliminated from the formulation and, instead the sequential relations between tasks are identified in the constraints. In the event-based continuous time formulation, the start of each task is given as the event. The binary variable x_{spijn}/ x_{fpijn} represents that if task i of project p is started/finished at event n with combination j. If the constraints are compared with the clock-based version, the assignment constraints and the resource constraints are similar to each other. In the time constraints, the continuous variables representing the start and finish times of task i of project p are used. Since these variables give the exact time of event n, it is not necessary to sum the variables with respect to time index as in the clock-based model. In order to sequence the continuous time variables, they apply additional constraints. Since the duration of time periods are unknown, "in process" variables are added to the event based model in addition to the start and finish variables.

Mouret *et al.* [1] offer a general framework for multi-stage batch scheduling problems and crude-oil operations scheduling. They give an example of four time representations that are similar to the existing studies in the literature in meaning but have different names;

(i) Multi-operation sequencing: Unit-specific event-based models
(ii) Multi-operation sequencing with synchronized start times: Global event-based models
(iii) Multi-operation sequencing with fixed start times: Clock-based models
(iv) Single operation sequencing

They offer a strengthened formulation by using the concepts from non-overlapping graph properties. In this study, events are accepted as the assignment of operation to the priority slot. S_{iv} represents the start time of operation v if it is assigned to priority-slot i, and Z_{iv} is a binary variable indicating whether operation v is assigned to priority-slot i or not. There are slight differences between the USEB and GEB models. In the GEB model, apart from the constraints in the USEB model, in order

to synchronize the time, their model includes the variables that represent global time points t_i. They employ time point sequence constraints (2.3) and synchronization constraints (2.4)-(2.5), since the starting times of all operations in the same event should be the same in GEB model.

$$t_{i-1} \leq t_i \qquad i = 1, \ldots, I-1 \tag{2.3}$$

$$S_{iv} \leq t_i \qquad i = 1, \ldots, I; v = 1, \ldots, V \tag{2.4}$$

$$S_{iv} \geq t_i - H(1 - Z_{iv}) \qquad i = 1, \ldots, I; v = 1, \ldots, V \tag{2.5}$$

In the clock-based time model, because the time of slots is known a priori, t_i is represented as a parameter that can be seen in equalities (2.6). H represents the time horizon. S_{iv} can be defined as a function of time that is given in constraints (2.7). They show that USEB model is superior to other models. Based on this study, we can say that the integer feasible space of USEB model is larger than the other models.

$$t_i = \frac{i-1}{n} H \qquad i = 1, \ldots, I \tag{2.6}$$

$$S_{iv} = t_i Z_{iv} \qquad i = 1, \ldots, I; v = 1, \ldots, V \tag{2.7}$$

Castro and Oliveira [23] and Castro and Grossmann [24] inspire from the scheduling problems and apply the event-based time approach to the two-dimensional (2D) orthogonal packing problems. There are different classes of packing problems such as strip packing (SP), knapsack and bin packing problems. Since the problems are similar to each other, they give the common constraints for them. In SP problems, there are rectangles with given width and height. The aim is to place these rectangles into a strip with given width so as to minimize the height of the strip. Castro and Oliveira [23] give a new hybrid discrete/continuous-space approach for mathematical model of SP problems which is discrete in the x-axis and continuous in the y-axis. They use a continuous variable to represent the Y coordinate of the event e and the grid x (Y_{xe}). The binary variable N_{ixe} identifies the assignment of the left edge of rectangle i to

event e on grid x. The differences between the Y coordinate of the event e and $e+1$ should be larger than the height of the assigned rectangle. Furthermore, if a rectangle is placed, through the width of this rectangle the Y coordinate does not change. We utilize the main constraints of this formulation in our DBAP formulation. Castro and Grossmann [24] use sequential variables and their modeling approach is based on the precedence relations between the rectangles.

These studies above show us that event based approach can be applied in different areas but mostly in scheduling problems and give an effective formulation for these type of problems. In the next chapter, we give the event-based formulations of the two applications and search a general framework for the formulations.

2.3. Optimum Number of Events

In the event-based formulations, we cannot estimate the optimum number of events a priori. However, the maximum number of events is restricted to the maximum number of activities. In some cases, we can make inference related to the number of events based on the structure of the data and the problem. For example, in SP problem the maximum number of events is the total number of rectangles that are placed into the strip. The minimum number of events can be calculated by dividing the width of the strip to the total sum of the width of the rectangles. Nevertheless, the number of events has large influence on the computational performance and if the difference between the minimum and the maximum number of events is large, finding the optimum solution by trial and error is a challenging work.

In order to solve these types of models, the number of events should be predefined [25]. At each event number, we obtain different optimal solution. The global optimum is reached at a sufficiently large number of events. For solving the problem, the number of events can be increased iteratively until the objective value converges to the same value. Although large number of events increases the opportunity of getting a global optimum solution, the model gets larger and solution time increases. This iterative

procedure results in large computational times. Efficient formulations have a crucial importance in overcoming this type of difficulties.

Castro and Oliveira [23] and Castro and Grossmann [24] use the same principle for finding a better solution. They start to solve a problem from the minimum number of events and iteratively they increase the number of events by one and add the best objective value found as a cut to the new problem's objective value. At each iteration, they solve the problem by a MILP solver. If the objective value does not change between the two consecutive event numbers, they stop the algorithm. However, this algorithm does not prove that the objective value and total number of events are optimal. The optimal value of two consecutive events can remain the same, but at the next step, the optimal value can improve. In addition to this approach, Mouret *et al.* [1] use the multiplicative approach by multiplying the number of events by two instead of increasing by one. These are heuristic approaches for finding better solutions and event numbers. Nevertheless, for a given event number the efficient solution of a problem is still a challenging work. Seid and Majozi [26] and Li and Floudas [27] try efficient techniques for the prediction of time points. They utilize the properties of the chemical processes and find the minimum and maximum number of events. Then, they apply a branch and bound algorithm where the root node corresponds to the solution with a mid-point of maximum and minimum number of events. Each node represents an event and every node includes at most one child except the root node. While in one branch the number of events increases, in the other branch the number of events decreases.

2.4. Solution Methods

In general, instead of giving alternative exact solution methods, studies in the event-based formulations take into account different cases in problem environment and compare the results with clock-based formulation by solving proposed models with MILP solver.

In order to get good results, some heuristic decomposition methods are applied in the literature. We can divide the decomposition techniques into two main groups.

In the first group, there are decomposition methods that utilize the simplified problem's solution and then try to improve this solution in order to obtain better feasible solutions. Wu and Ierapetritou [28] present a number of heuristic based decomposition techniques. In time decomposition, the time horizon is divided into smaller subsets. The earliest one is chosen and the production is calculated in that period. Then, the inventory is transferred to the next period. The next period's production is calculated based on the prior period's inventory. This process continuous until the end of time horizon. However, this approach results in suboptimal solutions. In event decomposition, the problem is started to solve with a small number of events, since it requires less computational effort. Then, task sequences are fixed and problem is solved with a higher number of events until the objective value converges. However, again the optimal solution is not guaranteed. Additionally, they apply Lagrangean decomposition in scheduling problems based on the event-based formulation of Ierapetritou and Floudas [9]. Wang and Guignard [21] propose a heuristic algorithm to solve batch sizing and scheduling problems. First of all, they simplify their problem by changing the parameters of the problem and they get a solution from the clock-based formulation. After that, they implement this solution to the event-based formulation in order to improve the solution quality. In clock-based solution, they assume that job durations are uniform. From the simplified problem, they obtain the sequences of the production schedule and then they plug this information into an event based formulation. The optimum values of the rest of the variables are taken from the continuous time approach.

In the second group, the production stages are divided into small, manageable sizes. Stefansson *et al.* [19] propose a decomposition heuristic in production scheduling problems. The authors decompose the production stages into two components. The first component's solution is the input for the second component. Hence, the relation between these two main components is important in order to find a good feasible so-

lution. By decomposing the production stages into two main components, they obtain good results in relatively shorter times. The clock-based and continuous time formulations are used in the proposed technique in order to compare the computational performance of both formulations. Li and Ierapetritou [29] apply a decomposition technique in integrated planning and scheduling problem based on the bi-level formulation of the problem. The upper level problem is a planning problem and the sub-problems are scheduling problems. The continuous time formulation for batch process scheduling proposed by Ierapetritou and Floudas [9] is solved in the sub-problems.

In general, heuristic decomposition techniques dominate the alternative solution techniques in the literature. Apart from heuristics, in order to improve the computational performance of the event-based formulations additional constraints can be implemented to formulations. Janak and Floudas [30] propose a number of preprocessing steps and new sets of tightening constraints for an effective USEB formulation of the short-term scheduling of batch plants. Additionally, Maravelias and Grossmann [31] use a hybrid technique for the scheduling of multipurpose batch plants. The assignment of units to tasks is made by the master problem and the constraint programming subproblem checks the feasibility of a specific assignment and obtains a feasible detailed schedule.

Based on the event-based formulations, as an alternative to the MILP solvers exact solution methods are very rare. Hence, in Chapter 5 we introduce a branch and price algorithm based on event decomposition in order to be an example in this field.

3. EVENT-BASED FORMULATIONS OF INTEGER PROGRAMS

In this chapter, we investigate the general properties of the event based formulations. Then, we choose two problems as our main concern: location and scheduling problem (LASP) and desired berth allocation problem (DBAP). First of all, we present the event-based formulation of LASP in WSN and their relation with clock-based formulation. Then, we focus on DBAP. Before giving the mathematical formulation of DBAP, its relatives 2D level strip packing (LSP) and strip packing (SP) problem formulations are given in order to show the evolution of the formulation steps.

3.1. General Properties of the Event-Based Formulations

The USEB models include less events than the GEB models, since total number of events equals the total number of events in one of the units. However, GEB models consider the events globally. If an event occurs in one of the units, the number of global events increases. Therefore, GEB models are more general than USEB models and can be useful when the resources and the outputs are common across all units. Generally, as USEB approach includes less integer variables, it requires less computational effort than the GEB approach. The time between two events in a unit is easily calculated in USEB approach, since after the completion of an activity in a unit, another one can start. An event's duration depends on the activity's processing time in that unit. On the contrary to USEB approaches, the time between the two global events is not explicit, since a global event can start before another event's ending across the different units.

Generally, the GEB models are derived from USEB models. By integrating some additional constraints to USEB formulation related to time sequencing and synchronization, we can obtain GEB formulations. In both of the approaches, we present the timing of events by a variable. Formulations include event ordering constraints and

time synchronization constraints. The event indexed variables replace time-indexed variables in clock-based approach. In the clock-based approach, timing is a parameter of the problem. If we want to define time duration, we can sum the binary variables indexed by time, since an increase in time index refers to an increase in time. However, an increase in the number of events does not correspond to an increase in time. Hence, in the event based approach we utilize the new continuous variables that represent the duration of an event or we take the differences of timing variables between two events. Apart from the new continuous variables, the formulations are similar.

3.2. Location and Scheduling Problem

A wireless sensor network includes tiny devices called sensors capable of sensing, data processing and communicating. The sensors are distributed to the sensor field to monitor the environment from the remote areas. A sensor can be in active or standby mode. They have different characteristics such as unit cost, sensing range and energy consumption level [32]. In real life applications, we need WSNs that have longer lifetimes. Hence, the efficient placement of sensors is an important issue. Lifetime of a WSN depends on the energy consumptions of the sensors, since WSN exists until the sensors run out of their energy. If we increase the total number of active sensors in a given period to satisfy the coverage quality requirements of the points in the sensor field, the total cost and energy consumption increase. In LASP, the goal is to find the optimal deployment of the sensors and the active periods that maximize the coverage lifetime of the WSN subject to coverage quality requirements, energy restrictions and budget constraint.

In our problem, the sensor field consists of J points that should be covered by the sensors. There are I possible points to place the sensors whose places are the same as J points. N types of sensors with different technical specifications such as the unit cost, energy consumption rate are utilized. Additionally, each point should be covered by at least d_j sensors and the budget constraint should be fulfilled.

Table 3.1. Parameters and decision variables for LASP.

Parameters	Definition
a_{ijn}	Indicates whether a type n sensor placed on point i can cover point j or not
e_n	Energy consumption rate for a type n sensor at active mode
E_n	Initial battery of a type n sensor
d_j	Minimum number of sensors that should cover point j
c_n	Unit cost for a type n sensor
B	Total available budget
E	Total number of events
M	Large number
Variables	Definition
W_e	Duration of event e
x_{in}	Indicates whether a type n sensor is placed on point i or not
z_{ine}	Indicates whether a type n sensor placed on point i is active at event e or not
s_{ine}	Active time length of a type n sensor placed on point i at event e

In the following sub-section, we give the event-based formulation of LASP. The clock-based mathematical model given in [33] is also represented in Appendix A.1 for the sake of completeness.

3.2.1. Event-Based Mathematical Formulation

In the event based approach, event refers to the starting of a sensor activation. The global event-based formulation of an optimal placement and activity scheduling to maximize coverage lifetime is given in the following. The definitions of the model parameters and variables can be found in Table 3.1.

$$\max \sum_{e=1}^{E} W_e \tag{3.1}$$

subject to

$$\sum_{i=1}^{I}\sum_{n=1}^{N} a_{ijn} z_{ine} \geq d_j \qquad j = 1, \ldots, J; e = 1, \ldots, E \tag{3.2}$$

$$\sum_{e=1}^{E} s_{ine} \leq \left\lfloor \frac{E_n}{e_n} \right\rfloor x_{in} \qquad i = 1, \ldots, I; n = 1, \ldots, N \tag{3.3}$$

$$z_{ine} \leq x_{in} \qquad i = 1, \ldots, I; n = 1, \ldots, N; e = 1, \ldots, E \tag{3.4}$$

$$\sum_{n=1}^{N} c_n \sum_{i=1}^{I} x_{in} \leq B \tag{3.5}$$

$$s_{ine} = W_e z_{ine} \qquad i = 1, \ldots, I; n = 1, \ldots, N; e = 1, \ldots, E \tag{3.6}$$

$$z_{ine}, x_{in} \in \{0, 1\} \qquad i = 1, \ldots, I; n = 1, \ldots, N; e = 1, \ldots, E \tag{3.7}$$

$$s_{ine}, W_e \geq 0 \qquad i = 1, \ldots, I; n = 1, \ldots, N; e = 1, \ldots, E \tag{3.8}$$

The objective function (3.1) maximizes the lifetime of sensor network. The variable W_e represents the duration of an event e and the variables W_es are summed with respect to overall events e in order to define lifetime LT in the clock-based formulation.

$$s_{ine} \leq W_e \qquad i = 1, \ldots, I; n = 1, \ldots, N; e = 1, \ldots, E \tag{3.9}$$

$$s_{ine} \leq M z_{ine} \qquad i = 1, \ldots, I; n = 1, \ldots, N; e = 1, \ldots, E \tag{3.10}$$

$$s_{ine} \geq W_e - M(1 - z_{ine}) \qquad i = 1, \ldots, I; n = 1, \ldots, N; e = 1, \ldots, E \tag{3.11}$$

In constraints (3.6), s_{ine} is equal to the product of W_e and z_{ine} and determines the active time length of a sensor at point i with type n if it is active at this period. Since s_{ine} is a product of two unknown variables, constraints (3.9)-(3.11) are used in order to linearize these constraints. The energy consumption constraints (A.5) in clock-based approach and (3.3) in event-based approach are slightly different in terms of variables used. As in constraints (3.3), the total active time length of type n sensor on point i can be at most the ratio of battery energy to consumption rate. Constraints (3.4) and

(3.5) are similar to constraints (A.6) and (A.7) in clock-based formulation. Event-based MILP formulation of LASP includes constraints (3.2)-(3.5), (3.7)-(3.11).

In the modeling step, the index of binary variables changes from t to e. In order to represent the duration of activation, the continuous s_{ine} variables are summed with respect to overall events e. In clock-based formulation, this is achieved by summing the z_{int} variables with respect to index t. The number of binary z_{int} variables equals the product of I, N and T. However, in the event-based formulation the number of binary variables z_{ine} equals the product of I, N and E. In other words, values of T and E affect heavily the complexity of the models in both approaches.

In this problem, GEB formulation successfully gives the same result with clock-based formulation. However, USEB formulation does not work well due to the coverage quality constraints (3.2). Each of the points has to be controlled in terms of the number of sensors that covers this point after the state of the network changes. In USEB approach, each one of the events can occur at different times. On the other hand, in GEB approach, the time ticks globally at each change in the system. Therefore, when an event occurs, the coverage quality can be controlled for each point. We conclude that it is not possible to model some type of problems by using all types of time representation due to the structure of their constraints. Especially, if a resource is common for all events, it is hard to formulate it as a USEB formulation. In this problem, the total energy for a type n sensor at point i should be controlled with each of the global events.

3.3. Two-Dimensional Problems

Recently, the continuous time formulation of the 2D packing problems inspiring from the works related to scheduling problems are given in [23,24]. These types of problems try to pack several small rectangles into one large main rectangle with respect to different objective functions in order to minimize the height or cost. In the orthogonal packing problem, the edges of the small items must be parallel to the edge of the main

rectangle such as bin or strip. The most popular classes of 2D orthogonal packing problems are SP, knapsack and bin-packing problems [24]. In this chapter, we shortly demonstrate the event-based formulations of the 2D level strip packing (LSP) and 2D strip packing (SP) problems. Then, we mainly focus on GEB and USEB formulations of DBAP.

3.3.1. Strip Packing Problems

In SP, we have K small rectangles with given width and height and we try to find the optimal placement of these small rectangles in a given largest rectangle, which is called a strip, so that the packing height is minimized. In the event-based formulations, the x-axis represents the width of the strip and it is divided into X equal sized blocks. The y-axis represents the height of the strip and it is continuous in the time domain. The y-axis shows the events occurrence. It is divided into E blocks with variable length. In LSP problems, the items are packed by horizontal levels. The height of a level is determined by the tallest item it contains. In a level, the x-coordinate that the item is assigned is not important. At the same level, y-coordinate remains the same along the x-axis. However, in SP, y-coordinate can take different values along the x-axis in order to arrange the rectangles. Hence, LSP problem is the easier version of SP problem. There are several heuristics in order to obtain better feasible solutions for SP by using the solution of LSP. First of all, we present the mathematical model of LSP in order to show the formulation of an easier version of SP.

3.3.1.1. Formulation of Level Strip Packing Problems. In 2D LSP, we try to pack the items in levels that are parallel to the bottom of the strip. Each level's height equals the maximum height of the rectangles that are packed in that level. In this problem, a level corresponds to an event. The sequences of the events cannot affect the optimal solution of the problem. As the ordering of the rectangles at each level does not affect total height, we can use some symmetry breaking constraints to decrease the number of alternative optimal solutions. Since levels divide the strip globally, the given formulation is called the GEB formulation of 2D LSP. If the USEB approach is

Table 3.2. Parameters and decision variables for LSP.

Parameters	Definition
w_k	Width of item k
h_k	Height of item k
X	Width of strip
Variables	Definition
δ_{ke}	Indicates whether the item k is assigned to event e or not
H_e	Height of event e

used, the number of events in one unit would be equal to the number of global events. Therefore, both formulations become similar. The global event-based formulation of LSP problem is given in the following. In Table 3.2, we give the definitions of the model variables and parameters.

$$\min \sum_{e=1}^{E} H_e \tag{3.12}$$

subject to

$$\sum_{e=1}^{E} \delta_{ke} = 1 \qquad k = 1, \ldots, K \tag{3.13}$$

$$\sum_{k=1}^{K} w_k \delta_{ke} \leq X \qquad e = 1, \ldots, E \tag{3.14}$$

$$H_e \geq h_k \delta_{ke} \qquad k = 1, \ldots, K; e = 1, \ldots, E \tag{3.15}$$

$$\delta_{ke} \in \{0, 1\} \qquad k = 1, \ldots, K; e = 1, \ldots, E \tag{3.16}$$

$$H_e \geq 0 \qquad e = 1, \ldots, E \tag{3.17}$$

In (3.12), we minimize the total sum of heights of all events. Constraints (3.13) are assignment constraints and guarantee that each item should be packed in an event.

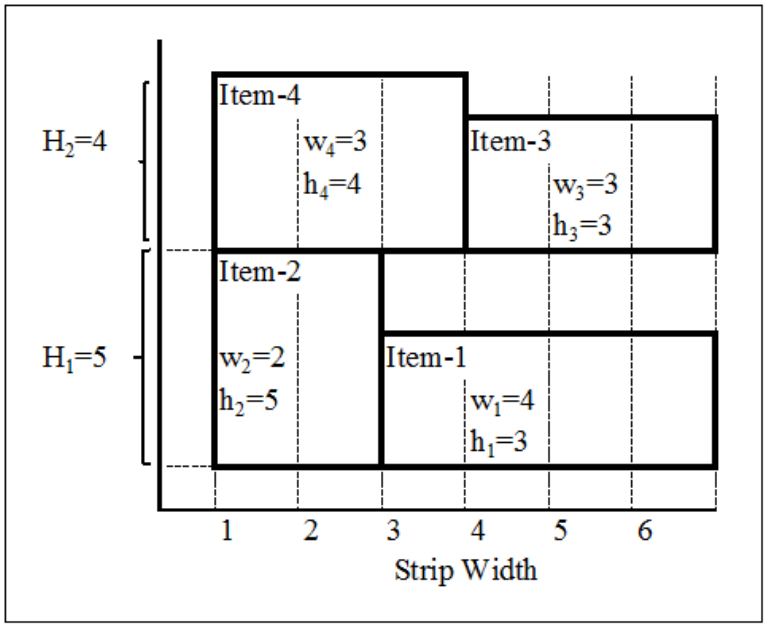

Figure 3.1. Representation of LSP solution.

In (3.14), at each event, the total sum of widths of items cannot exceed the strip width. Constraints (3.15) assert that an event's height should be larger than the largest item in that event.

An optimal solution is illustrated in Figure 3.1, which includes 4 items with given widths and heights. We try to pack these items into a strip with X=6. The optimal solution includes two events with H=9.

3.3.1.2. Formulation of Strip Packing Problems. A new model for 2D packing problems is proposed in [23]. This type of packing problems is discrete along the x-axis and continuous along the y-axis. In this section, we give the description of this model. An event corresponds to the assignment of the items to the strip. In addition to the parameters reported in Table 3.2, the decision variables of SP are given in Table 3.3.

Table 3.3. Decision variables for SP.

Variables	Definition
δ_{kxe}	Indicates whether item k's bottom left corner is assigned to event e on grid x or not
Y_{xe}	Y coordinate of event e on grid x
H	Strip height

$$\min H \tag{3.18}$$

subject to

$$\sum_{x=1}^{X-w_k+1} \sum_{e=1}^{E} \delta_{kxe} = 1 \qquad k = 1, \ldots, K \tag{3.19}$$

$$Y_{x(e+1)} - Y_{xe} \geq \sum_{k=1}^{K} \sum_{x'=x-w_k+1}^{x} h_k \delta_{kx'e} \qquad x = 1, \ldots, X; e = 1, \ldots, E-1 \tag{3.20}$$

$$H - Y_{xe} \geq \sum_{k=1}^{K} \sum_{x'=x-w_k+1}^{x} h_k \delta_{kx'e} \qquad x = 1, \ldots, X; e = E \tag{3.21}$$

$$\sum_{k=1}^{K} \sum_{x'=x-w_k+1}^{x} \delta_{kx'e} \leq 1 \qquad x = 1, \ldots, X; e = 1, \ldots, E \tag{3.22}$$

$$Y_{(x+1)e} \geq Y_{xe} \sum_{k=1}^{K} \sum_{x'=x-w_k+2}^{x} \delta_{kx'e} \qquad x = 1, \ldots, X-1; e = 1, \ldots, E \tag{3.23}$$

$$Y_{xe} \geq Y_{(x+1)e} \sum_{k=1}^{K} \sum_{x'=x-w_k+2}^{x} \delta_{kx'e} \qquad x = 1, \ldots, X-1; e = 1, \ldots, E \tag{3.24}$$

$$\delta_{kxe} \in \{0, 1\} \qquad k = 1, \ldots, K; x = 1, \ldots, X; e = 1, \ldots, E \tag{3.25}$$

$$Y_{xe} \geq 0 \qquad x = 1, \ldots, X; e = 1, \ldots, E \tag{3.26}$$

This formulation is constructed using USEB approach. The sections that the items are assigned are important in order to adjust the y coordinate across the sections.

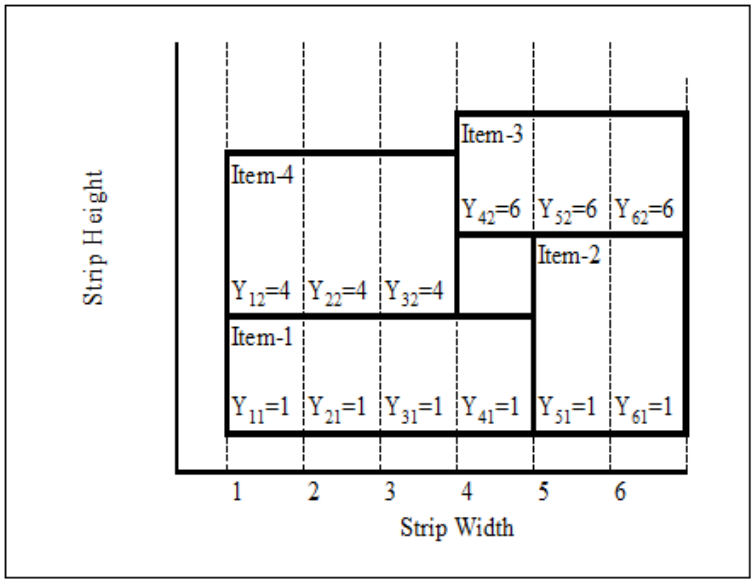

Figure 3.2. Representation of SP solution.

Therefore, another index, namely index x, is used to identify the section. Constraints (3.19) assure that every item should be assigned exactly once. Constraints (3.20) and (3.21) provide that if item k is assigned to section x and event e, along the width of this item the difference between the y-coordinates of two consecutive events $(Y_{x(e+1)} - Y_{xe})$ should be larger than the height of item k. Constraints (3.22) assure that the two items should not overlap.

$$-Y_{(x+1)e} + Y_{xe} \leq M\left(1 - \sum_{k=1}^{K} \sum_{x'=x-w_k+2}^{x} \delta_{kx'e}\right) \qquad x = 1, \ldots, X-1; e = 1, \ldots, E \tag{3.27}$$

$$Y_{(x+1)e} - Y_{xe} \leq M\left(1 - \sum_{k=1}^{K} \sum_{x'=x-w_k+2}^{x} \delta_{kx'e}\right) \qquad x = 1, \ldots, X-1; e = 1, \ldots, E \tag{3.28}$$

Finally, constraints (3.23)-(3.24) guarantee that y coordinate should remain the same along the width of the assigned rectangle. However, these constraints include non-linear terms. Hence, instead of constraints (3.23)-(3.24), the big-M constraints (3.27)-(3.28) are used in the MILP formulation of SP problem. In Figure 3.2, the items have the

same specifications. Although in LSP the optimal height of a strip is equal to 9, i.e. $H = 9$, in SP $H = 8$. Each unit includes two events. At the same event, y coordinates can be different in different sections. For example; while y coordinates of the first, second and third sections at event 2 are equal to 4, i.e. Y_{12}=Y_{22}=Y_{32}=4, y coordinates of the fourth, fifth and sixth sections at event 2 are equal to 6, i.e. Y_{42}=Y_{52}=Y_{62}=6.

3.3.2. Berth Allocation Problems

By using some additional constraints and modifying the objective function of SP, we can get another type of problem that is called berth allocation problem (BAP). In BAP, we have number of vessels whose arrival times and due times are known a priori. We try to assign vessels to berthing positions. After a vessels' arrival, quay cranes start to load /unload the containers from /onto the vessels. Hence, each vessel leaves the berth after its being processed and new vessels arrive to the emptied sections. A vessel is represented as a rectangle whose height is its processing time and whose width is its width. The berthing area is accepted as a linear facility and various vessels can berth simultaneously [34]. In our formulation, the x axis corresponds to the width of the berth. There are L equal size sections. The y axis measures the time. We have K vessels and E events and we try to minimize the total sum of the departure times of the vessels and lateness.

3.3.2.1. Formulation of Level Berth Allocation Problems. BAP is a complex problem and solving the problem usually requires considerable computational time. Instead, the simplified versions can be used to get approximate solutions. Therefore, we propose an event-based formulation for the level BAP.

Table 3.4. Parameters and decision variables for level BAP.

Parameters	Definition
w_k	Width of vessel k
p_k	Processing time of vessel k
a_k	Arrival time of vessel k
d_k	Due time of vessel k
f_k	Lateness penalty for vessel k
L	Width of berth
Variables	Definition
z_k	Deviation from due time of vessel k
c_k	Departure time of vessel k (completion time)
δ_{ke}	Indicates whether vessel k is assigned to event e or not
T_e	Time of event e

$$\min \sum_{k=1}^{K} c_k + \sum_{k=1}^{K} z_k f_k \tag{3.29}$$

subject to

$$\sum_{e=1}^{E} \delta_{ke} = 1 \qquad k = 1, \ldots, K \tag{3.30}$$

$$\sum_{k=1}^{K} w_k \delta_{ke} \leq L \qquad e = 1, \ldots, E \tag{3.31}$$

$$T_{e+1} - T_e \geq p_k \delta_{ke} \qquad k = 1, \ldots, K; e = 1, \ldots, E-1 \tag{3.32}$$

$$T_e \geq a_k \delta_{ke} \qquad k = 1, \ldots, K; e = 1, \ldots, E \tag{3.33}$$

$$c_k \geq (T_e + p_k)\delta_{ke} \qquad k = 1, \ldots, K; e = 1, \ldots, E \tag{3.34}$$

$$z_k \geq c_k - d_k \qquad k = 1, \ldots, K \tag{3.35}$$

$$\delta_{ke} \in \{0, 1\} \qquad k = 1, \ldots, K; e = 1, \ldots, E \tag{3.36}$$

$$T_e, c_k, z_k \geq 0 \qquad k = 1, \ldots, K; e = 1, \ldots, E \tag{3.37}$$

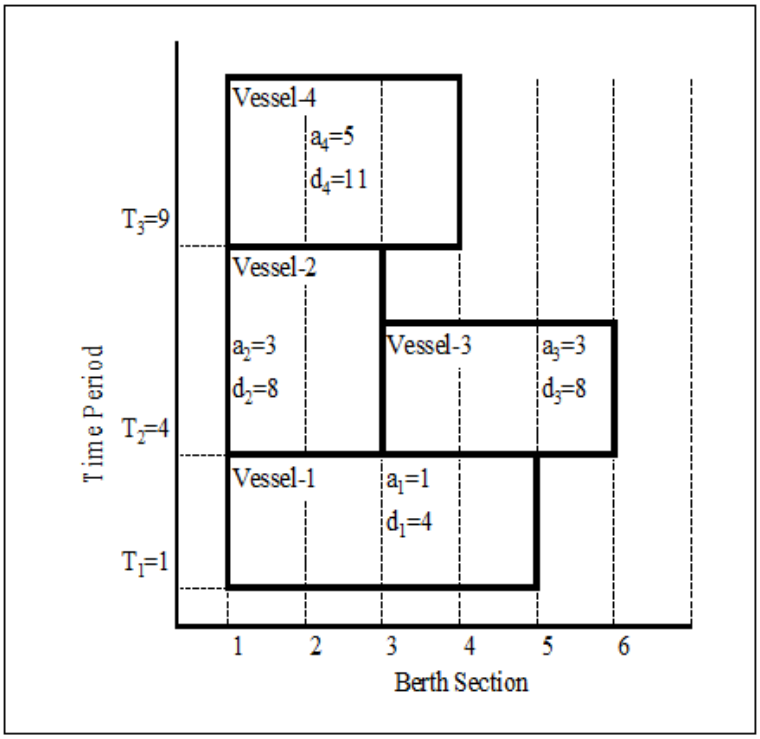

Figure 3.3. Representation of level BAP solution.

In this problem, the sequences of the events gain an importance in terms of optimal solution. We have levels that represent the global events. At each level, the sum of the widths of the vessels assigned at the same event should not exceed the berth width which is shown in constraints (3.31). If the sequencing of the events changes, the departure time of a vessel changes, hence the objective value changes. In order to determine the departure time of vessels, we need additional continuous variables that determine event times. Constraints (3.33) assure that if vessel k is included in event e, T_e should be larger than the arrival time of vessel k.

$$c_k \geq T_e + p_k - M(1 - \delta_{ke}) \qquad k = 1, \ldots, K; e = 1, \ldots, E \tag{3.38}$$

Constraints (3.34) ensure that the departure time of vessel k should be larger than the sum of T_e and p_k if it is assigned to event e. However, these constraints include non-linear terms. Hence, we use constraints (3.38) instead of (3.34) for linearization. In Figure 3.3, the items of Figure 3.1 are depicted with arrival and due times. Now,

Table 3.5. Decision variables for DBAP.

Variables	Definition
δ_{kle}	Indicates whether vessel k's bottom left corner is assigned to event e on berth section l or not
ST_{le}	Time of event e on section l

an optimal solution consists of three events.

3.3.2.2. Formulation of Berth Allocation Problem. In this part, we construct the formulations of DBAP obtained by using USEB and GEB approaches. In this version of BAP, which we call DBAP, we also penalize the deviations from the desired berthing positions. Hence additional parameter des_k represents the berth section desired by vessel k. In order to compare the results with the clock-based approach, the position assignment formulation based on [34] is used. It is given in Section A.2. In addition to variables and parameters given in Table 3.4, new variables used in DBAP are displayed in Table 3.5.

Unit-specific event-based formulation: In addition to constraints in SP, following constraints (3.45)-(3.47) are introduced to define the DBAP. Constraints (3.45) set the relation between the arrival time of the vessel and the time of section and event which the vessel is assigned to. If vessel k is assigned to event e on section l, time of event e on section l should be larger than the arrival time of vessel k. If vessel k is assigned to event e on section l, its departure time should be larger than or equal to the sum of the time spent by event e on section l and the processing time of vessel k. Therefore, the following constraints (3.46) insure this issue. In (3.47), the positive deviation from the due time is calculated.

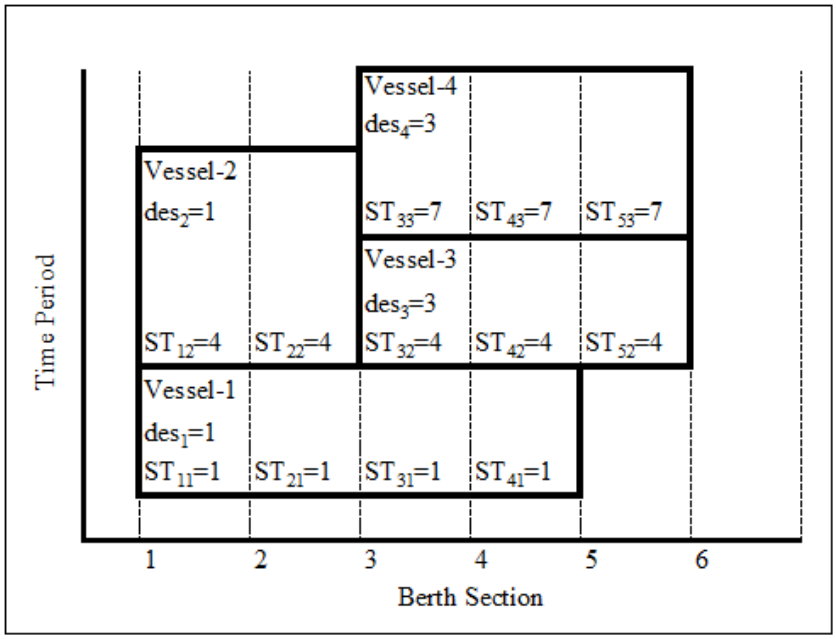

Figure 3.4. Representation of DBAP solution (USEB).

$$\min \sum_{k=1}^{K} c_k + \sum_{k=1}^{K} f_k z_k + \sum_{k=1}^{K} \sum_{l=1}^{L} \sum_{e=1}^{E} f_k |l - des_k| \delta_{kle} \tag{3.39}$$

subject to

$$\sum_{l=1}^{L-w_k+1} \sum_{e=1}^{E} \delta_{kle} = 1 \qquad k = 1, \ldots, K \tag{3.40}$$

$$\sum_{k=1}^{K} \sum_{l'=l-w_k+1}^{l} \delta_{kl'e} \leq 1 \qquad l = 1, \ldots, L; e = 1, \ldots, E \tag{3.41}$$

$$ST_{le} + \sum_{k=1}^{K} \sum_{l'=l-w_k+1}^{l} p_k \delta_{kl'e} \leq ST_{l(e+1)} \qquad k = 1, \ldots, K; e = 1, \ldots, E-1 \tag{3.42}$$

$$ST_{(l+1)e} \geq ST_{le} \sum_{k=1}^{K} \sum_{l'=l-w_k+2}^{l} \delta_{kl'e} \qquad l = 1, \ldots, L-1; e = 1, \ldots, E \tag{3.43}$$

$$ST_{le} \geq ST_{(l+1)e} \sum_{k=1}^{K} \sum_{l'=l-w_k+2}^{l} \delta_{kl'e} \qquad l = 1, \ldots, L-1; e = 1, \ldots, E \tag{3.44}$$

$$ST_{le} \geq a_k \delta_{kle} \qquad k = 1, \ldots, K; l = 1, \ldots, L; e = 1, \ldots, E \qquad (3.45)$$

$$c_k \geq (ST_{le} + p_k)\delta_{kle} \qquad k = 1, \ldots, K; l = 1, \ldots, L; e = 1, \ldots, E \qquad (3.46)$$

$$z_k \geq c_k - d_k \qquad k = 1, \ldots, K \qquad (3.47)$$

$$\delta_{kle} \in \{0, 1\} \qquad k = 1, \ldots, K; l = 1, \ldots, L; e = 1, \ldots, E \qquad (3.48)$$

$$ST_{le}, c_k \geq 0 \qquad k = 1, \ldots, K; l = 1, \ldots, L; e = 1, \ldots, E \qquad (3.49)$$

There are some differences between the clock and event-based formulations originated from constraints (A.3) in a clock-based formulation. In clock-based formulation, second type of binary variable σ_{kxy} provides the adjacency of vessel rectangles in both dimensions. In the event-based formulation, due to the continuous y axis, the time difference between two event points in this axis is indicated by continuous variable ST_{le} in constraints (3.42). The adjacency of continuous variables is provided by constraints (3.43) and (3.44). Constraints (3.43)-(3.44) and (3.46) are non-linear. Linear form of these constraints are given in the following constraints (3.50)-(3.52).

$$-ST_{(l+1)e} + ST_{le} \leq$$
$$M\left(1 - \sum_{k=1}^{K} \sum_{l'=l-w_k+2}^{l} \delta_{kl'e}\right) \qquad l = 1, \ldots, L-1; e = 1, \ldots, E \qquad (3.50)$$

$$ST_{(l+1)e} - ST_{le} \leq$$
$$M\left(1 - \sum_{k=1}^{K} \sum_{l'=l-w_k+2}^{l} \delta_{kl'e}\right) \qquad l = 1, \ldots, L-1; e = 1, \ldots, E \qquad (3.51)$$

$$c_k \geq ST_{le} + p_k -$$
$$M(1 - \delta_{kle}) \qquad k = 1, \ldots, K; l = 1, \ldots, L; e = 1, \ldots, E \qquad (3.52)$$

We replace constraints (3.45) and (3.52) with their strengthened versions. Constraints (3.53) and (3.54) are used in our computational results. Additionally, as the departure time of vessel k should be larger than or equal to the sum of arrival and processing

time of that vessel, we can derive the constraints (3.55).

$$ST_{le} \geq \sum_{k=1}^{K} \sum_{l'=l-w_k+1}^{l} a_k \delta_{kl'e} \qquad l = 1, \ldots, L; e = 1, \ldots E, \tag{3.53}$$

$$c_k \geq ST_{le} + p_k -$$
$$M(1 - \sum_{e'=e}^{E} \sum_{l'=l-w_k+1}^{l} \delta_{kl'e'}) \qquad k = 1, \ldots, K; l = 1, \ldots, L; e = 1, \ldots, E \tag{3.54}$$

$$c_k \geq a_k + p_k \qquad k = 1, \ldots, K \tag{3.55}$$

Improvements in the solution time is given in Chapter 6. Hence, MILP formulation of DBAP with USEB approach includes constraints (3.40)-(3.42), (3.47)-(3.49), (3.50)-(3.51) and (3.53)-(3.55).

Global event-based formulation: After constructing the USEB formulation, we can derive GEB formulation from USEB formulation.

$$\min \sum_{k=1}^{K} c_k + \sum_{k=1}^{K} f_k z_k +$$
$$\sum_{k=1}^{K} \sum_{l=1}^{L} \sum_{e=1}^{E} f_k |l - des_k| \delta_{kle} \tag{3.56}$$

subject to

$$\sum_{l=1}^{L-w_k+1} \sum_{e=1}^{E} \delta_{kle} = 1 \qquad k = 1, \ldots, K \tag{3.57}$$

$$ST_{l(e_2)} \geq \left(ST_{l(e_1)} + \sum_{k=1}^{K} \sum_{l'=l-w_k+1}^{l} p_k \delta_{kl'(e_1)} \right)$$
$$\left(\sum_{k=1}^{K} \sum_{l'=l-w_k+1}^{l} \delta_{kl'(e_2)} \right) \qquad l = 1, \ldots, L; e_1 = 1, \ldots, E-1;$$
$$e_2 = e_1 + 1, \ldots, E \tag{3.58}$$

$$\sum_{k=1}^{K} \sum_{l'=l-w_k+1}^{l} \delta_{kl'e} \leq 1 \qquad l = 1, \ldots, L; e = 1, \ldots, E \tag{3.59}$$

$$ST_{(l+1)e} \geq ST_{le} \sum_{k=1}^{K} \sum_{l'=l-w_k+2}^{l} \delta_{kl'e} \qquad l = 1, \ldots, L-1; e = 1, \ldots E \tag{3.60}$$

$$ST_{le} \geq ST_{(l+1)e} \sum_{k=1}^{K} \sum_{l'=l-w_k+2}^{l} \delta_{kl'e} \qquad l = 1, \ldots, L-1; e = 1, \ldots E \tag{3.61}$$

$$ST_{le} \geq \sum_{k=1}^{K} \sum_{l'=l-w_k+1}^{l} a_k \delta_{kl'e} \qquad l = 1, \ldots, L; e = 1, \ldots, E \tag{3.62}$$

$$T_{e+1} \geq T_e \qquad e = 1, \ldots, E-1 \tag{3.63}$$

$$T_e \geq ST_{le} \sum_{k=1}^{K} \sum_{l'=l-w_k+1}^{l} \delta_{kl'e} \qquad l = 1, \ldots, L; e = 1, \ldots, E \tag{3.64}$$

$$ST_{le} \geq T_e \sum_{k=1}^{K} \sum_{l'=l-w_k+1}^{l} \delta_{kl'e} \qquad l = 1, \ldots, L; e = 1, \ldots, E \tag{3.65}$$

$$c_k \geq (ST_{le} + p_k)\delta_{kle} \qquad k = 1, \ldots, K; l = 1, \ldots, L; e = 1, \ldots, E \tag{3.66}$$

$$z_k \geq c_k - d_k \qquad k = 1, \ldots, K \tag{3.67}$$

$$\delta_{kle} \in \{0, 1\} \qquad k = 1, \ldots, K; l = 1, \ldots, L; e = 1, \ldots, E \tag{3.68}$$

$$ST_{le}, T_e, c_k \geq 0 \qquad k = 1, \ldots, K; l = 1, \ldots, L; e = 1, \ldots, E \tag{3.69}$$

Since we should define the events as globally, we introduce the new continuous variable T_e for each global event e that represents the time of event e across all units. Constraints (3.64) and (3.65) are for synchronizing the continuous variables. If a vessel is assigned to event e and berth section l, the event occurs and time T_e remains equal for all sections. In USEB approach, the time remains the same only along the width of the assigned vessel. Constraints (3.63) are for sequencing. The structures of constraints (3.42) in unit based and (3.58) in global event based approach are slightly different. In USEB, vessels occupying a given section l are assigned to events which are ordered consecutively. However, in GEB vessels occupying a given section l are assigned to events which are ordered increasingly but not consecutively. In the GEB approach, the consecutive global events can occur in non-overlapping sections. Therefore, in order to determine the time difference between two events at given section l as in constraints (3.58) we should consider all events that occur before event e on section l. In GEB

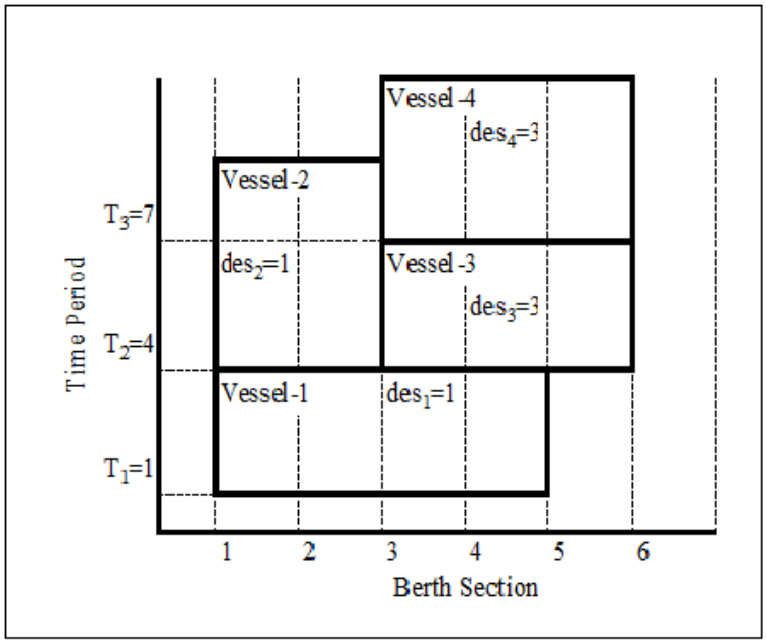

Figure 3.5. Representation of DBAP solution (GEB).

formulation, constraints (3.58), (3.60), (3.61), (3.64), (3.65) and (3.66) include non-linear terms. In MILP of GEB formulation, constraints (3.58), (3.64) and (3.65) are replaced with linear constraints (3.70), (3.71) and (3.72) and also constraints (3.50), (3.51) and (3.52) are used in order to linearize the constraints (3.60), (3.61) and (3.66).

$$ST_{l(e_1)} + \sum_{k=1}^{K} \sum_{l'=l-w_k+1}^{l} p_k \delta_{kl'(e_1)} \leq ST_{l(e_2)} +$$

$$M(1 - \sum_{k=1}^{K} \sum_{l'=l-w_k+1}^{l} \delta_{kl'(e_2)}) \qquad k = 1, \ldots, K; e_1 = 1, \ldots, E-1;$$

$$e_2 = e_1 + 1, \ldots, E \qquad (3.70)$$

$$T_e - ST_{le} \leq -M(\sum_{k=1}^{K} \sum_{l'=l-w_k+1}^{l} \delta_{kl'e} - 1) \qquad l = 1, \ldots, L; e = 1, \ldots, E \qquad (3.71)$$

$$-T_e + ST_{le} \leq 0 \qquad l = 1, \ldots, L; e = 1, \ldots, E \qquad (3.72)$$

4. ENHANCEMENTS OF THE FORMULATIONS

The LP relaxation of the event-based formulations is generally weaker than the clock-based formulations. In this chapter, we introduce new inequalities in order to strengthen the formulations because of the continuous variables and big-M constraints.

4.1. Location and Scheduling Problem

4.1.1. Symmetry-Breaking Constraints

If we change the placements of the events, we can obtain alternative optimal solutions. Sherali and Smith [35] state the importance of reducing symmetric solutions. If we have five events at optimality, we have 5! solutions based on the permutations of the events. By inspiring from Sherali and Smith [35], we offer new constraints in order to reduce the number of alternative solutions. Constraints (4.1) order the duration of the events where the first event's duration is the longest and the last event's duration is the shortest.

$$W_{e+1} \leq W_e \qquad e = 1, \ldots, E-1 \tag{4.1}$$

Constraints (4.2) force a hierarchy on the number of active sensors at each event.

$$\sum_{i=1}^{I}\sum_{n=1}^{N} z_{ine} \geq \sum_{i=1}^{I}\sum_{n=1}^{N} z_{in(e+1)} \qquad e = 1, \ldots, E-1 \tag{4.2}$$

Inequalities (4.3) sequence the events based on the sum of the point indices that are assigned to any event.

$$\sum_{i=1}^{I}\sum_{n=1}^{N} i z_{ine} \geq \sum_{i=1}^{I}\sum_{n=1}^{N} i z_{in(e+1)} \qquad e = 1, \ldots, E-1 \tag{4.3}$$

Based on the similar approaches above, we can hierarchically sequence the energy or cost consumption for events.

4.1.2. Tightening the Constraints

In order to strengthen the LP relaxation of the formulation, constraints (4.4) and (4.5) are also proposed.

$$\sum_{e'=1}^{e} s_{ine'} \leq \sum_{e'=1}^{e} W_{e'} \qquad i=1,\ldots,I, n=1;\ldots,N; e=1,\ldots,E \qquad (4.4)$$

$$\sum_{e'=e}^{E} s_{ine'} \leq T - \sum_{e'=1}^{e-1} W_{e'} \qquad i=1,\ldots,I; n=1,\ldots,N; e=1,\ldots,E \qquad (4.5)$$

Constraints (4.4) imply total active time of a type n sensor placed at point i that elapses until event e occurs can be at most the lifetime. Constraints (4.5) limit total active time of a type n sensor placed at point i elapsed after event e occurs. The right part of the constraints (4.5) implies the amount of time left which is calculated by substracting the sum of duration of events until event e occurs from the time horizon.

Türkoğulları [32] generates the valid inequalities for the clock-based formulation of location, scheduling and routing problem. Same constraints can be adapted for the event-based formulation as well. The readers can find the proof of these valid inequalities in [32]. We modify them by replacing z_{int} with s_{ine} and lifetime with $\sum_{e=1}^{E} W_e$.

$$\sum_{i=1}^{I}\sum_{n=1}^{N} c_n z_{ine} \geq C_{lb} \qquad e=1,\ldots,E \qquad (4.6)$$

Constraints (4.6) put a lower bound for the cost of active sensors in one event. C_{lb} is

the optimum value of the following binary integer programming problem.

$$\min \sum_{i=1}^{I} \sum_{n=1}^{N} c_n x_{in} \tag{4.7}$$

subject to

$$\sum_{i=1}^{I} \sum_{n=1}^{N} a_{ijn} x_{in} \geq d_j \qquad j = 1, \ldots, J \tag{4.8}$$

$$x_{in} \in \{0, 1\} \qquad i = 1, \ldots, I; n = 1, \ldots, N \tag{4.9}$$

Next inequalities constitute lower bounds on the energy consumptions:

$$\sum_{i=1}^{I} \sum_{n=1}^{N} E_n x_{in} \geq \sum_{e=1}^{E} E_{lb} W_e, \tag{4.10}$$

$$\sum_{e=1}^{E} \sum_{i=1}^{I} \sum_{n=1}^{N} e_n s_{ine} \geq \sum_{e=1}^{E} E_{lb} W_e. \tag{4.11}$$

Here, E_{lb} is the minimum required energy value that satisfies the quality constraints in one time unit. Basically, it is the optimum value of the binary integer programming problem:

$$\min \sum_{i=1}^{I} \sum_{n=1}^{N} e_n x_{in} \tag{4.12}$$

subject to

$$\sum_{i=1}^{I} \sum_{n=1}^{N} a_{ijn} x_{in} \geq d_j \qquad j = 1, \ldots, J \tag{4.13}$$

$$x_{in} \in \{0, 1\} \qquad i = 1, \ldots, I; n = 1, \ldots, N. \tag{4.14}$$

The left hand side of (4.10) is the total initial energy of the placed sensors. Hence, the left hand side of (4.11) is the total energy spent during E events. Both of them have to be larger than the lower bound on the total energy consumed over the lifetime of the network.

4.2. Desired Berth Allocation Problem

In order to strengthen the LP relaxation of the event-based mathematical formulations, Maravelias and Grossmann [12] propose additional constraints in batch plants. In 2D problems, the generated inequalities strengthen the formulation. In order to be coherent with the nomenclature, we use DBAP parameters and decision variables in the constraints. However, proposed constraints can be applied in all the 2D problems. Constraints (4.16) and (4.17) are generated using USEB approach and constraints (4.18) and (4.19) are generated using GEB approach. Constraints (4.15) can be applied for both of the approaches.

Constraints (4.15) force the sum of processing times of all vessels occupying section l during all events to be less than the time horizon:

$$\sum_{e=1}^{E}\sum_{k=1}^{K}\sum_{l'=l-w_k+1}^{l} p_k\delta_{kl'e} \leq T \qquad l=1,\ldots,L. \tag{4.15}$$

Similarly, constraints (4.16) limit the sum of processing times of the vessels occupying section l after event e occurs:

$$\sum_{e'=e}^{E}\sum_{k=1}^{K}\sum_{l'=l-w_k+1}^{l} p_k\delta_{kl'e'} \leq T - ST_{le} \qquad l=1,\ldots,L; e=1,\ldots,E. \tag{4.16}$$

Constraints (4.17) ensure that the sum of processing times of all vessels occupying section l before the eth event occurs is less than the time that event e occurs at section l (i.e. ST_{le});

$$\sum_{e'=1}^{e-1}\sum_{k=1}^{K}\sum_{l'=l-w_k+1}^{l} p_k\delta_{kl'e'} \leq ST_{le} \qquad l=1,\ldots,L; e=1,\ldots,E. \tag{4.17}$$

Similar constraints can be also derived using GEB approach after making only minor changes on the constraints. Constraints (4.18) guarantee that the sum of pro-

cessing times of all vessels occupying section l after event e occurs is less than the amount of time left, if event e occurs at one of the sections $l - w_k + 1$ through l;

$$\sum_{e'=e}^{E}\sum_{k=1}^{K}\sum_{l'=l-w_k+1}^{l} p_k\delta_{kl'e'} \leq T - ST_{le} + M(1 - \sum_{k=1}^{K}\sum_{l'=l-w_k+1}^{l}\delta_{kl'e}) \qquad l = 1,\ldots,L; e = 1,\ldots E. \tag{4.18}$$

The sum of processing times of all vessels occupying section l before the eth event occurs is less than the time that event e occurs at section l (i.e. ST_{le}), if one of the vessels occupies section l at event e:

$$\sum_{e'=1}^{e-1}\sum_{k=1}^{K}\sum_{l'=l-w_k+1}^{l} p_k\delta_{kl'e'} \leq ST_{le} + M(1 - \sum_{k=1}^{K}\sum_{l'=l-w_k+1}^{l}\delta_{kl'e}) \qquad l = 1,\ldots,L; e = 1,\ldots,E. \tag{4.19}$$

Although we do not utilize the following constraints in our computations, they are given for additional information. In the event-based formulations of batch processes, we can put a lower and upper bound on the number of operations in one unit. For DBAP, inequalities (4.20) limit the minimum number of vessels (V_{min}) that should be assigned in one of the events and constraints (4.21) put an upper bound on the maximum number of vessels (V_{max}) that are assigned to an event. The maximum number of vessels in one event can be calculated by assigning relatively small vessels until the berth has no place for other vessels. The minimum event number can be calculated based on E and V_{max}. For example, if we have 6 vessels and V_{max}=4 and

E=2, then V_{min} will be 2;

$$\sum_{k=1}^{K}\sum_{l=1}^{L}\delta_{kle} \geq V_{min} \qquad e = 1, \ldots, E, \tag{4.20}$$

$$\sum_{k=1}^{K}\sum_{l=1}^{L}\delta_{kle} \leq V_{max} \qquad e = 1, \ldots, E. \tag{4.21}$$

In constraints (4.22), for each section, the total number of vessels assigned to event e should be larger than the ones assigned to the prior event $e-1$;

$$\sum_{e'=1}^{e-1}\sum_{k=1}^{K}\delta_{kle'} \leq \sum_{e''=1}^{e}\sum_{k=1}^{K}\delta_{kle''} \qquad l = 1, \ldots, L; e = 1, \ldots, E. \tag{4.22}$$

5. A BRANCH AND PRICE ALGORITHM

As stated in Section 2.4, different event-based decomposition techniques are applied in scheduling problems. However, they try to generate a solution using heuristics. In this chapter, we develop a branch and price algorithm to solve the problem exactly.

In the proposed approach, we decompose the problem so that each subproblem finds an optimal schedule for an event. Hence, the problem turns out to be the selection of a collection of subsets from a large family in order to choose the best possible combination of events. Each new column corresponds to an event schedule that satisfies constraints of the subproblem. In LASP, an event's schedule corresponds to a set of active sensors that satisfies the quality constraints for each point. In order to simplify the problem in addition to E subproblems, one additional subproblem that determines the placed sensors based on the cost constraints is introduced. In DBAP, the optimal placement of vessels for an event corresponds to an event schedule. In the decomposition approach, DBAP includes E subproblems.

We reformulate the problems in terms of the weights of the integer solutions of the subproblems. At each node of the branch and bound tree, we apply column generation procedure to solve the restricted master problem (RMP). We obtain optimal dual variables from the LP relaxation of the RMP to solve pricing problems. Until the optimality condition is satisfied, we add new columns to the RMP. The LP relaxation solution of the restricted master problem may not be integer. Therefore, we branch on the fractional integer variables until integrality is obtained.

5.1. Reformulation of the Location and Scheduling Problem

For each event e, we let S_{1e} denote the set of solutions satisfying:

$$\sum_{i=1}^{I}\sum_{n=1}^{N} a_{ijn} z_{ine} \geq d_j \qquad j = 1, \ldots, J, \tag{5.1}$$

$$s_{ine} \leq W_e \qquad i = 1, \ldots, I; n = 1, \ldots, N, \tag{5.2}$$

$$s_{ine} \leq M z_{ine} \qquad i = 1, \ldots, I; n = 1, \ldots, N, \tag{5.3}$$

$$s_{ine} \geq W_e - M(1 - z_{ine}) \qquad i = 1, \ldots, I; n = 1, \ldots, N, \tag{5.4}$$

$$s_{ine}, W_e \geq 0 \qquad i = 1, \ldots, I; n = 1, \ldots, N, \tag{5.5}$$

$$z_{ine} \in \{0, 1\} \qquad i = 1, \ldots, I; n = 1, \ldots, N. \tag{5.6}$$

We also let S_2 denote the set of solutions satisfying:

$$\sum_{n=1}^{N} c_n \sum_{i=1}^{I} x_{in} \leq B \tag{5.7}$$

$$x_{in} \in \{0, 1\} \qquad i = 1, \ldots, I; n = 1, \ldots, N. \tag{5.8}$$

Then, we can restate the original problem as:

$$\max \sum_{e=1}^{E} W_e \tag{5.9}$$

subject to

$$(3.3), (3.4), (4.1), (4.4), (4.5) \tag{5.10}$$

$$(W_e, s_{ine}, z_{ine}) \in S_{1e} \qquad i = 1, \ldots, I; n = 1, \ldots, N; e = 1, \ldots, E \tag{5.11}$$

$$x_{in} \in S_2 \qquad i = 1, \ldots, I; n = 1, \ldots, N. \tag{5.12}$$

For subproblem e, the variable $z_{ine} \in S_{1e}$ if and only if it can be represented as:

$$z_{ine} = \sum_{p_1=1}^{P_{1e}} \lambda_e^{p_1} z_{ine}^{p_1} \qquad i=1,\ldots,I; n=1,\ldots,N; e=1,\ldots,E, \tag{5.13}$$

$$\sum_{p_1=1}^{P_{1e}} \lambda_e^{p_1} = 1 \qquad e=1,\ldots,E, \tag{5.14}$$

$$\lambda_e^{p_1} \in \{0,1\} \qquad e=1,\ldots,E; p_1=1,\ldots,P_{1e}. \tag{5.15}$$

Similarly, for subproblem $(E+1)$, $x_{in} \in S_2$ if and only if it can be represented as:

$$x_{in} = \sum_{p_2=1}^{P_2} \lambda^{p_2} x_{in}^{p_2} \qquad i=1,\ldots,I; n=1,\ldots,N, \tag{5.16}$$

$$\sum_{p_2=1}^{P_2} \lambda^{p_2} = 1, \tag{5.17}$$

$$\lambda^{p_2} \in \{0,1\} \qquad p_2=1,\ldots,P_2. \tag{5.18}$$

Here, P_{1e} is the number of elements (i.e. size) of S_{1e} and P_2 is the number of elements in S_2. Let $\lambda_e^{p_1}$ and λ^{p_2} be the new decision variables. $\lambda_e^{p_1}$=1 if the sensors in the p_1th solution of event e are active, $\lambda_e^{p_1}$=0 otherwise. λ^{p_2}=1 if the sensors in the p_2th solution of subproblem $(E+1)$ is placed on the network, λ^{p_2}=0 otherwise. If we replace z_{ine} and x_{in} with these new definitions, the master problem can be stated as:

$$\max \sum_{e=1}^{E} \sum_{p_1=1}^{P_{1e}} W_e^{p_1} \lambda_e^{p_1} \tag{5.19}$$

subject to

$$\sum_{e=1}^{E} \sum_{p_1=1}^{P_{1e}} s_{ine}^{p_1} \lambda_e^{p_1} - \sum_{p_2=1}^{P_2} \left\lfloor \frac{E_n}{e_n} \right\rfloor x_{in}^{p_2} \lambda^{p_2} \leq 0 \qquad i=1,\ldots,I; n=1,\ldots,N \tag{5.20}$$

$$\sum_{p_1=1}^{P_{1e}} z_{ine}^{p_1} \lambda_e^{p_1} - \sum_{p_2=1}^{P_2} x_{in}^{p_2} \lambda^{p_2} \leq 0 \qquad i=1,\ldots,I; n=1,\ldots,N; e=1,\ldots,E \tag{5.21}$$

$$\sum_{p_1=1}^{P_{1(e+1)}} W_{e+1}^{p_1}\lambda_{e+1}^{p_1} -$$
$$\sum_{p_1=1}^{P_{1e}} W_{e}^{p_1}\lambda_{e}^{p_1} \leq 0 \qquad e = 1,\ldots,E-1 \tag{5.22}$$

$$\sum_{e'=1}^{e}\sum_{p_1=1}^{P_{1e'}} s_{ine'}^{p_1}\lambda_{e'}^{p_1} -$$
$$\sum_{e'=1}^{e}\sum_{p_1=1}^{P_{1e'}} W_{e'}^{p_1}\lambda_{e'}^{p_1} \leq 0 \qquad i = 1,\ldots,I; n = 1,\ldots,N; e = 1,\ldots,E \tag{5.23}$$

$$\sum_{e'=e}^{E}\sum_{p_1=1}^{P_{1e'}} s_{ine'}^{p_1}\lambda_{e'}^{p_1} +$$
$$\sum_{e'=1}^{e-1}\sum_{p_1=1}^{P_{1e'}} W_{e'}^{p_1}\lambda_{e'}^{p_1} \leq T \qquad i = 1,\ldots,I; n = 1,\ldots,N; e = 1,\ldots,E \tag{5.24}$$

$$\sum_{p_1=1}^{P_{1e}} \lambda_{e}^{p_1} = 1 \qquad e = 1,\ldots,E \tag{5.25}$$

$$\sum_{p_2=1}^{P_2} \lambda^{p_2} = 1 \tag{5.26}$$

$$\lambda_e^{p_1}, \lambda^{p_2} \in \{0,1\} \qquad e = 1,\ldots,E; p_1 = 1,\ldots,P_{1e}; p_2 = 1,\ldots,P_{2e}. \tag{5.27}$$

This problem, which is the integer master problem, cannot be solved directly due to the exponential number of columns. We solve the LP relaxation of RMP by using column generation method. Additional columns for the RMP can be generated by solving the pricing problem. We can state the first E pricing subproblems for each event e as

$$\min z_{1e} \tag{5.28}$$

subject to

$$(5.1) - (5.6). \tag{5.29}$$

Here the objective function z_{1e} is

$$z_{1e} = \begin{cases} \sum_{i=1}^{I}\sum_{n=1}^{N}\left(\alpha_{in} + \sum_{e'=e}^{E}\mu_{ine'} + \sum_{e'=1}^{e}\nu_{ine'}\right)s_{ine} + \\ \sum_{i=1}^{I}\sum_{n=1}^{N}\gamma_{ine}z_{ine} + \left(-\epsilon_e - \sum_{i=1}^{I}\sum_{n=1}^{N}\sum_{e'=e}^{E}\mu_{ine'} + \right. \\ \left. \sum_{i=1}^{I}\sum_{n=1}^{N}\sum_{e'=e+1}^{E}\nu_{ine'} - 1\right)W_e, & e = 1 \\ \\ \sum_{i=1}^{I}\sum_{n=1}^{N}\left(\alpha_{in} + \sum_{e'=e}^{E}\mu_{ine'} + \sum_{e'=1}^{e}\nu_{ine'}\right)s_{ine} + \\ \sum_{i=1}^{I}\sum_{n=1}^{N}\gamma_{ine}z_{ine} + (\epsilon_{e-1} - \epsilon_e - \\ \left. \sum_{i=1}^{I}\sum_{n=1}^{N}\sum_{e'=e}^{E}\mu_{ine'} + \sum_{i=1}^{I}\sum_{n=1}^{N}\sum_{e'=e+1}^{E}\nu_{ine'} - 1\right)W_e, & e = 2,\ldots,E-1 \\ \\ \sum_{i=1}^{I}\sum_{n=1}^{N}\left(\alpha_{in} + \sum_{e'=e}^{E}\mu_{ine'} + \sum_{e'=1}^{e}\nu_{ine'}\right)s_{ine} + \\ \sum_{i=1}^{I}\sum_{n=1}^{N}\gamma_{ine}z_{ine} + \left(\epsilon_{e-1} - \sum_{i=1}^{I}\sum_{n=1}^{N}\sum_{e'=e}^{E}\mu_{ine'} + \right. \\ \left. \sum_{i=1}^{I}\sum_{n=1}^{N}\sum_{e'=e+1}^{E}\nu_{ine'} - 1\right)W_e, & e = E. \end{cases} \tag{5.30}$$

The last subproblem is

$$\min -\sum_{i=1}^{I}\sum_{n=1}^{N}\alpha_{in}\left\lfloor\frac{E_n}{e_n}\right\rfloor x_{in} - \sum_{i=1}^{I}\sum_{n=1}^{N}\sum_{e=1}^{E}\gamma_{ine}x_{in} \tag{5.31}$$

subject to

$$(5.7) - (5.8). \tag{5.32}$$

In all these subproblems, $\boldsymbol{\alpha}$, $\boldsymbol{\gamma}$, $\boldsymbol{\epsilon}$, $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$ are the dual variables corresponding to constraints (5.20), (5.21), (5.22), (5.23) and (5.24). $\boldsymbol{\zeta}$ and κ represent the dual variables of the convexity constraints (5.25) and (5.26). The master problem includes the coupling constraints and the convexity constraints. By taking the optimal values of dual variables corresponding to the coupling and convexity constraints in the LP relaxation of RMP, we solve the pricing subproblems. We are looking for the minimum

of $\min_{e=1,\ldots,E}\{z_{1e}+\zeta_e\}$ and $z_2+\kappa$. If it is smaller than 0, we continue adding new columns to our restricted master problem by taking the dual variables of the LP relaxation of RMP. Note that, the LP relaxation of the master problem provides a bound at least as tight as the bound provided by the LP relaxation value of the original problem.

5.2. Reformulation of the Desired Berth Allocation Problem

For each event e, we let S'_e denote the set of solutions satisfying

$$\sum_{k=1}^{K}\sum_{l'=l-w_k+1}^{l}\delta_{kle}\leq 1 \qquad l=1,\ldots,L \tag{5.33}$$

$$-ST_{(l+1)e}+ST_{le}\leq M\left(1-\sum_{k=1}^{K}\sum_{l'=l-w_k+2}^{l}\delta_{kl'e}\right) \qquad l=1,\ldots,L-1 \tag{5.34}$$

$$ST_{(l+1)e}-ST_{le}\leq M\left(1-\sum_{k=1}^{K}\sum_{l'=l-w_k+2}^{l}\delta_{kl'e}\right) \qquad l=1,\ldots,L-1 \tag{5.35}$$

$$ST_{le}\geq\sum_{k=1}^{K}\sum_{l'=l-w_k+1}^{l}a_k\delta_{kl'e} \qquad k=1,\ldots,K;l=1,\ldots,L \tag{5.36}$$

$$c_k\geq ST_{le}+p_k-M\left(1-\sum_{l'=l-w_k+1}^{l}\delta_{kle}\right) \qquad k=1,\ldots,K;l=1,\ldots,L \tag{5.37}$$

$$c_k\geq\sum_{l=1}^{L}(a_k+p_k)\delta_{kle} \qquad k=1,\ldots,K \tag{5.38}$$

$$c_k\leq M\sum_{l=1}^{L}\delta_{kle} \qquad k=1,\ldots,K \tag{5.39}$$

$$z_k\geq c_k-d_k \qquad k=1,\ldots,K \tag{5.40}$$

$$\delta_{kle}\in\{0,1\} \qquad k=1,\ldots,K;l=1,\ldots,L \tag{5.41}$$

$$ST_{le}\geq 0 \qquad l=1,\ldots,L \tag{5.42}$$

$$c_k,z_k\geq 0 \qquad k=1,\ldots,K. \tag{5.43}$$

Then, we restate the original problem as

$$\min \sum_{k=1}^{K} c_k + \sum_{k=1}^{K} f_k z_k + \sum_{k=1}^{K} f_k |l - des_k| \tag{5.44}$$

subject to

$$(3.40), (3.42), (4.15), (4.16), (4.17) \tag{5.45}$$

$$(c_k, z_k, ST_{le}, \delta_{kle}) \in S'_e \qquad k = 1, \ldots, K; l = 1, \ldots, L; e = 1, \ldots, E. \tag{5.46}$$

The variable $\delta_{kle} \in S'_e$ if and only if it is represented as

$$\delta_{kle} = \sum_{p=1}^{P'_e} \lambda_e^p \delta_{kle}^p \qquad k = 1, \ldots, K; l = 1, \ldots, L; e = 1, \ldots, E, \tag{5.47}$$

$$\sum_{p=1}^{P'_e} \lambda_e^p = 1 \qquad e = 1, \ldots, E, \tag{5.48}$$

$$\lambda_e^p \in \{0, 1\} \qquad e = 1, \ldots, E; p = 1, \ldots, P'_e. \tag{5.49}$$

P'_e denotes the number of elements in S'_e. Notice that λ_e^p is the decision variable of the master problem. $\lambda_e^p = 1$ if the pth solution of the eth event schedule is included in the solution. Replacing δ_{kle} with the definitions given above, we obtain the following problem, which is called as the integer master problem.

$$\min \sum_{k=1}^{K} \sum_{p=1}^{P'_e} c_k^p \lambda_e^p + \sum_{k=1}^{K} \sum_{p=1}^{P'_e} f_k z_k^p \lambda_e^p + \sum_{k=1}^{K} \sum_{l=1}^{L} \sum_{e=1}^{E} \sum_{p=1}^{P'_e} f_k |l - des_k| \lambda_e^p \delta_{kle}^p \tag{5.50}$$

subject to

$$\sum_{l=1}^{L-w_k+1} \sum_{e=1}^{E} \sum_{p=1}^{P'_e} \delta_{kle}^p \lambda_e^p = 1 \qquad k = 1, \ldots, K \tag{5.51}$$

$$\sum_{p=1}^{P'_e} ST^p_{le}\lambda^p_e + \sum_{k=1}^{K}\sum_{l'=l-w_k+1}^{l}\sum_{p=1}^{P'_e}\delta^p_{kl'e}\lambda^p_e p_k - \sum_{p=1}^{P'_{e+1}} ST^p_{l(e+1)}\lambda^p_{e+1} \leq 0 \qquad l=1,\ldots,L; e=1,\ldots,E-1 \quad (5.52)$$

$$\sum_{e=1}^{E}\sum_{p=1}^{P'_e}\sum_{k=1}^{K}\sum_{l'=l-w_k+1}^{l} p_k\delta^p_{kl'e}\lambda^p_e \leq T \qquad l=1,\ldots,L \quad (5.53)$$

$$\sum_{e'=e}^{E}\sum_{p=1}^{P'_{e'}}\sum_{k=1}^{K}\sum_{l'=l-w_k+1}^{l} p_k\delta^p_{kl'e'}\lambda^p_{e'} + \sum_{p=1}^{P'_e} ST^p_{le}\lambda^p_e \leq T \qquad l=1,\ldots,L; e=1,\ldots,E \quad (5.54)$$

$$\sum_{e'=1}^{e-1}\sum_{p=1}^{P'_{e'}}\sum_{k=1}^{K}\sum_{l'=l-w_k+1}^{l} p_k\delta^p_{kl'e'}\lambda^p_{e'} - \sum_{p=1}^{P'_e} ST^p_{le}\lambda^p_e \leq 0 \qquad l=1,\ldots,L; e=1,\ldots,E \quad (5.55)$$

$$\sum_{p=1}^{P'_e}\lambda^p_e = 1 \qquad e=1,\ldots,E \quad (5.56)$$

$$\lambda^p_e \in \{0,1\} \qquad e=1,\ldots,E; p=1,\ldots,P'_e \quad (5.57)$$

It is important to say that we modify constraints (3.55) in the original formulation and replaced them with constraints (5.38), since if the vessel is assigned, its departure time should be larger than or equal to the sum of the arrival and processing times in a given event schedule. Otherwise, the departure time of that vessel should be equal to zero as shown in constraints (5.39).

Due to its large number of variables, LP relaxation of the integer master problem is solved by column generation technique. We associate the dual variables $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$, $\boldsymbol{\xi}$,

$\boldsymbol{\pi}$ and $\boldsymbol{\omega}$ with constraints (5.51)-(5.55) and express the subproblem for event e as

$$\max z_{2e} \tag{5.58}$$

$$\text{subject to}$$

$$(5.33) - (5.43). \tag{5.59}$$

Here the objective function z_{2e} is defined as

$$z_{2e} = \begin{cases} \sum_{k=1}^{K}\sum_{l=1}^{L}\sum_{l'=l-w_k+1}^{l} p_k \left(\theta_{le} + \xi_l + \sum_{e'=1}^{e} \pi_{le'} + \sum_{e'=e+1}^{E} \omega_{le'}\right) \delta_{kl'e} + \sum_{k=1}^{K}\sum_{l=1}^{L-w_k+1} \beta_k \delta_{kle} + \\ \sum_{l=1}^{L} \left(\theta_{le} + \pi_{le} - \omega_{le}\right) ST_{le} - \\ \sum_{k=1}^{K} \left(c_k + f_k z_k + \sum_{l=1}^{L} f_k |l - des_k|\right) \delta_{kle}, \quad e = 1 \\ \\ \sum_{k=1}^{K}\sum_{l=1}^{L}\sum_{l'=l-w_k+1}^{l} p_k \left(\theta_{le} + \xi_l + \sum_{e'=1}^{e} \pi_{le'} + \sum_{e'=e+1}^{E} \omega_{le'}\right) \delta_{kl'e} + \sum_{k=1}^{K}\sum_{l=1}^{L-w_k+1} \beta_k \delta_{kle} + \\ \sum_{l=1}^{L} (\theta_{le} - \theta_{l(e-1)} + \pi_{le} - \omega_{le}) ST_{le} - \\ \sum_{k=1}^{K} \left(c_k + f_k z_k + \sum_{l=1}^{L} f_k |l - des_k|\right) \delta_{kle}, \quad e = 2, \ldots, E-1 \\ \\ \sum_{k=1}^{K}\sum_{l=1}^{L}\sum_{l'=l-w_k+1}^{l} p_k \left(\xi_l + \sum_{e'=1}^{e} \pi_{le'} + \sum_{e'=e+1}^{E} \omega_{le'}\right) \delta_{kl'e} + \sum_{k=1}^{K}\sum_{l=1}^{L-w_k+1} \beta_k \delta_{kle} + \\ \sum_{l=1}^{L} (-\theta_{l(e-1)} + \pi_{le} - \omega_{le}) ST_{le} - \\ \sum_{k=1}^{K} \left(c_k + f_k z_k + \sum_{l=1}^{L} f_k |l - des_k|\right) \delta_{kle}, \quad e = E \end{cases} \tag{5.60}$$

In DBAP, $\boldsymbol{\psi}$ corresponds to the dual variables of constraints (5.56). If $\max_{e=1,\ldots,E}\{z_{2e} + \psi_e\} > 0$, we decide to add a new variable to the restricted master problem. The steps

of the branch and price algorithm are formally given in Figure 5.1 for DBAP. The steps of branch and price algorithm for LASP are similar. In the algorithm, NL represents the node list that includes the objective values of the set of unexplored nodes. BL is a branching constraint list that includes the branching constraints of the set of unexplored nodes. $\overline{\delta}$ indicates the branching values of the original variables δ. We let z_{RMP} and z_{BEST} denote the optimal objective value of the RMP and the objective value of the best feasible solution. At a given node, we can obtain the values of the original variables by calculating the values of λ_e^p and the values of the variables obtained from the subproblems. We assume that P_e^c is the number of solutions obtained from the subproblem e at the current node.

5.3. Implementation Issues

There are many details that should be defined related to the implementation of the branch and price algorithm:

5.3.1. Initialization

The column generation procedure must start with a feasible solution at each node. We add artificial variables to greater than or equality type constraints and penalize them in the objective function.

5.3.2. Column Selection Strategy

At each iteration, one must decide if a column or more than one column is added to the restricted master problem. We can add a column for each event or just for one event. In our experiments, we try to add all alternative optimum solutions of the subproblem to the master problem. However, adding all alternative solutions increase the size of the problem and the computational time. Hence, we decide to add one column that has the largest reduced cost.

5.3.3. Branching Strategy

First of all, we calculate the real values of each variables based on expressions (5.13) and (5.47). Then we choose the variable whose value is the closest to 0.5. We branch on this variable by fixing it to 0 and 1. Branching rule can be achieved as follows in LASP:

In the original formulation setting z_{ine}= 1 or 0 determines if a type n sensor placed on point i is active or not at event e. When z_{ine}= 1, all existing columns that do not activate a type n sensor placed on point i at event e are deleted from the master problem. This can be achieved by setting $\lambda_e^{p_1} = 0$ for all $p_1 \in P_{1e}$ if $z_{ine}^{p_1}$ =0. Conversely, when z_{ine}= 0, all existing columns that activate a type n sensor placed on point i at event e are deleted from the master problem by setting $\lambda_e^{p_1} = 0$ for all $p_1 \in P_{1e}$ if $z_{ine}^{p_1}$ =1.

In DBAP, the branching rule is slightly different. In the original formulation, fixing $\delta_{kle} = 0$ prohibits the assignment of vessel k to event e on section l. In the restricted master problem, we can delete the columns that assign vessel k to event e on section l. This can be achieved as follows: $\lambda_e^p = 0$ for all $p \in P'_e$ if $\delta_{kle}^p = 1$.

Conversely, in the original formulation fixing $\delta_{kle} = 1$ allows the assignment of vessel k to event e on section l. We delete the columns from the restricted master problem that do not assign vessel k to event e on section l and assign vessel k to other events on section l. In the restricted master problem this can be achieved as follows: for event e if $\delta_{kle}^p = 0$ then $\lambda_e^p = 0$ for all $p \in P'_e$ and for events $e' \neq e$, if $\delta_{kle'}^p = 1$ then $\lambda_{e'}^p = 0$ for all $p \in P'_{e'}$.

Branching scheme should be compatible with the pricing subproblems. We modify the subproblem based on the branching constraints in order to prevent the regeneration of columns that are infeasible due to the branching constraints. We set the upper bound of the variable in the subproblem to zero if the value of the branching variable is zero,

or lower bound of the variable to one if the value of the branching variable is one.

5.3.4. Node Selection

Generally, depth first search and best first search are the most common node selection strategies. In this study, we choose best first strategy, since it is more efficient.

5.3.5. Finding a Feasible Solution

In the branch and price algorithm, the linear programming relaxation of the restricted master problem is tighter than the linear programming relaxation of the original formulation. However, finding a feasible solution becomes harder especially for large instances. Therefore, for both of the algorithms we propose greedy heuristics and at each iteration we find a feasible solution by using the constraint set of the search node that results from branching.

At each node of the search tree, the relaxation of restricted master problem is solved by column generation with branching constraints on that node. In order to prevent fractional solutions, some of the variables are set to 0 or 1 at each node. For LASP, at the initialization step, we activate or prohibit the activation of sensors based on the branching constraints. If the coverage quality constraints are satisfied for all events and the budget constraint is fulfilled, we calculate the lifetime of the network. However, if the coverage quality constraints are satisfied and the budget constraint is not fulfilled, the lifetime will be zero. Otherwise, for each event if the coverage quality constraints are not satisfied, we generate a feasible solution. The equations and the logic of the heuristic is inspired from the one proposed for the clock-based formulation in [33]. In order to decide for the set of active sensors, the uncoverage level u_{je} for each event and point is calculated as

$$u_{je} = \max\left(0, d_j - \sum_{i=1}^{I}\sum_{n=1}^{N} a_{ijn} z_{ine}\right). \tag{5.61}$$

We either activate the sensor that has been already placed or activate the new sensor. Since the deployment of a new sensor reduces the budget, first we control the remaining energy of already placed sensors and their coverage level by calculating the p value of the sensors using

$$p_{ine} = \sum_{j=1}^{J} a_{ijn} u_{je} g_{ine}. \tag{5.62}$$

Here, g_{ine} represents the remaining energy of the type n sensor located at point i at event e. If a positive p value cannot be found, then we should activate the new sensor to satisfy the coverage requirements for each point i. Hence, we calculate the r value using (5.63)

$$r_{ine} = \frac{\sum_{j=1}^{J} a_{ijn} u_{je}}{c_n} g_{ine} \tag{5.63}$$

and choose the sensor that has the highest r value. This process continuous until the coverage constraints are satisfied. If positive p or r value cannot be found, we consider another event. Otherwise, if the budget constraint is satisfied for the determined active sensor set, we activate them. After determining the active sensors that satify the coverage quality and budget constraints at event e, the minimum of the chosen sensors' remaining energies becomes equal to the duration of event e. The pseudocode of the heuristic algorithm is given in Figure 5.2.

In DBAP, first we determine the assigned and unassigned vessels based on the branching constraints at a given search node. Then, we start to determine an available location for unassigned vessels. At initial stage, we are looking an available location for the first vessel at $e = 1$. At the next stages for each vessel, we search a space at the last event on which previous unassigned vessel is assigned. At a given e, first of all we determine the formerly assigned vessels' departure times and ST_{le} values for all $l \in L$. Then, if the desired location of unassigned vessel k at event e is appropriate that means no vessel is assigned from berth section des_k to $des_k + w_k$, then we calculate

the departure time of vessel k if it is assigned to des_k at e. If the departure time is smaller than the due time, we assign it to des_k at e. Otherwise, we try to find a new appropriate section that gives smaller penalty values. We calculate the deviation z_k from the due time. If it is equal to one, we again assign it to des_k at e. However, if it is larger than one, we search a new location for the previous events whose deviation from the desired location of k is smaller than z_k. If the appropriate space cannot be found at the preceding events, we assign it to des_k at e. The above calculations are done under the assumption that vessel k can be assigned to des_k at e. On the other hand, if the assignment is not possible, we search an available location for previous events without any restriction. If we cannot find an available location, we increase the event number by one and we start searching an available location at this event. The algorithm is listed formally in Figure 5.3. In the algorithm E_{max} represents the maximum number of events that the vessels can be assigned. Searching an available location for each of the unassigned vessels starts at E_{max}.

Algorithm 5.1: BRANCH_AND_PRICE_DBAP

Begin

Initialization: $\overline{\delta}^1$=null; $z^1_{RMP} = 0; z_{BEST} = M; NL = z^1_{RMP}; BL = \overline{\delta}^1$

While ($|NL| > 0$)

$z^{n^*}_{RMP} = \min_{n=1,\ldots,|NL|}\{z^n_{RMP}\}$, delete $z^{n^*}_{RMP}$ from NL and $\overline{\delta}^{n^*}$ from BL

Rule out infeasible columns according to branching constraints $\overline{\delta}^{n^*}$

Fix the branching variables in pricing subproblems

Find a feasible solution using Algorithm 5.3 and update z_{BEST}

Set $RC = M$

While ($RC > 0$)

Solve RMP, obtain $\boldsymbol{\alpha}$, $\boldsymbol{\gamma}$, $\boldsymbol{\epsilon}$, $\boldsymbol{\mu}$, $\boldsymbol{\nu}$, $\boldsymbol{\psi}$ and set $RC = 0$

Solve each of the E subproblems, find $z_{e*} = \max_{e=1,\ldots,E}\{z_{2e} + \psi_e\}$, obtain the corresponding solution for subproblem e^*

If ($z_{e*} > 0$)

$RC = z_{e*}$, add a column to RMP using the solution of the subproblem e^*

End If

End While

Calculate original variables, $\delta_{kle} = \sum_{p=1}^{P^c_e} \lambda^p_e \delta^p_{kle}$

If ($z_{RMP} < z_{BEST}$) Then

If a solution is fractional Then

Find δ_{kle} that is closest to 0.5 (i.e. $\delta_{k^*l^*e^*}$)

Set $\overline{\delta}^n_{k^*l^*e^*} = 0$, $NL = NL \cup \{z_{RMP}\}$, $BL = BL \cup \{\overline{\delta}^n\}$

Set $\overline{\delta}^n_{k^*l^*e^*} = 1$, $NL = NL \cup \{z_{RMP}\}$, $BL = BL \cup \{\overline{\delta}^n\}$

Else

$z_{BEST} = z_{RMP}$

End If

End If

End While

End

Figure 5.1. Branch and Price algorithm for DBAP.

Algorithm 5.2: FEASIBLE_SOLUTION_LASP

Begin

Initialization

For $e \leq E$

While coverage constraints are not satisfied for event e

Calculate g_{ine} and p_{ine} and $p_{i^*n^*e} = \max_{i,n}\{p_{ine}\}$

If $p_{i^*n^*e} = 0$ Then calculate r_{ine} and $r_{i^*n^*e} = \max_{i,n}\{r_{ine}\}$

If $r_{i^*n^*e} = 0$ Then $W_e = 0$ and consider another event

End If

End If

If ($p_{i^*n^*e} > 0$ or $r_{i^*n^*e} > 0$)

If budget constraint is satisfied when $z_{i^*n^*e} = 1$ Then

Activate sensor at point i^* with type n^* at event e

Update the minimum of the remaining energies

Else If budget constraint is not satisfied Then

Prohibit the activation of sensor at point i^* with type n^* at event e

End If

Control the feasibility of the coverage quality constraints

If the coverage quality constraints are satisfied Then

W_e = Minimum remaining energy of the active sensors

Update the remaining energy of the sensors

Return coverage constraints are satisfied for event e

End If

End If

End While

End For

End

Figure 5.2. Algorithm for finding a feasible solution for LASP.

Algorithm 5.3: FEASIBLE_SOLUTION_DBAP

Begin

$E_{max} = 0$

For all unassigned vessels, $e = E_{max}$

While vessel k is not assigned

For all assigned vessels

If the vessel is assigned at event e Then

Calculate ST values and departure time of that vessel

End If

End For

If the desired location of vessel k is appropriate at event e Then

If the assignment of k at des_k at event e is not prohibited Then calculate c_k

If $d_k \geq c_k$ Then $\delta_{k(des_k)e} = 1$, update E_{max}, skip next vessel

Else calculate deviation $z_k = d_k - c_k$

If $z_k = 1$ Then $\delta_{k(des_k)e} = 1$, consider the next vessel

Else if $z_k \geq 1$ Then find space which gives smaller penalty than z_k

End If

If k is still unassigned Then $\delta_{k(des_k)e} = 1$, update E_{max}, consider the next vessel End If

End If

Else If the assignment is prohibited Then

Search empty space for previous events

If it is still unassigned Then $e = e + 1$ End If

End If

Else $e = e + 1$

End If

End While

End For

End

Figure 5.3. Algorithm for finding a feasible solution for DBAP.

6. COMPUTATIONAL EXPERIMENTS

6.1. Instance Generation

For LASP, we generate an instance of nine different problem sets. The sensor field is a $m \times m$ grid including $I = m \times m$ randomly distributed candidate sensor locations. I ranges from 16 to 100 and m ranges from 4 to 10 for small and medium size instances. Additionally, we generate two instances with 225 and 400 candidate sensor locations in order to see the performance of the formulations for large instances. The places of the points that should be covered are the same as the places of possible sensor locations on the network. There are two types of sensors ($N = 2$). Each point should be covered by at least two sensors ($d_j = 2$). We have three budget levels. The low budget level is calculated as the 75% of points is covered by type 1 sensor. The high budget level is calculated as all points are covered by type 2 sensor. The medium level budget is the average of the high and low budget levels. In Table 6.1, the sensor specifications are given. s_n, E_n, e_n, c_n respectively represent the sensing range, initial battery energy, energy consumption rate and unit cost of the type n sensor.

For DBAP, mainly we have two different data sets in order to evaluate the performance of the event-based formulation. In the first data set, the interarrival times of vessels are randomly distributed over the interval [0,2] and it is called as instances with tight arrival times. In the second data set, the interarrival times of vessels are distributed between [0,9] and it is called as instances with loose arrival times. For each of the two data sets, small problem instances with berth section L=12 and K=3, 6, 9, 12 and 15 vessels and large instances with L=24 and K=3, 6, 9, 12, 15, 20 and 25 are generated. For each of the data groups, we generate 5 instances. For both sets of instances, the vessel widths w_k are random and range from 3 to 8 and the processing times of vessels p_k are uniformly distributed between $0.2w_k$ and $1.5w_k$, since the processing time is proportional to vessel's width. Desired berth location of vessel k, des_k is randomly generated and ranges from $[1, L - w_k + 1]$. Time horizon is calculated as

Table 6.1. Sensor specifications.

Type(n)	s_n	E_n	e_n	c_n
1	1	500	5	1
2	2	500	10	1.5

multiplying the number of vessels by 10, since this value will be appropriate for each of the processing times of vessels. The penalty for lateness and deviation from the desired berth location are taken as f_k=100.

6.2. Experimental Results

All results obtained by the exact model presented in this paper have been implemented and solved using IBM ILOG CPLEX 12.2 by C# environment on Microsoft Visual Studio 2010 with a single thread and default options. For event-based formulations, the maximum computational time per event number in the incremental approach is set to 10800 CPU seconds. The resulting total time is used to solve clock-based versions in order to get fair results. We test the branch and price algorithm at the last event number found in the incremental approach and the time is set to 10800 CPU seconds. The hardware consists of a computer with Microsoft Windows Server 2003 R2 Enterprise X64 Edition operating system and Intel Xeon @3.40 GHz processor with 32 GB RAM.

In the event-based formulation, LP-relaxation is highly degenerate due to the big-M constraints. Hence, proving the optimality takes too much time even with small instances. In order to prevent this situation, we try to take M value as small as possible in our computational results. For LASP, in constraints (3.10), since the value of s_{ine} can be equal to at most $\left\lfloor \frac{E_n}{e_n} \right\rfloor$, we set $M = \left\lfloor \frac{E_n}{e_n} \right\rfloor$ for all n. In constraints (3.11), since the value of W_e can be equal to at most $\left\lfloor \frac{E_n}{e_n} \right\rfloor$ for all sensor types, we set $M = \max\limits_{n=1,\ldots,N} \left\{ \left\lfloor \frac{E_n}{e_n} \right\rfloor \right\}$. In DBAP, M value equals to the time horizon T.

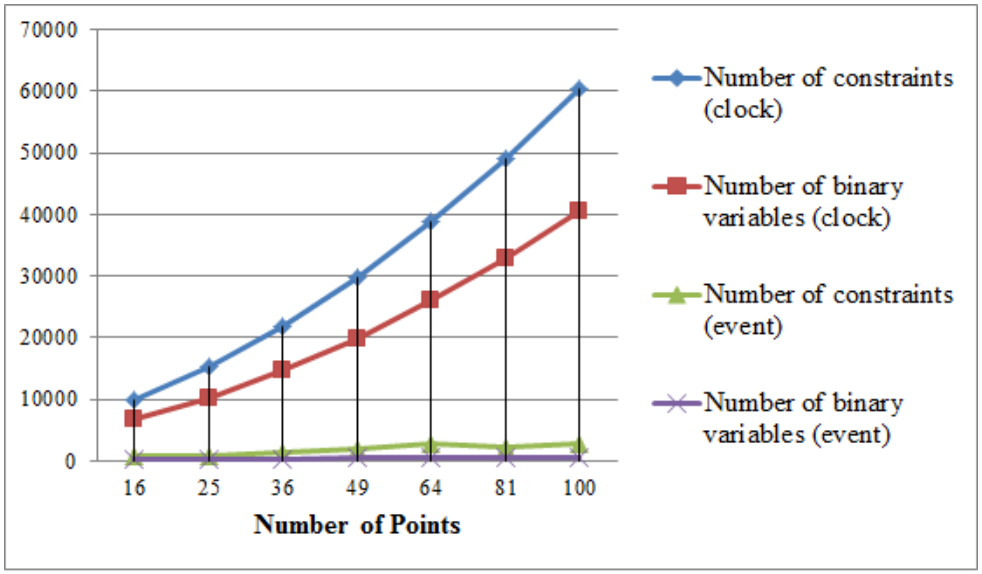

Figure 6.1. Clock-based vs. event-based: model sizes for LASP.

6.2.1. Comparison of the Model Size

We measure the size of the models based on the number of variables and constraints. In LASP, the number of variables and constraints are counted for different number of points at different budget levels. In the event-based approach, the last event number found in the iterative approach is used in the measurements. The details are given in Table B.1 and for the clock and event-based approaches, the first, second and third column represent the number of constraints, number of binary variables and number of continuous variables. In Figure 6.1, for the first budget level the number of constraints and binary variables are shown. We can see that a significant reduction in the number of periods leads to a significant decrease in the number of binary variables and constraints in the event-based approach.

For DBAP, we measure the size of the problems at $K = 6$, 9, 12 and 15 for two types of problem sets. In Table B.2 and B.3, the number of constraints, the number of binary variables and the number of continuous variables are given for two instance groups where vessels arrive with changing frequencies. From Figure 6.2, we can see that in the clock-based approach the average number of binary variables and constraints are

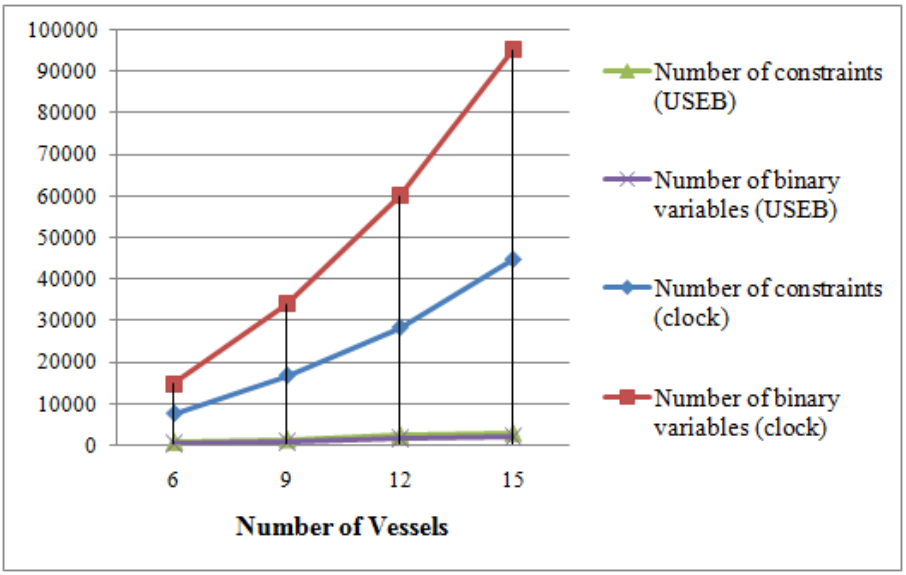

Figure 6.2. Clock-based vs. event-based: model sizes for DBAP.

significantly larger. In Figure 6.3, the event based formulation's size is compared with respect to two different instance groups. Although data sets include the same number of vessels and berth sections, the number of constraints and binary variables in tight arrival data set is smaller than the ones of the loose arrival data set. This is because in loose arrival data set the average number of events required for optimal solution is larger than the tight arrival data set.

6.2.2. The Strength of the New Valid Inequalities

In order to measure the strength of the tightening constraints (4.4), (4.5) and symmetry breaking constraints (4.1) for LASP, we compare the computational time of the instances for $J = 16$, 25 and 36 with and without the constraints. Although these constraints do not improve the LP relaxation, they improve the computational time of the problems. As we use the incremental approach for finding an optimal event number and add prior objective value as a cut, the CPU time is measured again at optimal number of events without cut. In Table 6.2, CPU^1 and CPU^2 represent the computational time for strengthened and original formulations. In LASP, the strength-

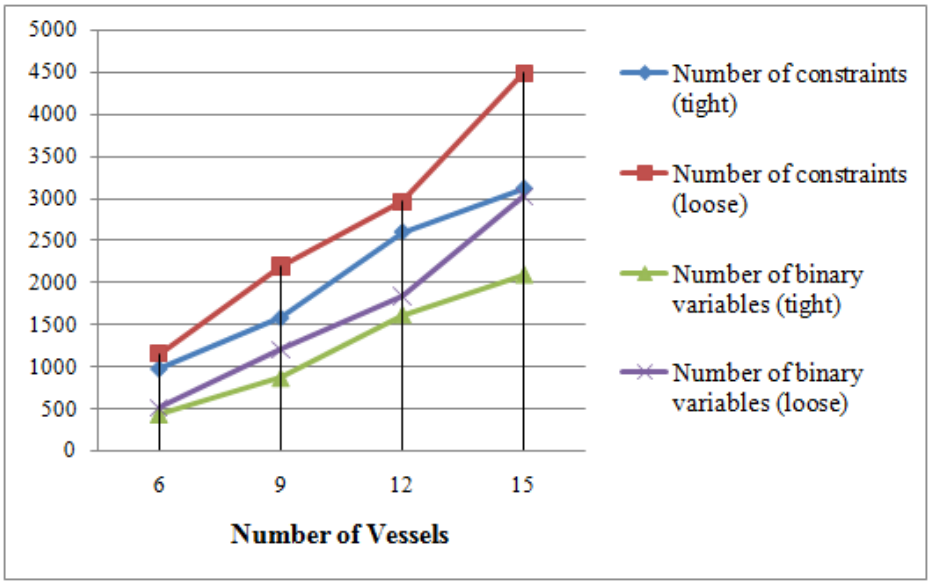

Figure 6.3. Model sizes for event-based formulations: DBAP.

ened constraints slightly improve the solution time, since the event-based formulation is very efficient even if we solve the problems without these constraints. For example, the optimal objective value of an instance J=25 at the first budget level is found in 6.21 CPU seconds with constraints and 34.84 CPU seconds without constraints. We use the strengthened version of the formulation, since on the average its performance is better than the original formulation.

In DBAP, constraints (3.53), (3.54), (3.55) and tightening constraints (4.15), (4.16) and (4.17) are tested for $K = 15$ and $L = 24$ with tight and loose arrival times. In our computations for finding an optimal number of events, we do not add the previous objective value as a cut for the next event, since it makes difficult to find an integer feasible solution and increases the branch and bound tree's size in DBAP. Therefore, we directly take the best event's computational time from Table 6.9 and Table 6.12. In Table 6.3, CPU^1_{ip} and CPU^1_{lr} represent the solution time of the mixed integer programming and the LP relaxation of the strengthened formulation. Z^1_{ip} and Z^1_{lr} correspond to optimal objective value and the LP relaxation objective value of the strengthened formulation. Same notations indexed by two are displayed for the original formulation. The time limit is set to 3600 seconds for only seeing the

Table 6.2. Strength of the new valid inequalities: LASP.

J	B	E	CPU^1	CPU^2
16	1	3	3.39	9.84
	2	3	1.82	1.29
	3	4	1.92	3.12
25	1	2	6.21	34.84
	2	3	0.85	0.66
	3	4	3.64	4.42
36	1	3	1.06	9.98
	2	4	0.84	6.42
	3	4	0.40	0.93
Avg			2.24	7.94

effects of the new constraints. We can see that the new valid inequalities improve considerably the LP relaxation bound and computational time. The LP relaxation's solution times are similar in both formulations. For tight arrival data set, at K=15 with L=24, one of the instances cannot be solved in given time limit without the new valid inequalities. However, after adding these constraints, we can solve the problem in 472.26 CPU seconds. Although the average solution time to solve the problems with the valid inequalities is 113.83 CPU seconds, on 4 out of 10 instances, the event-based formulation without constraints does not prove that the objective value is optimal in given time limit. For all instances, LP relaxation values are better than ones of the formulations without the valid inequalities.

6.2.3. Comparison of the Models

We compare the clock and event-based formulations using the solution times and quality of the best solutions. The quality of the best solution is important, since when both approaches are not solved to optimality, we prefer the approach that gives the best solution. We use the iterative approach to find the number of events which gives

Table 6.3. Strength of the new valid inequalities: DBAP.

Tight arrival times										
L	K	E	CPU^1_{ip}	Z^1_{ip}	CPU^1_{lr}	Z^1_{lr}	CPU^2_{ip}	Z^2_{ip}	CPU^2_{lr}	Z^2_{lr}
24	15	5	121.09	662	0.06	185.68	495.29	662	0.06	0
	15	6	9.51	450	0.07	297.06	167.03	450	0.05	100
	15	7	420.34	596	0.10	247.00	2681.17	596	0.07	0
	15	5	71.53	722	0.09	464.65	214.45	722	0.06	300
	15	6	472.26	708	0.10	323.21	3600.00	708	0.06	100
Loose arrival times										
L	K	E	CPU^1_{ip}	Z^1_{ip}	CPU^1_{lr}	Z^1_{lr}	CPU^2_{ip}	Z^2_{ip}	CPU^2_{lr}	Z^2_{lr}
24	15	10	3.97	697	0.27	690	3600.00	697	0.18	0
	15	7	2.25	483	0.19	466	61.56	483	0.07	0
	15	9	3.59	536	0.16	526	3600.00	536	0.09	0
	15	8	5.30	588	0.14	576	3600.00	588	0.09	0
	15	8	28.42	633	0.23	505	429.15	633	0.12	0
Avg			113.83	608	0.14	428.06	1943.19	608	0.08	50

the best solution. However, iterative approach is limited especially when complicated instances are considered, because none of the instances, for given number of events, is proved to be optimal under time limitations. Besides, when the difference between the initial number of events and the best event is large, iterative increment approach becomes computationally intractable.

In the clock-based formulation of LASP, we take different T values such as T=200, T=250 and T=300 for three budget levels. In the event-based formulation, the incremental approach is started with different initial event numbers (E_1) for each budget level. Since intuitively we expect an increase in the lifetime and event numbers with an increasing budget level, we set E_1=2 for the first budget level, E_1=3 for the second level and E_1=4 for the last level. Computational results for all instances are given in Table 6.4. The optimal objective value (Z_{lp}), solution time (CPU_{lp}), LP relaxation's

objective value (Z_{lr}) and LP relaxation's solution time (CPU_{lr}) are displayed for clock-based formulations. In the event-based formulations, the number of events that gives the best objective value (E_2), optimal objective value at the best event (Z_{lp}), cumulative solution time (CPU_1), solution time at the best event (CPU_2), LP relaxation's objective value (Z_{lr}) and LP relaxation's solution time (CPU_{lr}) are reported. The asterisks on the best event number represent that the out of memory exception is thrown at $E_2 + 1$. Hence, we give the results for final event number without exception.

For LASP, the event-based formulation is superior to the clock-based one. This is because while the solution time of the clock-based formulation is affected from the time horizon, the event-based formulation is affected from the number of events. When both of the formulations find an optimal solution in given time limit, the required computational effort is considerably lower in the event-based approach. Additionally, they give the same solution in the given time limit. Hence, this demonstrates that we properly find the optimal number of events by using the iterative approach. When the time limit is not enough for proving the optimality for both of the approaches, in most of the instances the event-based formulation finds better objective values. Generally, at the best events, the event-based formulation finds the optimal solution much faster. At the next event, the computational time, which is spending for proving that the best event is the prior event, is more than our time limitation. For example, for an instance $I = 25$ at the second budget level, we find the network lifetime as 200, when $E = 3$. But at $E = 4$, the solver neither finds better solution nor proves that 200 is the best in 10800 seconds. On the other hand, the clock-based formulation gives $LT = 144$ under the given running time limitation. Additionally, for an instance $I = 25$ at the third budget level we come up with a suboptimal solution. At E=4 and 5, we find the same objective value that is equal to 250 and the algorithm stops. However, the clock-based formulation finds the optimal solution as LT=300. Hence, the iterative increment of the number of events cannot guarantee the optimality of the solution. For large instances with I=225 and 400, although the clock-based formulations cannot find any feasible solutions under the given running time limitation, the event-based formulations find the best solutions using iterative approach. In the clock-based approach, the LP relaxation

Table 6.4. LASP results.

	Clock-based Formulation				Event-based Formulation					
I	Z_{lp}	CPU_{lp}	Z_{lr}	CPU_{lr}	E_2	Z_{lp}	CPU_1	CPU_2	Z_{lr}	CPU_{lr}
16	150	29.99	200	1.77	3	150	32.18	3.05	268.53	0.00
	200	135.80	250	1.37	3	200	19.03	1.39	275.00	0.01
	250	3474.71	300	0.80	4	250	47.42	1.12	361.53	0.01
25	150	10800.00	200	8.65	2	150	252.21	4.75	194.11	0.01
	144	10800	250	14.84	3	200	10800.76	0.71	282.35	0.01
	300	1608.90	300	12.15	4	250	10801.23	1.04	370.58	0.03
36	138	10800.00	200	26.46	3	150	10801.73	1.39	261.53	0.03
	200	1568.73	250	47.59	4	200	50.90	2.43	342.85	0.04
	200	98.31	300	81.35	4	200	10.42	0.46	342.85	0.04
49	74	10800.00	200	116.50	3*	150	10860.01	10800	266.66	0.04
	62	10800.00	250	190.73	3	200	146.57	138.81	266.66	0.03
	225	1450.938	300	268.64	5	225	493.95	16.06	408.33	0.10
64	119	10800.00	200	564.14	2*	100	10800	10800	192.36	0.14
	142	21600.28	250	1161.89	4	150	21600.76	10800	333.33	0.16
	200	1562.75	300	1118.82	4	200	134.25	12.35	333.33	0.16
81	0	10800.00	200	1207.34	2	100	10800.57	0.39	180.00	0.08
	150	2742.85	250	2578.90	3	150	193.87	1.62	240.00	0.09
	150	2974.55	300	2760.00	4	150	64.28	49.48	300.00	0.14
100	0	10800.00	200	7256.01	2	100	17.10	0.45	162.50	0.12
	125	5568.83	249.50	5362.83	4	125	26.85	7.87	262.50	0.34
	0	10800.00	300	10797.00	4	125	16.31	5.79	262.50	0.33
225	0	10800.00	200	10797.03	2	100	10803.83	3.10	196.87	1.64
	0	10800.00	250	10646.57	4*	200	9741.60	9732.27	341.67	1.36
	0	10800.00	300	10794.36	5	200	611.67	33.50	408.33	3.36
400	0	10800.00	200	10794.05	2	100	61.26	12.59	162.50	0.94
	0	10800.00	250	10791.49	4	125	889.82	182.46	262.50	16.76
	0	10800.00	273.71	10787.19	4	125	1527.71	1141.70	262.50	16.67
Avg	110.33	7585.80	249	3636.61		162.04	4133.57	1620.55	279.33	1.58

values are affected from the T value. As the network enlarges, the solution time of the LP relaxation increases. In the event-based approach, generally LP relaxation values are worse than ones of the clock-based formulations. However, the solution time is insignificant.

In DBAP, we test the models with two groups of data. Due to the more frequent incoming vessel arrivals, the first data set is more complex. The minimum number of events is calculated as $E_1 = \left\lceil \frac{\sum_{k=1}^{K} w_k}{L} \right\rceil$. First of all, we report the results for the experiments with small and medium size instances. In Tables 6.7, 6.8, 6.9, 6.10, 6.11 and 6.12, we report the results of USEB formulations for both data sets. The results of clock-based formulations are given in Tables C.1, C.2, C.3, C.4, C.5 and C.6. In Table 6.5 and Table 6.6 results of GEB approach for K=3, 6 and 9 with L=24 are summarized for both data sets. In our computations, we report the solution time required for finding an optimal solution (CPU_{lp}) and optimal objective value (Z_{lp}) of clock-based formulation. In the event-based approaches, we give total time, which is required for finding the best number of events (CPU_1) in the incremental approach and in order to see the computational performance of the event-based approach, the solution time at optimal number of events (CPU_2) is given. The objective value of the event-based formulations (Z_{lp}), the minimum number of events (E_1) and the best number of events (E_2) are reported. For all the approaches, the time to solve LP relaxation of the problems (CPU_{lr}) and optimal objective value of the LP relaxations (Z_{lr}) are reported. LP relaxations of the event-based approaches are calculated for the optimal event number.

For all instances, at the best event, the clock-based and GEB formulation require relatively long CPU times to obtain optimal solutions. The optimal number of events for USEB is smaller than the one for GEB. Hence, GEB takes significantly more time to find the optimal number of events. For the rest of the computational experiments, we compare the clock-based and USEB approach to see their performances. If we have problems which can be formulated by using both USEB and GEB approaches, USEB formulation will be advantageous in terms of computational time.

Table 6.5. Results for GEB formulation (tight).

	GEB Formulation						
K	E_1	E_2	CPU^1	CPU^2	Z_{lp}	CPU_{lr}	Z_{lr}
3	1	3	0.70	0.09	27	0.00	23.00
3	1	3	0.14	0.09	31	0.02	21.13
3	1	2	0.15	0.04	30	0.00	24.26
3	1	2	0.23	0.03	146	0.02	133.56
3	1	3	0.29	0.06	46	0.02	30.83
6	1	4	0.76	0.10	44	0.00	43.00
6	2	6	6.42	2.31	75	0.05	61.00
6	2	5	7.18	1.31	96	0.03	77.00
6	2	4	19.96	4.93	401	0.02	62.66
6	2	5	3.00	0.45	57	0.03	48.00
9	2	8	196.39	37.21	224	0.08	101.00
9	3	6	848.82	218.06	430	0.05	87.00
9	2	7	987.87	318.57	315	0.08	75.00
9	2	9	214.56	75.37	124	0.16	102.00
9	2	6	38.84	12.04	105	0.06	73.00
Avg			155.02	44.71	143.40	0.04	64.16

According to the results obtained for small instances, USEB is very efficient on these simple problems. For both of the data sets at K=3 and K=6, no significant difference occurs in the CPU times. For the first data set at K=9 with L=12, 3 out of 5 instances USEB formulation is superior than the clock-based ones. On the average, the cumulative time in USEB approach is smaller than the computational time of the clock-based approach. At K=12, except two instances, the event-based formulation finds better solutions. The clock-based formulation cannot prove optimality in given time limit. At K=15, both formulations do not prove optimality in given time limit. However, except an instance, USEB formulations find better feasible solutions. At K=9 with L=24, for all instances USEB models of these problems are solved in relatively

Table 6.6. Results for GEB formulation (loose).

	GEB Formulation						
K	E_1	E_2	CPU^1	CPU^2	Z_{lp}	CPU_{lr}	Z_{lr}
3	1	3	0.35	0.01	37	0.00	37
3	1	3	0.10	0.01	39	0.00	35
3	1	3	0.14	0.04	27	0.00	20
3	1	3	0.10	0.01	33	0.02	33
3	1	3	0.10	0.01	45	0.00	45
6	2	6	14.95	0.07	123	0.05	122
6	1	6	1.65	0.06	149	0.05	149
6	2	6	1.39	0.29	137	0.03	121
6	2	6	4.73	0.06	150	0.05	148
6	2	5	0.96	0.18	89	0.03	81
9	2	9	245.21	3.17	238	0.11	220
9	3	9	93.32	4.59	211	0.12	194
9	3	8	131.59	1.32	244	0.12	216
9	2	9	156.45	4.01	281	0.08	270
9	3	9	28.82	12.43	196	0.14	176
Avg			45.32	1.75	133.27	0.05	124.47

shorter time. For K=12, on 2 out of 5 instances, USEB formulations require less solution time. Comparing the average cumulative solution times of the event-based formulations with the solution times of clock-based formulations, USEB formulations perform better. For all instances with K=15, USEB performs better. The required solution time at the best event is significantly less than the cumulative time. For example, when K=15 for the last instance, at E=6 the CPU time is 472 seconds. However, the algorithm tests whether the solver finds the same solution at E=7 or not. Hence, the cumulative time increases exponentially. When K=20, on 2 out of 5 instances the clock-based formulation finds the optimal solution in given time limits. It takes much longer for the event-based approach to close the optimality gap.

Although the event-based formulation does not prove the optimality, optimal solutions are obtained in an earlier stage. For the rest of the instances, event-based approach finds better feasible solutions. When K=25, for all instances, the quality of the feasible solutions is better in USEB approaches.

For the second data set, for K=9 and K=12 with L=12, nearly in all instances, at the best event, USEB formulations require less computational effort. However, the total time spending for incremental approach is more than the clock-based formulations require. For K=15, the clock-based formulation requires less time. When K=9 and L=24, on 4 out of 5 instances and when K=12 on 3 out of 5 instances, USEB formulations require less time. Besides, for all instances at the best event USEB formulations perform better than clock-based formulations. For K=15, the total time is much more than the clock-based approach. Nevertheless, for all instances, at the best event numbers USEB formulations require less time. It should be noted that the difference between the minimum number of events and the best event is large in this set. Hence, a gradual increase in the event number causes an increase in the total time. For K=20 and K=25, although both formulations find the same solutions, clock-based formulations outperform USEB formulations in terms of computational time. It should be pointed out that if we know the optimal number of events or if we start the incremental approach close to the optimal number of events, USEB approach becomes more advantageous in getting good results. Generally, both of the approaches find similar LP relaxation values. However, in some of the instances USEB approach results in significantly better relaxation objective values. For example, in the second and third instances of the first data set for K=12 and L=24, USEB approach results in good lower bounds.

The event-based formulation performs better for LASP. As for DBAP, USEB approach is superior to the clock-based formulation for more complex data sets in terms of both solution quality and computational performance. In a loose arrival data set, although we obtain an optimal solution by applying both of the approaches, the USEB formulation requires much longer time. The cumulative solution time includes

the time for closing the optimality gap at each event and the time for finding the optimal event in incremental approach. It should be noted that in real life cases both approaches are restricted to solve the problems optimally. Then, the formulation which finds good feasible solution in reasonable time is advantageous for decision makers. Hence, the event-based approaches are capable of finding good feasible solutions under time limitations. Therefore, we can achieve to find better feasible solutions by using reasonable number of events ignoring the large optimality gaps in the event-based approaches.

6.2.4. Testing the Accuracy of the Branch and Price Algorithm

The accuracy of the branch and price algorithm is tested for the optimal number of events found by the incremental approach. Although the branch and price algorithm solves small instances at optimality, it results in poor performance for medium or large size instances. However, the solution obtained at the root node provides a very good upper bound on the optimal value of LASP and lower bound on the optimal value of DBAP. In order to provide good upper and lower bounds for both of the problems, simple heuristics are implemented in the branch and price procedure. For each of the iterations, using the constraint set of tree node with the best bound, the new feasible solution is generated and the best feasible solution is updated.

Computational results for LASP are given in Table 6.13. The upper bound (Z_{UB}) at the root node, initial feasible solution obtained by heuristic (Z_{lp}^1), final best feasible solution (Z_{lp}^2), required solution time at the root node (CPU^1) and total time required to solve the problem (CPU^2) are reported. The asterisks represent that root node's optimality is not proved in the given time limit. However, we can calculate an upper bound at each step of the column generation procedure.

For all the instances, although finding an upper bound at the root takes too much time, it is significantly better than the LP relaxation of the original formulation. Nearly for all instances, the calculated upper bound value is the optimal solution at

that event. Besides, the initial feasible solution provides a good lower bound for this problem. If we use the proposed heuristic without implementing other methods, it gives good feasible solution in negligible solution time. For example, at J=64 with medium budget level, we can find a lifetime of 150 units in 10800 seconds using event-based approach. The clock-based formulation finds LT=142 in given time limit. On the other hand, we can find the same lifetime with event-based approach in significantly short time by using this simple heuristic. Additionally, for large data set at J=225, even if heuristic finds the same solutions with event-based formulations in relatively shorter time, the clock-based formulations cannot find any feasible solutions in given time limit.

For DBAP, computational results for small and medium size problems are given in Table 6.14 and Table 6.15 with L=24 and in Table 6.16 with L=12 for tight arrival data set. For loose arrival data set, in Table 6.18 and Table 6.19 computational results for small and medium size problems with L=24 and in Table 6.17 computational results with L=12 are reported. The lower bound at the root node (Z_{LB}), initial feasible solution found by heuristic (Z^1_{lp}), final best feasible solution (Z^2_{lp}), required solution time at the root node (CPU^1) and total time required to solve the problem (CPU^2) are displayed. The branch and price algorithm takes too much time to prove the optimality. Only for all the instances for K=3 and K=6 are solved at optimally. The performance of the branch and price algorithm worsens as the size of the instances becomes larger.

For tight arrival data set, nearly for all instances, the lower bound of the root node is better than the LP relaxation of the original formulation. We cannot achieve the desired quality at the initial feasible solution. As the algorithm proceeds, the quality of the feasible solution improves. However, at the end of the time limit, final feasible solutions are not so close to the solution found by the solver. For loose arrival data set, in some instances, the lower bound at the root node is better than the LP relaxation bound. On the other hand, in the first data set, we achieve good lower bounds in larger number of instances. However, the performance of the heuristic is better in

the second data set. For example, for K=20 with L=24 and K=15 with L=12 we find initial feasible solutions which are optimal or close to optimal. In most of the instances, we obtain optimal solutions by using this simple heuristic. Generally, while the feasible solution found by the heuristic does not improve for LASP, the quality of the feasible solutions improves as long as the algorithm proceeds for DBAP. It should be noted that it is not necessary to know the optimal number of events in order to use these heuristics. Without using an incremental approach, we can use a large number of events without much affecting the solution time of the heuristic. Hence, in different cases, experimentally we can obtain good feasible solutions in very short time. We can implement these heuristics in other methods to improve the solution quality as well.

Table 6.7. Results for small and medium size instances with L=12 (tight).

	USEB Formulation						
K	E_1	E_2	CPU^1	CPU^2	Z_{lp}	CPU_{lr}	Z_{lr}
3	2	2	0.21	0.03	18	0.01	15.12
3	1	2	0.09	0.01	19	0.01	17.14
3	2	3	0.14	0.04	30	0.03	20.41
3	2	3	0.21	0.06	549	0.03	36.53
3	2	2	0.06	0.01	23	0.01	21.42
6	3	4	1.18	0.17	177	0.03	155.27
6	3	4	1.14	0.28	189	0.03	50.38
6	4	5	46.32	6.46	1618	0.04	366.46
6	3	5	29.51	3.98	575	0.03	352.40
6	3	5	0.56	0.14	66	0.01	45.08
9	4	6	543.57	78.01	846	0.04	295.02
9	4	6	1127.95	496.14	828	0.03	268.95
9	5	6	381.07	113.42	1146	0.03	396.80
9	4	5	380.18	96.51	1539	0.03	492.73
9	4	5	13.60	4.31	332	0.03	91.19
12	5	7	19185.95	6198.80	1823	0.06	255.86
12	6	8	11180.35	262.60	1154	0.06	575.10
12	5	7	32.56	9.67	211	0.03	135.14
12	5	6	14591.35	3770.20	3203	0.06	1018.00
12	6	7*	9198.01	7913.50	2973	0.07	497.15
15	6	8	22244.56	10800.00	1558	0.14	971.05
15	7	8	30914.74	10800.00	3110	0.23	674.52
15	7	8*	21600.00	10800.00	3114	0.09	197.00
15	7	9*	23325.61	10800.00	3851	0.19	1140.03
15	7	8*	21600.00	10800.00	2894	0.12	794.00
Avg			7055.96	2918.17	1273.84	0.06	355.31

Table 6.8. Results for small size instances with L=24 (tight).

	USEB Formulation						
K	E_1	E_2	CPU^1	CPU^2	Z_{lp}	CPU_{lr}	Z_{lr}
3	1	2	0.29	0.03	27	0.03	23.26
3	1	3	0.29	0.04	31	0.01	21.13
3	1	2	0.12	0.03	30	0.01	24.00
3	1	2	0.15	0.03	146	0.01	35.41
3	1	3	0.28	0.04	46	0.01	30.83
6	1	3	0.40	0.04	44	0.01	43.05
6	2	3	0.32	0.07	75	0.01	61.95
6	2	3	0.40	0.10	96	0.01	77.56
6	2	3	2.21	0.12	401	0.03	262.59
6	2	3	0.26	0.06	57	0.03	48.02
9	2	4	1.71	0.42	224	0.04	201.27
9	3	4	21.75	4.26	430	0.03	88.61
9	2	5	21.26	3.42	315	0.04	176.21
9	2	4	1.50	0.39	124	0.03	102.00
9	2	3	0.45	0.10	105	0.03	73.06
12	3	6	450.78	73.23	537	0.07	169.00
12	3	5	328.46	46.85	625	0.07	371.02
12	3	5	178.98	11.81	584	0.06	444.18
12	3	7	46.95	5.32	226	0.08	168.00
12	3	5	195.79	15.29	609	0.07	172.66
Avg			62.62	8.08	236.60	0.03	129.69

Table 6.9. Results for medium size instances with L=24 (tight).

	USEB Formulation						
K	E_1	E_2	CPU^1	CPU^2	Z_{lp}	CPU_{lr}	Z_{lr}
15	4	5	649.68	121.09	662	0.06	185.68
15	3	6	138.42	9.51	450	0.07	297.06
15	4	7	1402.20	420.34	596	0.10	247.00
15	3	5	659.62	71.53	722	0.09	464.65
15	4	6	3172.12	472.26	708	0.10	323.21
20	5	6	32400.00	10800.00	989	0.2	292.25
20	5	8	18521.58	3676.10	1064	0.34	646.05
20	5	10	47344.30	10800.00	704	0.39	312.00
20	5	6	22952.29	10800.00	3218	1.14	2322.00
20	5	7*	32400.00	10800.00	3080	0.42	1068.50
25	7	8	32400.00	10800.00	6483	3.45	1928.41
25	6	8	43201.10	10800.00	5281	0.58	830.76
25	6	7	32400.44	10800.00	7551	1.72	1601.59
25	5	9*	64800.73	10800.00	1859	0.47	557.00
25	6	8	43200.66	10800.00	4858	0.55	1716.00
Avg			25042.88	6798.06	2548.33	0.64	852.81

Table 6.10. Results for small and medium size instances with L=12 (loose).

	USEB Formulation						
K	E_1	E_2	CPU^1	CPU^2	Z_{lp}	CPU_{lr}	Z_{lr}
3	2	2	0.20	0.02	31	0.00	31.00
3	2	3	0.06	0.02	31	0.01	31.00
3	2	2	0.08	0.02	44	0.00	44.00
3	2	3	0.06	0.03	44	0.01	33.72
3	2	3	0.08	0.03	30	0.00	28.14
6	4	6	0.89	0.12	152	0.01	149.00
6	3	4	4.70	0.17	799	0.01	783.53
6	4	5	7.69	0.59	515	0.00	283.95
6	4	5	5.22	0.23	436	0.01	309.00
6	3	4	0.28	0.05	127	0.01	126.01
9	5	7	68.66	6.55	641	0.03	496.17
9	5	7	157.28	57.66	592	0.03	249.00
9	4	6	5.86	0.33	190	0.01	175.02
9	4	7	2.62	0.30	274	0.03	266.00
9	6	8	284.64	27.55	974	0.04	528.16
12	6	8	60.88	13.47	742	0.09	509.27
12	6	11	416.02	22.16	636	0.14	497.49
12	6	10	473.16	17.64	377	0.07	314.00
12	6	9	31.43	1.50	418	0.06	395.01
12	6	9	943.67	132.34	1092	0.11	636.57
15	7	11	493.27	22.94	564	0.15	544.00
15	7	12	1255.38	44.36	690	0.20	661.00
15	7	11	4519.47	163.08	779	0.14	622.00
15	8	14	767.00	38.89	688	0.25	638.00
15	7	12	703.80	12.50	582	0.15	556.00
Avg			408.10	22.50	457.92	0.06	356.28

Table 6.11. Results for small size instances with L=24 (loose).

	USEB Formulation						
K	E_1	E_2	CPU^1	CPU^2	Z_{lp}	CPU_{lr}	Z_{lr}
3	1	1	0.20	0.02	37	0.03	37.00
3	1	2	0.14	0.02	39	0.02	35.77
3	1	3	0.14	0.05	27	0.02	20.00
3	1	2	0.11	0.03	33	0.02	33.00
3	1	2	0.12	0.02	45	0.02	45.00
6	2	3	0.53	0.11	123	0.02	122.00
6	2	4	0.83	0.14	149	0.03	149.00
6	2	4	0.83	0.19	137	0.03	121.04
6	2	4	0.70	0.14	150	0.03	148.00
6	2	3	0.31	0.06	89	0.02	81.01
9	2	6	4.86	0.34	238	0.03	220.00
9	3	6	7.22	1.23	211	0.05	194.14
9	3	6	14.58	0.77	244	0.05	216.00
9	2	4	1.00	0.23	281	0.02	270.00
9	3	6	4.66	0.84	196	0.03	176.00
12	3	8	53.5	9.84	406	0.09	272.00
12	4	7	14.48	1.16	395	0.11	389.00
12	3	5	4.67	0.77	435	0.08	427.01
12	3	7	263.02	41.75	670	0.17	440.00
12	4	5	13.69	1.53	529	0.08	295.03
Avg			19.28	2.96	221.70	0.05	184.55

Table 6.12. Results for medium size instances with L=24 (loose).

	USEB Formulation						
K	E_1	E_2	CPU^1	CPU^2	Z_{lp}	CPU_{lr}	Z_{lr}
15	4	10	52.59	3.97	697	0.27	690.00
15	4	7	45.28	2.25	483	0.19	466.00
15	3	9	19.95	3.59	536	0.16	526.00
15	4	8	30.11	5.30	588	0.14	576.00
15	4	8	306.36	28.42	633	0.23	505.00
20	5	9	382.25	30.72	1162	0.36	1116.00
20	5	15	10460.56	1475.11	1372	0.87	1235.00
20	5	12	13320.91	630.23	1095	0.68	968.00
20	5	10	639.88	53.48	1013	0.48	1000.00
20	5	10	539.62	23.23	1108	0.46	1088.00
25	6	13	25991.45	486.88	1695	1.30	1558.00
25	7	16	47313.39	1388.97	1948	2.22	1933.00
25	6	13	18235.50	332.20	1611	1.08	1598.00
25	6	14	13988.09	1124.17	1574	1.67	1539.00
25	5	13	7173.19	844.84	1311	0.91	1280.00
Avg			9233.28	428.89	1121.73	0.73	1071.87

Table 6.13. B&P results of LASP.

J	B	E	Z_{UB}	Z_{lp}^1	Z_{lp}^2	CPU^1	CPU^2
16	1	3	150	100	150	87.56	477.82
	2	3	200	150	200	24.85	271.81
	3	4	250	200	250	23.90	302.01
25	1	2	166.67	100	100	6156.26	10800.00
	2	3	225	200	200	112.28	10800.00
	3	4	275	250	250	66.76	10800.00
36	1	3	150	100	150	1744.05	9743.53
	2	4	200	150	200	214.77	9850.72
	3	4	200	200	200	27.31	27.31
49	1	3*	247	100	100	10800.00	10800.00
	2	3	200	150	150	6046.66	10800.00
	3	5	225	200	200	292.88	10800.00
64	1	2*	200	100	100	10800.00	10800.00
	2	4*	328	150	150	10800.00	10800.00
	3	4	200	200	200	165.01	165.01
81	1	2	100	100	100	29.07	29.07
	2	3	150	150	150	136.06	136.06
	3	4	150	150	150	157.98	157.98
100	1	2	100	100	100	61.46	61.46
	2	4	125	100	100	286.98	10800.00
	3	4	125	100	100	290.74	10800.00
225	1	2	100	100	100	502.02	502.07
	2	4*	200	200	200	10800.00	10800.00
	3	5*	250	200	200	10800.00	10800.00
400	1	2	100	100	100	10277.00	10277.00
	2	4*	200	100	100	10800.00	10800.00
	3	4*	200	100	100	10800.00	10800.00
Avg			185.80	142.59	151.85	3789.04	6785.27

Table 6.14. Small size instances of DBAP with L=24 (tight).

L	K	E	Z_{LB}	Z_{lp}^1	Z_{lp}^2	CPU^1	CPU^2
24	3	2	27.00	27	27	1.11	1.12
	3	3	31.00	332	31	4.56	4.58
	3	2	30.00	33	30	0.97	1.09
	3	2	145.17	146	146	1.36	1.47
	3	3	46.00	48	46	4.22	4.23
24	6	3	44.00	44	44	8.50	8.53
	6	3	75.00	75	75	28.19	28.19
	6	3	89.00	1114	96	25.52	36.44
	6	3	292.11	696	401	57.84	297.83
	6	3	53.00	58	57	14.84	26.53
24	9	4	221.40	952	224	260.59	331.06
	9	4	146.37	1638	439	234.05	10800.00
	9	5	209.99	925	318	701.62	10800.00
	9	4	102.00	219	124	331.40	10800.00
	9	3	97.65	202	105	199.69	212.72
24	12	6	169.00	1235	835	816.45	10800.00
	12	5	435.14	1753	1238	735.54	10800.00
	12	5	467.85	1212	596	1316.32	10800.00
	12	7	190.17	831	330	1111.18	10800.00
	12	5	218.62	2028	704	732.14	10800.00
Avg			154.52	678.40	293.30	329.30	4367.69

Table 6.15. Medium size instances of DBAP with L=24 (tight).

L	K	E	Z_{LB}	Z_{lp}^1	Z_{lp}^2	CPU^1	CPU^2
24	15	5	226.86	1292	672	1006.46	10800.00
	15	6	328.52	1770	954	1395.74	10800.00
	15	7	247.00	1321	1012	1821.36	10800.00
	15	5	502.18	3294	1269	1794.71	10800.00
	15	6	352.25	2142	1217	1214.57	10800.00
24	20	6	328.59	3438	2114	2463.89	10800.00
	20	8	707.44	3821	2773	2990.42	10800.00
	20	10	312.00	3838	2212	5109.50	10800.00
	20	6	2452.06	10963	7604	5725.93	10800.00
	20	7	1150.00	7174	4844	3789.77	10800.00
24	25	8*	1435.00	19840	19840	10800.00	10800.00
	25	8	993.47	24900	21755	7191.11	10800.00
	25	7*	0.00	20114	20114	10800.00	10800.00
	25	9	557.00	6972	5849	4264.41	10800.00
	25	8	1859.12	15420	13810	8198.33	10800.00
Avg			763.43	8419.93	7069.27	4571.08	10800.00

Table 6.16. Small and medium size instances of DBAP with L=12 (tight).

L	K	E	Z_{LB}	Z_{lp}^1	Z_{lp}^2	CPU^1	CPU^2
12	3	2	18.00	19	18	0.50	0.52
	3	2	19.00	25	19	0.33	0.33
	3	3	30.00	31	30	1.00	1.14
	3	3	549.00	549	549	1.58	1.59
	3	2	23.00	23	23	0.27	0.27
12	6	4	175.12	490	177	11.55	20.87
	6	4	92.15	899	189	20.33	40.78
	6	5	617.06	3142	1618	22.91	81.55
	6	5	394.53	890	575	40.32	4218.26
	6	5	66.00	79	66	9.50	9.50
12	9	6	358.66	1668	943	69.69	10800.00
	9	6	357.10	3581	921	43.14	10800.00
	9	6	500.25	2678	1348	70.89	10800.00
	9	5	612.80	3467	1643	58.89	10800.00
	9	5	131.86	358	333	22.37	10800.00
12	12	7	356.93	7301	2320	159.39	10800.00
	12	8	681.61	4728	1460	183.28	10800.00
	12	7	174.00	1123	497	75.12	10800.00
	12	6	1212.15	9605	4217	291.69	10800.00
	12	7	721.58	5008	2975	167.59	10800.00
12	15	8	1024.83	4910	1969	273.45	10800.00
	15	8	842.88	10011	4725	389.25	10800.00
	15	8	356.91	8671	4322	423.34	10800.00
	15	9	1340.67	14311	8716	433.20	10800.00
	15	8	876.74	5847	4622	411.23	10800.00
Avg			461.31	3576.56	1771	127.23	6654.99

Table 6.17. Small and medium size instances of DBAP with L=12 (loose).

L	K	E	Z_{LB}	Z_{lp}^1	Z_{lp}^2	CPU^1	CPU^2
12	3	2	31.00	31	31	0.41	0.44
	3	3	31.00	31	31	0.81	0.81
	3	2	44.00	44	44	0.42	0.42
	3	3	43.25	243	44	1.67	1.86
	3	3	28.00	30	30	1.11	1.56
12	6	6	149.00	152	152	13.42	23.09
	6	4	799.00	1722	799	17.50	23.03
	6	5	314.70	1136	515	20.73	74.06
	6	5	332.56	1040	436	11.58	40.91
	6	4	126.00	127	127	2.30	6.09
12	9	7	542.49	2073	641	86.52	626.14
	9	7	249.00	1908	1184	55.89	10800.00
	9	6	175.00	197	190	21.72	197.69
	9	7	266.00	276	274	27.03	233.39
	9	8	574.03	983	974	70.09	10800.00
12	12	8	539.03	852	746	103.73	10800.00
	12	11	505.85	1169	636	193.75	10800.00
	12	10	314.00	684	377	101.03	10800.00
	12	9	395.00	525	418	107.94	2474.86
	12	9	678.28	2097	1313	152.70	10800.00
12	15	11	544.00	564	564	217.83	10800.00
	15	12	661.00	898	705	171.42	10800.00
	15	11	635.27	978	872	199.03	10800.00
	15	14	638.00	693	689	276.47	10800.00
	15	12	556.00	582	582	214.75	10800.00
Avg			366.86	761.40	494.96	82.79	4900.17

Table 6.18. Small size instances of DBAP with L=24 (loose).

L	K	E	Z_{LB}	Z_{lp}^1	Z_{lp}^2	CPU^1	CPU^2
24	3	1	37.00	37	37	0.84	0.87
	3	2	39.00	39	39	0.78	0.78
	3	3	25.00	28	27	2.28	3.14
	3	2	33.00	33	33	0.39	0.39
	3	2	45.00	45	45	0.41	0.41
24	6	3	122.00	123	123	22.91	26.36
	6	4	149.00	149	149	31.50	32.06
	6	4	136.50	435	137	41.55	58.14
	6	4	148.00	150	150	54.30	82.45
	6	3	81.00	89	89	50.14	65.83
24	9	6	220.00	240	238	223.50	10280.11
	9	6	207.93	211	211	234.83	400.36
	9	6	216.00	546	529	337.06	10800.00
	9	4	270.00	281	281	363.81	1467.68
	9	6	176.00	196	196	456.31	10800.00
24	12	8	272.00	518	406	1094.12	10800.00
	12	7	389.00	395	395	444.90	10800.00
	12	5	427.00	435	435	626.70	5031.23
	12	7	440.00	1292	670	696.85	10800.00
	12	5	301.63	826	529	454.52	10800.00
Avg			186.75	303.40	235.95	256.89	4112.49

Table 6.19. Medium size instances of DBAP with L=24 (loose).

L	K	E	Z_{LB}	Z_{lp}^1	Z_{lp}^2	CPU^1	CPU^2
24	15	10	690	706	706.00	1547.02	10800.00
	15	7	466	483	483	2402.22	10800.00
	15	9	526	536	536	2518.53	10800.00
	15	8	576	592	588	965.98	10800.00
	15	8	505	1768	1768	1773.14	10800.00
24	20	9	1116	1264	1163	1843.24	10800.00
	20	15	1235	1480	1480	9198.71	10800.00
	20	12	968	1098	1098	10411.12	10800.00
	20	10	1000	1013	1013	4433.53	10800.00
	20	10	1088	1108	1108	2737.23	10800.00
24	25	13*	0	3064	3064	10800.00	10800.00
	25	16*	0	1950	1950	10800.00	10800.00
	25	13	1598	1612	1612	10469.02	10800.00
	25	14	1539	1575	1575	8178.77	10800.00
	25	13*	0	1319	1319	10800.00	10800.00
Avg			753.80	1304.53	1297.53	5925.23	10800.00

7. CONCLUSION

Although there is no general agreement on which approach is better, generally the selection of time representation depends on the differences between the total number of decision variables needed for the clock-based representation and the total number of big-M constraints and the number of events needed for event-based representation. Our results show the advantages and disadvantages of each approach by testing them on two different problems. This work is an example for other studies related to the application of the event-based formulations apart from the process industry problems.

If a problem and data set require long horizon and small number of events, the event-based formulation can be useful as it is the one for LASP. However, if the total number of events converges to the maximum number of events, event-based formulations have poor performance. Although we cannot be sure whether the solution is optimal, the event-based formulation is the way of getting good results in shorter computation time. In DBAP, for more complex data sets, we obtain better solutions by using an event-based formulation. However, second data set is the easier version of the first data set and requires large number of events. Since finding the optimum number of events takes too much time in these types of data sets, finding an optimal solution with event-based formulations can be time-consuming. However, if the computation time is limited, better feasible solution can be obtained from event-based formulations. The decision maker should first analyze the problem and then choose the proper time representation.

We choose the event-based formulations in order to propose a branch and price algorithm, since they reduce the complexity of problems and result in an efficient formulation. Due to the big-M constraints and continuous variables, event-based formulations result in poor LP relaxations. Besides, finding an optimal solution for large-scale optimization problems takes too much time. Experiments show that the branch and price algorithm is inefficient for proving the optimality due to the highly degenerate

master problem relaxation. By implementing a heuristic method in a branch and price algorithm, it provides a feasible solution for complex problems. Thus, even if we cannot find optimum solutions, we can obtain good lower and upper bounds for the problems. At the root node, we obtain better lower bounds for DBAP and better upper bounds for LASP.

The performance of LASP, DBAP and other event-based formulations given in this thesis can be tested for more data sets. Additionally, different cases can be implemented in these problems and the formulations of more complex problems can be given. As a further study, different problems can be chosen to apply event-based formulation in order to compare the results with clock-based versions. General approach for finding the optimum number of events would be the subject of another study. In the branch and price algorithm, the performance of heuristics can be improved in order to compute good feasible solutions. Besides, the efficient solution of subproblems can provide more efficient algorithm. Other decomposition methods, such as the Lagrangean or Benders decomposition should be also considered.

APPENDIX A: CLOCK-BASED FORMULATIONS OF APPLICATIONS

A.1. Clock-based Formulation of Location and Scheduling Problem

The clock-based formulation of LASP is as follows as given in [33]:

$$\max LT \tag{A.1}$$

subject to

$$LT \leq T \tag{A.2}$$

$$\sum_{i=1}^{I}\sum_{n=1}^{N} a_{ijn} z_{int} \geq d_j(1-y_t) \qquad j=1,\ldots,J; t=1\ldots T \tag{A.3}$$

$$T(1-y_t) \geq LT+1-t \qquad t=1,\ldots,T \tag{A.4}$$

$$\sum_{t=1}^{T} z_{int} \leq \lfloor \frac{E_n}{e_n} \rfloor x_{in} \qquad i=1,\ldots,I; n=1\ldots N \tag{A.5}$$

$$z_{int} \leq x_{in} \qquad i=1,\ldots,I; n=1,\ldots,N; t=1\ldots T \tag{A.6}$$

$$\sum_{n=1}^{N} c_n \sum_{i=1}^{I} x_{in} \leq B \tag{A.7}$$

$$z_{int}, x_{in} \in \{0,1\} \qquad i=1,\ldots,I; n=1,\ldots,N; t=1\ldots T \tag{A.8}$$

$$LT \geq 0 \tag{A.9}$$

The objective function (A.1) aims to maximize the coverage lifetime. The coverage lifetime cannot exceed the planning horizon as shown in the constraints (A.2). Constraints (A.4) guarantee that period t is within the lifetime and additionally constraints (A.3) guarantee the coverage quality for each point. Constraints (A.5) ensure that the total number of active periods for a type n sensor placed at point i cannot exceed the ratio of battery energy to consumption rate. If the sensor is active at period t, it should be located at first as it is indicated in constraints (A.6). Constraints (A.7) imply that total replacement cost cannot be larger than the total budget.

Table A.1. Parameters and decision variables for the clock-based formulation of LASP.

Parameters	Definition
T	Planning horizon
Variables	Definition
LT	Lifetime of the network
z_{int}	Indicates whether a type n sensor placed on point i is active in period t or not
y_t	Indicates whether a period t is within lifetime LT or not

A.2. Clock-based Formulation of Desired Berth Allocation Problem

In the clock-based formulation, both the x and y axes are divided into equal sized sections. x axis represents the berth sections and is divided into L equal sized sections and y axis measures the time and is divided into T equal sized sections. Let block (x, y) be the block for time unit $x \in [1, T]$, and berth section $y \in [1, L]$. The clock-based

Table A.2. Parameters and decision variables for the clock-based formulation of DBAP.

Parameters	Definition
T	Planning horizon
Variables	Definition
δ_{klt}	Indicates whether the vessel k arrives at berth section l at time t or not
σ_{kxy}	Indicates whether the vessel k covers block (x, y) or not

formulation of DBAP is as follows as given in [34]:

$$\min \sum_{k=1}^{K} c_k + \sum_{k=1}^{K} f_k z_k + \sum_{k=1}^{K} \sum_{l=1}^{L} \sum_{t=1}^{T} f_k |l - des_k| d_{klt} \tag{A.1}$$

subject to

$$\sum_{l=1}^{L-w_k+1} \sum_{t=a_k}^{T-p_k+1} \delta_{klt} = 1 \qquad k = 1, \ldots, K \tag{A.2}$$

$$\sum_{x=t}^{t+p_k-1} \sum_{y=l}^{l+w_k-1} \sigma_{kxy} - p_k w_k - (\delta_{klt} - 1) M \geq 0 \qquad k = 1, \ldots, K; l = 1, \ldots, L; t = a_k, \ldots, T \tag{A.3}$$

$$\sum_{k=1}^{K} \sigma_{kxy} \leq 1 \qquad x = 1, \ldots, X; y = 1, \ldots, Y \tag{A.4}$$

$$p_k + \sum_{l=1}^{L-w_k+1} \sum_{t=a_k}^{T-p_k+1} t \delta_{klt} \leq c_k \qquad k = 1, \ldots, K \tag{A.5}$$

$$z_k \geq c_k - d_k \qquad k = 1, \ldots, K \tag{A.6}$$

$$\delta_{klt} \in \{0, 1\} \qquad k = 1, \ldots, K; l = 1, \ldots, L; t = 1 \ldots T \tag{A.7}$$

$$c_k, z_k \geq 0 \qquad k = 1, \ldots, K \tag{A.8}$$

$$\sigma_{kxy} \in \{0, 1\} \qquad k = 1, \ldots, K; x = 1, \ldots, X; y = 1 \ldots Y \tag{A.9}$$

In (A.1), we try to minimize the total departure time of vessels, total sum of positive deviations from due time of the vessels and the deviations from the desired locations by penalizing them. Constraints (A.2) ensure that each vessel must berth exactly once. Constraints (A.3) state that blocks occupied by each vessel rectangle must be adjacent. According to constraints (A.4), each block must be occupied by only one vessel. Constraints (A.5) guarantee that the departure time of a vessel should be at least the sum of processing time and berthing time. The positive deviation from due time is determined by means of constraints (A.6).

APPENDIX B: COMPARISON OF MODEL SIZE

Table B.1. Number of variables and constraints in LASP.

				Clock-based Formulation			Event-based Formulation		
J	B	T	E	# of cnst.	# of bin. var.	# of cont.	# of cnst.	# of bin. var.	# of cont.
16	1	200	3	9834	6632	1	659	128	99
	2	250	3	12284	8282	1	659	128	99
	3	300	4	14734	9932	1	868	160	132
25	1	200	2	15252	10250	1	702	150	102
	2	250	3	19052	12800	1	1028	200	153
	3	300	4	22852	15350	1	1354	250	204
36	1	200	3	21874	14672	1	1479	288	219
	2	250	4	27324	18322	1	1948	360	292
	3	300	4	32774	21972	1	1948	360	292
49	1	200	3	29700	19898	1	2012	392	297
	2	250	3	37100	24848	1	2012	392	297
	3	300	5	44500	29798	1	3288	588	495
64	1	200	2	38730	25928	1	1794	384	258
	2	250	4	48380	32378	1	3460	640	516
	3	300	4	58030	38828	1	3460	640	516
81	1	200	2	48964	32762	1	2270	486	326
	2	250	3	61164	40912	1	3324	648	489
	3	300	4	73364	49062	1	4378	810	652
100	1	200	2	60402	40400	1	2802	600	402
	2	250	4	75452	50450	1	5404	1000	804
	3	300	4	90502	60500	1	5404	1000	804
Avg				40108.00	26856.00	1	2432.67	463.43	360.81

Table B.2. Number of variables and constraints in DBAP (tight).

			Clock-based Formulation			USEB Formulation		
L	K	E	# of cnst.	# of bin. var.	# of cont.	# of cnst.	# of bin. var.	# of cont.
24	6	3	8543	15719	12	972	432	84
	6	3	7337	14513	12	972	432	84
	6	3	6993	14169	12	972	432	84
	6	3	7376	14552	12	972	432	84
	6	3	8207	15383	12	972	432	84
24	9	4	16903	34147	18	1579	864	114
	9	4	16478	33722	18	1579	864	114
	9	5	17183	34427	18	1961	1080	138
	9	4	16506	33750	18	1579	864	114
	9	3	17336	34580	18	1197	648	90
24	12	6	28393	60025	24	2784	1728	168
	12	5	28173	59805	24	2330	1440	144
	12	5	29730	61362	24	2330	1440	144
	12	7	28451	60083	24	3238	2016	192
	12	5	27680	59312	24	2330	1440	144
24	15	5	45613	95953	30	2699	1800	150
	15	6	45969	96309	30	3225	2160	174
	15	7	43218	93558	30	3751	2520	198
	15	5	46378	96718	30	2699	1800	150
	15	6	43299	93639	30	3225	2160	174
Avg			24488.30	51086.30	21.00	2068.30	1249.20	131.40

Table B.3. Number of variables and constraints in DBAP (loose).

			Clock-based Formulation			USEB Formulation		
L	K	E	# of cnst.	# of bin. var.	# of cont.	# of cnst.	# of bin. var.	# of cont.
24	6	3	6147	13323	12	972	432	84
	6	4	5536	12712	12	1282	576	108
	6	4	6045	13221	12	1282	576	108
	6	4	5665	12841	12	1282	576	108
	6	3	7443	14619	12	972	432	84
24	9	6	14288	31532	18	2343	1296	162
	9	6	14513	31757	18	2343	1296	162
	9	6	13847	31091	18	2343	1296	162
	9	4	13616	30860	18	1579	864	114
	9	6	14917	32161	18	2343	1296	162
24	12	8	26443	58075	24	3692	2304	216
	12	7	23044	54676	24	3238	2016	192
	12	5	23763	55395	24	2330	1440	144
	12	7	24567	56199	24	3238	2016	192
	12	5	25265	56897	24	2330	1440	144
24	15	10	35311	85651	30	5329	3600	270
	15	7	39523	89863	30	3751	2520	198
	15	9	40058	90398	30	4803	3240	246
	15	8	37749	88089	30	4277	2880	222
	15	8	37723	88063	30	4277	2880	222
Avg			20773.15	47371.15	21.00	2700.30	1648.80	165.00

APPENDIX C: RESULTS OF CLOCK-BASED FORMULATIONS

Table C.1. Clock-based results for small size instances with L=24 (tight).

	Clock-based Formulation			
K	CPU_{lp}	Z_{lp}	CPU_{lr}	Z_{lr}
3	0.20	27	0.03	23.42
3	0.33	31	0.01	21.84
3	0.30	30	0.01	24.14
3	1.08	146	0.03	33.73
3	1.15	46	0.03	30.65
6	0.55	44	0.06	43.12
6	1.66	75	0.11	61.72
6	1.44	96	0.12	78.16
6	24.00	401	0.14	63.62
6	1.34	57	0.06	49.10
9	19.52	224	0.22	102.25
9	424.42	430	0.30	88.75
9	62.64	315	0.22	78.13
9	9.16	124	0.23	102.74
9	3.23	105	0.26	75.12
12	258.42	537	0.47	170.76
12	820.37	527	0.45	172.56
12	142.92	584	0.41	146.29
12	28.37	226	0.31	170.22
12	594.42	609	0.56	172.98
Avg	119.78	231.70	0.20	85.46

Table C.2. Clock-based results for medium size instances with L=24 (tight).

	Clock-based Formulation			
K	CPU_{lp}	Z_{lp}	CPU_{lr}	Z_{lr}
15	5132.60	662	0.80	187.98
15	860.53	450	0.83	200.66
15	1870.50	596	0.81	249.43
15	2097.20	722	0.69	168.27
15	6715.10	708	0.78	225.60
20	32500.00	990	1.01	300.50
20	6495.90	1064	1.11	350.65
20	2974.10	704	1.11	316.29
20	23000.00	4052	1.22	328.45
20	32400.00	4316	1.12	275.39
25	32400.00	15475	2.22	438.48
25	43200.00	9563	1.92	433.50
25	32400.00	14833	1.97	406.18
25	49548.00*	2565	1.70	561.73
25	43200.00	9415	1.90	524.13
Avg	20986.26	4407.67	1.28	331.15

Table C.3. Clock-based results for small and medium size instances with L=12 (tight).

	Clock-based Formulation			
K	CPU_{lp}	Z_{lp}	CPU_{lr}	Z_{lr}
3	0.12	18	0.00	15.17
3	0.09	19	0.00	17.33
3	0.14	30	0.00	20.57
3	0.42	549	0.01	36.78
3	0.08	23	0.00	21.07
6	2.16	177	0.02	56.72
6	13.00	189	0.03	51.85
6	100.98	1618	0.05	67.52
6	72.53	575	0.03	54.76
6	1.00	66	0.02	46.01
9	124.55	846	0.06	97.41
9	373.73	828	0.09	73.50
9	1215.57	1146	0.09	99.07
9	1289.42	1539	0.09	94.78
9	36.46	332	0.09	91.88
12	20000.00	2507	0.17	153.86
12	10800.00	1559	0.15	180.08
12	16.54	211	0.12	140.17
12	14543.00*	4538	0.17	124.98
12	7334.23	2973	0.20	192.70
15	13194.53	1558	0.20	176.01
15	30900.00	4819	0.20	182.88
15	21600.00	3917	0.20	196.82
15	23300.00	5980	0.22	248.40
15	21600.00	3714	0.19	201.58
Avg	6660.74	1589.24	0.10	105.68

Table C.4. Clock-based results for small size instances with L=24 (loose).

	Clock-based Formulation			
K	CPU_{lp}	Z_{lp}	CPU_{lr}	Z_{lr}
3	0.15	37	0.01	37.00
3	0.21	39	0.01	35.38
3	0.25	27	0.01	20.35
3	0.09	33	0.01	33.00
3	0.12	45	0.01	45.00
6	0.81	123	0.06	122.10
6	0.67	149	0.06	149.00
6	4.56	137	0.10	121.77
6	0.59	150	0.06	148.07
6	0.76	89	0.06	81.45
9	7.00	238	0.18	220.70
9	9.59	211	0.25	194.74
9	13.31	244	0.26	216.83
9	1.95	281	0.14	270.13
9	7.89	196	0.20	176.80
12	100.76	406	0.43	273.14
12	4.62	395	0.42	389.32
12	11.00	435	0.32	427.29
12	113.29	670	0.45	341.47
12	68.00	529	0.46	296.42
Avg	17.28	221.70	0.18	180.00

Table C.5. Clock-based results for medium size instances with L=24 (loose).

	Clock-based Formulation			
K	CPU_{lp}	Z_{lp}	CPU_{lr}	Z_{lr}
15	13.48	697	0.57	690.17
15	21.38	483	0.71	467.40
15	9.09	536	0.46	526.98
15	9.45	588	0.59	577.17
15	132.46	633	0.71	506.07
20	115.46	1162	1.32	1117.13
20	127.14	1372	1.12	1236.05
20	125.76	1095	1.15	969.05
20	20.57	1013	1.09	1000.60
20	78.73	1108	1.15	1088.27
25	87.06	1695	1.67	1559.54
25	36.89	1948	1.51	1934.19
25	31.26	1611	1.30	1598.63
25	85.73	1574	1.45	1540.99
25	74.28	1311	1.29	1282.56
Avg	64.58	1121.73	1.07	1072.99

Table C.6. Clock-based results for small and medium size instances with L=12 (loose).

	Clock-based Formulation			
K	CPU_{lp}	Z_{lp}	CPU_{lr}	Z_{lr}
3	0.09	31	0.01	31.00
3	0.06	31	0.01	31.00
3	0.10	44	0.01	44.00
3	0.17	44	0.01	33.65
3	0.06	30	0.01	28.30
6	0.34	152	0.03	149.10
6	53.89	799	0.03	84.28
6	11.57	515	0.03	84.65
6	17.54	436	0.04	109.71
6	0.28	127	0.01	126.20
9	53.21	641	0.09	198.69
9	56.23	592	0.09	250.71
9	1.43	190	0.07	176.10
9	0.89	274	0.06	266.57
9	50.28	974	0.09	229.57
12	40.15	742	0.14	309.83
12	82.04	636	0.17	397.49
12	152.92	377	0.14	316.05
12	6.26	418	0.14	396.27
12	81.53	1092	0.17	338.88
15	10.01	564	0.18	544.97
15	8.31	690	0.28	662.48
15	142.57	779	0.21	523.84
15	24.92	688	0.25	639.90
15	13.79	582	0.25	557.17
Avg	32.34	457.92	0.10	261.22

REFERENCES

1. Mouret, S., I. E. Grossmann and P. Pestiaux, "Time Representations and Mathematical Models for Process Scheduling Problems", *Computers & Chemical Engineering*, Vol. 35, No. 6, pp. 1038 – 1063, 2011.

2. Méndez, C. A., J. Cerdá, I. E. Grossmann, I. Harjunkoski and M. Fahl, "State-of-the-Art Review of Optimization Methods for Short-Term Scheduling of Batch Processes", *Computers & Chemical Engineering*, Vol. 30, No. 6-7, pp. 913 – 946, 2006.

3. Floudas, C. and X. Lin, "Mixed Integer Linear Programming in Process Scheduling: Modeling, Algorithms, and Applications", *Annals of Operations Research*, Vol. 139, No. 1, pp. 131–162, 2005.

4. Floudas, C. A. and X. Lin, "Continuous-Time versus Discrete-Time Approaches for Scheduling of Chemical Processes: A Review", *Computers & Chemical Engineering*, Vol. 28, No. 11, pp. 2109 – 2129, 2004.

5. Koné, O., C. Artigues, P. Lopez and M. Mongeau, "Event-Based MILP Models for Resource-Constrained Project Scheduling Problems", *Computers & Operations Research*, Vol. 38, No. 1, pp. 3 – 13, 2011.

6. Mockus, L. and G. Reklaitis, "Mathematical Programming Formulation for Scheduling of Batch Operations Based on Nonuniform Time Discretization", *Computers & Chemical Engineering*, Vol. 21, No. 10, pp. 1147 – 1156, 1997.

7. Pan, M., X. Li and Y. Qian, "Continuous-Time Approaches for Short-Term Scheduling of Network Batch Processes: Small-Scale and Medium-Scale Problems", *Chemical Engineering Research and Design*, Vol. 87, No. 8, pp. 1037 – 1058, 2009.

8. Sundaramoorthy, A. and C. T. Maravelias, "Computational Study of Network-Based Mixed-Integer Programming Approaches for Chemical Production Scheduling", *Industrial & Engineering Chemistry Research*, Vol. 50, No. 9, pp. 5023–5040, 2011.

9. Ierapetritou, M. G. and C. A. Floudas, "Effective Continuous-Time Formulation for Short-Term Scheduling. 1. Multipurpose Batch Processes", *Industrial & Engineering Chemistry Research*, Vol. 37, No. 11, pp. 4341–4359, 1998.

10. Zhang, X. and R. Sargent, "The Optimal Operation of Mixed Production Facilities—a General Formulation and Some Approaches for the Solution", *Computers & Chemical Engineering*, Vol. 22, No. 9, pp. 1287 – 1295, 1998.

11. Mockus, L. and G. V. Reklaitis, "Continuous Time Representation Approach to Batch and Continuous Process Scheduling. 1. MINLP Formulation", *Industrial & Engineering Chemistry Research*, Vol. 38, No. 1, pp. 197–203, 1999.

12. Maravelias, C. T. and I. E. Grossmann, "A New General Continuous-Time State Task Network Formulation for Short-Term Scheduling of Multipurpose Batch Plants", *Industrial & Engineering Chemistry Research*, Vol. 42, No. 13, pp. 3056–3074, 2003.

13. Hazaras, M. J., C. L. Swartz and T. E. Marlin, "Flexible Maintenance within a Continuous-Time State-Task Network Framework", *Computers & Chemical Engineering*, Vol. 46, No. 1, pp. 167 – 177, 2012.

14. Kondili, E., C. Pantelides and R. Sargent, "A General Algorithm for Short-Term Scheduling of Batch Operations. I. MILP Formulation", *Computers & Chemical Engineering*, Vol. 17, No. 2, pp. 211 – 227, 1993.

15. Janak, S. L., X. Lin and C. A. Floudas, "Enhanced Continuous-Time Unit-Specific Event-Based Formulation for Short-Term Scheduling of Multipurpose Batch Processes: Resource Constraints and Mixed Storage Policies", *Industrial & Engineering Chemistry Research*, Vol. 43, No. 10, pp. 2516–2533, 2004.

16. Giannelos, N. F. and M. C. Georgiadis, "A Simple New Continuous-Time Formulation for Short-Term Scheduling of Multipurpose Batch Processes", *Industrial & Engineering Chemistry Research*, Vol. 41, No. 9, pp. 2178–2184, 2002.

17. Sundaramoorthy, A. and I. Karimi, "A Simpler Better Slot-Based Continuous-Time Formulation for Short-Term Scheduling in Multipurpose Batch Plants", *Chemical Engineering Science*, Vol. 60, No. 10, pp. 2679 – 2702, 2005.

18. Méndez, C., G. Henning and J. Cerdá, "An MILP Continuous-Time Approach to Short-Term Scheduling of Resource-Constrained Multistage Flowshop Batch Facilities", *Computers & Chemical Engineering*, Vol. 25, No. 4-6, pp. 701 – 711, 2001.

19. Stefansson, H., S. Sigmarsdottir, P. Jensson and N. Shah, "Discrete and Continuous Time Representations and Mathematical Models for Large Production Scheduling Problems: A Case Study from the Pharmaceutical Industry", *European Journal of Operational Research*, Vol. 215, No. 2, pp. 383 – 392, 2011.

20. Saharidis, G. K., M. Minoux and Y. Dallery, "Scheduling of Loading and Unloading of Crude Oil in a Refinery Using Event-Based Discrete Time Formulation", *Computers & Chemical Engineering*, Vol. 33, No. 8, pp. 1413 – 1426, 2009.

21. Wang, S. and M. Guignard, "Hybridizing Discrete- and Continuous-Time Models for Batch Sizing and Scheduling Problems", *Computers & Operations Research*, Vol. 33, No. 4, pp. 971 – 993, 2006.

22. Zapata, J. C., B. M. Hodge and G. V. Reklaitis, "The Multimode Resource Constrained Multiproject Scheduling Problem: Alternative Formulations", *AIChE Journal*, Vol. 54, No. 8, pp. 2101–2119, 2008.

23. Castro, P. M. and J. F. Oliveira, "Scheduling Inspired Models for Two-Dimensional Packing Problems", *European Journal of Operational Research*, Vol. 215, No. 1, pp. 45 – 56, 2011.

24. Castro, P. M. and I. E. Grossmann, "From Time Representation in Scheduling to the Solution of Strip Packing Problems", *Computers & Chemical Engineering*, Vol. 44, No. 1, pp. 45 – 57, 2012.

25. Castro, P., A. P. F. D. Barbosa-Póvoa and H. Matos, "An Improved RTN Continuous-Time Formulation for the Short-Term Scheduling of Multipurpose Batch Plants", *Industrial & Engineering Chemistry Research*, Vol. 40, No. 9, pp. 2059–2068, 2001.

26. Seid, R. and T. Majozi, "A Novel Technique for Prediction of Time Points for Scheduling of Multipurpose Batch Plants", *Chemical Engineering Science*, Vol. 68, No. 1, pp. 54 – 71, 2012.

27. Li, J. and C. A. Floudas, "Optimal Event Point Determination for Short-Term Scheduling of Multipurpose Batch Plants via Unit-Specific Event-Based Continuous-Time Approaches", *Industrial & Engineering Chemistry Research*, Vol. 49, No. 16, pp. 7446–7469, 2010.

28. Wu, D. and M. G. Ierapetritou, "Decomposition Approaches for the Efficient Solution of Short-Term Scheduling Problems", *Computers & Chemical Engineering*, Vol. 27, No. 8–9, pp. 1261 – 1276, 2003.

29. Li, Z. and M. G. Ierapetritou, "Integrated Production Planning and Scheduling Using a Decomposition Framework", *Chemical Engineering Science*, Vol. 64, No. 16, pp. 3585 – 3597, 2009.

30. Janak, S. L. and C. A. Floudas, "Improving Unit-Specific Event Based Continuous-Time Approaches for Batch Processes: Integrality Gap and Task Splitting", *Computers & Chemical Engineering*, Vol. 32, No. 4–5, pp. 913 – 955, 2008.

31. Maravelias, C. T. and I. E. Grossmann, "A Hybrid MILP/CP Decomposition Approach for the Continuous Time Scheduling of Multipurpose Batch Plants", *Computers & Chemical Engineering*, Vol. 28, No. 10, pp. 1921 – 1949, 2004.

32. Türkoğulları, Y. B., *Optimal Placement, Scheduling and Routing to Maximize Lifetime in Wireless Sensor Networks*, Ph.D. Thesis, Boğaziçi University, 2011.

33. Türkoğulları, Y. B., N. Aras, K. Altınel and C. Ersoy, "Optimal Placement and Activity Scheduling to Maximize Coverage Lifetime in Wireless Sensor Networks", *22nd International Symposium on Computer and Information Sciences (ISCIS 2007)*, pp. 1–6, 2007.

34. Ak, A., *Berth and Quay Crane Scheduling: Problems, Models and Solution Methods*, Ph.D. Thesis, Georgia Institute of Technology, 2008.

35. Sherali, H. D. and J. C. Smith, "Improving Discrete Model Representations Via Symmetry Considerations", *Management Science*, Vol. 47, No. 10, pp. 1396–1407, 2001.