2009

Hybrid Nested Partitions method with Intelligent Greedy Search for solving Weapon-Target Assignment Problem

Gunhyung Cho
Iowa State University

Hybrid Nested Partition method with intelligent greedy search for solving Weapon Target Assignment problem

by

Gunhyung Cho

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Industrial Engineering

Program of Study Committee:
Sigurdur Olafsson, Major Professor
Robert Ruben
Lizhi Wang

Iowa State University

Ames, Iowa

2009

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The Weapon-Target Assignment (WTA) problem is one of the most important problems of military applications of operations research. The objective of the WTA problem is to find proper assignments of weapons to targets which minimize the expected damage of defensive side. The WTA problem is known to be NP-complete. In this paper, hybrid Nested Partitions (NP) method is proposed to solve WTA problems. The proposed algorithm is named as "Hybrid NP method with intelligent greedy search". The NP method has been found to be very effective for solving complex large-scale discrete optimization problems. In addition to that, due to the inherent flexibility of the NP method, any other heuristic for generating good feasible solutions can be incorporated and improve the performance of the NP method. The intelligent greedy search is an improved version of greedy search which finds good solutions very quickly. The proposed algorithm combines the advantages of the NP method and intelligent greedy search. The simulation results show that the proposed algorithm is very efficient for solving the WTA problem.

CHAPTER 1. INTRODUCTION

1.1 Overview

1.1.1 Modern Military weapons' development

As the technologies have developed, modern military weapons have also improved. Especially, during World War II, stimulation occurred. Thus, during that period, the Cold War, the countries involved were engaged in the continuous competition developing weapons and counter-weapons. There were many kinds of weapons developed during that period such as nuclear missiles, ballistic missiles, and so on. The essential point is that those weapons are more accurate and destructive than the old ones. With the advent of these long-range and precise weapons, the paradigm of modern war changed. Moreover, the information technology which improved dramatically for decades has allowed for the weapons to be much more effective.

1.1.2 Weapon Target Assignment Problem

Under these situations, military experts have been trying to find ways of protecting their own countries from their adversaries. Among those endeavors, one endeavor is the guard systems of missiles called weapon-target assignment (WTA) problem. The WTA problem is devised to find the proper weapon to target assignments which guarantees minimum expected damages. Offensive attacks with diverse weapon platforms such as missiles, combat air fighters, and so on. Defenders have defense systems which incorporate various weapons.

According to the properties of each weapon systems, the results of engagement will vary. Consequently, many military experts have been working on the project, the WTA problem to assure their safety since the beginning of the 20th century.

The WTA problem is known to be NP-complete. Lloyd and Witsenhausen (1986) found that the SWTA (Static weapon-target assignment) problem is NP-complete. That is, the processing time of solving WTA problems is exponentially increased according to the size of the problem. Thus, when we have a large scale problem which has a number of variables, we couldn't obtain an optimal solution in a short time. Consequently, studies on WTA problem have been employed heuristic algorithms to solve the WTA problem.

Studies which have been done for resolving WTA problems have employed various algorithms. In the 1980s, traditional algorithms were mainly used, such as implicit enumeration algorithms, branch and algorithms, dynamic programming. These algorithms were limited for solving the WTA problems efficiently because they were difficult to be implemented in computer programs, especially when the number of variables is very large. Since then, several algorithms have been developed such as neural networks, genetic algorithms. Those algorithms have been used to solve some complicated WTA problems with large scale by E. Wachholder (1989) and Ressler (1992). However, the algorithms have some problems. Sometimes, the algorithms are unstable and not able to find global optimum. They just converge to local optimum which is not the best solution. Later, other algorithms such as taboo search, simulated annealing algorithms (SA) have been used to solve WTA problems by Cullenbine (2000) and H. R. Li (2000). The important property of those

algorithms is that they could be combined. That is, they could improve their performances by combining themselves. However, the efficiency was too low. Recently, the parallel ant colony algorithm and improved GA are used to solve WTA problems by Z. J. Lee, et al (2002, 2003). S Gao (2005) proposes particle swarm optimization algorithm to resolve the WTA problem. The limitation of recent three algorithms is that the efficiency of the algorithms is not verified. That is, those algorithms satisfy just some problems with limited number of variables.

As mentioned previously, algorithms which have been used since 1980s are limited in terms of their performance. Those algorithms cannot resolve large scale problems in an acceptable time and therefore the huge amount of processing time makes it impossible to apply them in the real war situation. The time available for assigning proper weapons to various targets is just seconds to minutes, from detecting targets to before getting hit by the targets. All the assignments should be finished within a very short time.

1.2 Problem Statement

This research takes the viewpoint of military experts who want to protect themselves from adversaries' attacks. The objective of the WTA problem is to find the optimal assignment of weapons to targets so that they can reduce the total expected damages as much as possible. To assist the decision of allocation, we formulated an asset based static WTA problem and devised a heuristic algorithm to solve the problem.

In this paper, we used an algorithm called the Nested Partition method (NP). The NP algorithm is a relatively novel optimization method that has been found to be very effective for solving discrete optimization problems. It means that the NP algorithm can resolve a large scale problem with a satisfactory performance within short processing time. The NP is based on the concept of adaptive sampling and can be used with other algorithms by combining. The NP consists of four steps; partitioning, sampling, calculating promising index and moving. By incorporating other algorithms into the NP framework, the performance of the NP method can be improved significantly. The ant colony algorithm has been applied in the sampling step by Sameh Al-shihabi (2004) and shown better performance than the pure NP. Another good example is a hybrid of NP, Binary Ant System, and Linear Programming for the Multidimensional Knapsack Problem written by Sameh Al-shihabi and Sigurdur Olafsson (2009).

However, the NP method which shows very good performances when solving large scale discrete optimization problems has never been used solve WTA problems before. Therefore, we decided to use the NP method to WTA problems, assuming that the NP method will solve WTA problems very efficiently in terms of the best fitness values and the processing time.

1.3 Research Objective

The purpose of this paper is to provide good algorithms to decision makers which can solve the WTA problem efficiently. The algorithms for the WTA problem have two requirements. They should find good solutions which can guarantee the minimum damages. In addition to

that, the processing time should be short enough to finish all the assignment before getting hit by enemy's attacks. We used a hybrid NP algorithm which applies greedy search algorithm in the sampling step. The greedy search incorporated into the proposed algorithm is not a pure greedy search. We devised and named it as "Intelligent greedy search". The concept of the intelligent greedy search is to start from intuitively good solutions, rather than just random solutions. The intelligent greedy search will be explained in chapter 4 in detail. This hybrid NP algorithm incorporating the intelligent greedy search shows very good performance in finding good solutions within relatively short time. The proposed algorithm in this paper yields very good results in terms of quickness and high-quality.

1.4 Thesis Organization

The remainder of this paper is organized into five chapters. In the next chapter the relevant studies are presented. Chapter 3 is devoted to model formulation and Chapter 4 presents the algorithms used in this paper. In Chapter 5, we will show the numerical examples that illustrate the implementation of the algorithms and prove the performance of the proposed algorithm is the best among those algorithms in this paper. Finally, Chapter 6 presents concluding remarks and describes the future directions for this research.

CHAPTER 2. LITERATURE REVIEW

2.1. Overview

The WTA problem is one of the most important problems in the military operations research area. Researches about the WTA problem started from the 1950s and 1960s. Manne (1958), Braford (1961) and Day (1966) investigated the modeling issues for WTA problem. Lloyd and Witsethausen, (1986) found that the WTA problem is NP-complete problem. This means that there are no exact optimization methods to solve the WTA problem. In other words, it is almost impossible to solve this problem directly because the scale of the problem increases the complexity of the problem exponentially. However, limited to some special cases, exact algorithms for the WTA problem were studied. DenBroader et al. (1987) and Katter (1986) considered the case of identical type of weapons. Another special case is that targets can receive at most one weapon (Chang et al., 1987 and Orlin, 1987). At this time, heuristics employed to solve the WTA problem are based on nonlinear network flow (Castanon et al., 1987), neural networks (Wacholder, 1989), and genetic algorithms (Grant et al., 1993). Among those algorithms, genetic algorithms are still used very frequently.

Recently, several heuristic algorithms have been used to solve the WTA problem. Those are greedy search, ant colony optimization (Lee et al., 2002), particle swarm optimization (Zeng et al., 2006). Those algorithms have demonstrated good performances in solving the WTA problem. Sometimes, those algorithms are combined with other algorithms to enhance their performance. For example, Lee (2002, b) et al. used ant colony optimization algorithm with

simulated annealing algorithm, and also they used genetic algorithm with simulated annealing algorithm in other paper.

2.2 Algorithms and Models used to solve WTA problem

Genetic algorithms (GAs) are one of the most frequently employed algorithms to solve the WTA problem, which have shown good performances as search algorithms (Back et al., 1997; Davis, 1991; Goldberg and Lingle, 1985; Lai and Coghill, 1998; Ngo and Li, 1998). Since GAs demonstrated superior performance over other algorithms in the previous literatures, researchers began to use GAs to resolve the WTA problems.

Lee (2002, a) et al. proposed an algorithm whose name is “a genetic algorithm with domain knowledge”. This algorithm employed simulated annealing search as a local search method in the genetic algorithm framework and included domain specific knowledge into the crossover operator and the local search mechanism to improve its performance. Consequently, this algorithm outperformed its competitors in his study.

Gao (2007) et al. also used an algorithm which is based on a genetic algorithm. The algorithm proposed by Gao et al is named as “An Immune Genetic Algorithm”. The immune system is used as a local search mechanism for genetic algorithm.

Other algorithm which shows good performances when solving the WTA problem is an ant colony optimization algorithm. Lee (2002, b) et al proposed an algorithm which is based on ant colony optimization incorporating immune system in its search mechanism. Ant colony

optimization (ACO) itself performs very well on diverse applications of optimization problem such as production scheduling, routing problem (Ikeda et al., 2005; K.L. Mak et al., 2007; B. Yu et al., 2009). In this paper, the immunity based ACO demonstrates good performance for solving WTA problems by exploiting the advantages of immune system and ACO.

Zeng (2006) et al. proposed an algorithm whose name is “discrete particle swarm optimization (DPSO)”. This algorithm sponges the advantages of particle swarm optimization (PSO) and genetic algorithm (GA). The PSO is very popular algorithm because of its simplicity of implementation and ability to converge to a good solution. In the DPSO algorithm, the greedy search strategy is employed to control the local search and enhance the efficiency of the algorithm and the concept of “permutation” is included to the update strategy. The performance of DPSO algorithm is assessed by comparison with other algorithms such as general GA and GA with greedy eugenics. Conclusively, the DPSO algorithm is found to be very effective to solve the WTA problem in terms of the processing time and the convergence to the reasonably good solutions.

There is an important concept for targeting problem. Kevin and Alan (2004) studied on this concept. The subject of their research is “shoot-look-shoot” which is the manner in which information that we can obtain during the engagement should be considered the process of target assignments. That is, if we detect a target, we do not shot at it several targets simultaneously, but shot independently after observing whether the target is being hit or not. The reason why the concept of “shoot-look-shoot” has become increasingly important is

related with the development of modern weapons. Modern weapons have been improved in the aspects of the range and accuracy. As much as the weapons are improved, the price of weapons also increased. Thus, we need to make a decision which guarantees not only an effective protection but also an efficient resource use.

The difference between this paper and some other papers are as follows. First, we used an algorithm, NP method, which is found to be a very effective algorithm, but has never been used to solve the WTA problem. Second, we added a constraint to the model used in this paper, which is incorporating the concept of "shoot-look-shoot". That is, we assumed that every target can be assigned at most one weapon. Some papers can allow multi weapon assignment on one target, but only one weapon to one target assignment is allowed in this paper.

In the following chapter, we present the mathematical model. Then, in the chapter 4, we present the proposed algorithm and other comparison algorithms.

CHAPTER 3. MODEL DESCRIPTION – WTA PROBLEM

3.1 Introduction

In this paper, we studied Asset-Based static weapon-target assignment problem (SWTA). The reason why we use the term 'asset-based' is that the objective of the problem is to protect valuable assets from enemy's attack. We regard the assets of the defense as an important factor in the objective function in the model.

In the Asset-Based SWTA problem, the offense launches various types of missiles or air fighters at valuable assets of the defense. As a counter action, the defense allocates its weapons to these missiles so that the expected damages can be minimized.

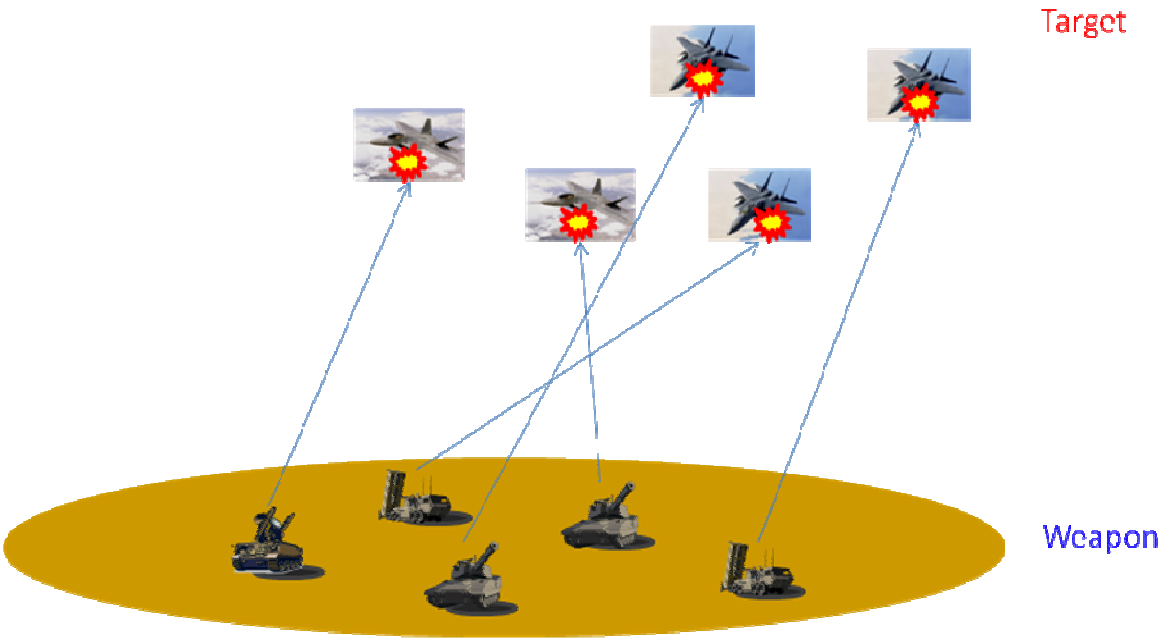

Figure 1. Illustration of Weapon Target Assignment Problem

This figure will help us to understand the scenario of the WTA problem used in this paper. The term used in this paper such as “weapon”, “target” is from the perspective of defensive side. Thus, in this paper, the “weapon” is a weapon of the defender which is supposed to be assigned to the “target” that is a weapon of the offensive. As the figure above, the types of weapons and targets may be different. Considering the characteristics of weapons and targets, the assignments have to be decided properly to achieve the goal.

The following key assumptions are made for the WTA problem in this paper. First, the number of weapons and targets are the same. This is a simplifying assumption that will rarely happen in practice. However, there is no loss of generality as dummy weapons or targets can be added if needed. Those dummy weapons or targets are not affecting the performance of the algorithms. For example, we use a dummy weapon when targets outnumber weapons. A dummy weapon has a very low kill probability. So, it will be assigned to a target which is relatively not threatening. This is same for a dummy target. Thus, by this assumption, following two sub-assumptions are made. All weapons must be assigned to targets and targets must be assigned by weapons. And the assignment must be one weapon to one target. The second sub-assumption is not just due to the first main assumption. Another reason for the second sub-assumption is coming from the concept of “shoot-look-shoot” policy (Kevin and Alan, 2004). The idea of “shoot-look-shoot” is as follows. For example, suppose we have a weapon whose single-shot probability of killing a target is 0.9, but the target is so dangerous that we can’t accept even 0.1’s survival probability. In this case, we can launch two shots to the target to increase the probability of killing up to 0.99 at the expense of two shots. Alternatively, we can achieve the kill probability of 0.99 by shooting at the target,

looking whether it is shot down, and shooting again if necessary. The later is more efficient shooting policy in terms of economy. Thus, we applied the concept of “shoot-look-shoot” on this problem.

The other key assumption is that the information which is necessary to formulate this problem is automatically obtained by an external information system. The information needed to solve the WTA problem are related with the characteristics of weapons and targets such as, weapon values, target values, kill probability and destroy probability. As soon as the military radar systems are detecting the targets, we assume that the information can be obtained from its database.

In this paper, we used randomized data sets which include all the information needed to formulate the WTA problem because not only it is impossible to get the real information about weapons and targets, but also the real data is not necessary to simulate my algorithm. We just need to verify the proposed algorithm finds good solutions quickly. Thus, randomized data sets were good enough to achieve my goal.

3.2 Notation

We used the asset based SWTA problem in this paper.

Following notations are used in this paper.

W_k : The value of asset k

p_{ij} : The probability that target *j* destroys weapon *i*

γ_{ij} : The probability that weapon *i* kill target *j*

m : The number of weapons

n : The number of targets

x_{ij} : The decision variable

For convenience, we named p_{ij} as the probability of destroying and γ_{ij} as the probability of killing. The objective function represents the total expected loss of assets of the defensive.

The problem is modeled as the following.

$$\min \sum_{k=1}^{m} w_k \sum_{j=1}^{n} \left[p_{kj} \prod_{i=1}^{m} (1-\gamma_{ij} x_{ij}) \right] \quad (1)$$

$$s.t. \sum_{i=1}^{m} x_{ij} = 1, \quad \text{j=1,2,...,n} \quad (2)$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad \text{i=1,2,...,m} \quad (3)$$

$$x_{ij} = \{0,1\}, \quad \text{i=1,2,...,m, j=1,2,...,n} \quad (4)$$

The constraint (2) means that each target can be attacked by only one weapon at a time. This is due to the concept of "shoot-look-shoot", as we explained. The second constraint (3) means that each weapon can be assigned to only one target at a time. The decision variable is x_{ij} which is only able to take one of two values, $\{0,1\}$. If x_{ij} =1, it indicates that weapon *i* is

allocated to target *j*. If not, the weapon *i* is not assigned to the target *j*. Solutions are presented as matrices with the size of *m* by *n*.

There are many assignment problems which can be solved easily. For example, as a famous optimization problem, traveling salesman problem has an assignment problem as a sub-problem. Another example is a postman problem which is also an assignment problem. Those two problems are applications of weighted 1-matching problem and can be resolved easily. The WTA problem is also a kind of assignment problems. However, this is known to be very difficult to solve. The reason why the WTA problem is NP-complete is in its complexity due to the objective function in the model. The WTA problems are formulated by non-linear programming problem which is very difficult to solve, comparing with linear and integer programming problem.

There are two kinds of static WTA models; asset based and target value based model. The reason why the asset based model is used in this paper is that this model concerns more realistic situation than the other model does. The objective function of the target value based model is as follows:

$$\min \sum_{i=1}^{N} V_i \prod_{j=1}^{N} (1-K_{ij})^{X_{ij}} \qquad (5)$$

In this model, V_i is the target value which means the expected damage value of target *i*, and $\prod_{j=1}^{N} (1-K_{ij})^{X_{ij}}$ is the probability of killing target *i* when assigned with weapon *j*.

This model assumes that target values are applied to all weapons exactly, regardless of the characteristics of weapons. However, this assumption has a big problem because it doesn't consider that every target and weapon has its own properties. It could happen that target A is very effective to weapon B, but almost useless to weapon C. Thus, this situation needs to be considered. The asset based model concerns this situation by adding the probability of destroying in the objective function. Since the latter model reflects the real situation better and yields more realistic results, we choose this model in this paper.

CHAPTER 4. METHODOLOGY

4.1 Nested Partitions method

4.1.1. Introduction

In this paper, the proposed algorithm is “Hybrid Nested partitions (NP) method with intelligent greedy search”. The NP method is relatively noble optimization method that shows very good performance when solving discrete optimization problems. There are many discrete optimization problems in practical world and the NP method has been useful in those various applications. The advantages of the NP method can be summarized following two words; flexibility and effectiveness. Basically, the NP method can be used to solve diverse application area problems. In addition, the NP method can incorporate any other optimization algorithms and make its performance better. These are the examples of the NP method’s flexibility. Speaking of NP method’s effectiveness, various applications have shown that the NP method efficiently solves complex large scale optimization problems which are intractable to solve by traditional optimization methods.

4.1.2. NP framework

Basically, the NP framework consists of four aspects: Partitioning, Sampling, Calculating promising index and Moving. Following is the pure NP algorithm’s framework in detail.

First, in the partitioning part, the NP method partitions the feasible region into sub-regions. And then, in the sampling part, the NP method generates sample solutions from each

partitioned region. After sampling, the NP method assesses the potential of each region by comparing the best performance values of sampled solutions from each sub-region and complimentary region. The last stage is moving. In this stage, we have two ways to choose. If the best performance value from a sub-region of the most promising region is better than that of the complimentary region, then the sub-region which has the best performance value is considered the next most promising region. If not, we go previous depth. This procedure is done iteratively until a stop criterion is satisfied.

```
<Procedure : Pure NP>
Begin
Depth← 0;
While (depth doesn't reach the bottom) do
        Partition a feasible region into sub-regions and complimentary region;
        Generate random sample solutions from each region;
        If  the performance of promising region
                Then
                        Accept new solution;
                        Promising region←current solution region;
                        Depth←Depth+1;
                Else;
                        Backtrack;
                        Depth←depth-1;
        End;
        Update process history;
        End;
End;
```

Figure 2. Pseudocode for the pure NP

Following figure will help us to understand the NP framework.

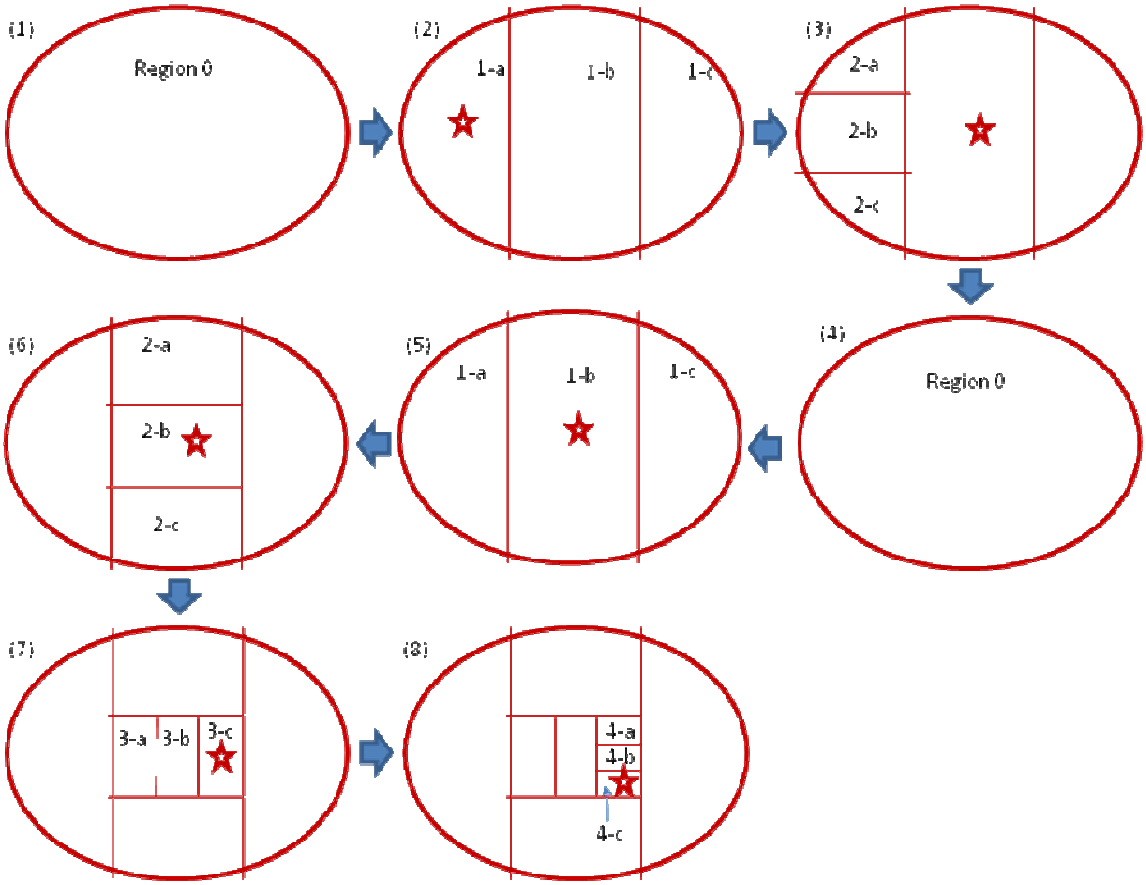

Figure 3. An example of the NP framework

The large circle means the whole feasible region. In order to find the optimal solution, the NP method partitions the feasible region into M sub-regions. Thus, we divide the feasible region into three sub-regions like (2). Then, we generate some feasible solutions from each sub-region and compare the performances of feasible solutions. We take the region '1-a' as the most promising region because it has the best solution at this juncture. We cycled the NP frame work one time from partitioning to moving. Continuously, we follow the procedure of NP framework. We partition the most promising region into three sub-regions and generate

feasible solutions from them like (3). However, at this time, sub-regions from the most promising region do not have the best solution. The best solution was in the complimentary region. Thus, we recognize that the region '1-a' we regarded as the most promising region was wrong and move back. We backtrack from (3) to (4) and restart the procedure. Iteratively, we do this procedure until the stop criterion is met. Finally, we obtain the optimal solution from the region '4-c'.

4.1.3. Characteristics of NP method

The NP method has important characteristics which explain why this algorithm is regarded as a good heuristic algorithm. First of all, this algorithm can be used in various application areas. Another characteristic is that this algorithm can exploit any other heuristic algorithms by incorporating them into the sampling part. By incorporating other algorithms into the sampling part, NP method can improve its performance. The NP method which incorporates other algorithms into its sampling part is called hybrid NP method. Genetic algorithms, tabu search, greedy search and act colony optimization are good examples of possible algorithms which can be incorporated into the NP framework. Those hybrid NP methods are better than either just a pure NP method or the other pure heuristic on their own.

The performance of the NP method depends on how well it makes correct move. That is, when the performance of sub-regions from the current most promising region is better than that from the complimentary region, we partition further and go ahead the procedure. If not, we go back to the previous depth and partition again from this previous depth. Thus, generated solutions have to be good enough to represent its region. Hybrid NP methods

improve their performance by increasing the probability of generating high-quality feasible solutions from each region.

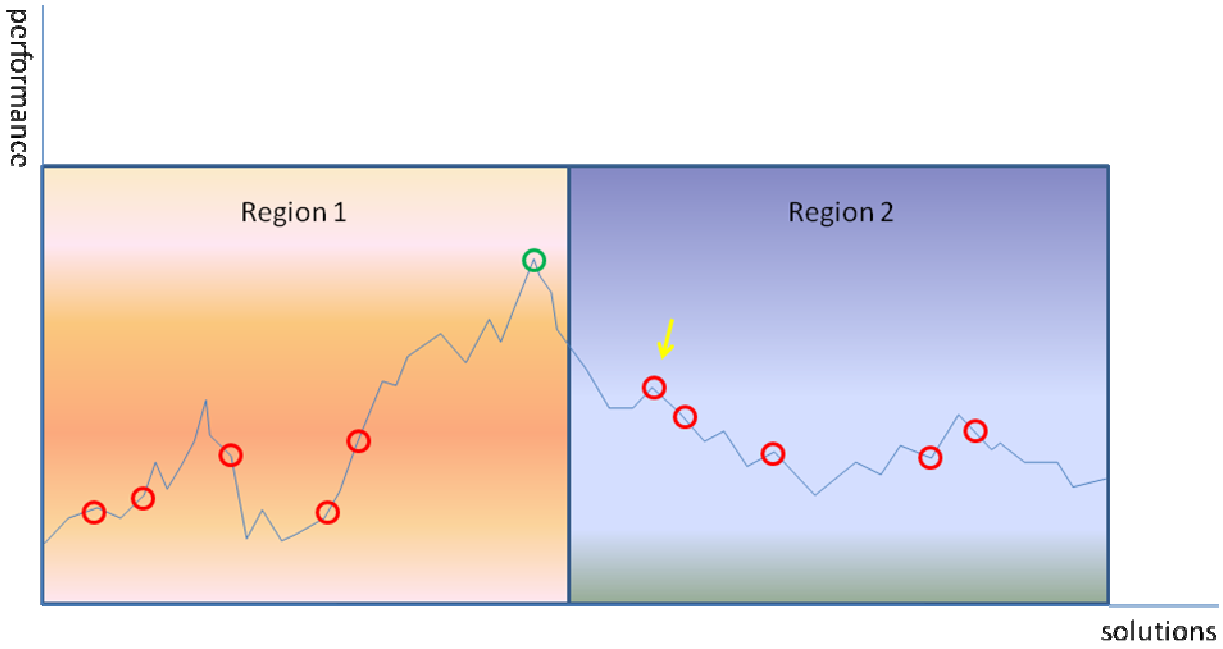

Figure 4. Illustration of the NP method

The following is an intuitive explanation of why hybrid NP methods are more effective than either the pure NP or the pure heuristics on their own.

First, in the pure NP, suppose we partition the feasible region into two sub-regions as the figure above, and from each region, the pure NP method generates 5 random sample solutions. The red circles are representing random feasible solutions. According to the figure, region 1 has to be selected as the most promising region because it has the optimal solution which is highlighted by green circle. However, comparing all the randomly generated solutions, region 2 has the highest quality feasible solution which is highlighted by yellow arrow.

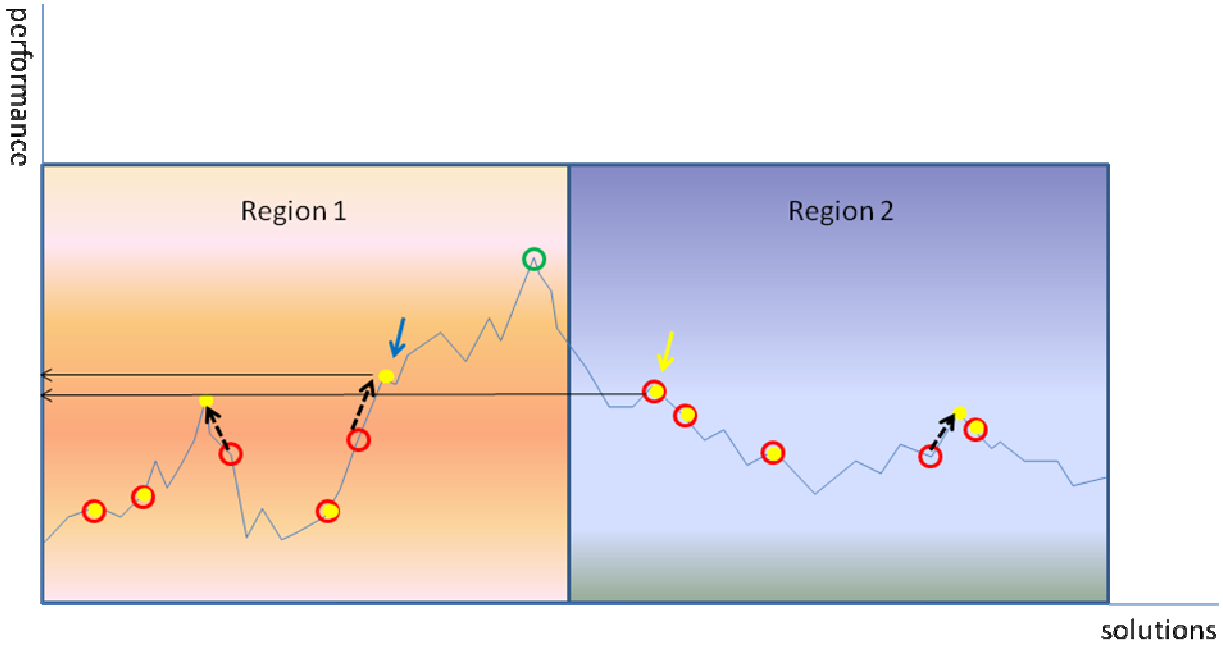

Figure 5. Illustration of hybrid NP method with greedy search (1)

Now, suppose we incorporate greedy search into the pure NP. The concept of greedy search is that we search neighborhood solutions from current solutions by giving small modifications until it turns out to be not better than original solution any more, not just generating random solutions. As a result of incorporating greedy search into the pure NP method, some of the randomly generated feasible solutions are improved. The yellow circles are modified solutions by greedy search. The best performance value from region 1 which is highlighted by blue arrow is better than that from the region 2 which is highlighted by yellow arrow. As shown from the figure above, by incorporating greedy search into the pure NP, we can increase the probability of making correct move. This explains that hybrid NPs better than pure NPs.

We mentioned that the hybrid NP is better than either the pure NP or the pure algorithms which were incorporated into the pure NP on their own. Now we will show why the hybrid NP is also better than the pure heuristics which is incorporated into the hybrid NP. Following example also uses greedy search and the hybrid NP method which incorporates greedy search into its sampling part. Suppose we generate 5 random feasible solutions from feasible region in the greedy search. Then, the randomly generated solutions are improved by the procedure of greedy search.

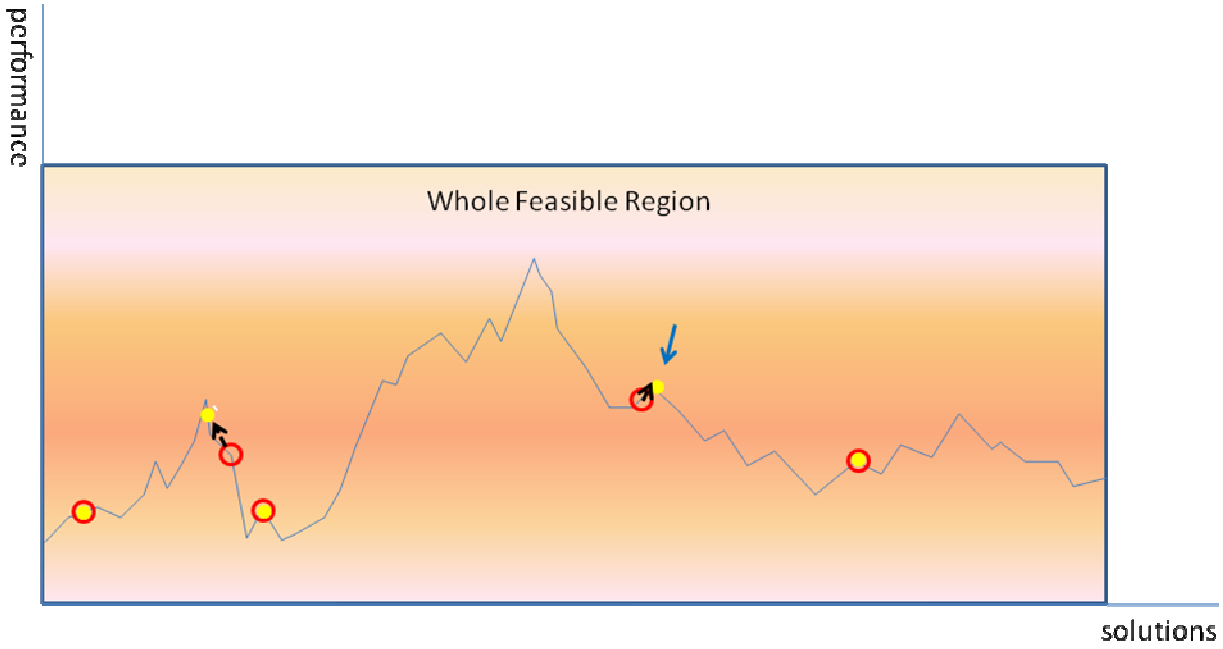

Figure 6. Illustration of Greedy search

The red circles are random feasible solutions and the yellow points are improved solutions from original feasible solutions by greedy search. Then, the greedy search regards the yellow point which is highlighted by blue arrow as the best solution because its performance is the

best among all imroved solutions. If we generate more random feasible solutions, we may obtain better solutions. However, since the greedy search is based on random search, we can't expect as good results as the algorithm spends time.

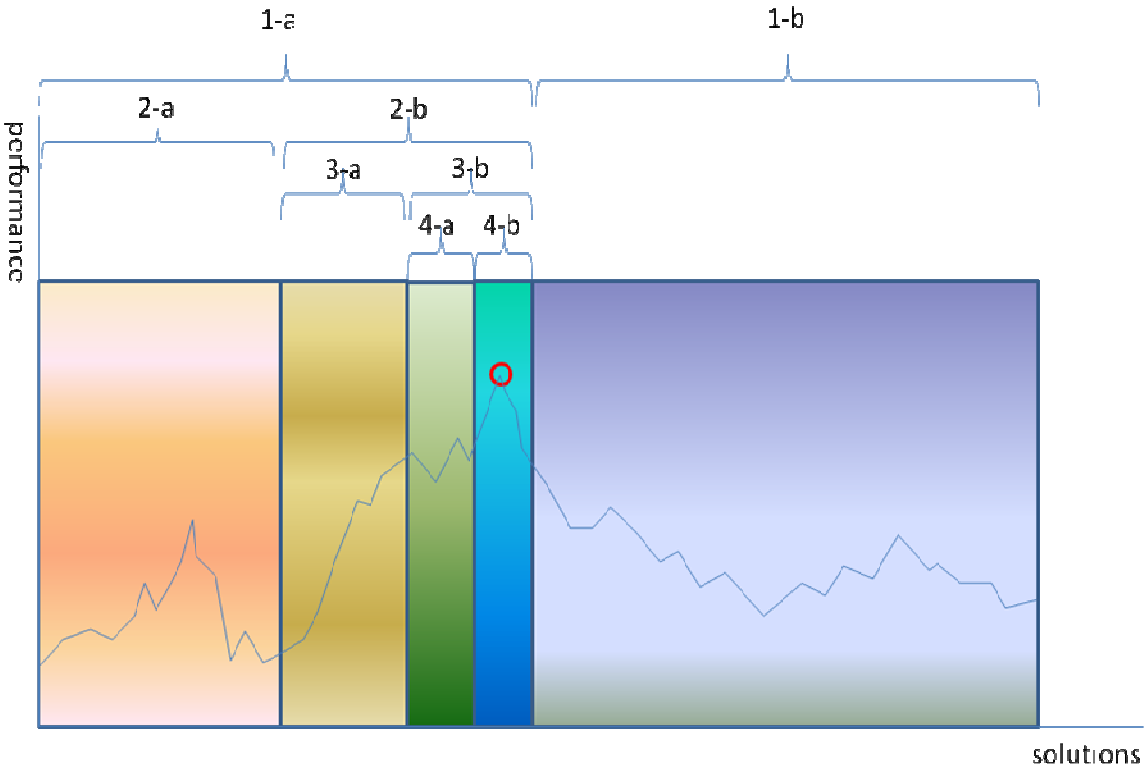

Figure 7. Illustration of hybrid NP method with greedy search (2)

The hybrid NP with greedy search is more efficient and effective algortihm than just pure greedy search. In the hybrid NP framework, the whole feasible region is divided into several sub-regions, and the most promising region which is expeted to have good solution is selected among them. Since the algorithm incorporates the greedy search, the pobability of correct move is increased. By doing this procedure iteratively, we finally obtain very good solution. The figure above shows the performance of the hybrid NP with greedy search intuitively.

4.2. Other algorithms used in this paper

4.2.1. Uniform search

Uniform search is one of the earliest algorithms, which is also called random search. This is very simple algorithm and not efficient algorithm. We used this algorithm in my study to compare the performance with Pure NP. The comparison explains the effectiveness of NP algorithm.

In this algorithm, when we try to find the optimal solution, we generate random feasible solutions from feasible region and choose the best solution among all random solutions generated. If the problem scale is large, it is difficult to find the optimal solutions by using this algorithm, because we generate solutions randomly without considering any factors which might be important to make a decision. The framework of the uniform search is as follows.

```
<Procedure : Uniform search>
Begin
Generate random feasible solutions π;
Compare all random feasible solutions and find the best solution among them;
End;
```

Figure 8. Pseudocode for the Uniform search

4.2.2. Greedy search

Greedy algorithm used in this paper is also simple algorithm, but this is more efficient algorithm than random search. In this algorithm, like uniform search, we generate some feasible solutions randomly. Then, we search better solutions from already generated random feasible solutions. The performance of greedy search is better than that of uniform search.

```
<Procedure : Greedy search>
Begin
Generate random feasible solutions π;
Improve all generated solutions by searching neighborhood solutions of π;
Compare all improved solutions  and find the best solution among them;
End;
```

Figure 9. Pseudocode for the greedy search

4.2.3. NP with greedy search

NP with greedy search is a hybrid NP method which applies greedy search in its sampling part. This algorithm shows better performance than the pure NP method. We will show the result of comparison in the next chapter. The result shows that hybrid NP method is better than just a pure NP method. The procedure of this algorithm is following

```
< Procedure : NP with Greedy search>
Begin
Depth← 0;
While (depth doesn't reach the bottom) do
        Partition feasible region into sub-regions and a complimentary region;
        Generate random sample solutions from each region;
```

```
        Apply greedy search to the sample solutions;
        If  the performance of promising region then
                Accept new solution;
                Promising region←current solution region;
                Depth←Depth+1;
        Else;
                Backtrack;
                Depth←Depth-1;
        End;
        Update process history;
        End;
End;
```

Figure 10. Pseudocode for the NP with greedy

4.2.4. Intelligent greedy search.

The intelligent greedy search is an improved version of a pure greedy search. We devised this algorithm by concerning the important property of the WTA problem in this paper. The difference between intelligent greedy search and a pure greedy search is as follows. When generating feasible solutions using greedy search, we randomly generate them as random search does. However, in the intelligent greedy search, we do not generate feasible solutions randomly. We consider the factors which affect the objective function value such as killing probability, destroying probability, and weapon and target values. Then, with this information, we generate intuitively good solutions which guarantee relatively good performance values. For example, the purpose of the asset based SWTA problem is to find the optimal assignments of weapons to targets which minimize the total expected damages.

The objective of this WTA problem in this paper is represented as following mathematical expression.

$$\min \sum_{k=1}^{m} w_k \sum_{j=1}^{n} \left[p_{kj} \prod_{i=1}^{m} (1 - \gamma_{ij} x_{ij}) \right]$$

(1) Threat of target (2) Target survival probability

After examining this objective function, we can recognize that an assignment which assigns a weapon with highest killing probability to a target with highest destroying probability first may yield good results. That is, the policy that eliminates the most threatened target first will minimize the expected damages. This is the concept of generating intuitively good solutions.

The intelligent greedy search process is as follows.

<Procedure : intelligent greedy search>
Begin
Generate one intuitively good solution π;
Generate n neighborhood solutions from π;
Improve all generated solutions by applying greedy search;
Compare all improved solutions and find the best solution;
End;

Figure 11. Pseudocode for the intelligent greedy search

First, we generate one intuitively good solution. Then, what is the intuitively good solution? As mentioned previously, the purpose of this WTA problem is to find the optimal solution which minimizes the expected damage. To achieve this goal, a concept of eliminating a target

which is the most destructive first can be an intuitively good solution. Like this concept, we can obtain the intuitively good solution by assigning a target with highest threat to a most effective weapon sequentially. The threat is represented by the destroy probability and the effectiveness of a weapon to a designated target is represented by the kill probability in the problem.

Suppose that we have an example which has 4 weapons and 4 targets. A randomized data set is for this example problem is as follows.

Table 1. An example of randomized data set (destroy probability)

Destroy probability	Target 1	Target 2	Target 3	Target 4
Weapon 1	0.79	0.70	0.79	0.63
Weapon 2	0.79	0.61	0.78	0.79
Weapon 3	0.68	0.63	0.75	0.61
Weapon 4	0.96	0.86	0.81	0.61

To find the intuitively good solution, we first rearrange the order of the targets by threat. The threat is represented with the probability of destroying which shows how efficient a target is to a certain weapon. The most threatened target is a target with the highest probability of destroying. In the table of destroy probability, target 1 is the most threatened target and the least threatened target is target 4. After rearranging the data set by the order of the targets' threat, we obtain the following modified data set.

Table 2. An example of randomized data set (destroy probability) - rearranged

Destroy probability	Target 1	Target 3	Target 2	Target 4
Weapon 1	0.79	0.79	0.70	0.63
Weapon 2	0.79	0.78	0.61	0.79
Weapon 3	0.68	0.75	0.63	0.61
Weapon 4	0.96	0.81	0.86	0.61

Now we decide the order of target assigning: Target 1 → 3 → 2 → 4. Then, we assign weapons to those targets concerning the efficiency of weapons, to obtain the intuitively good solution. The table of the kill probability provides the information about efficient allocations of weapons to targets.

Table 3. An example of randomized data set (kill probability)

Kill probability	Target 1	Target 3	Target 2	Target 4
Weapon 1	0.75	0.99	0.99	0.94
Weapon 2	0.94	0.90	0.83	0.95
Weapon 3	0.92	0.90	0.90	0.91
Weapon 4	0.53	0.71	0.58	0.99

For target 1, weapon 2 is the most efficient weapon because the probability of killing target 1 with weapon 2 is the highest. Thus, we assign weapon 2 to target 1. Once a weapon is assigned, it is eliminated from the available weapon set. That is, weapon 2 is not available

any more in this example because it is assigned to target 1. With this concept of assigning weapons to targets, we can obtain the most appropriate allocations as follows.

Table 4. The most effective weapons to each target

Kill probability	Target 1	Target 3	Target 2	Target 4
Weapon 1	0.75	0.99	0.99	0.94
Weapon 2	0.94	0.90	0.83	0.95
Weapon 3	0.92	0.90	0.90	0.91
Weapon 4	0.53	0.71	0.58	0.99

We assign weapon 2 to target 1, weapon 1 to target 3, weapon 3 to target 2, and finally weapon 4 to target 4. This is the solution that we want to find, named as the intuitively good solution. The solution is represented as following two-dimension matrix.

$$X_{ij} = \begin{Bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{Bmatrix}$$

Figure 12. An example of an intuitively good solution

Once we find the intuitively good solution, then we generate n neighborhood solutions from the intuitively good solution. And then, we improve the all solutions by applying greedy

search. The reason why we generate *n* neighborhood solutions is to increase the probability to find better solution. The more we search solutions, the more we can find better solutions. The way of generating neighborhood solutions is to change two rows which are chosen randomly in the solution matrix. For example, if we select 2, 3 rows in the solution above to generate a neighborhood solution, we obtain a neighborhood solution as follows.

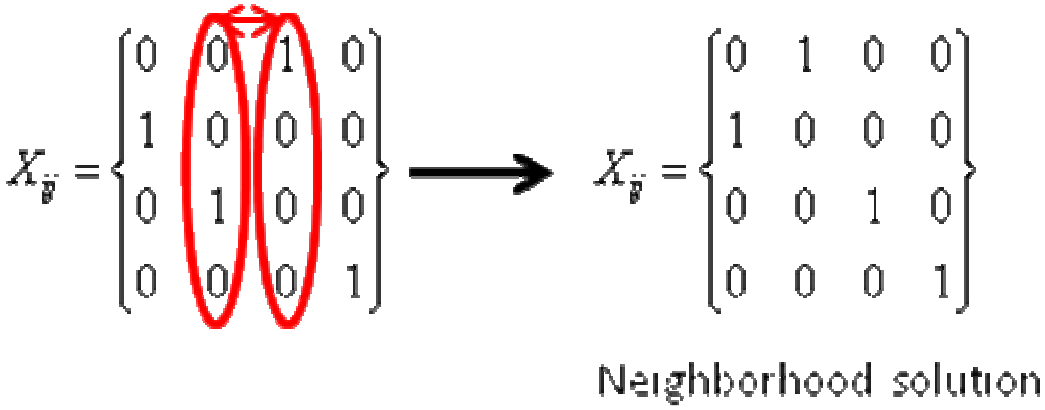

Figure 13. An example of generating a neighborhood solution

If all those procedures are done, we decide one solution which shows the best performance among all solutions by comparing all feasible solutions generated and improved. The solution found in this process may be just a local optimal solution, not the optimal solution.

The figure 14 shows well about how the intelligent greedy search works. The performance of the intelligent greedy search is the best among the algorithms

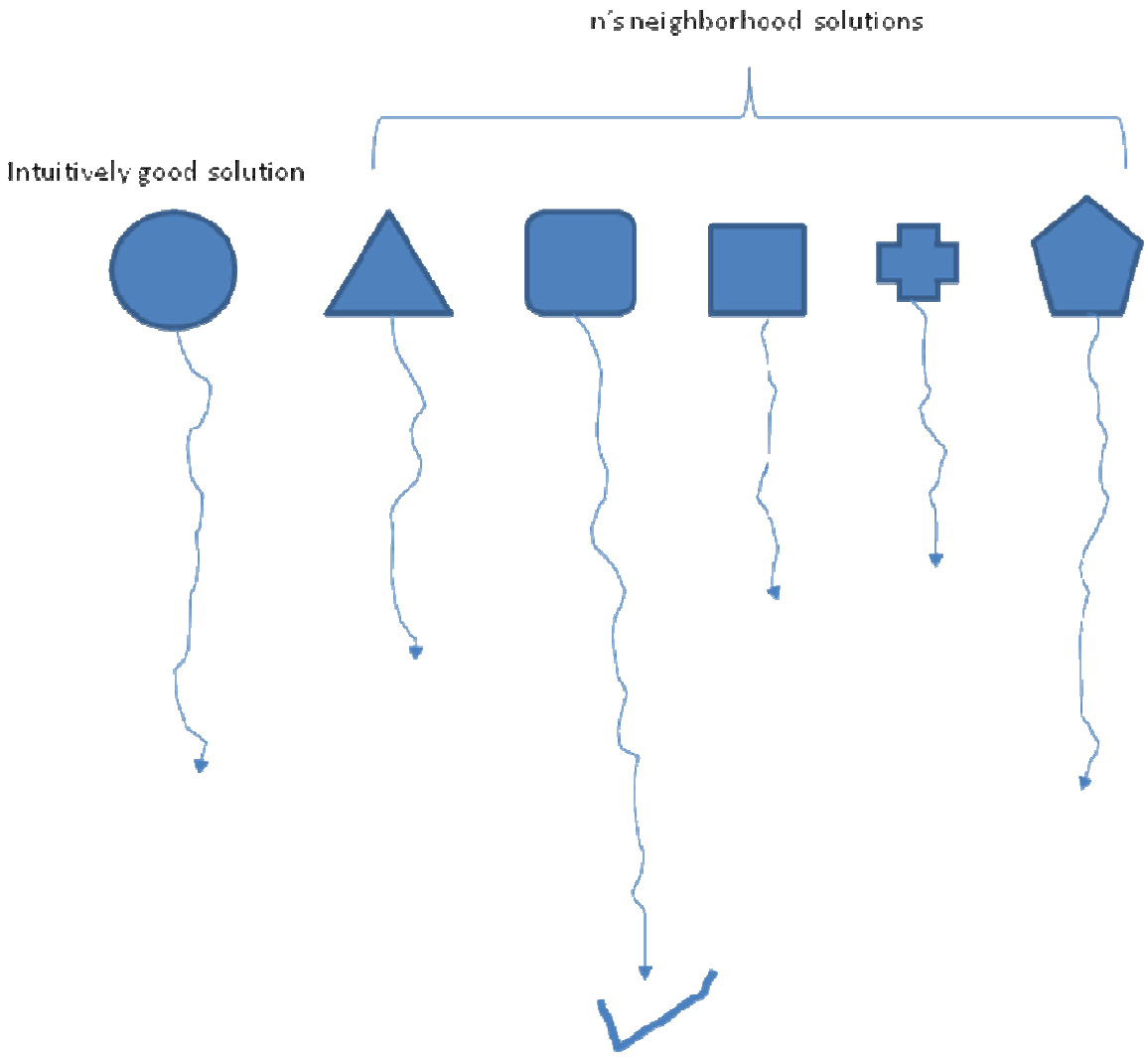

Figure 14. Illustration of intelligent greedy search

4.3. Proposed algorithm – Hybrid NP method (NP with intelligent greedy search)

The proposed algorithm in this paper is called NP with intelligent greedy search. As we know from the name of this algorithm, this is a hybrid NP method. This algorithm takes the advantages of NP method and intelligent greedy search simultaneously. Intelligent greedy search is the most efficient algorithm among those used in this paper as pure algorithms. We showed this in the following chapter.

The procedure of NP with intelligent greedy search is as follows:

```
<Procedure : NP with intelligent greedy search>
Begin
Rearrange the randomized data set according to the threat of targets;
Depth← 0;
While (depth doesn't reach the bottom) do
        Partition feasible region into sub-regions and a complimentary region;
        Generate intuitively good solutions π from each region;
        Generate neighborhood solutions from π;
        Improve each solution by applying greedy search and find one best
solutions from each region (promising region and complimentary region)
        If  the performance of promising region then
                Accept new solution;
                Promising region←current solution region;
                Depth←Depth+1;
        Else;
                Backtrack;
                Depth←Depth-1;
        End;
        Update process history;
        End;
End;
```

Figure 15. Pseudocode for NP with intelligent greedy search

From the beginning of this algorithm, we rearrange the randomized data set in accordance with the threat of targets. This step is related with the partitioning policy of this algorithm. Basically, in the NP method, we can choose two different partitioning policies; normal

partition and intelligent partition. The proposed algorithm in this paper takes an intelligent partitioning policy because it yields better results than just a normal partitioning policy. The superiority of an intelligent partitioning policy will be demonstrated in the next chapter.

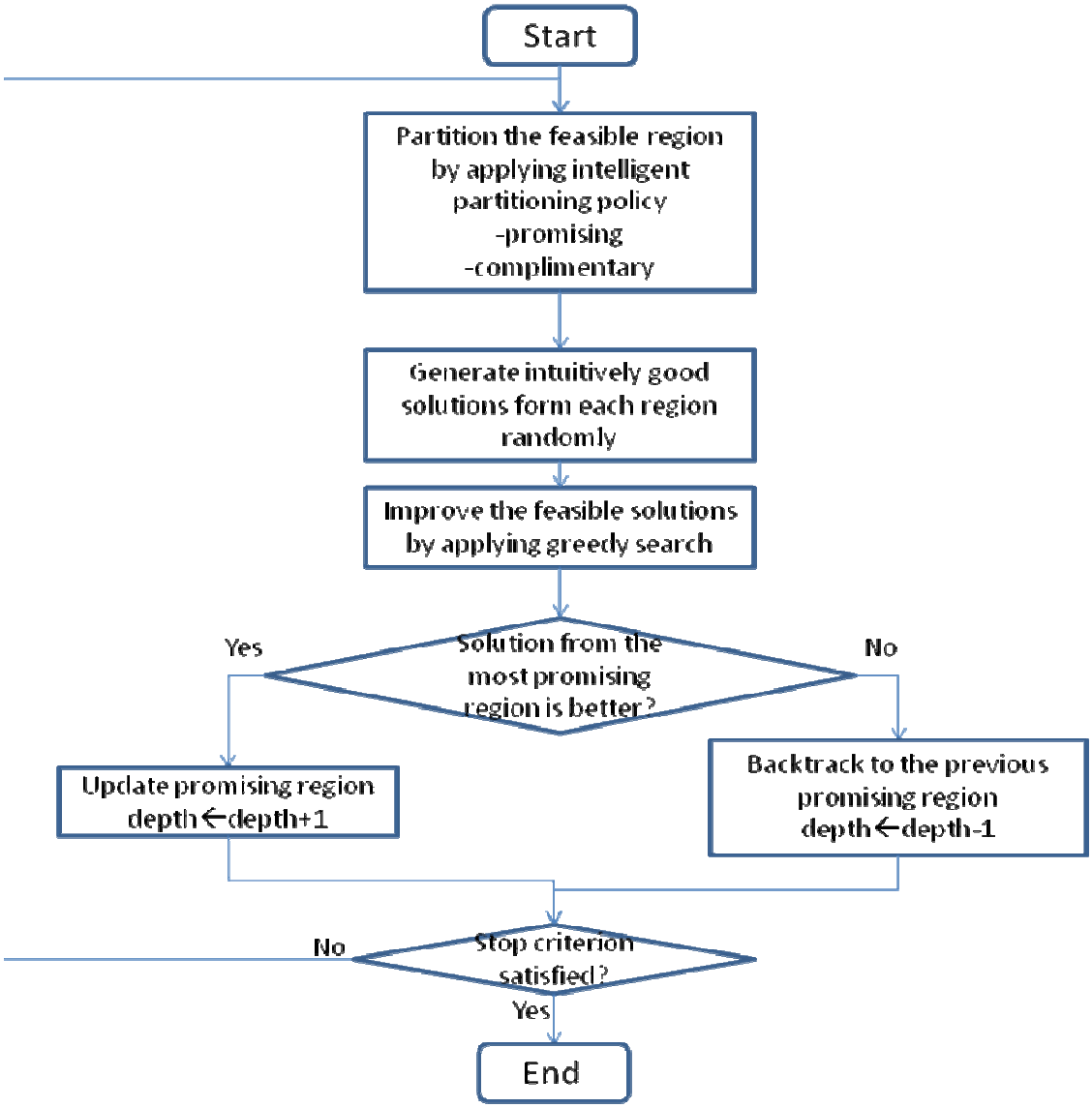

Figure 16. Flow diagram of the proposed algorithm

In the hybrid NP with intelligent greedy search algorithm, an intelligent greedy search is in charge of improving sample solutions generated from each partitioned region. By searching

neighborhood solutions, we find better solutions from original solution. This procedure plays an important role in the proposed algorithm. It helps us not to choose a bad region as a next promising region to partition further. That is, it increases the probability of making correct moves.

CHAPTER 5. NUMERICAL RESULTS

5.1. Introduction

The proposed algorithm in my study is NP with intelligent greedy search. In order to verify the proposed algorithm is a relatively efficient algorithm for solving WTA problem, we compared the proposed algorithm with several other algorithms.

We have three different scenarios in my experiment in accordance with the size of problems. That is, each scenario has different set of randomized data for weapons and targets. In the first scenario, we have 20 weapons and 20 targets which represent small size problem. Other two scenarios have 40 and 80 weapons and targets respectively, which represent medium and large scale problems.

The range of the objective function value is very large. For example, according to the algorithms used, the best performance values vary from around 10^10 to 10^(-10). So, when plotting the performance graph in a figure, sometimes it is very hard to read. In order to solve this problem, we took log values of the objective function. Actually, the objective function value, the total expected damage, is a positive value. However, the best performance value is sometimes shown as a negative value because we took log values of them. But, it doesn't mean that the actual value is negative.

Basically, the results of the experiments are averaged by 10 times of simulations because the random characteristics in simulations should not be included in the results. In addition to that, we established 95% of confidence interval to show that the significant differences between two comparison algorithms.

5.2. Performance Comparison among algorithms

5.2.1. Comparison between Uniform search and Pure NP

First, we compared the performance of uniform search and Pure NP method to show that NP method is promising algorithm to solve WTA problem. In this case, we used small size problem which has 20 weapons and targets.

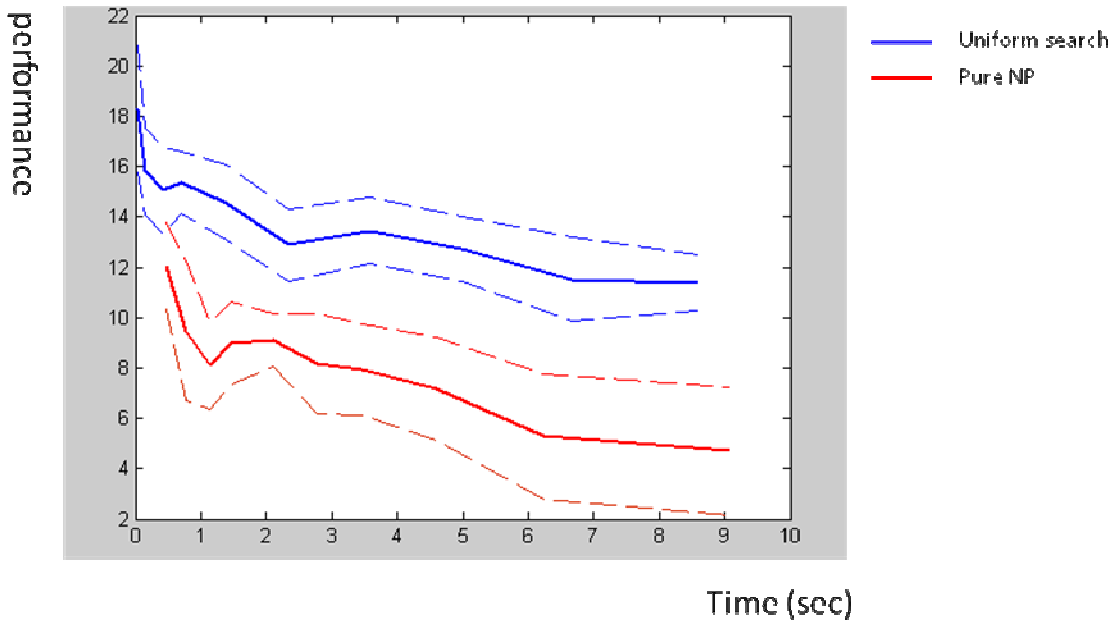

Figure 17. Uniform search VS Pure NP (N=20)

In this figure, the x-axis is the processing time and the y-axis is the objective function value at the point of time. The objective function value means the total expected damage. So, algorithms which find solutions with low objective function values are more promising algorithms. The blue line is the result of uniform search and the red line is for the pure NP. The dashed lines are respectively upper bound and lower bound of 95% of confidence interval. The red line is located below the blue line in this figure. That is, the performance of pure NP method is better than that of uniform search. Apparently, we found that the pure NP is significantly better than uniform search when solving WTA problem.

5.2.2. Comparison among Pure NP, Greedy search and NP with greedy search

In the previous section, we showed that a pure NP is better than a random search when solving WTA problem. Actually, the random search is very simple algorithm. We can't expect high performance from this random search. That is, it doesn't guarantee that any algorithms which are better than a random search are good enough to be a promising algorithm to solve WTA problem. So, at this time, we will compare the pure NP with a relatively advanced algorithm, a greedy search. As we explained in the previous chapter, a greedy search which is used in this paper is based on a uniform search, but having an improved framework. It obtains better performance by searching better solutions from neighborhood solutions. By comparing this advance algorithm with a pure NP method, we will show that even just a pure NP method is a promising algorithm to solve WTA problem.

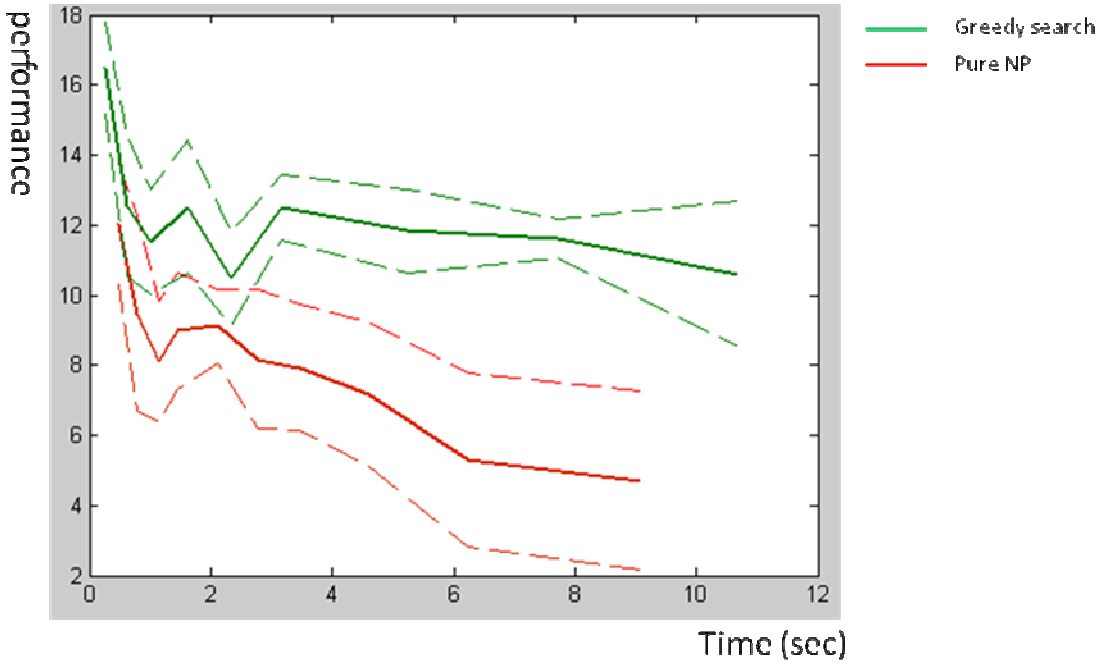

Figure 18. Greedy search VS Pure NP (N=20)

This figure shows that pure NP method is better than greedy search. When the processing time is less than 2 seconds, the 95% of confidence intervals of those two algorithms overlap one another. However, as the processing time is getting increased, the performance of the Pure NP is getting improved more than that of greedy search.

So far, we demonstrated that the Pure NP is a relatively efficient algorithm to solve WTA problem by comparing with uniform search and greedy search. From now on, we will show the superiority of hybrid NP method. In the previous chapter, we insisted that a hybrid NP method is better than either a pure NP or the pure algorithms which are incorporated into the hybrid NP method on their own. First, we will prove that hybrid NP method is better than a pure NP method.

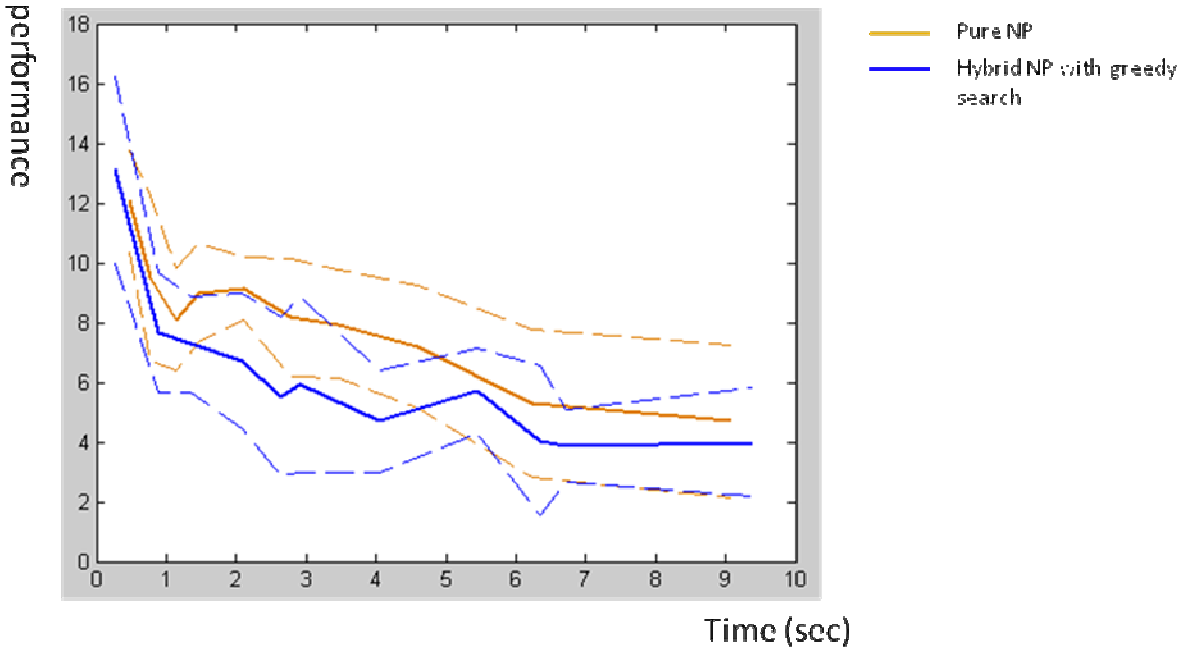

Figure 19. Pure NP VS NP with greedy search (N=20)

This figure is the result of comparing the best performance values of Pure NP and hybrid NP with greedy search. In the figure above, the performance of hybrid NP with greedy search is slightly better than that of the Pure NP. Taking a look at the average of the performance values of both algorithms which are presented as bold lines, hybrid NP is apparently having better performance values at every moment. However, their 95% of confidence intervals overlap entirely. Thus, we can't insist that hybrid NP with greedy search is significantly better than the Pure NP at this juncture.

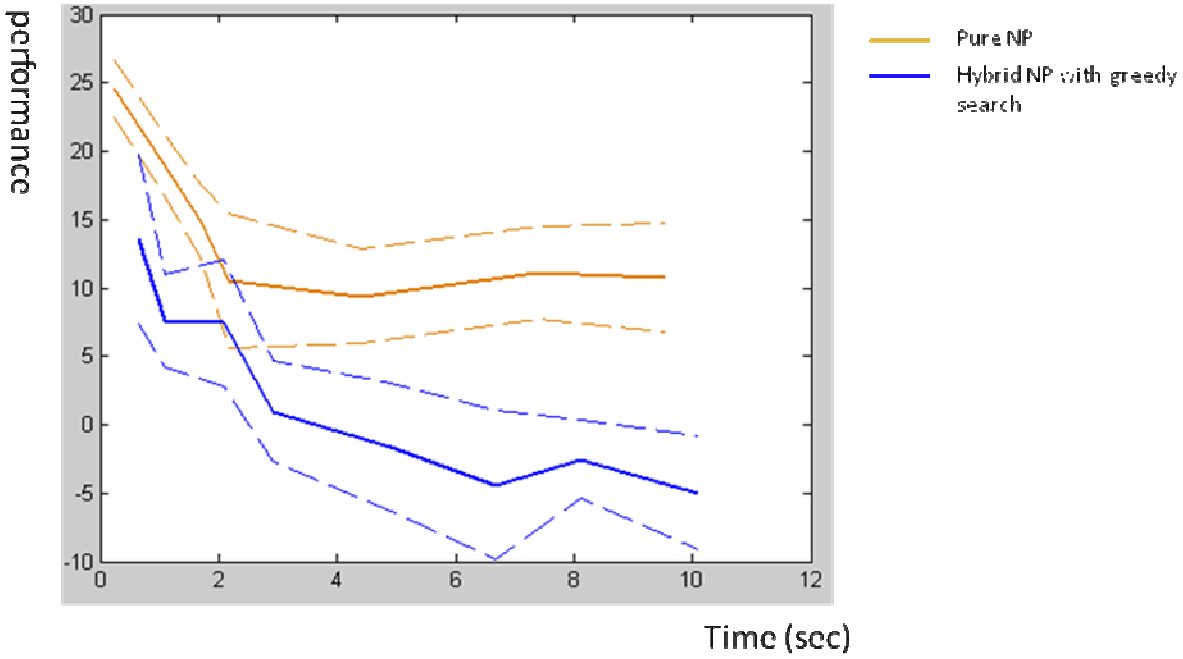

Figure 20. Pure NP VS NP with greedy search (N=40)

We changed the scale of the problem from small size to medium size. In this case, the number of weapons is 40 and the number of targets is 40. As seen in the figure above, those two algorithms show more apparent differences than when the problem size is small. The difference of the performances between the Pure NP method and hybrid NP method with greedy search tend to be getting increased according to the problem scale. With the example of medium scale problem, we showed that hybrid NP is better than pure NP.

Now we will show that hybrid NP is also better than the pure algorithms used in hybrid NP. The performance of greedy search and hybrid NP with greedy search is as the figure provided below.

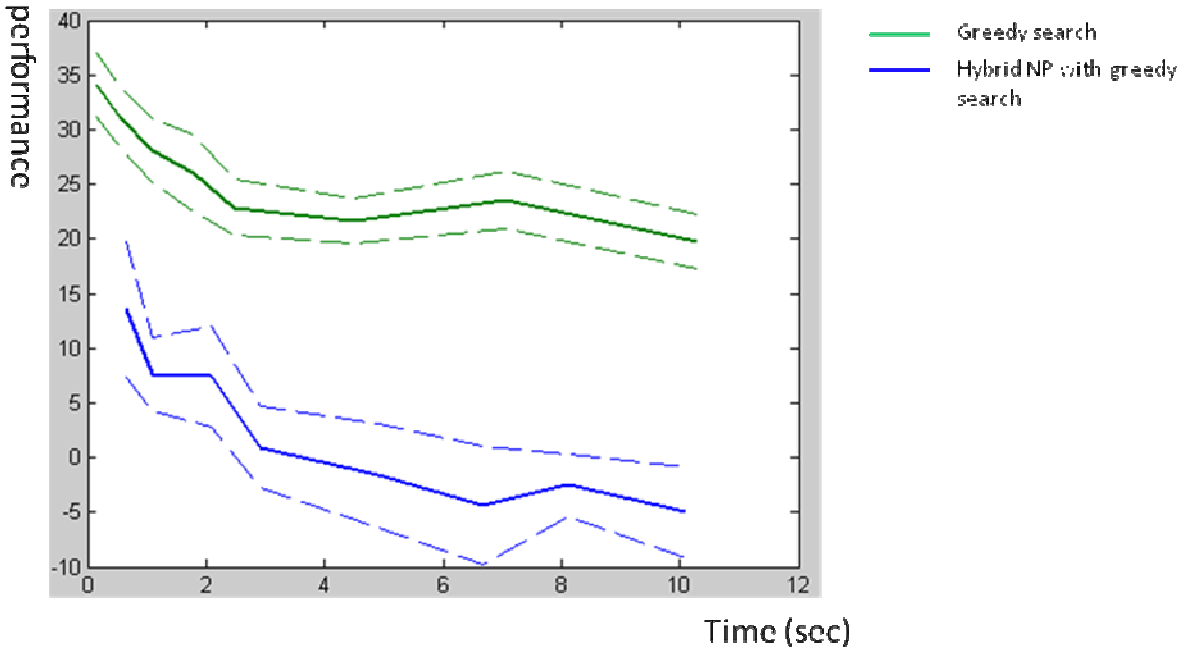

Figure 21. Greedy search VS NP with greedy search (N=40)

As we see the figure above, hybrid NP with greedy search is significantly better than greedy search.

Through the example of greedy search, we showed that hybrid NP method is better than a pure NP and the pure algorithm used in hybrid NP. As we see the performance of Hybrid NP method, we can say that the hybrid NP with greedy search is good algotirhm to solve WTA problem. However, the greedy search which is used in the sampling part of the hybrid NP method has a critical limitation. As we explained in the prevous chapter, the greedy search is based on a random search. It searches feasible solutions randomly, without considering anything related with the objective of the problem. Such fundanmental ineffectiveness of the greedy search framwork affects the performance of the algorithm. Thus, in order to overcome

such inefficiency, we slightly modified the greedy search. We named it as 'intelligent greedy search (iGreedy)'. The most important difference is that the intellignet greedy seach is concerning the characteristics of WTA problem.

5.2.3. Comparison among greedy search, intelligent greedy search and hybrid NP with intelligent greedy search

The intelligent greedy search is an improved version of greedy search. The performance of intelligent greedy search is far better than any other algorithms which are used in this paper so far.

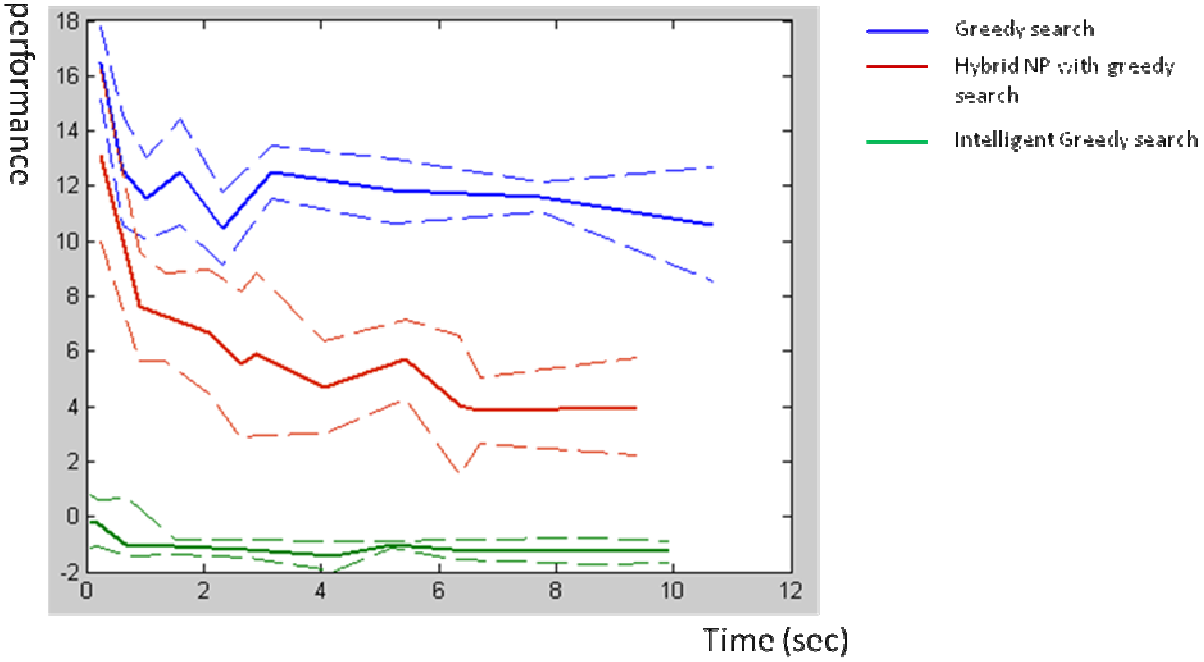

Figure 22. Greedy search, NP with greedy search and iGreedy (N=20)

This figure represents the performances of three different algorithms when the problem size is small. As seen the figure above, the intelligent greedy search shows outstanding

performances. It finds a very good sub-optimal solution within a very short time which is much better than the solutions found by other algorithms.

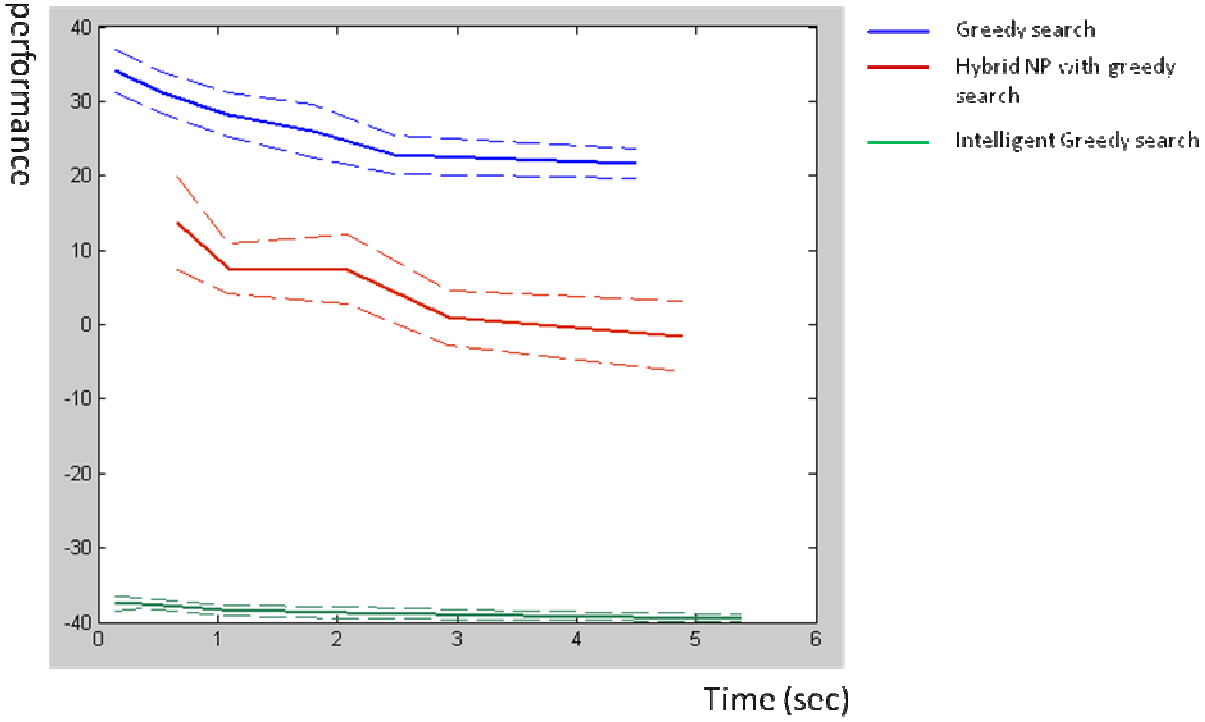

Figure 23. Greedy search, NP with greedy search and iGreedy (N=40)

The differences of the performances are getting increased as the problem size is becoming larger and larger. We found that the intelligent greedy search is more promising than any other algorithms used in this paper. In section 5.1.2, we showed that the hybrid NP method is better than the pure algorithms incorporated into the hybrid NP. Then, we can guess that the hybrid NP with intelligent greedy search is better that just the intelligent greedy search. To verify this, we simulated hybrid NP with intelligent greedy search and compared the performance with that of the intelligent greedy search.

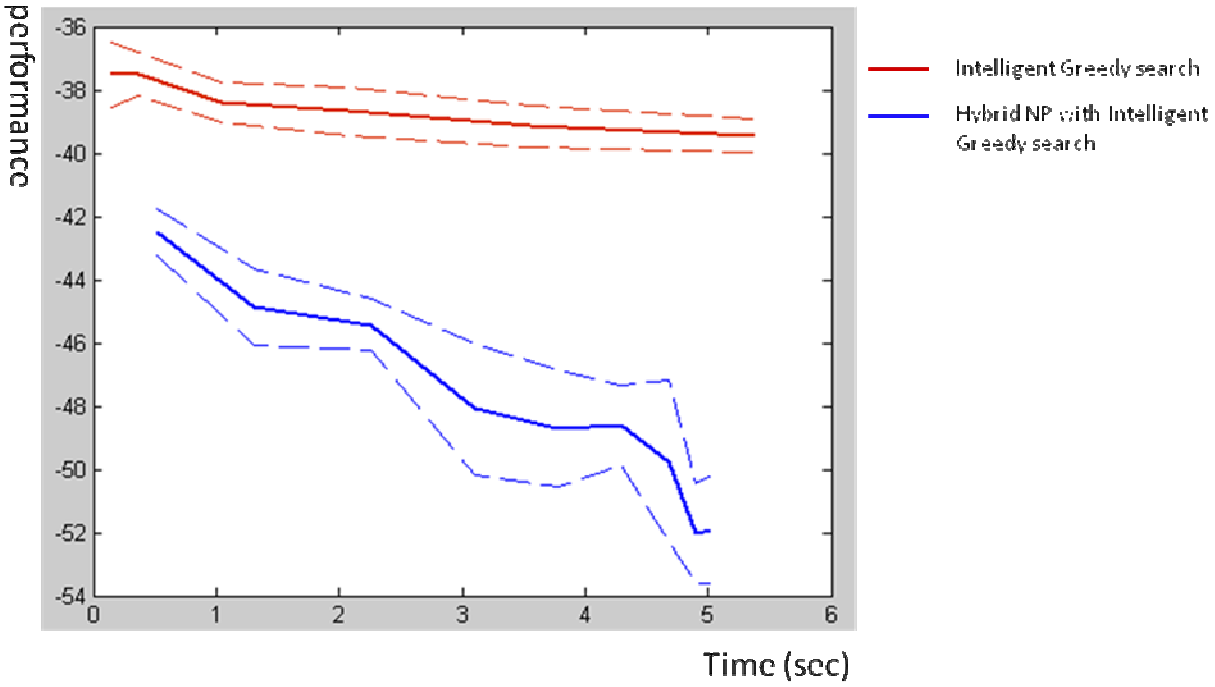

Figure 24. iGreedy VS NP with iGreedy (N= 40)

As we expected, the hybrid NP with intelligent greedy search shows better performance than just the pure intelligent greedy search.

5.3. Comparison of intelligent partitioning and normal partitioning

In the previous section, we demonstrated that NP with intelligent greedy search is the most efficient algorithm among those algorithms used as comparison objects in this paper. When we improve the performance of NP methods, we consider two stages from the NP framework: partitioning and sampling part. The proposed algorithm has been improved by only applying intelligent greedy search in its sampling part. Thus, one more thing is remained

to improve the performance of the proposed algorithm. We can take two different partitioning policies when using NP method such as normal partitioning and intelligent partitioning. Intuitively, an intelligent partitioning seems to have better performance than just a normal partitioning considering the term “intelligent” and “normal”. However, we cannot insist that an intelligent partitioning policy is better than a normal partitioning policy without any proofs. Thus, we verified this issue by comparing the performance of the two different proposed algorithms: first one used an intelligent partitioning policy and the other one uses a normal partitioning policy.

Basically, the structure of the partitioning part of NP method in this paper is as follows. The algorithm is assigning weapons to targets one by one. According to the assumption we made on the chapter 3, a target can’t be assigned by more than two weapons. In addition to that, when allocating weapons to targets, the NP method used in this paper assigns weapons to targets one by one, not assigning simultaneously all weapons to targets which should be assigned. Therefore, the order of assigning targets is very important. The partitioning policy is determined by the order of assignment. The intelligent partitioning policy is deciding the order of assignment in accordance with the importance of each target. That is, this is due to the idea that eliminating the most threatened targets first by assigning them with highly efficient weapons. On the other hand, the normal partitioning policy does not consider the threat of targets when ordering targets.

In the intelligent partitioning policy, the NP method assigns the most dangerous target first. The factor that decides which target is the most dangerous is the probability of destroying

because the destroy probability shows the effectiveness of a certain target to a certain weapon.

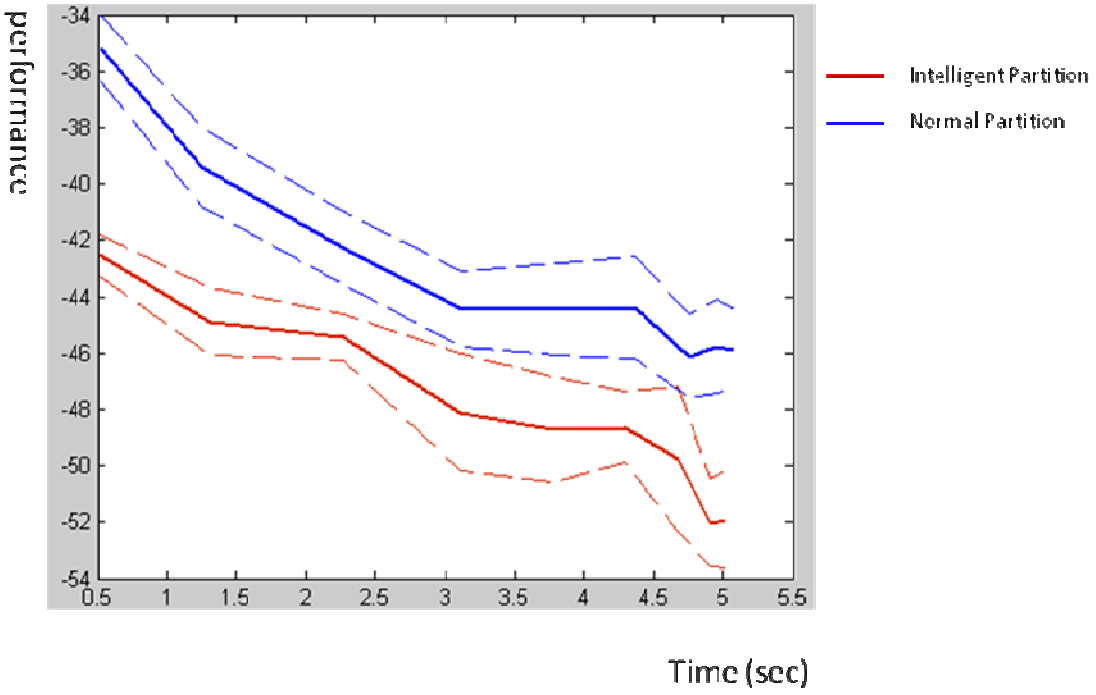

Figure 25. Performance of intelligent partitioning and normal partitioning policy

This figure is comparing the performances of two hybrid NP with intelligent greedy searches with different partitioning policies. As we see the figure above, the intelligent partitioning policy is better than the normal partitioning policy when solving this asset based static WTA problem.

Comparing the best fitness values of each algorithm in this paper, we can easily find that the proposed algorithm is the most promising algorithm to solve the WTA problem. The results of simulation in accordance with the size of the problem are as follows.

Table 5. Comparison of the best fitness values of each algorithm

Algorithm	*N=20*	*N=40*	*N=80*
Uniform	11.39	23.68	62.87
Pure NP	4.71	10.8	44.82
Greedy	10.48	19.8	62.6
NP with Greedy	3.86	-5.01	-1.74
Intelligent Greedy	-1.44	-39.6	-22.8
NP with iGreedy	-5.19	-52.02	-96.27

In this table, each value is the best result of simulations within certain processing time interval. The NP algorithms used in this paper are all employing the intelligent partitioning policy because it is found to be better than just a normal partitioning policy. As the table above, hybrid NP methods are always providing better performance than a pure NP and the pure algorithms incorporated into the hybrid NPs.

CHAPTER 6. CONCLUSION

6.1. Conclusion

In this paper, we used the asset based static WTA problem model and proposed an algorithm whose name is NP with intelligent greedy search to solve it. This algorithm is a hybrid NP method which incorporates intelligent greedy search into its sampling stage. We devised the intelligent greedy search which is an improved version of a pure greedy search. By concerning the fundamental characteristics of the WTA problem, we can improve its performance. From my simulations for the asset based static WTA problem, the proposed algorithm indeed has better performance than other existing algorithms in this paper do.

6.2. Future work

Two fundamental concerns are existed in the WTA problem. The first one is related with the problem model. The other one is about the algorithm.

There are two kinds of the WTA problem model; dynamic and static. In this paper, we only considered the static version which deals with only one time engagement. However, the battle information such as the number of targets and weapons available are likely to be changed according to time. Moreover, the probability of killing targets can also be changed because the distance which affects the kill probability may become shorter or longer by time. Thus, considering more realistic situation of engagement, studies on the dynamic version of WTA problem is needed.

Due to the characteristics of NP method, the capability of improvement of NP method is unlimited. Almost every algorithm can be incorporated into the NP framework to improve its performance. We used the intelligent greedy search to improve the performance of NP method in this paper. If we can incorporate other algorithms which are better than the intelligent greedy search used in this paper, we will find much more efficient algorithms to solve the WTA problem.

APPENDIX. SCENARIO RESULTS

Table 6. Average objective function values of 10 simulations (Uniform search)

Problem size	Time	0.040625	0.15	0.420313	0.7	1.404688	2.34375	3.579688	4.978125	6.665625	8.60625
N=20	Value	18.27311	15.81543	15.06074	15.37478	14.55623	12.87752	13.45224	12.74206	11.50878	11.39123
	Time	0.107813	0.43125	1.025	1.78125	2.853125	4.223438	5.767188	7.6125	9.745313	
N=40	Value	34.82183	32.04955	26.83235	26.79142	25.41125	25.64143	24.76916	24.26427	23.6768	
	Time	0.159375	0.451563	1.057813	2.998438	5.629688	9.134375	13.33438	18.47969	111.2109	171.5594
N=80	Value	84.29533	80.85906	76.9463	73.69619	69.90109	68.63404	67.81339	69.78151	62.8664	65.71188

Table 7. Average objective function values of 10 simulations (Pure NP method)

Problem size	Time	0.470313	0.78125	1.14375	1.454688	2.104688	2.76875	3.465625	4.592188	6.239063	9.09375	
N=20	Value	12.05581	9.478524	8.098668	8.988422	9.111718	8.166249	7.933887	7.167965	5.293208	4.709547	
	Time	0.223438	1.026563	1.710938	2.176563	4.414063	7.404688	9.55				
N=40	Value	24.54966	19.29373	14.8479	10.53307	9.385073	11.10011	10.80435				
	Time	2.475	3.679688	4.896875	6.301563	7.18125	8.5375	9.365625	13.76875	17.14375	47.0375	157.8047
N=80	Value	66.67391	65.24661	63.46027	61.98724	62.1277	59.66489	61.2247	59.47673	58.89367	49.94649	44.81934

Table 8. Average objective function values of 10 simulations (Greedy search)

Problem size	Time	0.254688	0.609375	1.021875	1.607813	2.335938	3.15625	5.234375	7.739063	10.67813		
N=20	Value	16.46747	12.56276	11.52797	12.5199	10.48097	12.52458	11.83753	11.60707	10.59823		
	Time	0.142188	0.529688	1.0875	1.790625	2.48125	4.5	7.057813	10.29219			
N=40	Value	34.06089	31.13731	28.14926	26.00027	22.84118	21.61313	23.58789	19.80402			
	Time	0.182813	0.767188	1.704688	2.975	4.026563	4.607813	6.101563	8.890625	15.29844	23.71563	186.525
N=80	Value	86.96524	81.47672	72.94277	70.91899	73.29106	72.05476	71.97997	70.71312	67.78021	65.42437	62.5985

Table 9. Average objective function values of 10 simulations (NP with Greedy)

Problem size	Time	0.257813	0.895313	1.346875	2.0875	2.634375	2.910938	4.054688	5.44375	6.353125	6.707813	9.4
N=20	Value	13.11256	7.657713	7.270917	6.706807	5.519322	5.913406	4.696088	5.716532	4.037296	3.867265	3.999819
	Time	0.6625	1.092188	2.08125	2.926563	4.895313	6.665625	8.120313	10.10313			
N=40	Value	13.59938	7.605716	7.442971	0.944977	-1.63685	-4.39274	-2.52718	-5.00812			
	Time	0.178125	0.753125	1.226563	2.75	6.754688	14.99688	16.41094	19.02188	23.37344	159.7344	
N=80	Value	76.2413	65.39589	57.22133	45.76934	29.31479	21.12772	26.32166	22.16452	20.24377	-1.73555	

Table 10. Average objective function values of 10 simulations (iGreedy)

Problem size	Time	0.071875	0.2	0.7	1.5	2.76875	4.229688	5.220313	6.235938	8.714063	9.945313
N=20	Value	-0.17578	-0.20973	-1.06844	-1.09516	-1.17289	-1.43726	-0.99154	-1.17244	-1.25805	-1.26008
	Time	0.15	0.370313	1.0375	2.085938	3.554688	5.3875	7.610938	10.22344		
N=40	Value	-37.5227	-37.4987	-38.3848	-38.6778	-39.1424	-39.4625	-39.5907	-39.4708		
	Time	0.189063	0.959375	2.301563	4.023438	6.092188	11.41563	18.26875	26.65156	192.0234	
N=80	Value	-17.4025	-18.2043	-19.6074	-19.07	-19.4775	-20.9296	-21.5219	-20.6008	-22.7886	

Table 11. Average objective function values of 10 simulations (NP with iGreedy)

Problem size	Time	0.046875	0.11875	0.254688	2.420313	5.264063	7.320313	8.325	9.25625
N=20	Value	-1.4823	-1.84551	-2.21148	-2.57182	-3.43394	-3.67817	-3.63841	-3.84374
	Time	0.515625	1.698438	3.254688	4.707813	6.0375	7.25	8.539063	9.910938
N=40	Value	-40.6076	-40.1418	-41.2259	-42.8952	-43.2095	-43.7451	-43.761	-44.424
	Time	8.128125	11.95313	16.23906	20.15625	38.16875	55.35	68.38125	146.6219
N=80	Value	-28.3045	-34.7196	-38.0822	-44.1713	-68.423	-79.7957	-84.121	-96.2679

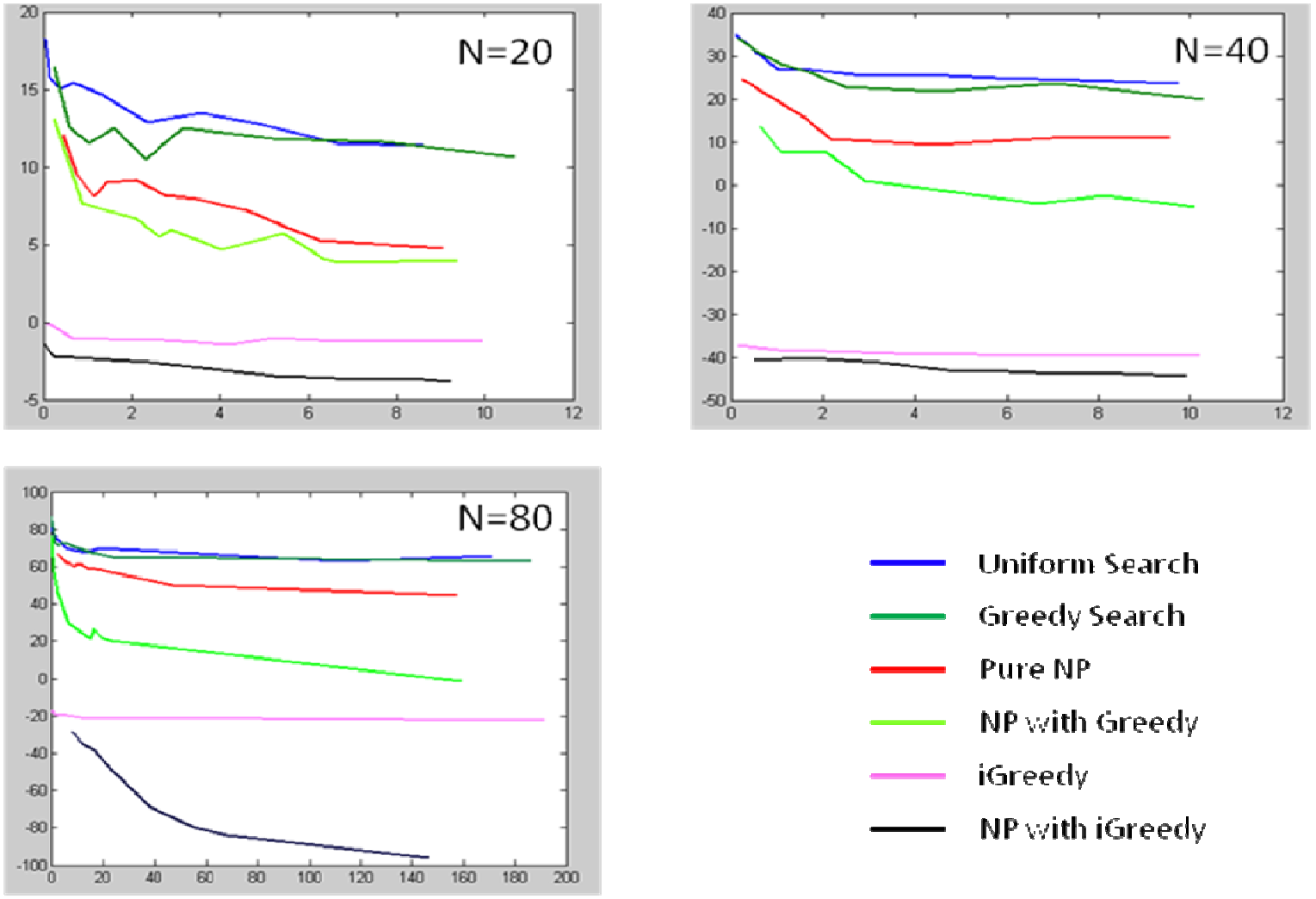

Figure 26. Comparisons of performance of each algorithm

BILIOGRAPHY

S.P. Lloyd, H.S. Witsenhausen (1986), "Weapon allocation is NP-complete", *in proceedings of the IEEE Summer Simulation Conference*

Kevin Glazebrook, Alan Washburn (2004), "Shoot-Look-Shoot : A Review and Extension", *Informs*, Vol. 52, No. 3, pp 454-463

Xiangping Zeng, Yunlong Zhu, Lin Nan, Kunyuan Hu and Xiaozian He (2006), "Solving Weapon-Target Assignment Problem Using Discrete Particle Swarm Optimization", *in proceedings of the 6th World Congress on Intelligent Control and Automation*

Gao Shang, Zhang Zaiyue, Zhang Xiaoru nad Cao Cungen (2007), "Immune Genetic Algorithm for Weapon-Target Assignment Problem", *Workshop on Intelligent Information Technology Application*

Z.J. Lee, S. F. Su and C.Y. Lee (2002), "A genetic algorithm with domain knowledge for weapon-target assignment problems", *Journal of Chinese Institute of Engineers, Transactions of the Chinese Institute of Engineers*, Series A, Vol. 25, No. 3, pp.287-295

Z.J. Lee, S. F. Su and C.Y. Lee (2003), "Efficiently solving general weapon-target assignment problem by genetic algorithms with greedy eugenics", *IEEE Transactions on Systems, Man, Cybernetics, Part B: Cybernetics*, Vol. 33, No. 1, pp.113-121

Z.J. Lee, S. F. Su and C.Y. Lee (2002), "An immunity-based ant colony optimization algorithm for solving weapon-target assignment problem", *Applied Soft Computing Journal*, Vol. 2, No. 1, pp.39-47

H.P. Cai, J.X. Liu, Y.W. Chen, and H. Wang (2006), "Survey of the research on dynamic weapon-target assignment problem", *Journal of Systems Engineering and Electronics*, Vol. 17, No. 3, pp.559-565

A.S. Manne (1956), "A target-assignment problem", *Operations Research* 6, pp.346-351

E. Wacholder (1989), "A neural network-based optimization algorithm for the static weapon-target assignment problem", *ORSA Journal on Computing* 4, pp.232-246

R,H. Day (1966), "Allocating weapons to target complexes by means of nonlinear programming", *Operations Research* 14, pp.992-1013

J.C. Braford (1961), "Determination of optimal assignment of a weapon system to several targets", *AEREITM-9*, Vought Aeronautics

DenBroader G. G., Jr., R.E. Ellison, and L. Emerling (1958), "On optimum target assignments", *Operations Research* 7, pp.322-326

J.D. Katter (1986), "A solution of the multi-weapon, multi-target assignment problem", *Working paper 26957*

Chang S.C., R.M. James, and J.J Shaw (1987), "Assignment algorithm for kinetic energy weapons in boost defense", *in proceedings of the IEEE 26th Conference on Decision and Control*

D. Orlin (1987), "Optimal weapons allocation against layered defenses", *Naval Research Logistics* 24, pp.605-616

Castanon D.A. (1987), "Advanced weapon-target assignment algorithm quarterly report", *TR-337, ALPHA TECH Inc.*

K.E. Grant (1993), "Optimal resource allocation using genetic algorithms", *Naval Review*, pp.174-175

Ikeda Yoshitaka, Nose Kazuo, and Hiramatsu Ayako (2005), "Optimization for Production Scheduling with Inequality Constraints using ACO", *in proceedings of the Annual Conference of the Institute of Systems, Control and Information Engineers*, Vol**.** 49, pp.67-68

K.L. Mak, P. Peng, X.X. Wang, and T.L. Lau (2007), "An ant colony optimization algorithm for scheduling virtual cellular manufacturing systems", *International Journal of Computer Integrated Manufacturing*, Vol. 20, Issue 6, pp.524-537

B. Yu, Z.Z. Yang, B. Yao (2009), "An improved ant colony optimization for vehicle routing problem", *European Journal of Operations Research*, Vol. 196, Issue 1, pp.171-176

L. s and S. Ólafsson (2000), Nested partition methods for global optimization, *Operations Research* 48 (3), pp. 390–407

Al-Shihabi S (2004), "Ants for sampling in the Nested Partition Algorithm." *Hybrid metaheuristic*, pp. 11–8

Al-Shihabi S, S. Ólafsson (2009), "A hybrid of Nested Partition, Binary Ant System, and Linear Programming for the multidimensional knapsack problem", *Computer & Operations Research*

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my thanks to those who helped me with various aspects of conducting research and the writing of this thesis. First and foremost, I thank Dr. Olafsson for his guidance and support throughout this research and the writing of this thesis. Especially, I used the NP method which is devised by him to conduct my research. If it were not for, I couldn't write my thesis well. I also thank Dr. Wang and Dr. Ruben who are my committee members, because they gave me good ideas on my research.

I also thank my wife who helped me to concentrate all my effort on my study and my lovely son, Jacob.