CHAPTER 3

REVIEWS ON COMBINATORIAL OPTIMIZATION PROBLEMS

In this chapter, the existing combinatorial optimization problems related to the proposed problems (WSP-N, WSP-E, and 2WSP) in Figure 3.1 are reviewed in details. The mathematical models and existing algorithms for each problem are described. The relationships among the problems are also discussed.

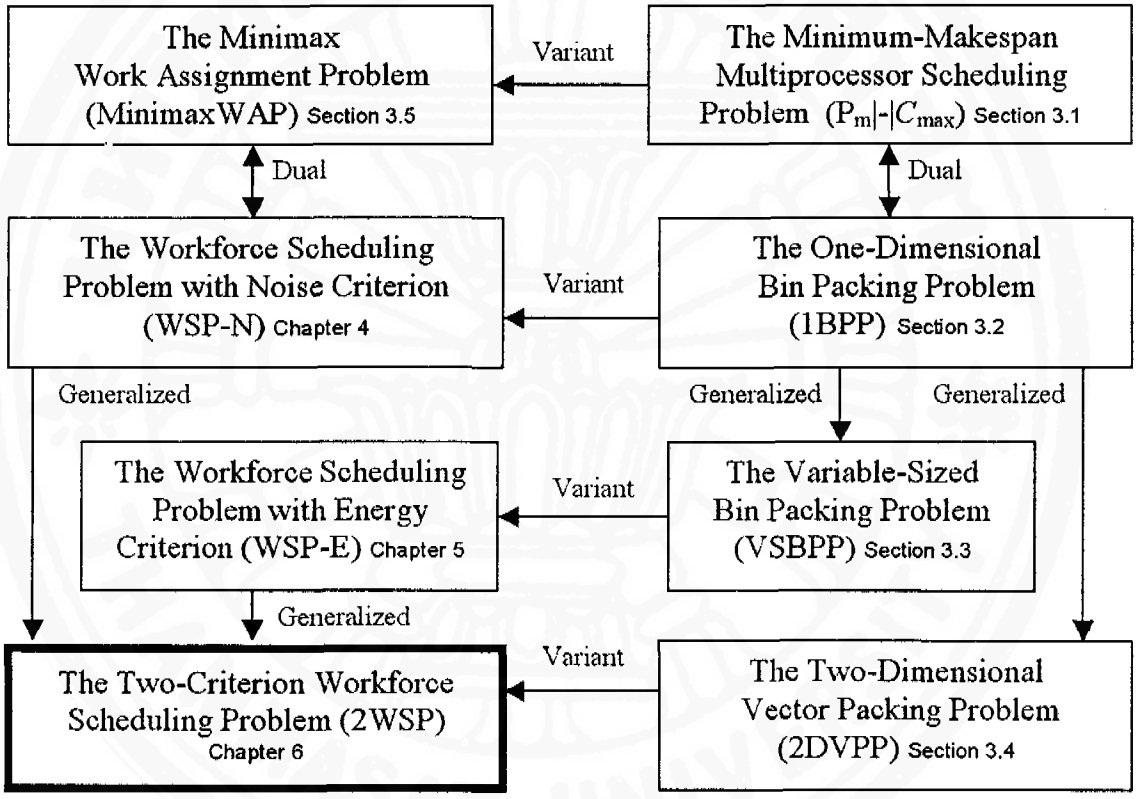

Figure 3.1 The relationship among the problems that are related to 2WSP

3.1 The Minimum-Makespan Multiprocessor Scheduling Problem ($P_m|-|C_{max}$)

First, a well-known scheduling problem, namely the Minimum-Makespan Multiprocessor Scheduling Problem or $P_m|-|C_{max}$ is reviewed. Given m identical parallel machines (multiprocessors) and n jobs with its processing time p_j, $P_m|-|C_{max}$ is to assign n jobs into m machines so that the maximum completion time (makespan) of all machines, C_{max}, is minimized. $P_m|-|C_{max}$ is one of the most classical combinatorial optimization problems and has appeared in the literature since 1960s. $P_m|-|C_{max}$ is known to be NP-hard (Garey and Johnson, 1979) and there exist a number of heuristics. Some of them are LPT (Longest Processing Time first) and MultiFit heuristics. Graham (1969) showed that the worst-case ratio of LPT heuristic is at most

$$\frac{C_{max}(LPT)}{C_{max}(OPT)} = \frac{4}{3} - \frac{1}{3m} \tag{3.1}$$

For the MultiFit heuristic by Coffman et al. (1978), its worst-case ratio is not greater than 1.220. MultiFit heuristic repeatedly uses the First-Fit Decreasing (FDD) heuristic, a simple algorithm for solving the one-dimensional bin packing problem (1BPP), to improve the solution of $P_m|-|C_{max}$. The 1BPP, which is reviewed in section 3.2, is well known as the dual problem of $P_m|-|C_{max}$. 1BPP is to identify the minimum number of bins that are required to pack all one-dimensional items with different size, where the height of bins is known and constant, and usually set to 1. A survey of 1BPP technique for solving $P_m|-|C_{max}$ can be found in Chen (2004). The mathematical model of is $P_m|-|C_{max}$ shown below.

Model of $P_m|-|C_{max}$

$$\text{Minimize} \quad C_{max} \tag{3.2}$$

$$\text{Subject to} \quad \sum_{j=1}^{n} p_j x_{ij} \le C_{max} \qquad \text{for } i = 1,\ldots, m \tag{3.3}$$

$$\sum_{i=1}^{m} x_{ij} = 1 \qquad \text{for } j = 1,\ldots, n \tag{3.4}$$

$$x_{ij} = \{0,1\} \qquad \text{for } i = 1,\ldots, m; j = 1,\ldots, n \tag{3.5}$$

Some of the variations of $P_m|-|C_{max}$ found in the literature are:

- *The Multidimensional Vector Scheduling Problem (MDVSP):* A set of jobs to be processed in a group of identical parallel machines has the multiple factors to be concerned (beside the processing time). For example, the processing cost, that is associated with each job, is concerned and the maximum cost sum of all machines is also required to be minimal. The approximation algorithms of solving MDVSP can be found in Chekuri and Khanna (1999). In fact, MDVSP is a dual problem of the Two-Dimensional Vector Packing Problem (2DVPP), which will be reviewed in Section 3.4.
- *The n Job-Groups $P_m|-|C_{max}$ Problem:* A set of jobs can be split into n subgroups, so-called n job-groups. In each subgroup, the processing time of all job in that subgroup is equal. When a machine needs to process a bunch of jobs of different subgroups, a setup time must be required when switching between the jobs of different subgroups. Sukto and Charnsethikul (2002) introduced this new variant and make use of list-scheduling heuristics and column generation technique to solve their proposed problem approximately. They also present the computational results in their paper. Column generation technique is well known for solving the classical one-dimensional Cutting Stock problem by solving many Knapsack problems as sub-problems. Interestingly, the MinimaxWAP is a more closely variant of the n Job-Groups $P_m|-|C_{max}$ Problem (with no setup time), rather than the $P_m|-|C_{max}$ itself. Again, the difference is also the existence of work period constraints that always separate the safety-based workforce scheduling problem from the combinatorial problems in Figure 3.1.

3.2 The One-Dimensional Bin-Packing Problem (1BPP)

Stated completely, the One-Dimensional Bin-Packing problem (1BPP) is to find the minimum number of fixed-height bins that is sufficient for being packed by a set of one-

dimensional items with various sizes. Since the 1BPP is also one of the first combinatorial optimization problems in the literature, there exist a number of papers concentrated on this problem and its variant. There are more than 100 papers of the problems since 1960s (Hochbaum, 1995), and the research of this problem still continues because of its vast application in industries, computer science, and businesses.

The developed methods for solving the problems range from optimization to heuristics. However, the problem is long known to be one of the first problems that were proved to be NP-complete; most papers thus concern on approximation approach. One way to formulate the mathematical model of 1BPP is shown below.

Model of the One-Dimensional Bin-Packing Problem (1BPP)

$$\text{Minimize} \quad \sum_{i=1}^{m} y_i \tag{3.6}$$

$$\text{Subject to} \quad \sum_{j=1}^{n} n_j x_{ij} \leq y_i \qquad \text{for } i = 1,\ldots, m \tag{3.7}$$

$$\sum_{i=1}^{m} x_{ij} = 1 \qquad \text{for } j = 1,\ldots, n \tag{3.8}$$

$$x_{ij}, y_i = \{0,1\} \qquad \text{for } i = 1,\ldots, m; j = 1,\ldots, n \tag{3.9}$$

Constraint 3.7 of 1BPP guarantees that the total height of items in any bin is at most 1 unit (e.g. meter). In Eiselt and Sandblom (2000), on page 149-151, they state the similarity between the 1BPP and the assignment of tasks to workers, in which they called the *Assembly Line Planning* problem. The existing algorithms for 1BPP are extensively reviewed next.

3.2.1 Approximation Algorithms

Johnson et al. (1974) proposed the first 1BPP heuristics namely First Fit Decreasing (FFD) and Best Fit Decreasing (BFD). The items in a list are sorted from large to small weights. FFD then tries to pack each item from the list into the smallest-indexed bin that fits. BFD differs from FFD by packing each item from the list into any *non-empty* bin that has smallest residual capacity. They proved that both heuristics will never use more than $1.222 \cdot L^* + 4$ bins where L^* is the minimum number of bins.

Scholl et al. (1997) reviewed the WFD and B2F heuristics and also proposed the FFD-B2F heuristics. Unlike BFD, Worst Fit Decreasing (WFD) heuristic tries to pack each item into any non-empty bin that has the *largest* residual capacity. For B2F heuristic, it works as FFD until a bin is completed, i.e., the residual capacity is lower than the weight of the lightest unassigned item. Then B2F tries to exchange the lightest item assigned in this bin for two *lighter* unassigned items such that a wasted residual capacity gets as small as possible. For the implementation of FFD-B2F, the reader can further study in their paper.

Gupta and Ho (1999) proposed an algorithm called Minimum Bin Slack (MBS) heuristic. MBS tries to find the group of items that fill each bin completely. Unlike FFD, MBS completes each bin first before initialize another empty bin. Their computational results indicated that MBS is superior, in term of solution quality, to FFD and BFD. MBS also finds an optimal solution for problem instances for which a two-bin solution exists.

Fleszar and Hindi (2002) modified MBS so that it gives a good initial solution which is subsequently improved by the variable neighborhood search meta-heuristics (together

called Perturbation MBS' + VNS algorithm). Their computational experiments claim that their hybrid algorithm outperforms the original MBS and is competitive to an optimization procedure namely BISON by Scholl et al. (1997).

Alvim et al. (2002) proposed a hybrid improvement heuristic (HI_BP) for 1BPP. HI_BP follows the outline based from Alvim et al. (1999). The dual strategy is applied in HI_BP, i.e. a dual problem of 1BPP is solved which may results a feasible solution of 1BPP at the number of bins equal to the lower bound. HI_BP can be implemented as follows.

Step 1. Apply the reduction procedure of Martello and Toth (1990a, 1990b).

Step 2. Use BFD to identify the upper bound (UB). The lower bounds (LB) by Martello and Toth (1990a, 1990b) and Fekete and Schepers (2001) are used. Stop if UB = LB.

Step 3. Apply the dual strategy using three greedy heuristics with the number of bins equal to LB.

Step 4. Apply the dual strategy using load balancing/unbalancing methods to improve bin usability.

Step 5. Apply the dual strategy using a tabu search to knock down capacity violations.

Step 6. If the solution from Step 3, 4, and 5 is feasible, then stop. Otherwise LB = LB + 1 and go to step 3.

An excellent survey of approximation algorithms of 1BPP is given by Coffman et al. (1999).

3.2.2 Exact Algorithms

Unlike heuristics, only a few exact algorithms have been presented in the literature, which are explained in details as follows.

Martello and Toth (1990a, 1990b) proposed the most frequently referred branch-and-bound algorithm so-called MTP. They proposed two first strong lower bounds (LBs) so-called L_2 and L_3 (beside from an obvious lower bound $L_1 = \lceil \sum w_j/c \rceil$ for all item j with a height of w_j and bins with a capacity of c.) L_2 is identified by separating all items into three subsets according to item's height. They first proposed a reduction procedure and a dominance rule which are both used to identify L_3. It can be shown that $L_2 \leq L_3$. But L_3 is more computationally expensive than L_2. At each decision node, L_2 and L_3 are identified; and heuristics: FFD, BFD, and WFD are solved and the solution with the minimum bin is set to UB. A branch is an item to be packed as done in FFD heuristic. A branch can be dominated (and fathomed) by the other branch by the dominance rule. A node is fathomed when one of its LBs is greater than or equal to UB. Steps in MTP are as follows.

Step 1. Apply the reduction procedure.

Step 2. Identify LB = max (L_2, L_3) and UB of root node.

Step 3. Apply the branch and bound method.

Vance et al. (1994) proposed a column generation based algorithm for 1BPP where columns correspond to feasible bins. The branching strategy is based on enforcing either join or separate assignment for item pairs. Chen and Srivastava (1996) developed two new lower bounds, which are the generalization of the lower bounds of MTP. The worst-case performance of the new lower bounds is analyzed and tested through experiments. Scholl et al. (1997) developed a hybrid procedure so-called BISON. They proposed three more lower bounds (L_4, L_5, and L_6). Each LB is strong in each of its special case of problems. At each decision node, FFD, BFD, WFD, B2F and FFD-B2F are used to improve the current UB. They also introduced three simple dominance rules and a tabu search. The tabu search solves a dual problem of 1BPP with the UB non-empty bins so that the residual capacity of

all UB bin is minimized and gives a feasible solution for 1BPP. A branching scheme of BISON is different from that of MTP. A branch is a group of items that fills a bin completely. BISON is bin-oriented; while MTP is item-oriented. Branches are categorized into two classes with different priorities. A branch can be dominated by the other branch by the dominance rules. Their computational results indicate that BISON outperforms MTP. Steps in BISON are as follows.

Step 1. Apply the reduction procedure (same as in MTP).

Step 2. Identify LB = max (L_2, L_4, L_5) and UB of root node.

Step 3. If UB=LB, then stop. Otherwise let LB=max(LB, L_3, L_6). Stop if UB=LB.

Step 4. Apply the dual strategy using a tabu search in order to decrease UB. Stop if UB=LB.

Step 5. Apply the branch and bound method.

Valerio de Carvalho (1999) formulates 1BPP as an arc flow problem, which is solved by a branch-and-price procedure with column generation technique. Fekete and Schepers (2001) developed new classes of lower bounds based on dual feasible function. Worst-case analysis as well as computational results shows that one of their classes clearly outperforms the lower bounds of MTP.

3.2.3 Meta-heuristic Algorithms

Falkenauer and Delchambre (1992) first designed a genetic algorithm with a cost function suitably for 1BPP, namely Grouping Genetic Algorithm (GGA). Falkenauer (1996) continued to improve GGA with the reduction procedure and dominance rule of MTP, which results as a hybrid grouping genetic algorithm (HGGA). Based on their computational tests, HGGA is superior to both GGA and MTP alone.

Alvim et al. (1999) devised a local search (LS) based on the progressive reduction of the number of bins used by a previously constructed solution. The differencing method for number partitions is used as the operator in LS. Based on their computational tests, LS is as efficient as HGGA but with much less computational time. Levine and Ducatelle (2003) devised a hybrid procedure which applies an ant colony optimization (ACO) approach for 1BPP. Their procedure, namely Hybrid Ant Colony Optimization (HACO), combined ACO with a simple local search based on MTP's dominance rule. Experimentally, HACO outperforms MTP and HGGA on the problems under consideration.

3.2.4 Variants of 1BPP

Since there are a number of algorithms in solving 1BPP; it is difficult to review most of the existing algorithms in most papers; however, some of the variation and generalization of 1BPP are partly summarized by Coffman et al. (1999) as follows.

- *The Variable Sized BPP*: The 1BPP with different-size bins. In each size of bin, there are unlimited numbers of bins. See the reviews of this problem in the next section
- *The Two-Dimensional Bin Packing Problem (2BPP):* The bins in this problem have two *dependent* dimensions: fixed height and width. The problems exist in cutting different rectangular parts from a steel coil. The parts cannot overlap each other parts. The variations of 2BPP are numerous and still are an active area of research. Three dimensions also appear in the recent literature. To the author's knowledge, the mathematical model of this problem is still unknown.
- *The Two-Dimensional Vector Packing Problem (2DVPP):* Please see the details of this problem in Section 3.4.
- *The Dual Bin Packing Problem (DBPP):* Instead of minimizing the number of bin required, In the DBPP, there is an unlimited number of one-dimensional bins of identical height, its objective is to pack items in as many bins as possible so that the total

height of each bin is at least equal to it height (Labbe et al., 1995). The higher dimension of DBPP also exists in the literature; it is called the Vector Covering Problem (VCP). In VCP, the bins and items have multiple and independent dimensions. Each dimension of bins has a limit.

- *The On-line Bin Packing Problem:* So far it is assumed that the height or size information of all items before packing them into the bins. This is in off-line mode. In real situation, it is unable to know the height of all upcoming items. Decision makers are required to periodically make decision on the current information of all items they currently have. Once the decision is made and the items have been packed. These items cannot be removed and included in the packing decision in the next period. This situation is called on-line mode.

3.3 The Variable-Sized Bin Packing Problem (VSBPP)

The Variable-Sized Bin Packing Problem (VSBPP) is the 1BPP with different-size bins. In each size of bin, there are unlimited numbers of bins. Later in Chapter 5, the Workforce Scheduling Problem with Energy Criterion (WSP-E) can be viewed as a variant of VSBPP. For the algorithms of VSBPP, only off-line algorithms are reviewed. In off-line scenario, all items arrive since the beginning and the packer knows the size of all items; while in on-line problems, the items arrive one by one, and the packer needs to pack each item without knowledge of the later items.

Friesen and Langston (1986) devised three algorithms, called NFL (next fit using largest bins only), FFDLR (first fit decreasing using largest bins, at end repack to smallest possible bins), and FFDLS (first fit decreasing using largest bins, but shifting as necessary). The three algorithms have the asymptotic worst-case performance bounds of 2, 3/2, and 4/3, respectively.

Murgolo (1987) gave a fully polynomial time approximation scheme for VSBPP which, for any given possible ε, produces a scheme with asymptotic worst case ration $1+\varepsilon$ and has time complexity polynomial in the number of items, the number of bins, and $1/\varepsilon$, respectively.

3.4 The Two-Dimensional Vector Packing Problem (2DVPP)

The bins of the Two-Dimensional Vector Packing Problem (2DVPP) have two *independent* dimensions. The example of 2DVPP is how to pack items to fixed-height bins with weight limits. The items occupy the height of bins while accumulate the total weight of the bin. 2DVPP first arises as a sub problem in resource constrained scheduling problem (Garey et al., 1976). We review the literature of 2DVPP because the Two-Criterion Workforce Scheduling Problem (2WSP) can be viewed as a variant of 2DVPP.

The 2DVPP is the generalization of the 1BPP to two dimensions. Thus, the 2DVPP is NP-hard. Moreover, the *d*-dimensional vector packing problem is the generalization of one-dimensional bin packing to *d* dimensions. Instead of being a single number, in the d-dimensional vector packing problem each item is a *d*-dimensional vector with coordinates in [0, 1]. The goal is to pack all items into a minimum number of bins provided with *d* corresponding capacities all equal to one. Since this problem is closely related to the proposed 2WSP. The literature on the 2DVPP will be reviewed by published year.

Spieksma (1994) proposed a heuristic adapted from the first fit decreasing rule, and also lower bounds used in the traditional branch-and-bound method. They showed that

computing one of their proposed lower bound is similar to computing the largest number of nodes of a clique of a 2-threshold graph. Some computational results are reported. Woeginger (1997) proved that there is no asymptotic polynomial time approximation scheme (PTAS) for 2DVPP. Coffman et al. (1999) comprehensively surveyed the approximation method of the 1BPP and most of its variant and generalization, including the 2DVPP. Chekuri and Khanna (1999) study the approximability of multi-dimensional generalization of three classical packing problems: multiprocessor scheduling, bin packing, and the knapsack problem. They proposed the new algorithms for these multi-dimensional (or multi-constraint) packing problems. Caprara and Toth (2001) introduce the mathematical model of 2DVPP, which can be solved by a traditional branch-and-bound method with linear relaxation. The model of 2DVPP is as follows.

Model of 2DVPP

$$\text{Minimize} \quad \sum_{i=1}^{m} y_i \tag{3.10}$$

$$\text{Subject to} \quad \sum_{j=1}^{n} w_j x_{ij} \leq y_i \qquad \text{for } i = 1,\ldots, m \tag{3.11}$$

$$\sum_{j=1}^{n} v_j x_{ij} \leq y_i \qquad \text{for } i = 1,\ldots, m \tag{3.12}$$

$$\sum_{i=1}^{m} x_{ij} = 1 \qquad \text{for } j = 1,\ldots, n \tag{3.13}$$

$$x_{ij}, \leq y_i \qquad \text{for } i = 1,\ldots, m \tag{3.14}$$

$$x_{ij}, y_i = \{0,1\} \qquad \text{for } i = 1,\ldots, m; j = 1,\ldots, n \tag{3.15}$$

Note that Constraint 3.14 is redundant; since Constraint 3.11 has already implied $x_{ij} \leq y_i$. They also introduce another formulation that is close to the *cutting stock formulation*. They also show that the 2DVPP can be converted to a *compatibility graph*, and use the graph theory to prove some lower bound. They have analyzed several lower bounds for 2DVPP. In particular, they determine an upper bound on the worst-case performance of class of lower bounding procedures derived from 1BPP. They also prove that the lower bound associated with the huge linear programming relaxation dominates all the other lower bounds being considered. They then introduce heuristic and exact algorithms, and report extensive computational results on several instance classes, showing that in some cases the heuristic approach allows for fast solution of the problem, while in other cases one has to use the exact algorithm. They claim that their results compare favorably with previous approaches to the problem.

Caprara et al. (2001) introduce a new case of 2DVPP in which the vectors of weights associated with the items are totally ordered, i.e., given any two weight vectors a_i, a_j, either a_i is *componentwise* not smaller than a_j or a_j is componentwise not smaller than a_i, and construct an asymptotic polynomial-time approximation scheme for this case. As a corollary, they also obtain such a scheme for the bin packing problem with cardinality constraint (i.e., the number of items in each bin is limited to a constant value), whose

existence was still an open question to their knowledge. Kellerer and Kotov (2003) present an $O(n\log n)$ time algorithm for 2DVPP with absolute performance guarantee 2.

A famous extension of 2DVPP namely the Robot Selection and Work Station Assignment Problem (RSWAP) is deserved to be reviewed. The RSWAP is first discussed by Han et al. (1994). Consider workplace layout an automated manufacturing system in which a number of workstations is to be served by a number of different types of robots. Each robot is constrained by its available processing time and work envelope, and each workstation has a known processing time demand and space requirement for each type of robot.

The RSWAP is to find the minimum cost mix of robots and assign each workstation to a single robot within the resource constraints. The RSWAP can be viewed as the multi-type two-dimensional vector packing problem. The robots are equal to the bins, and the workstations are the items. The mathematical model of RSWAP is as follows.

Han et al. (1994) proposed three solution algorithms: a simple greedy heuristic, a method based on *simulated annealing* (SA), and an exact algorithm based on Column Generation (CG) with a branch-and-bound method. Computational results are reported. Zhao et al. (1996) develop Genetic Algorithm (GA) for RSWAP for a CIM system. A multi-chromosome GA combined with heuristic bin packing algorithm is implemented for solving RSWAP. Computational results are reported. Han and Cook (1998) present an efficient (polynomial time bound) heuristics. The method consists of two stages. The first stage employs an LP-based bounding procedure to produce a tight solution bound, whereas the second stage repetitively invokes a random search heuristic using a greedy evaluation function. Based on their computational results, their heuristic outperforms the SA and CG methods.

3.5 The Minimax Work Assignment Problem (MinimaxWAP)

To our knowledge, to use job rotation to improve the safety of workers is first introduced by Nanthavanij and Yenradee (1999). They proposed the Minimax Work Assignment problem. In this study, the abbreviation of this problem is MinimaxWAP. MinimaxWAP can be stated as follows. For a workplace that has m workers performing n tasks, the noise level measured at the area of performing task j is equal to L_j. There is a set of work periods in an 8-hour day. MinimaxWAP is to identify the worker-task assignment that minimizes the maximum TWA of any worker in that assignment. (Section 2.1.2 explains the definition of TWA and its recommended limit.) Their original mathematical model of MinimaxWAP is formulated with the following assumptions.

1. The total time required for each task in a day is eight hours.
2. The total working time per day for each worker is eight hours.
3. The time of doing each task is measured in terms of the number of work periods.
4. The number of working hours per work period is constant, an integer number, and a devisor of 8. A fractional work period is not permissible.
5. Each task requires only one worker to perform per work period.
6. Each worker can perform any task with equal efficiency.

Notations used for the original model of MinimaxWAP are

h	number of working hours per work period
$\overline{L}_j$	noise level (dBA) measure at the area of performing task j
m	number of workers, which is equal to the number of tasks, n

p	number of work periods per day, where $p = 8/h$
x_{ijk}	1 if worker i is assigned to perform task j during work period k; 0 otherwise
n_j	noise weight per work period of task j, where $n_j = \frac{h}{8} 2^{\frac{\bar{L}j-90}{5}}$
TWA_{max}	The maximum TWA of all workers

Original Model of MinimaxWAP

$$\text{Minimize} \quad TWA_{max} \tag{3.16}$$

$$\text{Subject to} \quad 16.61 \log\left(\sum_{j=1}^{n}\sum_{k=1}^{p} n_j x_{ijk}\right) + 90 \le TWA_{max} \quad \text{for } i = 1,\ldots, m \tag{3.17}$$

$$\sum_{j=1}^{n} x_{ijk} \le 1 \quad \text{for } i = 1,\ldots, m;\ k = 1,\ldots, p \tag{3.18}$$

$$\sum_{i=1}^{m} x_{ijk} \le 1 \quad \text{for } j = 1,\ldots, n;\ k = 1,\ldots, p \tag{3.19}$$

$$\sum_{i=1}^{m}\sum_{k=1}^{p} x_{ijk} \ge p \quad \text{for } j = 1,\ldots, n \tag{3.20}$$

$$x_{ijk} = \{0,1\} \quad \text{for } i = 1,\ldots, m;\ j = 1,\ldots, n;\ k = 1,\ldots, p \tag{3.21}$$

Constraint 3.17 guarantees that the TWA of all workers cannot exceed TWA_{max}, which will be minimized. (The equations in Constraint 3.17 belong to Equation 2.3 in Section 2.1.2.) Constraints 3.18 and 3.19 represent Assumption 6. Constraint 3.20 makes sure that all tasks are performed in every period and throughout the 8-hour day. Note that this original model is nonlinear, which is very difficult to be solved optimally. They solved an example problem, which has 5 workers, 5 machines, and 4 periods (so-called as 5-5-4 or *m-n-p*), by using LINGO and achieve only the "local" optimal solution after about 800,000 iterations.

Later, Nanthavanij and Kullpattaranirun (2001) proposed two more near-optimal methods, which are a genetic algorithm and a heuristic method, and made the comparisons with the optimization method with LINGO. Based on their result, the genetic algorithm approach is practically superior to all existing methods.

Recently, Yaoyuenyong and Nanthavanij (2003) transform the non-linear model of MinimaxWAP to the linear model through a simple replacement as $N_{max} = 10^{\left(\frac{TWA\max-90}{16.61}\right)}$; or initially $TWA_{max} = 16.61 \log(N_{max}) + 90$.

In this linear model, the number of workers does not have to equal to the number of tasks ($m \neq n$), the linear model is shown below. Their integer linear (ILP) model of MinimaxWAP has fewer constraints than the original model does. Comparatively, this ILP model can guarantee the "global" solution while the original could not due to its nonlinear constraints.

After thoroughly studied the structure of the ILP model of MinimaxWAP, they also found that MinimaxWAP is actually the variation of $P_m|\text{-}|C_{max}$. A group of industrial workers in the MinimaxWAP are viewed as a set of multiprocessors (identical parallel machines) of $P_m|\text{-}|C_{max}$. Tasks with noise weight of MinimaxWAP are viewed as jobs with processing time of $P_m|\text{-}|C_{max}$. The major difference between the two problems is, the

MinimaxWAP requires one worker for one task at each of all work periods. $P_m|\text{-}|C_{max}$ does not have the constraint of work periods and the number of jobs in each machine is not necessary equal. (MinimaxWAP requires the equal number of workers performing each task of p) This difference is the key property that makes all of the safety-based problems different from all of the combinatorial optimization problems.

Integer Linear Model of MinimaxWAP

$$\text{Minimize} \quad N_{max} \tag{3.22}$$

$$\text{Subject to} \quad \sum_{j=1}^{n}\sum_{k=1}^{p} n_j x_{ijk} \le N_{max} \qquad \text{for } i = 1,\ldots, m \tag{3.23}$$

$$\sum_{j=1}^{n} x_{ijk} = 1 \qquad \text{for } i = 1,\ldots, m;\ k = 1,\ldots, p \tag{3.24}$$

$$\sum_{i=1}^{m} x_{ijk} = 1 \qquad \text{for } j = 1,\ldots, n;\ k = 1,\ldots, p \tag{3.25}$$

$$x_{ijk} = \{0,1\} \qquad \text{for } i = 1,\ldots, m; j = 1,\ldots, n;\ k = 1,\ldots, p \tag{3.26}$$

Table 3.1 Comparison between the MinimaxWAP and $P_m|\text{-}|C_{max}$ models

Notations	MinimaxWAP	$P_m\|\text{-}\|C_{max}$
Objective	N_{max}: The maximum weighted TWA of all workers, $N_{max} = 1$ when $TWA_{max} = 90$ dBA	C_{max}: The maximum makespan of all machines (processors)
n_j vs. p_j	Noise weight measured at task j	Processing time of job j
Decision variable x	x_{ijk} =1 if worker i is assigned to perform task j during work period k; 0 otherwise	x_{ij} =1 if machine i is assigned to process job j ; 0 otherwise
$i = 1,\ldots, m$	Worker i and the number of workers is m	Machine i and the number of machines is m
$j = 1,\ldots, n$	Task j and the number of tasks is n	Job j and the number of jobs is n
$k = 1,\ldots, p$	Work Period k and there is p periods.	(No dimension of period)

They also developed a heuristic namely the Modified LPT (M-LPT) heuristic. M-LPT was tested to be as efficient as the existing GA algorithm but with shorter computational time. Another safety-based problem, which has a similar ILP model of MinimaxWAP, is proposed by Carnahan et al. (2000). However, this problem aims to minimize the maximum Job Severity Index (JSI) of a group of workers who are assigned to do a set of lifting tasks. They applied the genetic algorithms to solve this problem successfully.

3.6 The Dual Problem of MinimaxWAP

Nanthavanij and Yenradee (2000) developed a dual model of MinimaxWAP. In this study, this dual model is in fact the Workforce Scheduling Problem with Noise Criterion (WSP-N). The objective of WSP-N is to find the *minimum number* of workers that are required to perform all given tasks and the TWA of all workers does not exceed 90 dB*A*, and also identify their daily work assignments.

Given the constant number of workers available, MinimaxWAP is to find the assignment that the maximum TWA of all workers is minimized. In contrast, given the number of tasks, WSP-N is to find the minimum number of workers and their assignments that are necessary for performing all tasks so that the maximum TWA of all workers is at most 90 dB*A*.

They successfully solved a set of problems of WSP-N using LINGO. But it is time consuming. Thus, they recommended the development of genetic algorithm or heuristics for solving the large-size of WSP-N. This has become the first part of the author's work in Chapter 4, in which our algorithms for WSP-N are proposed. The original mathematical model of WSP-N is shown in Chapter 4.