CHAPTER 5

THE WORKFORCE SCHEDULING PROBLEM WITH ENERGY CRITERION (WSP-E)

The Workforce Scheduling Problem with Energy Criterion (WSP-E) can be viewed as a variant of the Variable-Sized Bin Packing problem (VSBPP) in Section 3.3. Because workers with different working energy capacities are similar to bins with various capacities. The first section of Chapter 5 is the description and the mathematical model of WSP-E. Next, a lower bound is described. Then, three proposed algorithms are explained, followed by the hybrid procedure. Most concepts of WSP-E algorithms are similar to those of WSP-N. Readers are recommended to read Chapter 4 before this chapter. In order to illustrate the implementation of all algorithms, a numerical example is presented. Finally, all algorithms and hybrid procedure are tested in the computational experiment.

5.1 Problem Descriptions and Mathematical Model

Assumptions used in the formulation of WSP-E are as follows.

1.For a set of n physical tasks being considered, their energy costs are known.

2.All tasks are to be performed eight hours per day.

3.The number of available workers m is known and is more than the number of tasks. Not all workers need to be chosen for inclusion in the workforce scheduling and job rotation. With respect to work efficiency, all m workers are identical.

4.The daily working energy capacities of the m workers are known and are unequal.

5.A workday is divided into p equal work periods.

6.In each work period, a worker can be assigned to only one task and a task needs only one worker to perform.

7.A worker does not have to work in every work period.

Notation used for WSP-E are

m number of available workers
n number of physical tasks
p number of work periods per Day
e_j energy cost per work period of task j
E_i daily working energy capacity of worker i where $E_1 \geq E_2 \geq \ldots \geq E_m$
x_{ijk} 1 if worker i is assigned to task j during work period k
0 otherwise
y_i 1 if worker i is chosen from the workforce to perform any task
0 otherwise

WSP-E can be mathematically expressed as follows.

Model of WSP-E

$$\text{Minimize} \quad \sum_{i=1}^{m} y_i \tag{5.1}$$

$$\text{Subject to} \quad \sum_{j=1}^{n}\sum_{k=1}^{p} e_j x_{ijk} \le E_i \qquad \text{for } i = 1,\ldots, m \tag{5.2}$$

$$\sum_{j=1}^{n} x_{ijk} \le 1 \qquad \text{for } i = 1,\ldots, m;\ k = 1,\ldots, p \tag{5.3}$$

$$\sum_{i=1}^{m} x_{ijk} = 1 \qquad \text{for } j = 1,\ldots, n;\ k = 1,\ldots, p \tag{5.4}$$

$$\sum_{j=1}^{n}\sum_{k=1}^{p} x_{ijk} \le py_i \qquad \text{for } i = 1,\ldots, m \tag{5.5}$$

$$x_{ijk}, y_i = \{0,1\} \qquad \text{for } i = 1,\ldots, m; j = 1,\ldots, n; k = 1,\ldots, p \tag{5.6}$$

5.2 Lower Bound

The notations for WSP-E are as follows.

t_j a subtask of task j (task j has p copies of t_j)
I a set of subtasks of all tasks, e.g. if $p = 2$, then $I = \{t_1, t_1, t_2, t_2, \ldots, t_n, t_n\}$. Note that $|I| = np$.
$e(t_j)$ energy cost of t_j
$w(i, k)$ a set that may contain a subtask that is assigned to worker i in period k, where
$w(i, k) = \{t_j\}$ if t_j is assigned to worker i in period k, or
$w(i, k) = \varnothing$ if no subtask is assigned to worker i in period k
W_i a set of all subtasks assigned to worker i, i.e.,
$W_i = w(i, 1) \cup w(i,2) \cup \ldots \cup w(i, p)$
SE_i sum of energy costs of all $t_j \in W_i$
RE_i residual energy capacity of worker i where $RE_i = E_i - SE_i$

A lower bound (L_e) for WSP-E is defined as follows.

$$L_e = \min_{n \le d \le m} \left\{ d : \sum_{h=1}^{d} E_h \ge \sum_{t_j \in I} e(t_j) \right\} \tag{5.7}$$

5.3 The Modified FFD Heuristic (MFFD-E)

MFFD-E is similar to MFFD-N in Section 4.3. But MFFD-E considers $e(t_j)$ of tasks and E_i of workers instead. Note that MFFD-N considers $n(t_j)$ and N. MFFD-E is implemented as follows.

Step 0. Initialize I as defined. Sort all subtasks in non-increasing order of their $e(t_j)$ values. Let $w(1, 1) = \{t_1\}$, h=1, and $I = I \setminus \{t_1\}$.

Step 1. Consider the next t_j in the current order of the remaining I.
Step 2. Let a set H contains all worker $i = 1,\ldots, h$ whose $SE_i + e(t_j) \le E_i$.
Step 3. If $H \neq \varnothing$, choose the lowest-indexed worker in H, called worker v, who has any feasible period k that $w(v, k) = \varnothing$ and $\{t_j\} \cap \bigcup_{i=1}^{h} w(i,k) = \varnothing$.
Step 4. If $H = \varnothing$ or $H \neq \varnothing$ but there is no feasible period, let $h = h + 1$, $v = h$, and find any period k that $\{t_j\} \cap \bigcup_{i=1}^{h} w(i,k) = \varnothing$.
Step 5. Let $w(v, k) = \{t_j\}$ and $I = I \setminus \{t_j\}$. Go to Step 1 until $I = \varnothing$.
Step 6. Stop. MFFD-E gives the solution of h workers.

5.4 The DFDS-x Heuristic for WSP-E

The DFDS-x Heuristic for WSP-E is similar to the DFDS-x Heuristic for WSP-N in Section 4.4 but considers $e(t_j)$ of tasks and E_i of workers instead.

5.4.1. Dual Fit Decreasing (DFD) for WSP-E

Step 0. Set $h = m$. Let L_e be the lower bound identified in Section 5.2.
Step 1. Initialize I as defined. Sort all subtasks in non-increasing order of their $e(t_j)$ values. Let $w(1, 1) = \{t_1\}$ and $I = I \setminus \{t_1\}$.
Step 2. Consider the next t_j in the current order of the remaining I.
Step 3. From all h workers, find worker v
(a) who has any period k that $w(v, k) = \varnothing$ and $\{t_j\} \cap \bigcup_{i=1}^{h} w(i,k) = \varnothing$, and
(b) whose RE_v is a current maximum.
If there is a tie, break the tie arbitrarily.
Step 4. Let $w(v, k) = \{t_j\}$ and $I = I \setminus \{t_j\}$. Go to Step 2 until $I = \varnothing$.
Step 5. If $RE_i \ge 0$ for $i = 1,\ldots, h$, then go to Step 6. Otherwise, go to Step 8.
Step 6. If $h = L_e$, stop. The optimal solution of L_e workers is obtained.
Step 7. $h = h - 1$. Go to Step 1.
Step 8. Stop. The solution obtained is equal to $h + 1$ workers.

5.4.2. Dual Fit Decreasing with Swaps (DFDS) for WSP-E

In Step 5 of DFD, the solution is feasible if $RE_i \ge 0$ for all workers. However, if the current solution is *infeasible*, it is possible to make the solution feasible by some exchanges of the subtasks currently assigned among workers (similar to the concept in Section 4.4.2) Let $RE_{\min} = \min\{RE_1,\ldots, RE_h\}$, $RE_{\max} = \max\{RE_1,\ldots, RE_h\}$. The implementation of DFDS is the same as that of DFD except in Step 5. It is replaced by Steps 5.1 and 5.2.

Step 5.1 If $RE_{\min} < 0$, apply these procedures consecutively: ED, EI, ED2, and EI2.
Step 5.2 If $RE_{\min} \ge 0$, then go to Step 6 (as in DFD). Otherwise go to Step 8.

The procedures are described in details as follows. Note that the procedures of a single period or two periods are described since p is usually equal to four. Though, the procedures of three or more periods can be developed with the same idea.

Procedure ED

This procedure tries to decrease RE_{max} by exchanging the subtasks of workers in a single period. For any worker z whose $RE_z = RE_{max}$, find any worker i ($i = 1,\ldots, h$) that $i \neq z$ in any period k ($k = 1,\ldots, p$) such that Inequalities 5.8 and 5.9 hold.

$$RE_i + e(t_x) - e(t_y) < RE_{max} \quad (5.8)$$
$$e(t_x) > e(t_y) \quad (5.9)$$

Where $w(i, k) = \{t_x\}$ and $w(z, k) = \{t_y\}$. If there exists such worker i and period k, then swap their subtasks as $w(i, k) = \{t_y\}$ and $w(z, k) = \{t_x\}$.

Procedure EI

This procedure tries to increase RE_{min} by exchanging the subtasks of workers in a single period. For any worker z whose $RE_z = RE_{min}$, find any worker i ($i = 1,\ldots, h$) that $i \neq z$ in any period k ($k = 1,\ldots, p$) such that Inequalities 5.10 and 5.11 hold.

$$RE_i + e(t_x) - e(t_y) > RE_{min} \quad (5.10)$$
$$e(t_x) < e(t_y) \quad (5.11)$$

Where $w(i, k) = \{t_x\}$ and $w(z, k) = \{t_y\}$. If there exists such worker i and period k, then swap their subtasks as $w(i, k) = \{t_y\}$ and $w(z, k) = \{t_x\}$.

Procedure ED2

This procedure tries to decrease RE_{max} by exchanging the subtasks of workers in two periods. For any worker z whose $RE_z = RE_{max}$, find any worker i ($i = 1,\ldots, h$) that $i \neq z$ between any two periods k_1 and k_2 where $k_1 \neq k_2$ such that Inequalities 5.12 and 5.13 hold.

$$RE_i + e(t_{x1}) + e(t_{x2}) - e(t_{y1}) - e(t_{y2}) < RE_{max} \quad (5.12)$$
$$e(t_{x1}) + e(t_{x2}) > e(t_{y1}) + e(t_{y2}) \quad (5.13)$$

Where $w(i, k_1) = \{t_{x1}\}$, $w(i, k_2) = \{t_{x2}\}$, $w(z, k_1) = \{t_{y1}\}$,and $w(z, k_2) = \{t_{y2}\}$. If there exists such worker i, periods k_1, and period k_2; then swap their subtasks as $w(i, k_1) = \{t_{y1}\}$, $w(i, k_2) = \{t_{y2}\}$, $w(z, k_1) = \{t_{x1}\}$, and $w(z, k_2) = \{t_{x2}\}$.

Procedure EI2

This procedure tries to increase RE_{min} by exchanging the subtasks in two periods. For any worker z whose $RE_z = RE_{min}$, find any worker i ($i = 1,\ldots, h$) that $i \neq z$ between any two periods k_1 and k_2 where $k_1 \neq k_2$ such that Inequalities 5.14 and 5.15 hold.

$$RE_i + e(t_{x1}) + e(t_{x2}) - e(t_{y1}) - e(t_{y2}) > RE_{min} \quad (5.14)$$
$$e(t_{x1}) + e(t_{x2}) < e(t_{y1}) + e(t_{y2}) \quad (5.15)$$

Where $w(i, k_1) = \{t_{x1}\}$, $w(i, k_2) = \{t_{x2}\}$, $w(z, k_1) = \{t_{y1}\}$,and $w(z, k_2) = \{t_{y2}\}$. If there exists such worker i, periods k_1, and period k_2; then swap their subtasks as $w(i, k_1) = \{t_{y1}\}$, $w(i, k_2) = \{t_{y2}\}$, $w(z, k_1) = \{t_{x1}\}$, and $w(z, k_2) = \{t_{x2}\}$.

5.4.3.DFDS with Random Search (DFDS-x)

The DFDS-x heuristic for WSP-E is exactly similar to DFDS-x for WSP-N described in Section 4.4.3. DFDS-x is implemented by changing Step 5.2 of DFDS as follows. Let c be a counter which is set to 0.

Step 5.2.1 If $RE_{\min} \geq 0$, then go to Step 6 (as in DFD).
Step 5.2.2 $c = c + 1$
Step 5.2.3 If $c > x$, then go to Step 8 (as in DFD).
Step 5.2.4 For each $k = 1,\ldots, p$, randomly choose workers a and b where $a \neq b$.
Step 5.2.5 Exchange their subtasks of $w(a, k)$ and $w(b, k)$.
Step 5.2.6 Go to Step 5.1 of DFDS.

5.5 The Branch-and-Bound Method (BB-E)

The BB-E method is a branch-and-bound method designed to solve WSP-E for optimum. The concept of BB-E is not exactly similar to BB-N in Section 4.6. Please note the differences of implementing BB-E as follows.

Let I be a set of p copy of each t_j for all tasks, i.e., $| I | = np$. Sort all t_j in I in non-increasing order of their $e(t_j)$. W_i is a set of subtasks assigned to worker i, i.e., $W_i = w(i, 1) \cup w(i,2) \cup \ldots \cup w(i, p)$. W_i is feasible if $SE_i \leq E_i$.

BB-E also uses the dominance rule developed by Martello and Toth (1990a, 1990b) to eliminate any node that can be dominated. Supposing node $W_i(a)$ and $W_i(b)$ are both feasible sets for worker i, let Y_a be the number of workers in the optimal solution obtained by forcing the solution to always include node $W_i(a)$. Then, it is said that $W_i(a)$ dominates $W_i(b)$ if $Y_a \leq Y_b$. To check the dominance rule, $W_i(a)$ dominates $W_i(b)$ if and only if there exists a partition of $W_i(b)$ into subsets $Q_{(1)},\ldots,Q_{(f)}$. Also, there exists a subset $\{t_{(1)},\ldots,t_{(f)}\}$ of $W_i(a)$ such that $e(t_{(h)}) \geq \sum_{t_j \in Q_{(h)}} e(t_j)$ for $h = 1,\ldots,f$.

At the root node, the upper bound (UB) is firstly set to m, if the better UB is not known. The branching method at any node W_i starts by firstly generating all *feasible* sets that contain *at least one* t_j in the remaining I. Then, eliminate any set that can be *dominated* by the dominance rule. The remaining sets are the alternative *son* nodes of node W_i.

For example, there are r nodes branched from the root node denoted as $W_1(1)$, $W_1(2),\ldots,$ $W_1(r)$ that are generated from I for worker 1. All these r nodes cannot be dominated by any node of r nodes. Supposing node $W_1(2)$ is chosen to be explored, then all son nodes of $W_1(2)$ will be generated from the remaining $I_1 = I \setminus W_1(2)$. Again, supposing, at the lower level, node $W_2(5)$ is chosen next, then all son nodes of $W_2(5)$ can be generated from the remaining $I_2 = I_1 \setminus W_2(5)$, and so on. For any node $W_i(a)$ at level i, its son nodes are generated from $I_i = I_{i-1} \setminus W_i(a)$.

The bounding rule of BB-E is established by identifying L_{energy}. At node $W_i(a)$ of level i, let I_i contain the remaining *unassigned* t_j where $I_i = I_{i-1} \setminus W_i(a)$. Let $L_{\text{energy}}(I_i)$ be the lower bound of node $W_i(a)$ where

$$L_{\text{energy}}(I_i) = i + \min_{d \leq m} \left\{ d : \sum_{h=i+1}^{d} E_h \geq \sum_{t_j \in I_i} e(t_j) \right\} \qquad (5.16)$$

Node $W_i(a)$ is fathomed if $UB \leq L_{\text{energy}}(I_i)$. At any bottom node $W_i(x)$ of level i ($I_i = \varnothing$), if $UB > i$, then a new UB is set to i.

5.6 Hybrid Procedure

The proposed hybrid procedure for solving WSP-E is implemented as follows.

Step 1. Firstly, L_e is identified.
Step 2. Next, UB is identified by MFFD-E. Stop if $UB = L_e$.
Step 3. Apply the dual strategy using the DFDS-x Heuristic for WSP-E. If DFDS-x can find a feasible solution ($RE_{min} \geq 0$) with UB workers, then decrease UB until $UB = L_e$ or DFDS-x fails to find a feasible solution.
Step 4. Improve the current UB by BB-E. Stop if $UB = L_e$, or BB-E can guarantee the optimality of the current UB within the time limit.

5.7 Numerical Example

Let us consider the following example. Suppose that three physical tasks are to be performed in an eight-hour day that is divided into four equal work periods. Energy costs per work period, e_j, of the three tasks are 1100, 700, and 600 kilocalories (kcal/2h), respectively. Also, suppose that there are five available workers in the worker team. The working energy capacities, E_i, of the five workers are 2800, 2700, 2500, 2200, and 1800 kcal/8h, respectively.

5.7.1 Computation of Lower bound

For the above example (where $m = 5$, $n = 3$, and $p = 4$), The set I can be constructed and all subtasks are sorted according to its $e(t_j)$ values as $I = \{t_1, t_1, t_1, t_1, t_2, t_2, t_2, t_2, t_3, t_3, t_3, t_3\}$. Since $e(t_1) > e(t_2) > e(t_3)$. Thus,

$$\sum_{t_j \in I} e(t_j) = 9600 \text{ and } \sum_{h=3}^{d} E_i = (8000, 10200, 12000) \text{ for } d=(3,4,5).$$

Thus, $L_e = \min\{4, 5\} = 4$ workers (specifically worker 1 to worker 4).

5.7.2 Solution of MFFD-E Heuristic

MFFD-E yields the set of daily work assignments for five workers (i.e., $UB = 5$) as shown in Table 5.1. The superscript represents the order of assignment. Whenever it is infeasible to assign any subtask to the existing workers, a new worker is added and that subtask can be assigned to the new worker in any feasible period. Since $UB > L_e$, it cannot be guaranteed that the minimum number of workers for this problem is five workers.

Table 5.1 Work assignments by MFFD-E

Worker	Work Period				Energy (kcal/8h)	
	1	2	3	4	SE_i	E_i
$i = 1$	$t_1\,(1100)^1$	$t_1\,(1100)^2$	$t_3\,(600)^9$	-	2800	2800
$i = 2$	-	-	$t_1\,(1100)^3$	$t_1\,(1100)^4$	2200	2700
$i = 3$	$t_2\,(700)^5$	$t_2\,(700)^6$	$t_2\,(700)^7$	-	2100	2500
$i = 4$	$t_3\,(600)^{10}$	$t_3\,(600)^{11}$	-	$t_2\,(700)^8$	1900	2200
$i = 5$	-	-	-	$t_3\,(600)^{12}$	600	1800

5.7.3 Solution of DFDS Heuristic

Next, let DFD find the work assignment solution for first four workers. Since a solution of all five workers is given by MFFD-E already. Table 5.2 shows the assignment

of subtasks to four workers generated by DFD. Clearly, the resulting work assignment solution is infeasible since RE_{min} is still less than zero (RE_4 = - 200). When applying DFDS, it is found that between workers 2 (i = 2) and worker 4 (z = 4) at periods k_1 = 2 and k_2 = 3; these workers and periods will satisfy Inequalities 5.14 and 5.15. Thus, before swapping, $w(2, 2) = \{t_1\}$, $w(2, 3) = \varnothing$, $w(4, 2) = \{t_3\}$, and $w(4, 3) = \{t_2\}$. EI2 procedure can iecrease RE_{min} by swapping subtasks as $w(2, 2) = \{t_3\}$, $w(2, 3) = \{t_2\}$, $w(4, 2) = \{t_1\}$, and $w(4, 3) = \varnothing$.

As a result, RE_4 is still RE_{min} but is now equal to zero. Thus, the feasible set of assignments for four workers is found and is optimal since L_e = 4. Random search is unnecessary since the optimal solution was already found. Table 5.3 shows the optimal work assignment solution for the four workers.

Table 5.2 Work assignments by DFD

Worker	Work Period				Energy (kcal/8h)		
	1	2	3	4	E_i	SE_i	RE_i
i = 1	t_1 (1100)^1	t_2 (700)^5	t_3 (600)^9	-	2800	2400	400
i = 2	t_2 (700)^6	t_1 (1100)^2	-	t_3 (600)^{10}	2700	2400	300
i = 3	t_3 (600)^{11}	-	t_1 (1100)^3	t_2 (700)^7	2500	2400	100
i = 4	-	t_3 (600)^{12}	t_2 (700)^8	t_1 (1100)^4	2200	2400	- 200

Table 5.3 Work assignments by DFDS

Worker	Work Period				Energy (kcal/8h)		
	1	2	3	4	E_i	SE_i	RE_i
i = 1	t_1(1100)	t_2 (700)	t_3 (600)	-	2800	2400	400
i = 2	t_2 (700)	t_3 (600)	t_2 (700)	t_3 (600)	2700	2600	100
i = 3	t_3 (600)	-	t_1 (1100)	t_2 (700)	2500	2400	100
i = 4	-	t_1(1100)	-	t_1(1100)	2200	2200	0

5.7.4 Solution of BB-E

To demonstrate BB-E, let us solve this numerical example once again from the beginning. As before, $I = \{t_1^1, t_1^2, t_1^3, t_1^4, t_2^1, t_2^2, t_2^3, t_2^4, t_3^1, t_3^2, t_3^3, t_3^4\}$. The superscript of subtask indicates the index of subtask copies. Thus, t_2^1 is the first copy of subtask t_2. Figure 5.1 shows the decision tree generated by BB-E.

At the root node:

- $I = \{t_1^1, t_1^2, t_1^3, t_1^4, t_2^1, t_2^2, t_2^3, t_2^4, t_3^1, t_3^2, t_3^3, t_3^4\}$.
- At the root node, L_e = 4. Although the optimal solution of 4 workers is known. Assume that initial UB for BB-E is 5.
- For worker 1, E_1 = 2800. All feasible sets can be generated from I as $W_1(1) = \{t_1^1, t_1^2, t_3^1\}$, $W_1(2) = \{t_1^1, t_2^1, t_2^2\}$, and $W_1(3) = \{t_2^1, t_2^2, t_2^3, t_2^4\}$. It is seen that assignment $W_1(1)$ *cannot* dominate assignments $W_1(2)$ and $W_1(3)$ and vice versa. Thus, there are 3 son nodes of root node.

At node $W_1(1) = \{t_1^1, t_1^2, t_3^1\}$:

- $I_1 = I \backslash W_1(1) = \{ t_1^3, t_1^4, t_2^1, t_2^2, t_2^3, t_2^4, t_3^2, t_3^3, t_3^4\}$
- From Equation 5.16, $L_{energy}(I_1)$ = 4.

- Since $UB > L_{energy}(I_1)$, node $W_1(1)$ *cannot* be fathomed
- For worker 2, $E_2 = 2700$. Generate all feasible set from I_1 that cannot be dominated as $W_2(1) = \{t_1^3, t_1^4\}$, $W_2(2) = \{t_1^3, t_2^1, t_2^2\}$, and $W_2(3) = \{t_3^2, t_2^1, t_2^2, t_2^3\}$. There are 3 son nodes of node $W_1(1)$

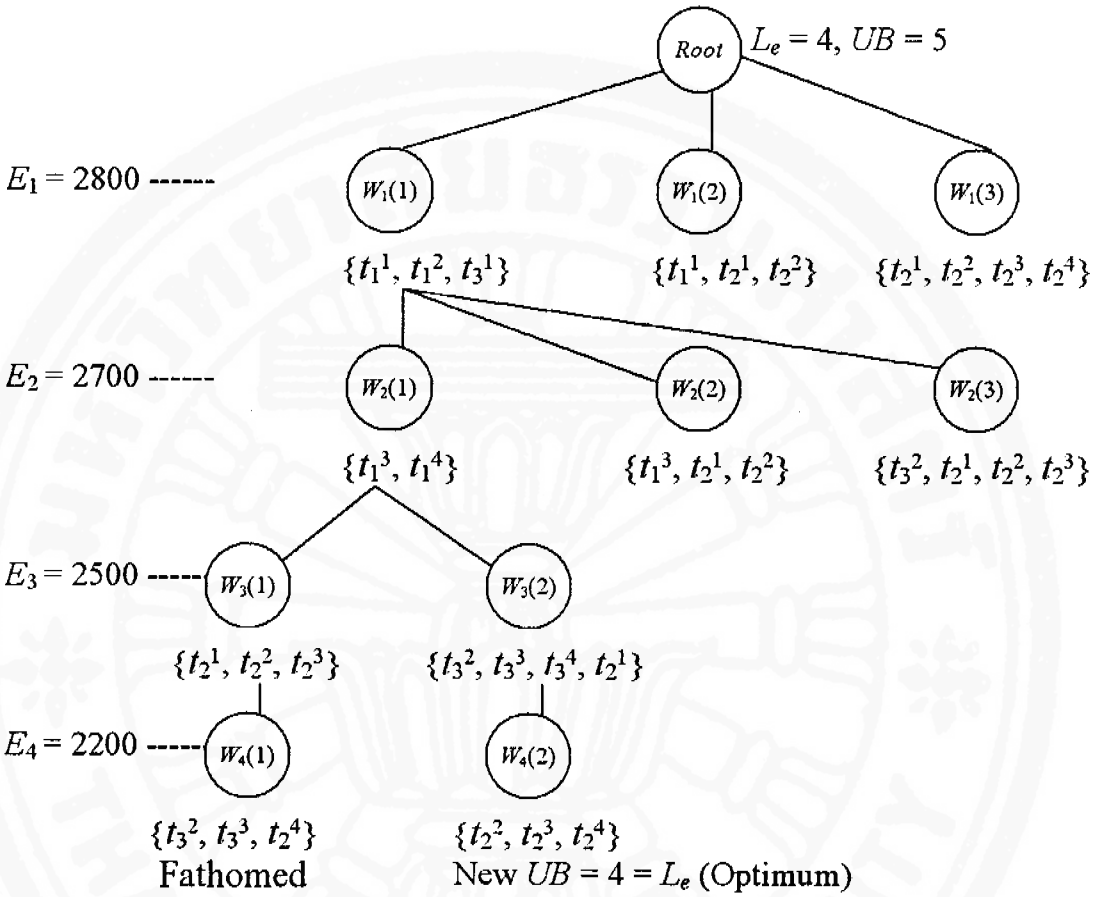

Figure 5.1 Decision tree generated by BB-E

At node $W_2(1) = \{t_1^3, t_1^4\}$:

- $I_2 = I_1 \backslash W_2(1) = \{t_2^1, t_2^2, t_2^3, t_2^4, t_3^2, t_3^3, t_3^4\}$
- From Equation 5.16, $L_{energy}(I_2) = 4$.
- Since $UB > L_{energy}(I_2)$, node $W_2(1)$ *cannot* be fathomed
- For worker 3, $E_3 = 2500$. Generate all feasible set from I_2 that cannot be dominated as $W_3(1) = \{t_2^1, t_2^2, t_2^3\}$ and $W_3(2) = \{t_3^2, t_3^3, t_3^4, t_2^1\}$. There are 2 son nodes of node $W_2(1)$

At node $W_3(1) = \{t_2^1, t_2^2, t_2^3\}$:

- $I_3 = I_2 \backslash W_3(1) = \{t_2^4, t_3^2, t_3^3, t_3^4\}$
- From Equation 5.16, $L_{energy}(I_3) = 4$.
- Since $UB > L_{energy}(I_3)$, node $W_3(1)$ *cannot* be fathomed
- For worker 4, $E_4 = 2200$. Generate all feasible set from I_3 that cannot be dominated as $W_4(1) = \{t_3^2, t_3^3, t_2^4\}$. There is only one son nodes of node $W_3(1)$.

At node $W_4(1) = \{t_3^2, t_3^3, t_2^4\}$:

- $I_4 = I_3 \backslash W_3(1) = \{t_3^4\}$

- From Equation 5.16, $L_{energy}(I_4) = 5$.
- Since $UB = L_{energy}(I_4)$, node $W_4(1)$ is fathomed and also node $W_3(1)$ is fathomed.

At node $W_3(2) = \{t_3^2, t_3^3, t_3^4, t_2^1\}$:
- $I_3 = I_2 \setminus W_3(2) = \{t_2^2, t_2^3, t_2^4\}$
- From Equation 5.16, $L_{energy}(I_3) = 4$.
- Since $UB > L_{energy}(I_3)$, node $W_3(2)$ *cannot* be fathomed
- For worker 4, $E_4 = 2200$. Generate all feasible set from I_3 that cannot be dominated as $W_4(2) = \{t_2^2, t_2^3, t_2^4\}$. There is only one son nodes of node $W_3(2)$.

At node $W_4(2) = \{t_2^2, t_2^3, t_2^4\}$:
- $I_4 = I_3 \setminus W_4(2) = \varnothing$
- From Equation 5.16, $L_{energy}(I_4) = 4$.
- Since $I_4 = \varnothing$ and no son node can be further generated. A new UB is then set to $L_{energy}(I_4)$. Since the new $UB = L_e$, the optimal solution is found at node $W_4(2)$. BB-E then terminated.

Table 5.4 shows the optimal solution generated by the BB-E method. In Tables 5.3 and 5.4, it is noted that the two optimal work assignment solutions while yielding the same number of workers (four workers) have different work assignments.

Table 5.4 Work assignments by BB-E

	Work Period				Energy (kcal/8h)		
Worker	1	2	3	4	E_i	SE_i)	RE_i
$i = 1$	-	t_1 (1100)	t_1 (1100)	t_3 (600)	2800	2800	0
$i = 2$	t_1 (1100)	-	-	t_1 (1100)	2700	2200	500
$i = 3$	t_3 (600)	t_3 (600)	t_3 (600)	t_2 (700)	2500	2500	0
$i = 4$	t_2 (700)	t_2 (700)	t_2 (700)	-	2200	2100	100

To compare the solution quality of this numerical example, all algorithms yield the optimal solution of 4 workers, except MFFD-E. However, in term of difference between RE_{max} and RE_{min}, DFDS produces the solution with the smallest gap. That is, all workers use about the same energy expenditure. See Table 5.5 for detailed comparison. This is an advantage of using the DFDS method.

Table 5.5 Comparison of algorithms in the WSP-E numerical example

Algorithm	Number of Workers	RE_{max}	RE_{min}
MFFD-E	5	1200	0
DFDS	4	400	0
BB-E	4	500	0

5.8 Computational Experiments

5.8.1 Test problems

Three sets (A, B, and C) of test problems (WSP-E) were randomly generated. Each set consisted of 100 problems which were divided into five levels of the number of tasks (n), i.e., 10, 20, 30, 40, and 50 tasks, respectively. For each n, there were 20 test problems. The number of work periods per day p was assumed to be four equal periods. For the experiment, the author considered physical tasks ranging from moderate to very heavy (Astrand and Rodahl , 1986). Energy costs e_j's were randomly generated using a uniform distribution between these following ranges:

Set A: $e_j \sim [300, 900]$ – moderate to heavy
Set B: $e_j \sim [600, 1200]$ – heavy to very heavy
Set C: $e_j \sim [300, 1200]$ – moderate to very heavy

A worker population of 1,000 male workers was then generated and the available workforce for each test problem was randomly drawn from that worker population. The working energy capacities of the 1,000 male workers were generated using a normal distribution with its mean and standard deviation of 2400 and 243.2 kcal/8h, respectively (NIOSH, 1981). To generate an initial number of workers m that is sufficient and feasible for each test problem, the 5th percentile energy capacity (12.5 kcal/min or 2000 kcal/8h) was used as a representative for a worker's energy capacity. The initial number of workers could be computed from the following formula.

$$m = \frac{p}{2000}\sum_{j=1}^{n} e_j \qquad (5.17)$$

To select workers for job rotation, m workers were randomly chosen from the 1000-worker population.

5.8.2 Experiment

All three algorithms and the hybrid procedure were coded in Visual Basic Application on Microsoft Excel and run on a 2.66 GHz, 512 MB RAM, Pentium IV personal computer. Each of the 300 test problems was solved by each of the three algorithms and the hybrid procedure with the same initial number of workers m. The computation time was measured in seconds. The time limit for BB-E when used either separately or as part of the hybrid procedure was set at 1,000 seconds.

The algorithm (and the hybrid procedure) is said to have solved the test problem to optimality when (1) $UB = L_e$, or (2) the BB-E method can find the optimal solution within the given time limit.

5.8.3 Results

Table 5.6 shows the number of test problems that each algorithm can guarantee an optimal solution by itself. When considering individual algorithms separately, it is seen that DFDS-1000 is the most efficient heuristic since it could guarantee the optimality for 252 problems (or 84.00%). MFFD-E is the least efficient heuristic, with only 9 optimal problems guaranteed. BB-E could guarantee only 53 problems perhaps because it has reached a 1000-second time limit before it could prove the optimality. As expected, the hybrid procedure is superior to any of the three algorithms when they are utilized separately. The hybrid procedure could find an optimal solution for 263 test problems (or 87.67%). For the remaining 37 test problems, the hybrid procedure exceeded the 1000-second time limit.

Table 5.6 Number of optimal solutions by WSP-E algorithms

Set	MFFD-E	DFDS-1000	BB-E		Hybrid Procedure	
	($UB = L_e$)	($UB = L_e$)	($UB = L_e$)	(BnB*)	($UB = L_e$)	(BnB*)
A (100)	7	99	12	0	99	0
B (100)	1	65	10	6	65	11
C (100)	1	88	25	0	88	0
Total	9	252	47 + 6 = 53		252 + 11 = 263	

*BnB means that L_e is too weak and the current best UB is optimal.

It is of interest to further investigate how the three algorithms and the hybrid procedure perform in each problem set with respect to the problem size. Table 5.7 shows the detailed summary of the results. Specifically, it shows the number of test problems that each algorithm can reach the best solution known (called *hits*), whether or not it (the algorithm) can guarantee if the solution is an optimal solution. A breakdown of numbers of hits shows the efficiency of individual algorithms in each problem size (five levels) of each set of the test problems. It is seen that DFDS-1000 was able to score 263 hits since there were additional 11 test problems that it could not guarantee the optimality but the lowest UB is in fact the optimal solution (as later verified by BB-E). Furthermore, the maximum differences (max dif) between the lowest UB generated by each algorithm and L_e for individual solutions are presented in Table 5.7. When BB-E is unable to verify the optimality, there will be a difference between UB and L_e. In fact, the lowest UB could be optimal and L_e is infeasible. However, the hybrid procedure fails to confirm that L_e is infeasible.

Table 5.7 Computational results for WSP-E algorithms

Set	n	Rep	Algorithm When Used Separately						Hybrid Procedure	
			MFFD-E		DFDS-1000		BB-E			
			hits	max dif	hits	max dif	hits	max dif	hits	max dif
A	10	20	4	2	20	0	12	1	20	0
	20	20	2	4	20	0	0	7	20	0
	30	20	1	5	19	1	0	9	19	1
	40	20	0	6	20	0	0	11	20	0
	50	20	0	6	20	0	0	17	20	0
B	10	20	1	4	18	1	16	1	18	1
	20	20	0	8	18	2	0	5	18	2
	30	20	0	15	15	3	0	15	15	3
	40	20	0	17	15	2	0	22	15	2
	50	20	0	23	10	1	0	29	10	1
C	10	20	1	2	19	1	18	1	19	1
	20	20	0	4	18	1	7	7	18	1
	30	20	0	6	17	1	0	11	17	1
	40	20	0	7	19	1	0	18	19	1
	50	20	0	7	15	1	0	23	15	1
Total Hits			9		263		53		263	
Max Dif			23		3		29		3	

The maximum differences between UB and L_e from MFFD-E, DFDS-1000, BB-E method, and the hybrid procedure are 23, 3, 29, and 3, respectively. Once again, it is seen that the hybrid procedure outperforms the other algorithms in all three problem sets.

Among the 37 test problems that the hybrid procedure could not prove the optimality within the 1000-second time limit, the maximum difference between *UB* and L_e of these 37 problems is not greater than 3. This result shows that the consecutive use of the algorithms can help to quickly improve (reduce) *UB* to perhaps its lowest.

From the comparison of computation times that individual algorithms and the hybrid procedure require to solve WSP-E (see Table 5.8), MFFD-E is the fastest solution algorithm with its maximum computation time of only 1.0 second. DFDS-1000 is very fast if performing only one start (DFDS-1). With 1000 moves of random search, it required much longer time. For BB-E, the 1,000-second time limit was reached in all five problem sizes of the three problem sets. It is also seen that the average time of the hybrid procedure was much less than that of the BB-E method, owing to the consecutive use of the algorithms.

It is seen that the hybrid procedure shows an outstanding performance in solving WSP-E. When comparing it to the other three algorithms based on the number of optimal solutions that can be guaranteed, it outperforms all of them. Of the 300 test problems, the hybrid procedure succeeded in solving 263 problems (or 87.67%), followed closely by DFDS-1000 (252 problems or 84%), BB-E (53 problems or 17.67%), and MFFD-E (9 problems or 3%). When considering the number of hits (the capability to reach the best solution known), DFDS-1000, although unable to verify, had also solved additional 11 problems to optimality; yielding the total of 263 hits (equal to that of the hybrid procedure).

Table 5.8 Computational times (in seconds) for WSP-E algorithms

			Algorithm When Used Separately						Hybrid Procedure	
			MFFD-E		DFDS-1000		BB-E			
Set	*n*	Rep	Avg.	Max.	Avg.	Max.	Avg.	Max.	Avg.	Max.
A	10	20	0.0	0.0	0.5	2.0	430.0	1000.0	1.0	2.0
	20	20	0.0	0.0	0.9	7.0	1000.0	1000.0	1.0	7.0
	30	20	0.1	1.0	1.0	11.0	1000.0	1000.0	51.0	1011.0
	40	20	0.1	1.0	2.8	5.0	1000.0	1000.0	3.0	5.0
	50	20	0.4	1.0	4.3	7.0	1000.0	1000.0	5.0	7.0
B	10	20	0.0	1.0	18.0	64.0	341.0	1000.0	187.0	1043.0
	20	20	0.0	0.0	26.9	137.0	1000.0	1000.0	129.0	1137.0
	30	20	0.1	1.0	136.9	393.0	1000.0	1000.0	433.0	1393.0
	40	20	0.2	1.0	164.0	618.0	1000.0	1000.0	414.0	1618.0
	50	20	0.2	1.0	310.3	929.0	1000.0	1000.0	811.0	1929.0
C	10	20	0.0	1.0	2.1	30.0	131.0	1000.0	52.0	1030.0
	20	20	0.0	0.0	1.3	8.0	856.0	1000.0	101.0	1008.0
	30	20	0.1	1.0	4.6	18.0	1000.0	1000.0	155.0	1016.0
	40	20	0.1	1.0	20.6	286.0	1000.0	1000.0	190.0	1286.0
	50	20	0.2	1.0	146.4	471.0	1000.0	1000.0	397.0	1471.0
Average Time			0.1		56.0		850.7		187.3	
Maximum Time			1.0		929.0		1000.0		1929.0	

When comparing the average computation times between BB-E and the hybrid procedure, the use of DFDS-1000 in the hybrid procedure to improve the initial *UB* yields a 78% improvement in the computation efficiency. Since DFDS-1000 could quickly find a good *UB*, BB-E in the hybrid procedure could prove the optimality within the given time limit; thus, resulting in a much less average computation time than that of BB-E alone.

In each set of the 300 test problems, the hybrid procedure achieved the highest hits and could, by itself, proved that all are the optimal solutions. For the 100 problems in Set A where $e_j \sim [300, 900]$, MFFD-E and BB-E performed poorly, scoring only 7 and 12 hits, respectively. DFDS-1000 and the hybrid procedure both performed equally well irrespective of the problem size, being able to score 99 hits. This might be because most of the randomly generated e_j's are small when compared with E_i's, which would make it relatively easy to assign the tasks to workers. From the results, all 20 test problems with $n = 50$ in Set A have the optimal solutions (the minimum number of workers) equal to n, which also equal to L_e.

In Set B where $e_j \sim [600, 1200]$, MFFD-E and BB-E scored only 1 and 16 hits, respectively, in problems with $n = 10$. DFDS-1000 could guarantee the optimality for 65 of the 100 test problems. With the hybrid procedure, BB-E could further guarantee in 11 additional test problems that the current *UB*s obtained by DFDS-1000 were in fact optimal. Thus, DFDS-1000 and the hybrid procedure could both score 76 hits. From Table 5.7, the problem size does have a negative effect on the performance of both solution procedures.

In Set C where $e_j \sim [300, 1200]$, MFFD-E could find an optimal solution of only one test problem (with $n = 10$) while BB-E could score 25 hits. Once again, DFDS-1000 and the hybrid procedure both scored equally at 88 hits. The problem size also does have some negative effect on their performances.

The main reason why DFDS-1000 demonstrated such an outstanding performance lies in its three algorithms: DFD, swap procedures, and random search. In several cases, the swap procedures could improve the current work assignment solution for the current *UB* workers such that the number of required workers could be reduced. In random search, where the current work assignment solution is moved, only *one* pair of workers is chosen to exchange their assigned energy costs in each period. This random search helps to move the work assignment solution to a new point in the feasible solution space, so that the new search for an optimal solution can be implemented again. It is noted that the number of moves is chosen to be 1,000 times based on the result from the author's preliminary experiment.

Since BB-E makes use of the branch-and-bound scheme, the computation time increases progressively with the problem size, especially when solving WSP-E. Thus, the 1000-second time limit was used in the experiment to terminate the solution search in large-sized problems. Although BB-E can guarantee the optimal solution when L_e is too weak or infeasible, its performance decreases drastically when n is more than 10 tasks in all three problem sets. From the author's observation, problems in Set A (in which the task level ranges from moderate to heavy) are very easy to solve irrespective of the problem size. This perhaps is due to the fact that most of the energy costs are comparatively smaller than one-fourth of the worker's average energy capacity (2400 kcal/8h). Problems in Set C (in which the task level ranges from moderate to very heavy) are also relatively easy to solve, perhaps for the same reason as that in Set A. For problems in Set B, the level of task ranges from heavy to very heavy. The large energy cost of the task makes it difficult to yield a work assignment for a worker that has the sum of energy costs close to his/her daily energy capacity.

DFDS-1000 dominates MFFD-E and BB-E in all three problem sets, owing to its efficient algorithms in improving (decreasing) the *UB*. However, it lacks the capability to guarantee the optimality of a solution when L_e is too weak. The hybrid procedure combines the strengths of DFDS-1000 and BB-E together and results in an efficient and also robust solution procedure.