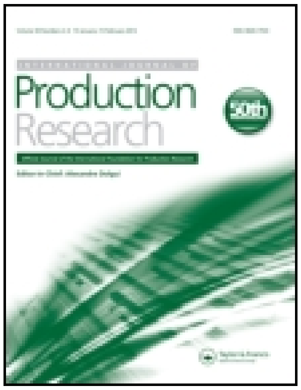


International Journal of Production Research
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/tprs20

# The burn-in test scheduling problem with batch dependent processing time and sequence dependent setup time

W.L. Pearn [a] , J.S. Hong [a] & Y.T. Tai [b]
[a] Department of Industrial Engineering & Management , National Chiao Tung University , Taiwan
[b] Department of Information Management , Kainan University , Taoyuan , Taiwan
Published online: 26 Jun 2012.

*International Journal of Production Research*
Vol. 51, No. 6, 15 March 2013, 1694–1706



# The burn-in test scheduling problem with batch dependent processing time and sequence dependent setup time

W.L. Pearn[a], J.S. Hong[a] and Y.T. Tai[b]*

[a]*Department of Industrial Engineering & Management, National Chiao Tung University, Taiwan;*
[b]*Department of Information Management, Kainan University, Taoyuan, Taiwan*

(*Received 7 November 2011; final version received 17 April 2012*)

The burn-in test scheduling problem (BTSP) is a variation of the complex batch processing machine scheduling problem, which is also a generalisation of the liquid crystal injection scheduling problem with incompatible product families and classical identical parallel machine problem. In the case we investigated on the BTSP, the jobs are clustered by their product families. The product families can be clustered by different product groups. In the same product group, jobs with different product families can be processed as a batch. The batch processing time is dependent on the longest processing time of those jobs in that batch. Setup times between two consecutive batches of different product groups on the same batch machine are sequentially dependent. In addition, the unequal ready times are considered in the BTSP which involves the decisions of batch formation and batch scheduling in order to minimise the total machine workload without violating due dates and the limited machine capacity restrictions. Since the BTSP involves constraints on unequal ready time, batch dependent processing time, and sequence dependent setup times, it is more difficult to solve than the classical parallel batch processing machine scheduling problem with compatible product families or incompatible product families. These restrictions mean that the existing methods cannot be applied into real-world factories directly. Consequently, this paper proposes a mixed integer programming model to solve the BTSP exactly. In addition, two efficient solution procedures which solve the BTSP are also presented.

**Keywords:** batch; incompatible product family; mixed integer programming; burn-in test

## 1. Introduction

In order to enhance efficiency of production, batch processing machines are very often involved in factory applications, in particular in the semiconductor (Lee *et al*. 1992, Lee and Uzsoy 1999, Pearn *et al*. 2004a) and thin film transistor liquid crystal display (TFT-LCD) manufacturing processes (Chung *et al*. 2009, Tai and Lai 2011). The burn-in test scheduling problem is an essential batch processing machine scheduling problem, which was first proposed by Lee *et al*. (1992). In an integrated circuit (IC) final test process of the semiconductor manufacturing process, burn-in test operation includes parallel batch processing machines which are often used to detect and screen out those infant defects. This process is essential, especially on those newly developed process or new designed devices to ensure their reliability to avoid ‘dead on arrival’. In a TFT-LCD manufacturing process, the ageing test operation (also referred to as the burn-in test operation) is used to detect the early failures before jobs are delivered (Chung *et al*. 2009). Lee and Uzsoy (1999) considered the batch processing machine scheduling problem with compatible product families in which the processing times are dependent on the longest processing time of all the jobs in the batch. However, the variety of burn-in specifications raised from customers’ requirements, such as temperatures and processing times in the burn-in operation, are increasing. Setup times between two consecutive batches of different product groups on the same batch machine are sequentially dependent. Consequently, in this paper, we consider a burn-in test scheduling problem (BTSP) which is a multi-dimensional parallel batch processing machines scheduling problem (see Figure 1) involving the constraints of batch dependent processing time, sequence dependent setup time, unequal ready time, due dates and limited machine capacity. The BTSP is more complicated than other classical parallel batch processing machine problems proposed by Chung *et al*. (2009) and Tai and Lai (2011).

Batch processing machine scheduling problems are commonly categorised into two types: compatible product family and incompatible product family. It should be noted that the BTSP considers the essential problem

*Corresponding author. Email: yttai@mail.knu.edu.tw

ISSN 0020–7543 print/ISSN 1366–588X online

http://dx.doi.org/10.1080/00207543.2012.694488
http://www.tandfonline.com

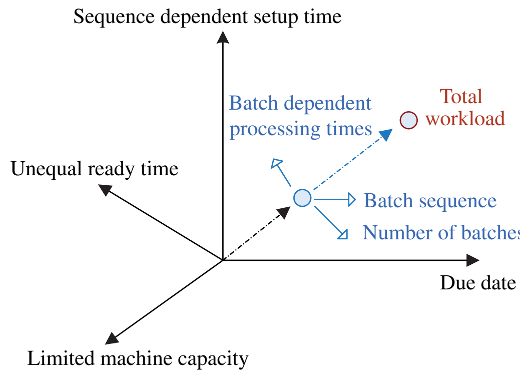


Figure 1. The multiple dimensions of the BTSP.



characteristics involving the two essential problem types simultaneously. In the first batch processing machine scheduling problem with compatible product family, it is assumed that jobs belonging to different product families may be simultaneously processed. The batch processing time is determined by the longest job processing time in that batch. The first researchers to address the problem arising in a burn-in oven of the final test in the semiconductor industry are Lee *et al*. (1992). In addition, the existing research works on the batch processing machine scheduling problems with compatible product family characteristics involve the single batch processing machine scheduling problem (Uzsoy 1994, Erramilli and Mason 2006, Chou *et al*. 2006, Kashan *et al*. 2006, Wang *et al*. 2007) and the parallel batch processing machine scheduling problem (Chang *et al*. 2004, Van Der Zee 2004, 2007, Chung *et al*. 2009). However, the existing research works do not take unequal ready times and setup time into considerations.

In the second batch processing machine scheduling problem with incompatible product family, some literature has investigated the solution procedures for the single batch processing machine to obtain exact or approximated solutions. Uzsoy (1994) is the first one considered the single batch processing machine scheduling problem with incompatible job families to minimise total completion time and makespan. Mehta and Uzsoy (1998), Dobson and Nambimadom (2001), Perez *et al*. (2005) and Tangudu and Kurz (2006) investigated the single batch processing machine with the incompatible product family characteristic to obtain exact or approximated solutions. For the parallel batch processing machines scheduling problems, Uzsoy (1994, 1995), Koh *et al*. (2004, 2005), Mönch *et al*. (2005, 2006), Malve and Uzsoy (2007), Castro and Novais (2009) and Tai and Lai (2011) presented the solution procedures to solve the scheduling problem in which one batch only contain the jobs clustered in the same product family. For the problems, the batch processing times are dependent on their jobs product family in that batch.

In this paper, the BTSP with the essential problem characteristics involves compatible and incompatible product family simultaneously. In the classical batch scheduling problems with compatible product families, the sequence dependence setup time is not considered due to fixed burn-in temperature. In the classical batch scheduling problems with incompatible product families, constant processing time is considered in one batch. However, the BTSP involves constraints of unequal ready times, limited machine capacity, batch dependent processing times and sequence dependence setup time, and is a variation of the classical parallel batch processing machine scheduling problem considered by Chung *et al*. (2009) and Tai and Lai (2011). The longest processing time dependent on the batch formation is the ultimate processing time. Once batches containing multiple jobs are formed and are processed on parallel batch processing machines successively, setup times between two consecutive batches of different product groups on the same burn-in test machine are incurred. In this paper, the objective of the BTSP is to schedule jobs without violating the constraints of unequal ready times, due dates, batch depended processing time and sequence dependent setup times, while the total machine workload is minimised. To the best of the authors' knowledge, the parallel batch processing machine scheduling problem with constraints that we investigated has not been considered by other researchers.

In this paper, we first formulate the BTSP as a mixed integer linear programming (MILP) model in order to obtain the exact solutions. If the computation time is a primary concern, two heuristic algorithms are presented and therefore could be an effective way of scheduling the jobs in the burn-in test operation with consideration of the batch dependent processing time, unequal ready time, due date, sequence dependent setup time and limited machine capability. This paper is organised as follows: Section 2 describes the burn-in test scheduling problem and Section 3 presents the MILP model for the BTSP. The practical illustration is shown in Section 4. Two heuristic algorithms are provided in Section 5. The test problem design and computation tests are presented in Section 6. Finally, Section 7 includes the conclusions.

## 2. The burn-in test scheduling problem in an IC final test process

An IC final testing process is the final stage of semiconductor manufacturing. In the burn-in test operation, many thousands of IC chips are arrayed in the oven with various settings of temperature and voltage to detect infant defects (Pearn *et al.* 2004a). Notably, there are several classifications (or grades) for the semiconductor products: military grade, industrial grade and consumer grade. Different devices have their own life time and infant mortality period. Consequently, the burn-in test time and temperature may vary accordingly. In the BTSP we investigated the jobs clustered by their product families. Product families are clustered by their product groups. In the same product group, jobs with different product families can be processed as a batch. The batch processing time is dependent on the longest processing times of those jobs in the batch. In a final testing house, the jobs in the burn-in test operation come from the upstream operations which is called 'final test' (FT) (Pearn *et al.* 2004a); thus, the ready times of jobs are usually unequal, which depend on the completion time of the final test operation. The formed batches must be processed on any of the parallel batch processing machines and be completed before their due dates. Setup times between two consecutive batches of different product groups on the same burn-in test machine are sequentially dependent. Machines are arranged in parallel and jobs to be processed are typically in hundreds of product families.



### 2.1 *The scheduling problem description for the BTSP*

The BTSP involves the batch formation and batch sequence can be stated as follows. The job set $C = \{c_{ij} | i = 1, 2, 3, \ldots, I, j = J(i)\}$ including $\sum_{i=1}^{I} J(i)$ jobs and each job $c_{ij}$ clustered by product family $i$, and $I$ product families was clustered into $G$ product groups $G = \{g_y | y = 0, 1, 2, \ldots, Y\}$, where $y = 0$ represents a pseudo product group. The jobs are formed into batches $H_{yb}$ ($b^{th}$ batch associated with product group $y$) in which that product groups of jobs are mutually incompatible to be processed on the batch machine set $K = \{k_m | m = 1, 2, \ldots, M\}$ containing $M$ identical parallel batch machines. Let $W$ represent the predetermined machine capacity and express the same time unit as that of processing time and setup time. Let the number of batches be denoted by $B$. Due to the considerations of real-world production restrictions in the burn-in test, job $c_{ij}$ has a ready time denoted by $r_{ij}$, a processing time denoted by $p_i$, and a latest starting time denoted by $e_{ij}$ which relates to the due date $d_{ij}$, where $e_{ij} = d_{ij} - p_i$. It should be noted that the processing time of batch $H_{yb}$ in the BTSP is depend on the longest processing time of the composite jobs with product family ($i$) in that batch; therefore, $pt_{yb} = \max\{p_i | i \in g_y\}$. The batch ready time is the latest ready times of those composite jobs; the batch latest starting time is the smallest starting time of those composite jobs. Consequently, the combination of jobs in a batch can directly determine the tightness of the time window in that batch.

To simplify the modelling of BTSP, we use information of product group $G$ to represent the information of product family directly, $C = \{c_{yj} | y = 0, 1, 2, \ldots, Y, j = J(y)\}$. Thus, variables $r_{ij}$, $p_i$, $e_{ij}$, and $d_{ij}$ can be transformed into variables $r_{yj}$, $p_{yj}$, $e_{yj}$ and $d_{yj}$ respectively. Let $t_{ybm}$ represent a starting time of batch $H_{yb}$ processed on machine $k_m$. The scheduled batch starting time ($t_{ybm}$) should not be less than the batch ready time and not be greater than the batch latest starting time. Furthermore, one batch can be processed on a machine on the condition that the accumulated number of jobs in that batch does not exceed the machine's capacity (maximum number of jobs can be processed simultaneously on a machine) ($b^{\max}$). Each batch is processed without pre-emption on one machine. No job can co-exist in more than one batch. In addition, let $s_{yy'}$ be the sequence dependent setup time between any two consecutive batches associated with different product groups ($y$ and $y'$) on the same machine. The BTSP can be solved to form batches appropriately as well as to find a schedule for those batches that satisfies the ready time and due date restrictions without violating the machine capacity constraints, while achieving the objective of minimising total workload.

## 3. A mixed integer programming model formulation

The mixed integer linear programming (MILP) model, with considerations of the real-world production environment, is proposed to obtain the detailed schedule for the BTSP. The objective of this MILP model is to minimise the total workload of burn-in test on the premise of satisfying the practical constraints. Before the mathematical model is presented, the decision variables used in the formulation are listed below.

$$x_{yjb} = \begin{cases} 1 & \text{if job } c_{yj} \text{ is assigned to batch } H_{yb}, \\ 0 & \text{otherwise.} \end{cases}$$

$$h_{yjm} = \begin{cases} 1 & \text{if job } c_{yj} \text{ is assigned to machine } k_m, \\ 0 & \text{otherwise.} \end{cases}$$

$$f_{ybm} = \begin{cases} 1, & \text{if the batch } H_{yb} \text{ associated with product group } y \text{ is processed on machine } k_m; \\ 0, & \text{otherwise.} \end{cases}$$

$$z_{yby'b'm} = \begin{cases} 1 & \text{if batch } H_{yb} \text{ associated with product group } y \text{ is scheduled immediately} \\ & \quad \text{following batch } H_{y'b'} \text{ associated with product group } y' \text{ on machine } k_m, \\ 0 & \text{otherwise.} \end{cases}$$

The mathematical formulation of the BTSP is shown as follows.

$$\text{Minimise} \sum_{m=1}^{M} \left\{ \sum_{y=0}^{Y} \sum_{b=1}^{B} pt_{ybm} + \sum_{y=0}^{Y} \sum_{b=1}^{B} \left( \sum_{y'=0}^{Y} \sum_{b'=1}^{B} z_{yby'b'm} s_{yy'} \right) \right\} \tag{1}$$

subject to

$$\sum_{b=1}^{B} x_{yjb} = 1, \quad \text{for all } y, j, \tag{2}$$

$$\sum_{m=1}^{M} h_{yjm} = 1, \quad \text{for all } y, j, \tag{3}$$

$$\sum_{j=1}^{J_i} x_{yjb} \le b^{\text{MAX}}, \quad \text{for all } y, b, \tag{4}$$

$$f_{0mm} = 1, \quad \text{for all } m, \tag{5}$$

$$\sum_{m=1}^{M} f_{ybm} \le \sum_{j=1}^{J_i} x_{yjb} \quad \text{for all } y, b, \tag{6}$$

$$\sum_{m=1}^{M} f_{ybm} \le 1, \quad \text{for all } y, b, \tag{7}$$

$$h_{yjm} \ge 1 + Q_1(x_{yjb} + f_{ybm} - 2), \quad \text{for all } y, j, b, m, \tag{8}$$

$$pt_{ybm} \ge p_{yj} + Q_1(x_{yjb} + h_{yjm} - 2), \quad \text{for all } y, j, b, m, \tag{9}$$

$$\sum_{b=1}^{B} pt_{ybm} + \sum_{y=0}^{Y} \sum_{b=1}^{B} \left( \sum_{y'=0}^{Y} \sum_{b'=1}^{B} z_{yby'b'm} s_{yy'} \right) \le W, \quad \text{for all } m, b' \ne b, \tag{10}$$

$$Q_1 f_{ybm} \geq t_{ybm}, \quad \text{for all } y, b, m, \tag{11}$$

$$\sum_{m=1}^{M} t_{ybm} \geq r_{yj} x_{yjb}, \quad \text{for all } y, j, b, \tag{12}$$

$$\sum_{m=1}^{M} t_{ybm} \leq d_{yj} \sum_{b=1}^{B} x_{yjb} - \sum_{m=1}^{M} pt_{ybm}, \quad \text{for all } y, j, b, \tag{13}$$

$$t_{ybm} + pt_{ybm} + s_{yy'} - t_{y'b'm} - Q_2(z_{yby'b'm} - 1) \leq 0, \quad \text{for all } y, y', b, b', m, b' \neq b, \tag{14}$$

$$(z_{yby'b'm} + z_{y'b'ibm}) + Q_2(f_{ybm} + f_{y'b'm} - 2) \leq 1, \quad \text{for all } y, y', b, b', m, b' \neq b, \tag{15}$$

$$(z_{yby'b'm} + z_{y'b'ibm}) - Q_2(f_{ybm} + f_{y'b'm}) \leq 0, \quad \text{for all } y, y', b, b', m, b' \neq b, \tag{16}$$

$$(z_{yby'b'm} + z_{y'b'ybm}) - Q_2(f_{y'b'm} - f_{ybm} + 1) \leq 0, \quad \text{for all } y, y', b, b', m, b' \neq b, \tag{17}$$

$$\sum_{y=0}^{Y} \sum_{b=1}^{B} z_{yby'b'm} + 1 = \sum_{y=0}^{Y} \sum_{b=1}^{B} f_{ybm}, \quad \text{for all } m, b, b' = 1, \ldots, B, b \neq b', \tag{18}$$

$$\sum_{H_{yb} \neq H_{y'b'}} z_{yby'b'm} \leq 1, \quad \text{for all } y, y', b, m, b' = 1, \ldots, B, b \neq b', \tag{19}$$

$$\sum_{H_{yb} \neq H_{y'b'}} z_{y'b'ybm} \leq 1, \quad \text{for all } y, y', b, m, b' = 1, \ldots, B, b \neq b', \tag{20}$$

$$x_{yjb} \in \{0, 1\}, \quad \text{for all } y, j, b, \tag{21}$$

$$z_{yby'b'm} \in \{0, 1\}, \quad \text{for all } y, y', b, b', m, b' \neq b, \tag{22}$$

$$f_{ybm} \in \{0, 1\}, \quad \text{for all } y, b, m. \tag{23}$$

The objective function (1) states that the total setup times and processing times in the burn-in test operation is minimal in order to minimise total workload. Constraints in (2) and (3) ensure that each job is exactly assigned to one batch and processed on one machine. Constraints in (4) are the batch size constraint, which requires that the sum of number of jobs contained in one batch processed simultaneously should not exceed the maximal batch size. Constraints in (5) guarantee that the only pseudo-job $c_{0j}$ is scheduled on one machine. Constraints in (6) states that if a product group $y$ assigned to batch $H_{yb}$, then there must have job(s) in product group $y$ assigned to batch $H_{yb}$. The BTSP could process different product families with the same product group simultaneously. Constraints in (7) ensure that each batch $H_{yb}$ consisted by one product group. The term $Q_1$ is a constant as it is sufficiently large in value to satisfy $Q_1(x_{yjb} + f_{ybm} - 2) \leq 0$ and $Q_1(x_{yjb} + h_{yjm} - 2) \leq 0$. Constraints in (8) are contingent constraints, if job $c_{yj}$ with product group $y$ assigned to batch $H_{yb}$ and $H_{yb}$ being processed on machine $k_m(f_{ybm} = 1)$, then job $c_{yj}$ processed by machine $k_m(h_{yjm} = 1)$. Constraints in (9) states that the processing time of batch $H_{yb}$ is the longest processing time of the jobs in batch $H_{yb}$. Constraint (10) is a capacity constraint, for each machine $k_m$, the machine workload does not exceed the machine capacity. Constraints in (11) are also contingent constraints. That is, if batch $H_{yb}$ associated with product group $y$ is not assigned to machine $k_m(f_{ybm} = 0)$, then the starting time of batch $H_{yb}$ on machine $k_m(t_{ybm})$ should be not greater than 0. Constraints in (12) and (13) indicate that the starting times of each batch are greater than or equal to the ready times and are not greater than the latest starting times of those, respectively. The batch ready time is the latest ready time of all the jobs clustered in one batch. The batch latest starting time is determined by the due dates of each job in the batch minus the processing time of the batch. Term $Q_2$

Table 1. The category of product family.

| Product group | Product family | Number of jobs | Product group | Product family | Number of jobs |
|---|---|---|---|---|---|
| A | 1 | 2 | C | 5 | 2 |
| A | 2 | 2 | C | 6 | 1 |
| B | 3 | 1 | C | 7 | 2 |
| B | 4 | 2 | | | |

Table 2. Ready times and due dates of the 12 jobs.

| Job | Product group | Ready time | Due date | Job | Product group | Ready time | Due date |
|---|---|---|---|---|---|---|---|
| $c_{11}$ | A | 30 | 950 | $c_{42}$ | B | 200 | 700 |
| $c_{12}$ | A | 200 | 920 | $c_{51}$ | C | 22 | 742 |
| $c_{21}$ | A | 230 | 950 | $c_{52}$ | C | 129 | 749 |
| $c_{22}$ | A | 23 | 943 | $c_{61}$ | C | 140 | 960 |
| $c_{31}$ | B | 150 | 870 | $c_{71}$ | C | 200 | 920 |
| $c_{41}$ | B | 100 | 820 | $c_{72}$ | C | 70 | 790 |

is the chosen constant as it is sufficiently large in value to satisfy the $z_{yby'b'm}=0$ or 1 which is required for constraints in (14)–(17). Constraints in (14) ensure the satisfaction of the inequality $t_{ybm}+pt_{ybm}+s_{yy'}\leq t_{y'b'm}$, where batch $H_{yb}$ immediately precedes batch $H_{y'b'}(z_{yby'b'm}=1)$. Constraints in (15)–(17) are the precedence constraints provided and their applicability has been demonstrated by Pearn *et al.* (2002). These constraints state their sequence relations. Constraints in (18) indicate that the number of batches processed on machine $k_m$ is equal to one more of the number of the combination of two batches that are processed consecutively. Constraints in (19) ensure that at most one batch $H_{yb}$ associated with product group $y$ can be scheduled behind batch $H_{y'b'}$ associated with product group $y'$ directly for all the batches, which are scheduled on the same machine $k_m$. Similarly, constraints in (20) ensure that at most one batch $H_{y'b'}$ associated with product group $y'$ can be scheduled behind batch $H_{yb}$ associated with product group $y$ directly for all the batches, which are scheduled on the same machine $k_m$. Constraints in (21)–(23) indicate that $x_{yjb}$, $z_{yby'b'm}$ and $f_{ybm}$ are binary integer variables. In the MILP model, the total number of variables is $I^2B^2M+2IBM+N_IB+N_IM$ and the total number of constraint equations is $(9/2)I^2B(B-1)M+IBM+3IB+2N_IBM+2N_IB+2N_I+3M$, where $G_i$ is the job number in product group $i$, $I$ is the total number of product groups, $N_I=G_0+G_1+G_2+\cdots+G_{I-1}$ and $B$ is the number of batches.

## 4. Solutions for the BTSP

To demonstrate the applicability of the scheduling mechanism for the BTSP, we consider a case in the burn-in test operation of a testing house for illustration purposes. In the following example, there are two parallel batch machines ($k_1$ and $k_2$) and 12 independent jobs clustered into seven product families and their family processing times are 180, 150, 160, 160, 145, 155 and 150 minutes, respectively. These seven product families are categorised into three product groups as shown in Table 1. The ready times and due dates for the 12 jobs is shown in Table 2. Furthermore, the setup times between different product groups are sequence dependent and presented in Table 3. In this illustrative example, the maximal number of jobs in one batch is two. Initially, it is assumed that each job will be contained in an individual batch ($B=12$). These 12 jobs should be clustered into appropriate batch numbers while the MILP model is performed; then, those batches are scheduled on the two identical parallel batch machines. The batch processing time is not affected by the machine processing it, but it is associated with job families composed in that batch. Finally, the machine capacity is set to 1200 minutes.

One optimal solution solved by the MILP for the BTSP with incompatible product groups is described as follows. Seven batches associated with different product groups are formed, namely $H_{A1}$, $H_{A4}$, $H_{B4}$, $H_{B5}$, $H_{C1}$, $H_{C2}$, and $H_{C3}$. Batch $H_{A1}$ is the first batch associated with product group A and is composed of $c_{11}$ and $c_{12}$ of productfamily 1. Therefore, the ready time and due date for $H_{A1}$ are 200 and 920, respectively. In the example we investigated, three batches of group C ($H_{C1}$, $H_{C2}$, and $H_{C3}$) are processed on machine $k_1$; four batches with two product groups, group A ($H_{A1}$ and $H_{A4}$) and B ($H_{B4}$ and $H_{B5}$) are processed on machine $k_2$ with one setup time because batches with different product groups are scheduled on this machine. The machine schedules of the optimal solution are depicted in Figure 2, where the total workload is 1155 minutes and the computation time on PC (Intel T2300 1.66 GHz with 512MB RAM) is 2671 CPU seconds.

To explore the run time growth with example problem size increasing, we ran eight example problems and show the computation results in Table 4, which contains various settings of number of jobs and product groups. It is noted that the computation time increases with the example problem size rapidly. In industrial applications, there

Table 3. Setup times of the three product groups.

| From \ To | U | A | B | C |
|---|---|---|---|---|
| U | – | 20 | 20 | 20 |
| A | 0 | – | 60 | 150 |
| B | 0 | 15 | – | 60 |
| C | 0 | 35 | 90 | – |

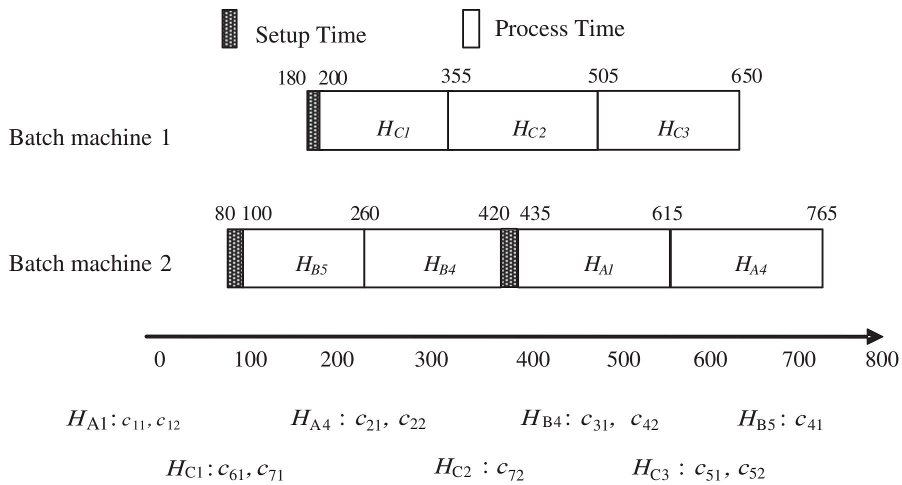


Figure 2. The Gantt chart of the illustrative example for the BTSP.

Table 4. Summary of example problems and computation times.

| Number of jobs | Number of product groups | Number of machines | CPU seconds |
|---|---|---|---|
| 9 | 3 | 2 | 356 |
| 12 | 3 | 2 | 2671 |
| 12 | 3 | 3 | 6210 |
| 16 | 4 | 4 | 8735 |
| 18 | 4 | 4 | 11501 |
| 18 | 4 | 5 | 17851 |
| 21 | 4 | 5 | * |
| 24 | 5 | 5 | * |

Remarks: *No solutions can be found within reasonable amount of computation time.

are hundreds of jobs which need to be scheduled for the daily production plan. Therefore, for large-scale BTSP, the optimal solutions may not be obtained easily within reasonable amount of computation time.

## 5. Heuristic algorithms

For the large-scale BTSP, the mixed integer linear programming model is computationally inefficient. Therefore, two heuristic algorithms are proposed to solve the BTSP, if the computation time is the primary concern. In this paper, we presented two two-phase heuristic algorithms for the BTSP, referred to as delay window-time parallel saving algorithm (*DWPSA*) and delay window-time generalised saving algorithm (*DWGSA*), in order to generate efficient solutions for the large-scale BTSP.

Notably, the batch processing machine scheduling problem can be decomposed into two parts, batch forming and batch scheduling, which is suggested by Mehta and Uzsoy (1998) and Mönch *et al.* (2005). In this paper, the two heuristic algorithms essentially consist of two phases each. Phase I forms appropriate batches to accommodate unequal ready times on the parallel burn-in test machines. In this phase, we incorporate the merits of the DELAY heuristic solution procedure proposed by Lee and Uzsoy (1999) and further modified by Chung *et al.* (2009) in order to accommodate the parallel processing batch machine with incompatible product families in the paper. The ready time of the selected jobs to be put in the forming batch need to be no later than the formed ready time of batch plus the processing time multiplied by a parameter. The main purpose is to avoid the jobs to cause excessive delays and to avoid jobs already waiting to be processed to cause the violation of due date constraint. However, this constraint might limit only few jobs could be put on one batch for some cases when variation of ready time of jobs is large. In such cases, the total processing time will increase dramatically since more batches are formed in this phase.

In the BTSP, we modify the constraint to limit the ready times of the selected jobs that should be no later than the latest starting time of the forming batch. In Phase II, it applies the idea of network transformation of wafer probing scheduling problem (WPSP) and liquid crystal injection scheduling problem, provided by Pearn *et al.* (2004b) and Tai and Lai (2011), respectively. In this phase, it incorporates the modified saving based algorithms to schedule those batches formed in Phase I in order to reduce the total workload. In addition, we incorporate and modify the saving function to allow the batch with shorter process window time to be processed earlier. The two proposed heuristic algorithms are described as follows.



### 5.1 *The delay window-time parallel saving algorithm (DWPSA)*

**Phase I** (batch formation)

**Step 0:** Let the available set ($\{AS\}$) be the set of jobs which are available to be selected as a batch. Sort all jobs associated with processing times in descending order of magnitude as an unscheduled-job list (*US*). Set $b = 1$.

**Step 1:** Choose the first job ($c_{yj}$) on the unscheduled-job list into the available set, $\{AS\}$. Set the decision time point (*tp*) as the ready time of the first job ($r_{yj}$). Let $E_y$ is the smallest latest starting time of the jobs in the available set in which product group is $y$.

**Step 2:** Check whether there is a job ($c_{yj'}$) with the same product group $y$ on the unscheduled-job list which satisfies constraints $r_{yj'} \leq E_y$ and number of batches in the available set is less than the maximum number of jobs in one batch ($b^{\text{MAX}}$). If a job ($c_{yj'}$) satisfies the conditions, then put the job ($c_{yj'}$) into the available set. Update the value of $E_y$ and go to Step 2, where $E_y = \{e_{yj} | c_{yj} \in \{AS\}\}$. Otherwise go to Step 3.

**Step 3:** Form the batch. Put the jobs in available set into batch $H_{yb}$ where $H_{yb} = \{AS\}$. Remove those jobs which are formed in batch $H_{yb}$ from the unscheduled-job list. Select the largest ready times of all the jobs in the formed batches as the batch ready time and select the smallest latest starting times of all the jobs in the formed batch as the batch latest starting time. Set $b = b + 1$.

**Step 4:** Check the unscheduled-job list, if the list still had unscheduled jobs. Clear all the jobs in the available set $\{AS\} = \phi$, $E_y = 0$, and go to Step 2. Otherwise, the phase of batch formation is completed.

**Phase II** (batch schedule)

Phase II is to schedule those batches formed in Phase I. In the *DWPSA* algorithm, we apply the parallel savings algorithm, which proposed by Golden (1977) and modified by Pearn *et al.* (2004b). The *DWPSA* initially calculates the savings consisting of three terms: the setup time, processing time and the time window restrictions. For the time window restriction, we proposed a new calculation considering the 'processing time window' ($e_{yj} - r_{yj}$) and force the batches with shorter processing time window be processed first. The *DWPSA* initially calculates the savings of all pairs of jobs and sorts those savings in the descending order of magnitude. The *DWPSA* creates a multiple of $M$ batch processing machines schedules simultaneously. In constructing the schedule, the $M$ pairs of the largest saving values are assigned to the $M$ available machines. Notably, the inserted job looks most promising for maximal waiting time, setup time reduction, and machine capacity ($W$) constraints. The parallel savings algorithm then searches downward from the savings list, for a batch which could be merged into the endpoints of current constructing schedules (at the front endpoint or rear endpoint) with the largest saving. The assigned procedure based on the saving procedure is repeated until all batches formed in Phase I are scheduled. The saving function in the $DWPSA$ can be expressed as Equation (24).

$$WPSA_{H_{yb}H_{y'b'}} = \begin{cases} 0 & \text{if } WPSA_{H_{yb}H_{y'b'}} < 0 \text{ or } y = y', b = b', \\ \alpha_1 (s_{Uy} + s_{y'U} - s_{yy'}) + 0.01 \times \beta_1 pt_{yb} \\ \quad + \gamma_1 \left( \frac{(e_{y'b'} - r_{y'b'})}{e_{yb}} - \frac{(e_{yb} - r_{yb})}{e_{y'b'}} \right) & \text{otherwise.} \end{cases} \tag{24}$$

In the saving calculation of the $DWPSA$, three parameters, $\alpha_1$, $\beta_1$ and $\gamma_1$, and three the ranges $0 \le \alpha_1 \le 1$, $0 \le \beta_1 \le 1$ and $0 \le \gamma_1 \le 1$, are added to the savings function as the weight of the 'savings term', 'processing time term' and 'time windows restrictions term', respectively. Parameter $\alpha_1$ represents weight of setup time savings, which results from consecutively processing two batches $H_{yb}$ and $H_{y'b'}$ as a batch pair. It can help to avoid a long setup time being incurred. Parameter $\beta_1$ can help that the batches with longer processing times are forced to be processed earlier than the others with the shorter ones. Finally, parameter $\gamma_1$ is used in the time windows restrictions term to prevent the batches being processed after their due dates in order to enhance customer satisfaction. In addition, the scaled values of 0.01 are added for the second term in Equation (24); it is mainly used to make the appropriate measurements because batch processing times are longer in the BTSP, adding this scaled value to make clear distinction of each saving value. Furthermore, $s_{yy'}$ represents the setup time between any two consecutive batches $H_{yb}$ and $H_{y'b'}$ associated with different product groups, respectively; $e_{yb}$ and $r_{yb}$ represent the latest starting time and the ready time of batch $H_{yb}$ with product group $y$, respectively; $U$ denotes the idle status of machine.



### 5.2 *The delay window-time generalised saving algorithm (DWGSA)*

Phase I of the $DWGSA$, which is referred to as the batch forming phase, also applies the modified DELAY algorithm as $DWPSA$. In Phase II, we incorporate the merit of DGSA heuristic solution procedure proposed by Tai and Lai (2011) for the LCISP. The inserting batch ($H_{yb}$) is chosen for scheduling, maximising the savings while minimising the insertion cost. This inserting batch selection criterion can avoid this algorithm to create a new schedule on another machine with a high setup time (Pearn *et al.* 2004b). In addition, unlike the insertion strategy of the $DWPSA$, the $DWGSA$ considers only the end of points, but also those positions between two consecutive batches. Let $\lambda(u_{pos-1}, H_{yb}, u_{pos})$ be the additional setup cost when batch $H_{yb}$ is inserted between position $(pos-1)$ and $pos$ in schedule $PS$. Let $\lambda^*(u_{pos-1}, H_{yb}, u_{pos})$ be the minimal insertion cost value.

$$\lambda(u_{pos-1}, H_{yb}, u_{pos}) = s_{u_{pos-1}y} + s_{yu_{pos}} - \delta_1 \times s_{u_{pos-1}u_{pos}} \quad 1 \le \delta_1 \le 2 \tag{25}$$

$$\lambda^*(u_{pos-1}, H_{yb}, u_{pos}) = \min[\lambda(u_{pos-1}, H_{yb}, u_{pos})] \tag{26}$$

Batch $H_{yb}$ is chosen, which maximises the savings $\sigma^*(u_{pos-1}, H_{yb}, u_{pos})$ while minimising the insertion cost $\lambda^*(u_{pos-1}, H_{yb}, u_{pos})$ and which avoids the algorithm to create a new schedule on another machine with a high setup time $\delta_2 \times s_{Uy}$. In addition to taking into account machine capacity, the latest starting time constraints of all batches must also be examined for violation before a batch is inserted. The procedure is repeated until all batches are scheduled or all schedules are full and cannot be expanded.

$$\sigma(u_{pos-1}, H_{yb}, u_{pos}) = \delta_2 \times s_{Uy} - \lambda^*(u_{pos-1}, H_{yb}, u_{pos}) \quad 1 \le \delta_2 \le 2 \tag{27}$$

$$\sigma^*(u_{pos-1}, H_{yb}, u_{pos}) = \max[\sigma(u_{pos-1}, H_{yb}, u_{pos})] \tag{28}$$

## 6. Test problem design and computation results comparisons

To analyse and compare the performance of those algorithms on various BTSP with different characteristics, we randomly generate 72 problems with different characteristics. The experimental design involves five essential characteristics: product group ratio, tightness of due date, processing time variation, setup time variation and

Table 5. Experimental factors for the BTSP problems.

| Factor | Value considered | Number of values |
|---|---|---|
| Product group ratio | 4, 6 | 2 |
| Tightness of due date | T, S | 2 |
| Processing time variation | L, M, S | 3 |
| Setup time variation | L, S | 2 |
| Maximal number of job allowed in one batch | 5, 6, 7 | 3 |
| Total problem instances | | 72 |



maximal number of jobs allowed in one batch. In these 72 test problems, they all have 80 job numbers. For the BTSP, the concentration of product group has great impact on the total incurred setup time in machine schedule. If a product group contains a large number of families (jobs), the total setup time may be decreased in the machine schedule. On the other hand, if a product group contains a small number of families (jobs), this product group will contribute a larger value of setup time. In this paper, we define an index, called product group ratio, representing the concentration of jobs in a product group, which is the division of the number of product families by the number of product groups. The larger the product group ratio the stronger the family concentration.

In these 72 test problems, we set the product group ratio at 4 and 6. Since the tightness of due date may impact on the solutions of BTSP, for those jobs with large time windows, they could be processed as a batch in jobs with the same product group and later ready time. In addition, for batches with later due dates (larger time windows), they allow postponement of scheduling to minimise the setup time. On the test problems, we set two levels of the tightness of the due dates, four and six times the processing time. In the factor of processing time variation, it can be characterised by three magnitudes, large (L), medium (M), and small (S). In the factor of setup time variation, two magnitudes involving large (L) and small (S) are characterised. Accordingly, the processing times in a problem instance are generated from uniform distributions in [150,440], [190,390] and [150,430] for large, medium and small variations, respectively. The large setup time variation is 2860 and the small setup time variation is 1361. Machine number is set as five and maximal number of job allowed in one batch is 5, 6 and 7 jobs in order to get 72 problem configurations. The five different experimental factors are listed in Table 5. In addition, the ready times are generated from uniform distributions in $[0, 1440]$ and the machine capacity is 3200 minutes. In those problem instances, once the batch processing begins, it is non-pre-emptive until the batch is completely processed. Processing and ready times are measured in minutes. All jobs should be formed as batches and be processed completely by the minimal total workload.

In order to solve those BTSP problem instances with distinct experimental factors using the two heuristic algorithms, the program codes of the two heuristic algorithms are written in Visual Basic 6.0. In the two heuristic algorithms, the number of formed batches is determined by Phase I, which is referred to as batch formation. It may be affected by the factors, including product group ratio, processing time variation and maximal number of job allowed in one batch. Tables 6 and 7 present the solutions of the 72 problem configurations generated by the proposed $DWPSA$ and $DWGSA$ algorithms with five parallel batch machines in each.

In Tables 6 and 7, the '6LTS' represents the batches with large processing time variation, tight due date tightness, and small setup time variations, which 80 jobs are clustered in six product groups. In addition, term $B'$ indicates the number of formed batches obtained from Phase I of the two heuristic algorithms for various problem instances.

In our testing, the run times for problems are very fast. Tables 6 and 7 display the run time required for the two algorithms in CPU seconds. We have found that none of them required more than 0.13 CPU seconds. In comparing the two developed algorithms, the test results showed that the $DWGSA$ receives 41 best solutions and seven tight solutions (out of 72) than those of the $DWPSA$. We particularly note that in those problems with loose capacity loading, the $DWGSA$ significantly outperformed the $DWPSA$. In the 72 problem instances, the product group ratio has great impact on the capacity loading. For example, the capacity loading is tighter when product group ratio is set to six rather than set to four in the 80 job problems due to the difference of the number of formed batch. In Tables 6 and 7, the results indicated that $DWGSA$ significantly outperformed $DWPSA$ for problems with loose capacity loading (product group ratio being four), as $DWGSA$ received 29 best solutions and four tight solutions out of 36 test problems.

Table 6. Run times and total workload results for the problem instances using the *DWPSA* heuristic algorithm with $\alpha_1 = 0.6$, $\beta_1 = 0.5$, and $\gamma_1 = 0.5$.

| $b^{max}$ | 5 | | | 6 | | | 7 | | |
|---|---|---|---|---|---|---|---|---|---|
| Problem configuration | Total workload | $B'$ | Time (sec) | Total workload | $B'$ | Time (sec) | Total workload | $B'$ | Time (sec) |
| 4MSL | 5310 | 17 | 0.144 | 5205 | 16 | 0.150 | 3805 | 12 | 0.154 |
| 4MSS | 5310 | 17 | 0.149 | 5190 | 16 | 0.151 | 3800 | 12 | 0.139 |
| 4MTL | 5350 | 17 | 0.145 | 5055 | 16 | 0.150 | 3835 | 12 | 0.139 |
| 4MTS | 5345 | 17 | 0.144 | 5055 | 16 | 0.151 | 3800 | 12 | 0.144 |
| 4SSL | 5215 | 17 | 0.144 | 5090 | 16 | 0.144 | 3770 | 12 | 0.144 |
| 4SSS | 5215 | 17 | 0.159 | 5090 | 16 | 0.144 | 3765 | 12 | 0.147 |
| 4STL | 5215 | 17 | 0.149 | 5120 | 16 | 0.132 | 3860 | 12 | 0.149 |
| 4STS | 5215 | 17 | 0.139 | 5155 | 16 | 0.132 | 3765 | 12 | 0.144 |
| 4LSL | 5215 | 17 | 0.139 | 5090 | 16 | 0.130 | 3765 | 12 | 0.144 |
| 4LSS | 5215 | 17 | 0.160 | 5090 | 16 | 0.155 | 3765 | 12 | 0.155 |
| 4LTL | 5195 | 17 | 0.169 | 5225 | 16 | 0.169 | 3785 | 12 | 0.157 |
| 4LTS | 5195 | 17 | 0.149 | 5230 | 16 | 0.139 | 3810 | 12 | 0.142 |
| 6MSL | 5240 | 18 | 0.179 | 4770 | 16 | 0.182 | 4095 | 14 | 0.169 |
| 6MSS | 5190 | 18 | 0.176 | 4770 | 16 | 0.178 | 4095 | 14 | 0.172 |
| 6MTL | 5190 | 18 | 0.188 | 4810 | 16 | 0.178 | 4095 | 14 | 0.178 |
| 6MTS | 5190 | 18 | 0.180 | 4830 | 16 | 0.185 | 4095 | 14 | 0.182 |
| 6SSL | 5270 | 18 | 0.191 | 4855 | 16 | 0.185 | 4155 | 14 | 0.165 |
| 6SSS | 5280 | 18 | 0.186 | 4855 | 16 | 0.191 | 4155 | 14 | 0.199 |
| 6STL | 5310 | 18 | 0.186 | 4855 | 16 | 0.191 | 4155 | 14 | 0.203 |
| 6STS | 5265 | 18 | 0.186 | 4855 | 16 | 0.177 | 4155 | 14 | 0.190 |
| 6LSL | 5550 | 18 | 0.186 | 5115 | 16 | 0.177 | 4615 | 14 | 0.189 |
| 6LSS | 5525 | 18 | 0.191 | 5115 | 16 | 0.190 | 4635 | 14 | 0.194 |
| 6LTL | 5590 | 18 | 0.186 | 5265 | 16 | 0.196 | 4705 | 14 | 0.196 |
| 6LTS | 5525 | 18 | 0.186 | 5275 | 16 | 0.169 | 4705 | 14 | 0.177 |

Table 7. Run times and total workload results for the problem instances using the *DWGSA* heuristic algorithm with $\delta_1 = 1$ and $\delta_2 = 1$.

| $b^{max}$ | 5 | | | 6 | | | 7 | | |
|---|---|---|---|---|---|---|---|---|---|
| Problem configuration | Total workload | $B'$ | Time (sec) | Total workload | $B'$ | Time (sec) | Total workload | $B'$ | Time (sec) |
| 4MSL | 5290 | 17 | 0.093 | 5155 | 16 | 0.103 | 3765 | 12 | 0.110 |
| 4MSS | 5305 | 17 | 0.101 | 5170 | 16 | 0.111 | 3780 | 12 | 0.109 |
| 4MTL | 5310 | 17 | 0.115 | 5055 | 16 | 0.105 | 3765 | 12 | 0.095 |
| 4MTS | 5290 | 17 | 0.094 | 5090 | 16 | 0.099 | 3765 | 12 | 0.094 |
| 4SSL | 5195 | 17 | 0.094 | 5070 | 16 | 0.095 | 3730 | 12 | 0.099 |
| 4SSS | 5210 | 17 | 0.099 | 5085 | 16 | 0.099 | 3745 | 12 | 0.099 |
| 4STL | 5215 | 17 | 0.104 | 5090 | 16 | 0.102 | 3730 | 12 | 0.114 |
| 4STS | 5215 | 17 | 0.104 | 5180 | 16 | 0.093 | 3730 | 12 | 0.108 |
| 4LSL | 5195 | 17 | 0.100 | 5070 | 16 | 0.110 | 3730 | 12 | 0.102 |
| 4LSS | 5210 | 17 | 0.108 | 5085 | 16 | 0.109 | 3745 | 12 | 0.104 |
| 4LTL | 5175 | 17 | 0.108 | 5255 | 16 | 0.108 | 3745 | 12 | 0.106 |
| 4LTS | 5175 | 17 | 0.109 | 5280 | 16 | 0.097 | 3745 | 12 | 0.102 |
| 6MSL | 5220 | 18 | 0.117 | 4790 | 16 | 0.117 | 4075 | 14 | 0.102 |
| 6MSS | 5205 | 18 | 0.116 | 4800 | 16 | 0.116 | 4090 | 14 | 0.111 |
| 6MTL | 5230 | 18 | 0.116 | 4770 | 16 | 0.116 | 4075 | 14 | 0.111 |
| 6MTS | 5305 | 18 | 0.111 | 4785 | 16 | 0.111 | 4090 | 14 | 0.113 |
| 6SSL | 5310 | 18 | 0.123 | 4885 | 16 | 0.123 | 4155 | 14 | 0.123 |
| 6SSS | 5370 | 18 | 0.100 | 4870 | 16 | 0.100 | 4150 | 14 | 0.100 |
| 6STL | 5250 | 18 | 0.115 | 4875 | 16 | 0.115 | 4215 | 14 | 0.125 |
| 6STS | 5265 | 18 | 0.116 | 4890 | 16 | 0.116 | 4310 | 14 | 0.096 |
| 6LSL | 5560 | 18 | 0.111 | 5135 | 16 | 0.111 | 4575 | 14 | 0.108 |
| 6LSS | 5525 | 18 | 0.109 | 5220 | 16 | 0.109 | 4575 | 14 | 0.103 |
| 6LTL | 5580 | 18 | 0.111 | 5265 | 16 | 0.111 | 4885 | 14 | 0.111 |
| 6LTS | 5690 | 18 | 0.126 | 5280 | 16 | 0.126 | 4885 | 14 | 0.114 |

Table 8. Total workload results for the problem instances using the *DWPSA* and *DWGSA* heuristic algorithms.

| | *DWPSA* | | | *DWGSA* | | |
|---|---|---|---|---|---|---|
| Problem configuration | Total workload | $B'$ | Time (sec) | Total workload | $B'$ | Time (sec) |
| 4MSL | 5195 | 16 | 0.150 | 5155 | 16 | 0.093 |
| 4MSS | 5195 | 16 | 0.167 | 5170 | 16 | 0.093 |
| 4MTL | 5075 | 16 | 0.167 | 5055 | 16 | 0.102 |
| 4MTS | 5075 | 16 | 0.149 | 5090 | 16 | 0.109 |
| 4SSL | 5110 | 16 | 0.149 | 5070 | 16 | 0.095 |
| 4SSS | 5110 | 16 | 0.155 | 5085 | 16 | 0.095 |
| 4STL | 5130 | 16 | 0.155 | 5090 | 16 | 0.092 |
| 4STS | 5125 | 16 | 0.158 | 5110 | 16 | 0.092 |
| 4LSL | 5110 | 16 | 0.155 | 5070 | 16 | 0.100 |
| 4LSS | 5110 | 16 | 0.163 | 5085 | 16 | 0.101 |
| 4LTL | 5235 | 16 | 0.152 | 5215 | 16 | 0.107 |
| 4LTS | 5230 | 16 | 0.158 | 5215 | 16 | 0.094 |
| 6MSL | 4790 | 16 | 0.195 | 4790 | 16 | 0.117 |
| 6MSS | 4790 | 16 | 0.175 | 4805 | 16 | 0.121 |
| 6MTL | 4790 | 16 | 0.181 | 4770 | 16 | 0.122 |
| 6MTS | 4790 | 16 | 0.195 | 4785 | 16 | 0.119 |
| 6SSL | 4875 | 16 | 0.195 | 4875 | 16 | 0.117 |
| 6SSS | 4875 | 16 | 0.203 | 4870 | 16 | 0.109 |
| 6STL | 4875 | 16 | 0.202 | 4875 | 16 | 0.115 |
| 6STS | 4875 | 16 | 0.198 | 4890 | 16 | 0.111 |
| 6LSL | 5135 | 16 | 0.199 | 5135 | 16 | 0.111 |
| 6LSS | 5135 | 16 | 0.193 | 5150 | 16 | 0.119 |
| 6LTL | 5285 | 16 | 0.187 | 5265 | 16 | 0.112 |
| 6LTS | 5295 | 16 | 0.191 | 5280 | 16 | 0.117 |

Therefore, to confirm the results, we took an additional 24 problems with $b^{\max}=6$ and six parallel batch machines. These problems all have loose capacity loadings. The results of the experiments (see Table 8) indicated that the *DWGSA* algorithm perform remarkably well. The *DWGSA* approach achieved 16 best solutions and four tight solutions (out of 24) than those of the *DWPSA* algorithm in terms of total workload.

## 7. Conclusions

In this paper burn-in test scheduling problem (BTSP) has many real-world applications, involving the constraints of batch dependent processing time, sequence dependent setup time, unequal ready time, due dates and limited machine capacity. The BTSP is more complicated than other classical parallel batch processing machine problems with compatible or incompatible product families. In this paper, a mixed integer programming model was proposed to solve the BTSP exactly. If the computation time was a primary concern, two solution procedures based on the modified batch formation technologies and network algorithms were also developed to solve large-scale problems efficiently. In addition, a design on the test problems considering five factors involving product group ratio, tightness of due date, processing time variation, setup time variation, and maximal number of job allowed in one batch was conducted for the computation tests and comparisons. The computation results showed that the *DWGSA* performed well, particularly for loose capacity loading. In addition, all proposed algorithms solved the large-scale LCISP quite efficiently within a reasonable amount of computation time. In further research, some metaheuristic solution procedures, such as particle swarm optimisation and generic algorithm might be applied to solve the BTSP.

## References

Castro, P.M. and Novais, A.Q., 2009. Scheduling multistage batch plants with sequence-dependent changeovers. *American Institute of Chemical Engineers*, 55 (8), 2122–2137.

Chang, P. Y., Damodaran, P., and Melouk, S., 2004. Minimizing makespan on parallel batch processing machines. *International Journal of Production Research*, 42 (19), 4211–4220.

Chou, F.D., Chang, P.C., and Wang, H.M., 2006. A hybrid genetic algorithm to minimize makespan for the single batch machine dynamic scheduling problem. *International Journal of Advanced Manufacturing Technology*, 31 (3), 350–359.

Chung, S.H., Tai, Y.T., and Pearn, W.L., 2009. Minimizing makespan on parallel batch processing machines with non-identical ready time and arbitrary job sizes. *International Journal of Production Research*, 47 (18), 5109–5128.

Dobson, G. and Nambimadom, R.S., 2001. The batch loading and scheduling problem. *Operations Research*, 49 (1), 52–65.

Erramilli, V. and Mason, S.J., 2006. Multiple orders per job compatible batch scheduling. *IEEE Transactions on Electronics Packaging Manufacturing*, 29 (4), 285–296.

Golden, B.L., 1977. Evaluate a sequential vehicle routing algorithm. *AIIE Transactions*, 9 (2), 204–208.

Kashan, A. H., Karimi, B., and Jolai, F., 2006. Effective hybrid genetic algorithm for minimizing makespan on a single-batch-processing machine with non-identical job sizes. *International Journal of Production Research*, 44 (12), 2337–2360.

Koh, S.G., *et al.*, 2004. Scheduling parallel batch processing machines with arbitrary job sizes and incompatible job families. *International Journal of Production Research*, 42 (19), 4091–4107.

Koh, S.G., *et al.*, 2005. Scheduling a single batch processing machine with arbitrary job sizes and incompatible job families. *International Journal of Production Economics*, 98 (1), 81–96.

Lee, C.Y. and Uzsoy, R., 1999. Minimising makespan on a single batch processing machine with dynamic job arrivals. *International Journal of Production Research*, 37 (1), 219–236.

Lee, C.Y., Uzsoy, R., and Martin-Vega, L.A., 1992. Efficient algorithms for scheduling semiconductor burn-in operations. *Operations Research*, 40 (4), 764–775.

Malve, S. and Uzsoy, R., 2007. A genetic algorithm for minimizing maximum lateness in parallel identical batch processing machines with dynamic job arrivals and incompatible job families. *Computer & Operations Research*, 34 (10), 3016–3028.

Mehta, S.V. and Uzsoy, R., 1998. Minimizing total tardiness on a batch processing machine with incompatible job families. *IIE Transactions*, 30 (2), 165–178.

Mönch, L., *et al.*, 2005. Heuristic scheduling of jobs on parallel batch machines with incompatible job families and unequal ready times. *Computers & Operations research*, 32 (11), 2731–2750.

Mönch, L., Zimmermann, J., and Otto, P., 2006. Machine learning techniques for scheduling jobs with incompatible families and unequal ready times on parallel batch machines. *Engineering Applications of Artificial Intelligence*, 19 (3), 235–245.

Pearn, W.L., Chung, S.H., and Yang, M.H., 2002. Minimising the total machine workload for the wafer probing scheduling problem. *IIE Transactions*, 34 (2), 211–220.

Pearn, W.L., *et al.*, 2004a. A case study on the multistage IC final testing scheduling problem with reentry. *International Journal of Production Economics*, 88 (3), 257–267.

Pearn, W.L., Chung, S.H., and Yang, M.H., 2004b. Algorithms for the wafer probing scheduling problem with sequence-dependent set-up time and due date restrictions. *Journal of the Operational Research Society*, 55 (11), 1194–1207.

Perez, I.C., Fowler, J.W., and Carlyle, W.M., 2005. Minimising total weighted tardiness on a single batch process with incompatible job families. *Computers & Operations Research*, 32 (2), 327–341.

Tai, Y.T. and Lai, C.M., 2011. The liquid crystal injection scheduling problem (LCISP). *International Journal of Production Research*, 49 (2), 467–484.

Tangudu, S.K. and Kurz, M.E., 2006. A branch and bound algorithm to minimise total weighted tardiness on a single batch processing machine with ready times and incompatible job families. *Production Planning and Control*, 17 (7), 728–741.

Uzsoy, R., 1994. Scheduling a single batch processing machine with non-identical job sizes. *International Journal of Production Research*, 32 (7), 1615–1635.

Uzsoy, R., 1995. Scheduling batch processing machines with incompatible job families. *International Journal of Production Research*, 33 (10), 2658–2708.

Van Der Zee, D.J., 2004. Dynamic scheduling of batch servers with compatible product families. *International Journal of Production Research*, 42 (22), 4803–4826.

Van Der Zee, D.J., 2007. Dynamic scheduling of batch-processing machines with non-identical product sizes. *International Journal of Production Research*, 45 (10), 2327–2349.

Wang, H.M., Chang, P.C., and Chou, F.D., 2007. A hybrid forward/backward approach for single batch scheduling problems with non-identical job sizes. *Journal of the Chinese Institute of Industrial Engineers*, 24 (3), 191–199.