ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 2000
Seria: AUTOMATYKA z. 129 Nr kol. 1474

Tadeusz SAWIK
University of Mining & Metallurgy, Kraków, Poland

DISCRETE PROGRAMMING MODEL FOR SCHEDULING SMT LINES[1]

Summary. The paper presents a new mixed integer programming formulation for blocking scheduling of SMT (Surface Mount Technology) lines for printed wiring board assembly. The SMT line consists of several processing stages in series, separated by finite intermediate buffers, where each stage has one or more identical parallel machines. A board which has completed processing on a machine may remain there and block the machine until a downstream machine becomes available for processing. The objective is to determine an assembly schedule for a mix of board types so as to complete the boards in a minimum time. Numerical examples are presented to illustrate applications of the model proposed.

MODEL PROGRAMOWANIA DYSKRETNEGO DO SZEREGOWANIA ZADAŃ W LINIACH MONTAŻU ELEKTRONICZNEGO

Streszczenie. W pracy przedstawiono nowy model programowania dyskretnego do szeregowania operacji montażu powierzchniowego kart elektronicznych w liniach SMT (ang. Surface Mount Technology). Linia SMT zbudowana jest z szeregowo połączonych stadiów rozdzielonych buforami międzyoperacyjnymi, z maszynami równoległymi w niektórych stadiach. Wyrób wykonany w pewnym stadium może blokować maszynę, jeśli wszystkie bufory przed następnym stadium będą zajęte. Należy wyznaczyć najkrótszy harmonogram montażu zadanej partii wyrobów. Przykłady liczbowe ilustrują możliwość zastosowania opracowanego modelu w montażu elektronicznym.

1. Introduction

Surface Mount Technology (SMT) has been widely used for the last decade in the manufacture of printed wiring boards. SMT assembly involves the following basic processes: screen printing of solder paste on the bare board, automated placement of components,

[1]This work was partially supported by AGH project # 10.10.200.81, KBN research grant # 8 T11F 015 13 and Motorola Advanced Technology Center (USA).

robotic or manual placement of large components, and solder reflow. A typical SMT line consists of several assembly stations in series and/or in parallel, separated by finite intermediate buffers. A conveyor system transfers the boards between the stations.

An SMT line is a practical example of a flexible flow line with limited intermediate buffers and parallel machines [6]. The line produces several different board types. Each board must be processed by at most one machine in each stage. A board which has completed processing on a machine in some stage is transferred either directly to an available machine in the next stage or to a buffer ahead of that stage. The problem objective is to determine the shortest production schedule for a mix of boards so as to complete all the boards in a minimum time.

In SMT lines blocking scheduling problem may often arise, e.g. [3]. When no intermediate buffer storage is available the board may remain on the machine and block it until a downstream machine becomes available. This, however, prevents another board from being processed on the blocked machine.

Various configurations of SMT lines can be encountered in the electronics assembly. For example, the single-pass lines, where one-pass through the line is required to complete a board or the double-pass reentrant lines, where the double-sided boards run twice through the same line, first to assemble the bottom side and then to assemble the top side, [2, 8].

Integer programming formulations have been widely used to express the assembly line design and balancing problems (e.g. [1, 5]). Their application, however, in scheduling flexible flow lines such as SMT lines is very limited. This paper provides the reader with a mixed integer programming formulation for scheduling flexible flow lines with finite capacity buffers. The formulation can be applied for constructing the optimal blocking schedules by using commercially available software for mixed integer programming. This has been illustrated in the paper with numerical examples. The example problems have been modelled using an advanced algebraic modelling language AMPL with CPLEX solver that runs on Windows platform.

The paper is organized as follows. In the next section mixed integer programming formulation is presented for blocking scheduling of a flexible flow line. Numerical examples and some computational results are provided in Section 3, and conclusions are given in the last section.

2. Mixed integer program for scheduling flexible flow lines with blocking

In this section a mixed integer programming model is presented for blocking scheduling of a flexible flow line with limited intermediate buffers.

A unified modelling approach is adopted with the buffers viewed as machines with zero processing times. As a result the scheduling problem with buffers can be converted into one with no buffers but with blocking, e.g., [4, 7]. The blocking time of a machine with zero processing time denotes board waiting time in the buffer represented by that machine. We assume that each board must be processed in all stages, including the buffer stages. However, zero blocking time in a buffer stage indicates that the corresponding board does not need to wait in the buffer. Let us note that for each buffer stage board completion time is equal to its departure time from the previous stage since the processing time is zero.

Notation used to formulate the problems is shown in Table 1, where buffers and machines are jointly called processors.

The flexible flow line under study consists of m processing stages in series. Each stage i, $(i = 1, \ldots, m)$ is made up of $n_i \geqslant 1$ identical parallel processors. The system produces v boards of various types. Each board must be processed without preemption on exactly one processor in each of the stages sequentially. That is, each board must be processed in stage 1 through stage m in that order. The order of processing the boards in every stage is identical and determined by an input sequence in which the boards enter the line, i.e., a so-called permutation flowshop is considered.

Let $p_{ik} \geqslant 0$ be the processing time in stage i of board k, $(k = 1, \ldots, v)$. For every board k denote by c_{ik} its completion time in each stage i, and by d_{ik} its departure time from stage i.

Processing without preemption indicates that board k completed in stage i at time c_{ik} had started its processing in that stage at time $c_{ik} - p_{ik}$. Board k completed in stage i at time c_{ik} departs at time $d_{ik} \geqslant c_{ik}$ to an available processor in the next stage $i + 1$. If at time c_{ik} all n_{i+1} processors in stage $i + 1$ are occupied, then the processor in stage i is blocked by board k until time $d_{ik} = c_{i+1k} - p_{i+1k}$ when board k starts processing on an available processor in stage $i + 1$.

The objective is to determine an assignment of boards to processors in each stage over a scheduling horizon in such a way as to complete all the boards in a minimum time, that is, to minimize the makespan $C_{max} = \max_{k \in K}(c_{mk})$, where c_{mk} denotes the completion time of board k in the last stage m.

Table 1

Notation

		Indices
i	=	processing stage, $i \in I = \{1, \ldots, m\}$
j	=	processor in stage i, $j \in J_i = \{1, \ldots, n_i\}$
k	=	board, $k \in K = \{1, \ldots, v\}$
		Input parameters
m	=	number of processing stages
n_i	=	$\lvert J_i \rvert$ – number of parallel processors in stage i
p_{ik}	=	processing time for board k in stage i
v	=	number of boards
Q	=	a large number not less than schedule length
		Decision variables
C_{max}	=	schedule length
c_{ik}	=	completion time of board k in stage i
d_{ik}	=	departure time of board k from stage i
x_{ijk}	=	1, if board k is assigned to processor $j \in J_i$ in stage $i \in I$; otherwise $x_{ijk} = 0$
y_{kl}	=	1, if board k precedes board l; otherwise $y_{kl} = 0$

The mixed integer program for scheduling flexible flow line with blocking is presented below.

Model FF: *Scheduling flexible flow line with limited intermediate buffers*

Minimize

$$C_{max} \tag{1}$$

subject to

Assignment constraints for stages with parallel processors

$$\sum_{j \in J_i} x_{ijk} = 1;\ i \in I, k \in K : |J_i| > 1 \tag{2}$$

$$\sum_{k \in K} p_{ik} x_{ijk} \leqslant \sum_{k \in K} p_{ik}/|J_i| + \min_{k \in K}(p_{ik});\ i \in I, j \in J_i : |J_i| > 1 \tag{3}$$

Board completion constraints

$$c_{1l} \geqslant p_{1l} + \sum_{k \in K: k<l} p_{1k} y_{kl} + \sum_{k \in K: k>l} p_{1k}(1 - y_{lk});\ l \in K : |J_1| = 1 \tag{4}$$

$$c_{1k} \geqslant p_{1k};\ k \in K : |J_1| > 1 \tag{5}$$

$$c_{ik} - c_{i-1k} \geqslant p_{ik};\ i \in I,\ k \in K : i > 1 \tag{6}$$

Board non-interference constraints for stages with single processors

$$c_{ik} + Q y_{kl} \geqslant d_{il} + p_{ik};\ i \in I, k, l \in K : k < l \text{ and } |J_i| = 1 \tag{7}$$

$$c_{il} + Q(1 - y_{kl}) \geqslant d_{ik} + p_{il};\ i \in I, k, l \in K : k < l \text{ and } |J_i| = 1 \tag{8}$$

Board non-interference constraints for stages with parallel processors

$$c_{ik} + Q(2 + y_{kl} - x_{ijk} - x_{ijl}) \geqslant d_{il} + p_{ik};\ i \in I, j \in J_i, k, l \in K : k < l \text{ and } |J_i| > 1 \tag{9}$$

$$c_{il} + Q(3 - y_{kl} - x_{ijk} - x_{ijl}) \geqslant d_{ik} + p_{il};\ i \in I, j \in J_i, k, l \in K : k < l \text{ and } |J_i| > 1 \tag{10}$$

No-store constraints

$$c_{ik} = d_{i-1k} + p_{ik};\ i \in I, k \in K : i > 1 \tag{11}$$

Completion time constraints

$$c_{mk} = d_{mk};\ k \in K \tag{12}$$

$$c_{mk} \leqslant C_{max};\ k \in K \tag{13}$$

$$C_{max} \geqslant \sum_{k \in K} p_{ik}/|J_i| + \sum_{h \in I: h \neq i} min_{k \in K}(p_{hk});\ i \in I \tag{14}$$

Variable elimination constraints

$$x_{ijk} = 0;\ i \in I, j \in J_i, k \in K : |J_i| = 1 \tag{15}$$

$$y_{kl} = 0;\ k, l \in K : k \geqslant l \tag{16}$$

Variable nonnegativity and integrality constraints

$$c_{ik} \geqslant 0;\ i \in I, k \in K \tag{17}$$

$$d_{ik} \geqslant 0;\ i \in I, k \in K \tag{18}$$

$$x_{ijk} \in \{0, 1\};\ i \in I, j \in J_i, k \in K \tag{19}$$

$$y_{kl} \in \{0, 1\};\ k, l \in K \tag{20}$$

The objective function (1) represents the schedule length to be minimized. Constraint (2) ensures that in every stage with parallel processors each board is assigned to exactly one processor and (3) equalizes in every stage the workload assigned to each parallel processor. Constraint (4) or (5) ensures that each board is processed in the first stage, and (6) guarantees that it is also processed in all downstream stages. Constraints (7) and (8) are board non-interference constraints for single processors, and (9) and (10) for parallel processors. No two boards can be performed on the same processor simultaneously. For a given sequence of boards only one constraint of each pair (7) and (8) or (9) and (10) is active, and only if both boards k and l are assigned to the same processor. Equation (11) indicates that processing of each board in every stage starts immediately after its departure from the previous stage. Equation (12) ensures that each board leaves the line as soon as it is completed in the last stage. Finally (13) defines the maximum completion time, and (14) imposes a lower bound on it.

Model **FF** for scheduling flexible flow line with limited intermediate buffers is a general formulation and includes various special cases [7]. For example, if $|J_i| = 1,\ \forall\ i \in I$ model **FF** reduces to scheduling flowshop with single processors, including buffers, and if $p_{ik} > 0,\ \forall i \in I,\ k \in K$ model **FF** can be applied for scheduling flexible flow line with no in-process buffers.

Model **FF** can also be applied for scheduling reentrant flow lines where a board visits a set of stages more than once, e.g. [7]. In order to extend model **FF** for scheduling a double-pass reentrant line, the number of boards is doubled to $2v$. A pair of boards $(k, k+v),\ k = 1, \ldots, v$ represents the bottom and the top side of board k. The release time for board $k+v$ cannot be less than the completion time of board k, i.e., additional board completion constraints should be added for each board $k+v,\ k = 1, \ldots, v$

$$c_{1,k+v} \geqslant c_{m,k} + p_{1,k+v};\ k = 1, \ldots, v$$

3. Numerical examples

In this section numerical examples are presented to illustrate application of the model proposed.

The SMT line configuration for the example is provided in Fig. 1. The line consists of

$m = 5$ stages, where stage $i = 1$ is a screen printer, each stage $i = 3, 5$ represents 2 parallel machines for automatic placement of components and each stage $i = 2, 4$ represents 2 intermediate buffers .

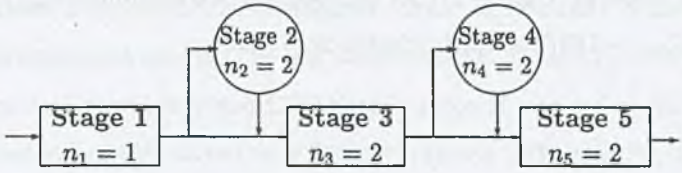

Fig. 1. An SMT line with parallel machines and in-process buffers
Rys. 1. Linia SMT z maszynami równoległymi i buforami międzyoperacyjnymi

The production batch consists of 3 board types, and the processing times p_{ik} for the boards are shown below (for the buffer stages $i = 2, 4$ all processing times are equal to zero)

$$[p_{ik}] = \begin{bmatrix} 10, & 10, & 10 \\ 0, & 0, & 0 \\ 56, & 59, & 74 \\ 0, & 0, & 0 \\ 53, & 54, & 55 \end{bmatrix}$$

The assembly schedules were determined for the following 3 cases:

- Unit-size batch scheduling, where only one board of each type is assembled.
- Cyclic scheduling, where 10 boards of each type are assembled and the boards of different types are scheduled alternately in a cyclic order. The optimal cycle of board types is obtained along with the optimal schedule for all boards.
- Batch scheduling, where 10 boards of each type are assembled and the boards of a given type are scheduled consecutively. The optimal sequence of board types is obtained along with the optimal schedule for all boards.

The lower bounds LBC_{max}, (14) on makespan for the example problems are shown below for scheduling 1 or 10 boards of each type, respectively

$$LBC_{max} = \max_{i \in I} \{ \sum_{k \in K} p_{ik}/n_i + \sum_{h \in I: h \neq i} min_{k \in K}(p_{hk}) \} = 157.5$$

$$LBC_{max} = \max_{i \in I}\{\sum_{k \in K} 10p_{ik}/n_i + \sum_{h \in I: h \neq i} min_{k \in K}(p_{hk})\} = 1008$$

The assembly schedules obtained are shown on Gantt charts in Fig. 2. The solution values obtained are following: $C_{max} = 178$ for unit-size batch scheduling, $C_{max} = 1020$ for cyclic scheduling, $C_{max} = 1027$ for batch scheduling.

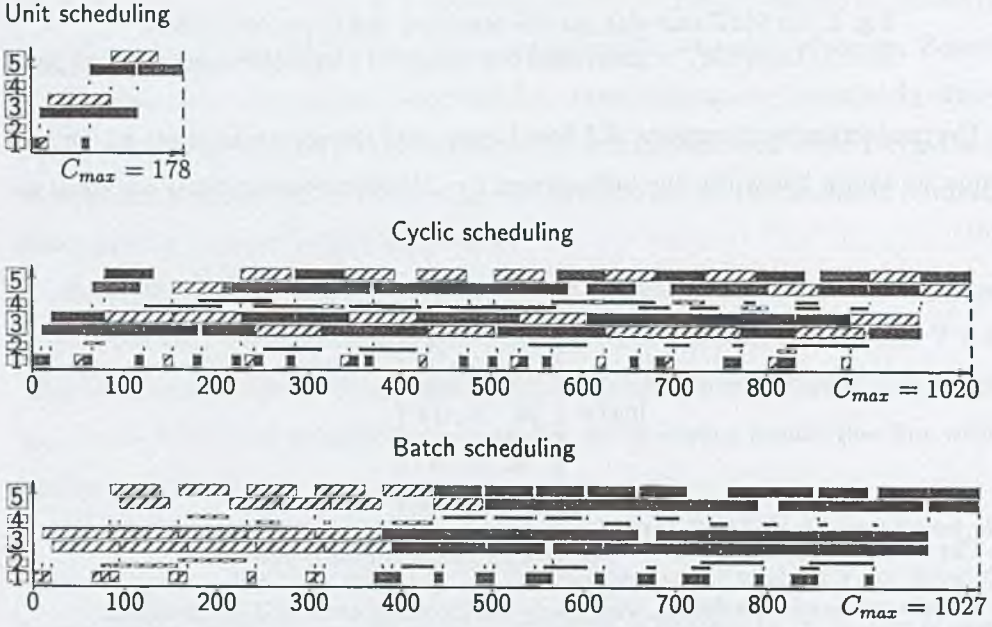

Fig. 2. Assembly schedules
Rys. 2. Harmonogramy montażu

Table 2

Example Characteristics and Solution Results

Problem	Var.	Bin.	Cons.	Nonz.	C_{max}	Nodes	CPU [sec]
Unit	58	27	103	363	178	10	0.098
Cyclic	787	486	8311	35544	1020	4314	1807
Batch	841	540	8581	37164	1027	27301	2035

The characteristics of mixed integer programs for the example problems and the solution results are summarized in Table 2. The size of mixed integer programming models for the example problems is represented by the total number of variables, *Var.*, number of binary variables, *Bin.*, number of constraints, *Cons.*, and number of nonzero coefficients, *Nonz.*, in the constraint matrix. The last two columns of Table 2 give the number of nodes in the branch-and-bound tree and CPU time in seconds required to find the optimal solution. The examples were solved on a Compaq Presario 1830 with Pentium III, 450 MHz using AMPL with CPLEX v.6.5.2 solver.

4. Conclusion

The paper shows that mixed integer programming can be used for modelling a hard problem of scheduling flexible flow lines with limited intermediate buffers. In particular, the model proposed can be used for blocking scheduling of SMT lines in electronics assembly. The approach enables various configurations of SMT lines to be modelled, e.g., double-pass lines, double-conveyor lines, etc. The optimal blocking schedules can be found by using commercially available software for discrete programming. Computational experiments with the approach have indicated ([7]) that mixed integer programming can be applied for scheduling printed wiring board assembly.

BIBLIOGRAPHY

1. Ghosh S., Gagnon R.J.: A comprehensive literature review and analysis of the design, balancing and scheduling of assembly systems. International Journal of Production Research, vol. 27, 1989, pp. 637–670.
2. Graves S.C., Meal H.C., Stefek D., Zeghmi A.H.: Scheduling of reentrant flow shops. Journal of Operations Management, vol. 3, 1983, pp. 197–207.
3. Hall N.G., Sriskandarajah C.: A survey of machine scheduling problems with blocking and no-wait in process. Operations Research, vol. 44, 1996, pp. 510–525.
4. McCormick S.T., Pinedo M.L., Shenker S., Wolf B.: Sequencing in an assembly line with blocking to minimize cycle time. Operations Research, vol. 37, 1989, pp. 925–936.
5. Sawik T.: Integer programming models for the design and balancing of flexible assembly systems. Mathematical and Computer Modelling, vol. 21, no. 4, 1995, pp. 1–12.

6. Sawik T.: Production Planning and Scheduling in Flexible Assembly Systems, Springer-Verlag, Berlin, 1999.
7. Sawik T.: Mixed integer programming for scheduling flexible flow lines with limited intermediate buffers. Mathematical and Computer Modelling, 2000, forthcoming.
8. Tirpak T.M.: Design-to-manufacturing information management for electronics assembly, In: M. Shaw, (ed.), Information Based Manufacturing Technology and Management, Kluwer 1999.

Recenzent: Prof. dr hab. inż. M. Zaborowski

Streszczenie

W pracy przedstawiono nowy model programowania całkowitoliczbowego mieszanego do szeregowania operacji montażu powierzchniowego kart elektronicznych w liniach SMT (ang. Surface Mount Technology). Linia SMT zbudowana jest z szeregowo połączonych stadiów rozdzielonych buforami międzyoperacyjnymi, z maszynami równoległymi w niektórych stadiach. Każdy wyrób (karta elektroniczna) przechodzi przez wszystkie stadia. Wyrób wykonany w pewnym stadium może blokować maszynę oczekując na zwolnienie bufora przed następnym stadium. W modelu matematycznym bufory traktowane są jako dodatkowe maszyny z zerowymi czasami wykonywania wyrobów, lecz z możliwością blokowania. Blokowanie takiej maszyny oznacza oczekiwanie przez wyrób w buforze. Należy wyznaczyć najkrótszy harmonogram montażu dla zadanej partii różnych typów wyrobów. Opracowany model może być również zastosowany do szeregowania zadań w przypadku montażu dwustronnych kart elektronicznych, wymagających dwukrotnego przejścia przez linię. Zamieszczono przykłady liczbowe ilustrujące zastosowania opracowanego modelu matematycznego do wyznaczania harmonogramów montażu. Do obliczeń użyto pakietu programowania dyskretnego AMPL/CPLEX v.6.5.2. Otrzymane wyniki wskazują na możliwość zastosowania modelu do harmonogramowania montażu elektronicznego w liniach SMT.