ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 2002
Seria: AUTOMATYKA z. 134 Nr kol. 1554

Tadeusz SAWIK
Akademia Górniczo-Hutnicza

SCHEDULING BATCHES OF PRINTED WIRING BOARDS IN SURFACE MOUNT TECHNOLOGY LINES[1]

Summary. The paper presents a mixed integer programming approach for batch scheduling of printed wiring board assembly in surface mount technology (SMT) lines. A typical SMT line consists of several assembly stations in series and/or in parallel, separated by finite intermediate buffers. The problem objective is to minimize makespan of an assembly schedule for a mix of board types, where identical boards are scheduled consecutively. Numerical examples modeled after real-world SMT lines illustrate the approach.

SZEREGOWANIE PARTII WYROBÓW ELEKTRONICZNYCH W LINIACH MONTAŻU POWIERZCHNIOWEGO

Streszczenie. W pracy przedstawiono model programowania całkowitoliczbowego mieszanego do harmonogramowania montażu powierzchniowego partii wyrobów elektronicznych w liniach SMT (ang. Surface Mount Technology). Linia SMT zbudowana jest z szeregowo połączonych stadiów z maszynami równoległymi i buforami międzyoperacyjnymi. Należy wyznaczyć najkrótszy harmonogram montażu wielu partii różnych typów wyrobów, w którym wyroby jednego typu montowane są kolejno. Zastosowania opracowanego modelu ilustrują przykłady liczbowe oparte na rzeczywistych danych z przemysłu elektronicznego.

1. Introduction

Printed wiring board assembly is typically performed on an automated Surface Mount Technology (SMT) line which includes three different processes in the following sequence: solder printing, component placement and solder reflow. A typical SMT line consists of several assembly stations in series and/or in parallel, separated by finite intermediate buffers and connected with a conveyor system that transfers the boards between the stations, see [6].

[1] This work was partially supported by AGH and KBN (Poland) and by the Motorola Advanced Technology Center(USA)

An SMT line is a practical example of a flexible flow line with limited intermediate buffers and parallel machines, e.g. [4]. The limited intermediate buffers result in a *blocking scheduling* problem, e.g. [2, 3], where a completed board may remain on a machine and block it until a downstream machine becomes available.

In practice scheduling of SMT line is based on daily demands and a simple approach to executing daily production plan is the use of batch scheduling, where boards of one type are scheduled consecutively. In a high-volume production, however, the production plan is often split into several identical sets of smaller batches of boards that are scheduled repeatedly. The smallest possible set of boards in the same proportion as the daily board mix requirements is called Minimal Part Set (MPS), e.g. [1].

Research on scheduling algorithms for flexible assembly lines with finite capacity buffers is mostly restricted to heuristics which seek good solutions within reasonable computation times, e.g. [3]. This paper however, provides the reader with an exact mixed integer programming formulation for batch scheduling of printed wiring board assembly in SMT lines, e.g. [4, 5, 6]. The formulation proposed is capable of finding optimal batch schedules for various SMT line configurations by using commercially available software for mixed integer programming.

The paper is organized as follows. In the next section a mixed integer programming formulation is presented for batch scheduling in a flexible assembly line with machine blocking. Numerical examples modeled after real-world SMT lines and some computational results are provided in Section 3, and conclusions are given in the last section.

2. Mixed integer program for batch scheduling

In this section a mixed integer programming model is presented for batch scheduling in a flexible assembly line with limited intermediate buffers.

A unified modeling 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. [3, 4].

Notation used to formulate the problem is shown in Table 1, where buffers and machines are referred to as processors.

The flexible assembly line under study consists of m processing stages in series. Each stage i, $(i = 1, \dots, m)$ is made up of $m_i \geq 1$ identical parallel processors. Let J_i be the circular set of indices of parallel processors at stage i. The system produces various types of boards. Let $G = \{1, \dots, p\}$, $K = \{1, \dots, n\}$, and $K_g = \{\sum_{f \in G: f<g} b_f + 1, \dots, \sum_{f \in G: f<g} b_f + b_g\}$ be the ordered sets of indices, respectively of all batches of boards, all individual boards, and all boards of type $g \in G$, (b_g, $n = \sum_{g=1}^{p} b_g$, and p denote, respectively the number of boards of type g, the total number of boards, and the number of batches in the schedule.)

All boards are scheduled in batches of boards of the same type and within the batch individual boards are processed consecutively. No setups are required between different boards or different batches of boards. Each board must be processed without preemption on exactly one processor in each of the stages sequentially. The

Table 1

Notation

		Input parameters
b_g	=	size of batch g (number of boards of type g)
m	=	number of processing stages, $i \in I = \{1, \dots, m\}$
m_i	=	number of parallel processors at stage i, $j \in J_i = \{1, \dots, m_i\}$
n	=	total number of boards, $k \in K = \{1, \dots, n\}$
p	=	number of batches (board types), $g \in G = \{1, \dots, p\}$
r_{ig}	=	processing time at stage i of board type g
		Decision variables
C_{max}	=	schedule length
c_{ik}	=	completion time of board k at stage i
d_{ik}	=	departure time of board k from stage i
x_{ijk}	=	1, if board k is assigned to processor $j \in J_i$ at stage $i \in I$; otherwise $x_{ijk} = 0$
y_{fg}	=	1, if batch f precedes batch g; otherwise $y_{fg} = 0$

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.

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.

Let $r_{ig} \geq 0$ be the processing time at stage i of each board type $g \in G$. Processing without preemption indicates that board $k \in K_g$ completed at stage i at time c_{ik} starts its processing in that stage at time $c_{ik} - r_{ig}$. Board $k \in K_g$ completed at stage i at time c_{ik} departs at time $d_{ik} \geq c_{ik}$ to an available processor in the next stage $i + 1$. If at time c_{ik} all m_{i+1} processors at stage $i + 1$ are occupied, then the processor at stage i is blocked by board k until time $d_{ik} = c_{i+1k} - r_{i+1g}$ when board $k \in K_g$ starts processing on an available processor at stage $i + 1$.

The objective is to determine an input sequence of batches of boards and an assignment of boards to processors in each stage over a scheduling horizon to complete all the boards in 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.

The mixed integer program for batch scheduling in a flexible assembly line with finite in-process buffers is presented below.

Minimize

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

subject to

Board assignment constraints

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

$$x_{i,next(j,J_i),k+1} = x_{ijk}; \; i \in I, j \in J_i, g \in G, k \in K_g : k < last(K_g), m_i > 1 \tag{3}$$

Board completion constraints

$$c_{1k} \geq r_{1g};\ g \in G, k \in K_g \tag{4}$$

$$c_{ik} - c_{i-1k} \geq r_{ig};\ i \in I, g \in G, k \in K_g : i > 1 \tag{5}$$

Board departure constraints

$$c_{ik} \leq d_{ik};\ i \in I, k \in K : i < m \tag{6}$$

$$c_{mk} = d_{mk};\ k \in K \tag{7}$$

Board non-interference constraints

$$c_{ik} + (Q_{ifg} + H_{ifk})(2 + y_{fg} - x_{ijk} - x_{ijlast(K_g)}) \geq d_{ilast(K_g)} + r_{if} + H_{ifk};$$
$$i \in I, j \in J_i, f, g \in G, k \in K_f : f < g \tag{8}$$

$$c_{il} + (Q_{igf} + H_{igl})(3 - y_{fg} - x_{ijlast(K_f)} - x_{ijl}) \geq d_{ilast(K_f)} + r_{ig} + H_{igl};$$
$$i \in I, j \in J_i, f, g \in G, l \in K_g : f < g \tag{9}$$

$$c_{ifirst(K_f)} + (Q_{ifg} + T_{igl})(2 + y_{fg} - x_{ijfirst(K_f)} - x_{ijl}) \geq d_{il} + r_{if} + T_{igl};$$
$$i \in I, j \in J_i, f, g \in G, l \in K_g : f < g \tag{10}$$

$$c_{ifirst(K_g)} + (Q_{igf} + T_{ifk})(3 - y_{fg} - x_{ijk} - x_{ijfirst(K_g)}) \geq d_{ik} + r_{ig} + T_{ifk};$$
$$i \in I, j \in J_i, f, g \in G, k \in K_f : f < g \tag{11}$$

Buffering constraints

$$c_{ik} = d_{i-1k} + r_{ig};\ i \in I, g \in G, k \in K_g : i > 1 \tag{12}$$

Maximum completion time constraints

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

$$d_{ik} + \sum_{h \in I : h > i} r_{hg} \leq C_{max};\ i \in I, g \in G, k \in K_g : i < m \tag{14}$$

$$c_{ml} - c_{1l} \leq C_{max} - \sum_{f \in G : f < g} b_f r_{1f} y_{fg} - \sum_{f \in G : f > g} b_f r_{1f}(1 - y_{gf}) - (l - \sum_{f=1}^{g-1} b_f) r_{1g}$$
$$-(\sum_{f=1}^{g} b_f - l) r_{mg} - \sum_{f \in G : f < g} b_f r_{mf}(1 - y_{fg}) - \sum_{f \in G : f > g} b_f r_{mf} y_{gf};$$
$$g \in G, l \in K_g : m_1 = 1, m_m = 1 \tag{15}$$

Batch processing constraints

$$c_{ik+m_i} \geq d_{ik} + r_{ig};\ i \in I, g \in G, k \in K_g : k + m_i \leq last(K_g), m_i > 1 \tag{16}$$

$$c_{ik+1} \geq c_{ik};\ i \in I, g \in G, k \in K_g : k < last(K_g), m_i > 1 \tag{17}$$

$$c_{ik+1} \geq d_{ik} + r_{ig};\ i \in I, g \in G, k \in K_g : k < last(K_g), m_i = 1 \tag{18}$$

Variable elimination constraints

$$y_{fg} = 0;\ k, l \in K : f \geq g \tag{19}$$

Variable nonnegativity and integrality constraints

$$c_{ik} \geq 0;\ i \in I, k \in K \tag{20}$$

$$d_{ik} \geq 0;\ i \in I, k \in K \tag{21}$$

$$x_{ijk} \in \{0,1\};\ i \in I, j \in J_i, k \in K \tag{22}$$

$$y_{fg} \in \{0,1\};\ f, g \in G \tag{23}$$

The objective function (1) represents the schedule length to be minimized. Assignment constraint (2) ensures that in every stage each board is assigned to exactly one processor, and (3) assigns successive boards of one type alternatively to different parallel processors ($next(j, J_i)$ is the next processor after $j \in J_i$ in the circular set J_i of parallel processors at stage i). Constraint (4) ensures that each board is processed in the first stage, and (5) guarantees that it is also processed in all downstream stages. Constraint (6) indicates that each board cannot be departed from a stage until it is completed in this stage, and equation (7) ensures that each board leaves the line as soon as it is completed in the last stage. Constraints (8),(9),(10) and (11) are board non-interference constraints. No two boards can be performed on the same processor simultaneously. For a given sequence of batches only one constraint (8) or (9) is active, and only if both boards $k \in K_f$ and $last(K_g)$ or $l \in K_g$ and $last(K_f)$ are assigned to the same processor. Likewise, either (10) or (11) is active, and only if both boards $l \in K_g$ and $first(K_f)$ or $k \in K_f$ and $first(K_g)$ are assigned to the same processor. Equation (12) indicates that processing of each board in every stage starts immediately after its departure from the previous stage. Constraint (13) defines the maximum completion time of all boards. Constraint (14) relates board departure times to makespan directly. Every board must be departed from a stage sufficiently early in order to have all of its remaining tasks completed within the remaining processing time. Constraint (15) ensures that each board is processed within the time interval remaining after processing of all preceding boards and before processing of all succeeding boards. Flow time $c_{ml} - (c_{1l} - r_{1g})$ of each board $l \in K_g$ cannot be greater than the makespan C_{max} minus sum of processing times of all preceding boards in the first stage

$$\sum_{f \in G: f<g} b_f r_{1f} y_{fg} + \sum_{f \in G: f>g} b_f r_{1f}(1 - y_{gf}) + (l - 1 - \sum_{f=1}^{g-1} b_f) r_{1g},$$

and sum of processing times of all succeeding boards in the last stage

$$(\sum_{f=1}^{g} b_f - l) r_{mg} + \sum_{f \in G: f<g} b_f r_{mf}(1 - y_{fg}) + \sum_{f \in G: f>g} b_f r_{mf} y_{gf}.$$

Batch processing constraints (16),(17) along with (3) ensure that boards of one type are processed consecutively in each stage with parallel processors, whereas consecutive processing of identical boards in each stage with a single processor is imposed by (18).

Parameters H_{ifk}, T_{ifk} and Q_{ifg} in constraints (8)–(11) are calculated as below.

$$H_{ifk} = \max\{0, \lfloor (k - \sum_{g=1}^{f-1} b_g - m_i)/m_i \rfloor\} r_{if}; i \in I, f \in G, k \in K_f \tag{24}$$

$$T_{ifk} = \max\{0, \lfloor(\sum_{g=1}^{f} b_g - k - m_i + 1)/m_i\rfloor\} r_{if}; i \in I, f \in G, k \in K_f \qquad (25)$$

$$Q_{ifg} = \sum_{i\in I}\sum_{g\in G} b_g r_{ig}/m_i - \sum_{h\in I:h<i} r_{hf} - \sum_{h\in I:h>i} r_{hg};\ i \in I, f, g \in G \qquad (26)$$

where H_{ifk} and T_{ifk} denote respectively, head and tail of board $k \in K_f$ in batch f at stage i, and Q_{ifg} is a large number not less than the schedule length calculated for stage i when batch f precedes batch g.

The proposed mixed integer program includes various cutting constraints that were identified exploiting typical SMT line configurations and some properties of batch processing on parallel machines. The constraints may significantly reduce computational effort required to find the optimal solution.

3. Numerical examples

In this section numerical examples are presented, and some computational results are reported to illustrate the mixed integer programming approach. The examples are modeled after real-world SMT lines [6]. The assembly schedules for the examples were calculated on a laptop Compaq Presario 1830 with Pentium III, 450 MHz using AMPL modeling language and CPLEX v.7.1 solver.

3.1. Example 1: Factory with single stations

The SMT line configuration for Example 1 is shown in Figure 1. The line consists of a loader, screen printer, four placement machines and a vision inspection machine, in series separated by intermediate buffers.

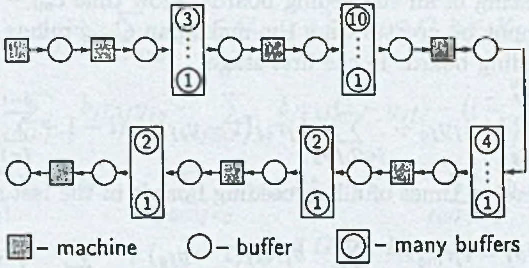

Fig. 1. Factory with single stations

Rys. 1. Linia z pojedynczymi maszynami

The line represents a typical low-volume, medium-variety production system. For the industry scenario that was studied, 13 different board types are assembled in small to medium size batches. Table 2 lists the processing times for boards, and Table 3 presents the input data for selected problem instances that represent five daily production orders.

Table 2

Example 1: Processing times in seconds

Board type	Processing stage						
	1	3	7	11	15	19	23
1	20	25	123	45	38	62	45
2	20	25	155	156	28	58	50
3	20	25	67	56	36	35	45
4	20	25	93	95	-	51	40
5	20	25	76	111	41	63	50
6	20	25	87	93	52	48	45
7	20	25	34	78	92	55	45
8	20	25	66	28	34	-	30
9	20	25	141	90	49	-	40
10	20	25	86	83	56	22	45
11	20	25	98	84	36	43	45
12	20	25	176	175	76	65	50
13	20	25	-	17	67	28	45

Table 3

Example 1: Input Data

Problem no.	Daily Mix							
	Board type	Batch size	Board type	Batch size	Board type	Batch size	Board type	Batch size
1	7	13	9	6	-	-	-	-
2	2	23	9	1	-	-	-	-
3	7	2	11	66	-	-	-	-
4	5	34	7	2	8	22	9	2
5	1	42	4	2	7	4	10	14

The characteristics of mixed integer programs for the example problems and the solution results are summarized in Table 4. The size of the mixed integer programs 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

Table 4

Example 1: Computational Results

Problem	Var.	Bin.	Cons.	Nonz.	LB	C^*_{max}	Nodes	CPU**
1	1085	590	2687	9649	1722	1722	13	1.4
2	1370	745	3442	12284	3953	3967	0	1.1
3	3878	2109	9861	35022	6789	6789	0	9.9
4	3427	1866	17964	73788	4869	5016	12	38
5	3541	1928	18579	76280	6923	6925	1	51

* optimal makespan, ** CPU time for proving optimality

nonzero coefficients, *Nonz.*, in the constraint matrix. The last four columns of Table 4 present the lower bound LB on the makespan, the optimal makespan C_{max}, the node number in the branch-and-bound tree at which the optimal solution was found, and CPU time in seconds required to prove optimality of the solution. The lower bound was calculated as below.

$$LB = \max_{i \in I}\{\sum_{g \in G} b_g r_{ig}/m_i + min_{g \in g}(\sum_{h \in I: h<i} r_{hg}) + min_{g \in G}(\sum_{h \in I: h>i} r_{hg})\} \qquad (27)$$

3.2. Example 2: Factory with parallel stations

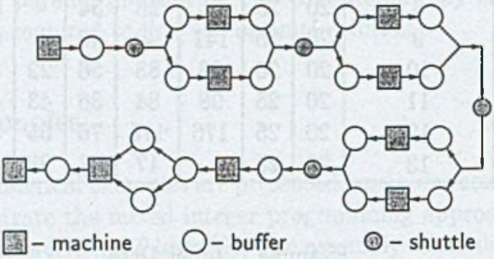

Fig. 2. Factory with parallel stations

Rys. 2. Linia z maszynami równoległymi

Table 5

Example 2: Processing times in seconds

Board type	Processing stage						
	1	5	9	13	17	19	21
1	22	207	213	204	80	40	62
2	22	208	220	204	80	40	62
3	22	207	224	191	80	40	62
4	22	207	213	204	80	40	62
5	22	207	220	204	80	40	62
6	22	184	196	199	80	40	62

The SMT line configuration for Example 2 is shown in Figure 2. The line consists of a screen printer, three sets of two parallel placement machines and four shuttles routing the boards to the next available placement machine, a vision inspection machine and a single placement machine, in series separated by intermediate buffers. The line represents a typical high-volume, low-variety production system, in which six different board types are produced in medium to large size batches. Table 5 lists the processing times for boards, and Table 6 presents the input data for selected problem instances that represent five daily production orders and the corresponding minimum part sets (MPS). The characteristics of mixed integer programs for the MPS problems and the solution results are summarized in Table 7. Figure 3

Table 6

Example 2: Input Data

Problem no.	Daily Mix/MPS							
	Board type	Batch size	Board type	Batch size	Board type	Batch size	Board type	Batch size
1	3	240/6	4	200/5	5	480/12	-	-
2	1	80/2	2	120/3	3	240/6	5	480/12
3	1	180/6	2	210/7	3	510/17	-	-
4	3	300/3	4	400/4	5	500/5	-	-
5	3	1080/27	6	400/10	-	-	-	-

Table 7

Example 2: Computational Results

Problem	Var.	Bin.	Cons.	Nonz.	*LB*	C^*_{max}	Nodes	CPU**
1	1269	670	4778	18250	3127	3233	20	20
2	1272	673	6265	25433	3137	3247	46	200
3	1654	873	6276	23892	3915	3993	8	26
4	664	351	2432	9400	1914	1992	21	1.6
5	2037	1074	5380	17975	4583	4695	11	5.4

* optimal makespan, ** CPU time for proving optimality

shows Gantt chart with the optimal batch schedule obtained for Problem 2, where letter B stands for Buffer and M stands for Machine for screen printing, component placement, or vision inspection. Buffering or machine blocking is indicated with a narrow bar. The input sequence of board types is 5,2,1,3, and the optimal makespan $C_{max} = 3247$.

4. Conclusion

This paper has presented an exact approach for batch scheduling in flexible assembly lines with limited intermediate buffers. The approach based on a mixed integer programming formulation is capable of optimal scheduling SMT lines by using commercially available software for integer programming. The computation time has been reduced by introducing various cutting constraints exploiting SMT line configurations and consecutive processing of identical boards as well as a specific MPS scheduling mode. The approach can be applied to a variety of different real-world assembly line configurations and production scenarios with only small modifications to the constraint formulations and input data definitions, see [5, 6].

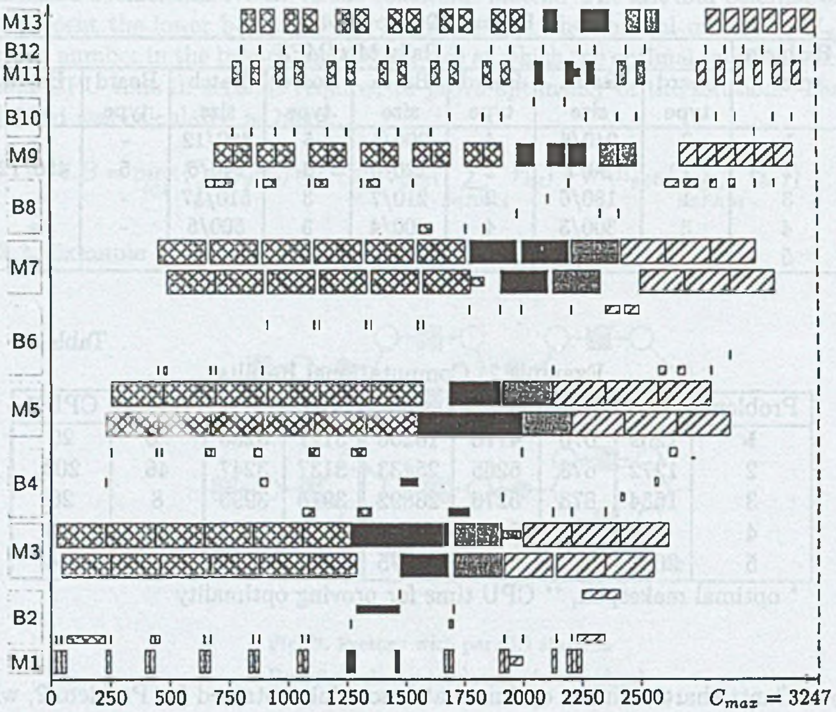

Fig. 3. Batch schedule for SMT line with parallel stations

Rys. 3. Uszeregowanie partii wyrobów dla linii z maszynami równoległymi

REFERENCES

1. Deane R.H. and Moon S.H.: Work flow control in the flexible flow line. *International Journal of Flexible Manufacturing Systems*, no.3/4, 1992, pp.217-235.
2. Hall N.G. and Sriskandarajah C.: A survey of machine scheduling problems with blocking and no-wait in process. *Operations Research*, vol.44, 1996, pp.510–525.
3. McCormick, S.T., Pinedo, M.L., Shenker, S. and Wolf, B.: Sequencing in an assembly line with blocking to minimize cycle time. *Operations Research*, vol.37, 1989, pp.925–936.
4. Sawik T.: Mixed integer programming for scheduling flexible flow lines with limited intermediate buffers, *Mathematical and Computer Modelling*, vol. 31, 2000, pp.39–52.
5. Sawik T.: Mixed integer programming for scheduling surface mount technology lines. *International Journal of Production Research*, vol. 39, 2001, pp. 3219–3235.
6. Sawik T., Schaller A. and Tirpak T.M.: Scheduling of printed wiring board assembly in surface mount technology lines. *Journal of Electronics Manufacturing*,

special issue on *Production Planning and Scheduling in Electronics Manufacturing*, vol. 11, no.1, 2002, pp.1–17.

Recenzent: Dr hab.inż. Mirosław Zaborowski, Prof. Pol. Śl.

Streszczenie

W pracy przedstawiono 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 o ograniczonych pojemnościach, z maszynami równoległymi w niektórych stadiach. Wyrób wykonany w pewnym stadium może blokować maszynę oczekując na zwolnienie bufora przed następnym stadium. Należy wyznaczyć najkrótszy harmonogram montażu wielu partii różnych typów wyrobów, gdzie wyroby jednego typu są montowane kolejno, jeden po drugim. 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. Do modelu matematycznego wprowadzono także różne ograniczenia odcinające, które wyznaczono analizując typowe konfiguracje linii SMT oraz pewne właściwości uszeregowań partii wyrobów na maszynach równoległych. Zamieszczone przykłady liczbowe z rzeczywistymi danymi zaczerpniętymi z przemysłu elektronicznego ilustrują możliwość zastosowania opracowanego modelu matematycznego do wyznaczania harmonogramów produkcji w elastycznych liniach montażowych.