Scheduling of printed wiring board assembly in surface mount technology lines

Article in Journal of Electronics Manufacturing · March 2002

DOI: 10.1142/S096031310200028X

Tadeusz J Sawik
AGH University of Science and Technology in Kraków

Journal of Electronics Manufacturing, Vol. 11, No. 1 (2002) 1–17

SCHEDULING OF PRINTED WIRING BOARD ASSEMBLY IN SURFACE MOUNT TECHNOLOGY LINES

TADEUSZ SAWIK*
University of Mining and Metallurgy,
Department of Computer Integrated Manufacturing,
Al. Mickiewicza 30, 30-059 Kraków, Poland
ghsawik@cyf-kr.edu.pl

ANDREAS SCHALLER
Motorola Advanced Technology Center — Europe,
Hagenauer Str. 44, 65203 Wiesbaden, Germany
Andreas.Schaller@motorola.com

THOMAS M. TIRPAK
Motorola Advanced Technology Center,
1301 East Algonquin Road, Room 1014, Schaumburg, IL 60196, USA
att002@email.mot.com

Received 2 January 2001
Accepted 9 August 2001

The paper presents a new mathematical programming approach for scheduling of printed wiring board assembly in SMT (Surface Mount Technology) lines. Various configurations of SMT lines encountered in the electronics industry are described and compared. An 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 minimum time. New mixed integer programming formulations are presented for blocking scheduling of various configurations of SMT lines that are found in the electronics industry. The proposed models can be used for optimization of assembly schedules by using commercially available software for discrete programming. Numerical examples are provided to illustrate the proposed approach. The influence of process time variability and machine breakdowns on an SMT line's performance is discussed.

Keywords: Electronics manufacturing; flexible assembly lines; blocking scheduling; mixed integer programming.

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, 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.

*Author to whom correspondence is to be sent.

An SMT line is a practical example of a flexible flow line with limited intermediate buffers and parallel machines.[1] The line typically produces several different board types. Each board must be processed by at most one machine in each stage because of different routings for different board types. 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.

Various configurations of SMT lines can be encountered in electronics assembly. For example, there are single-pass lines, where one-pass through the line is required to complete a board or double-pass re-entrant 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.[2,3]

The two major short-term planning problems in electronics assembly are loading and scheduling. Given a mix of boards to be produced, the objective of the loading problem is to allocate assembly tasks and component feeders among the placement stations with limited working space, so as to balance the station workloads.[4] An important issue in printed wiring board assembly is dynamic balancing of SMT lines that accounts for the intermittent availability of machines and variability of processing times, which are caused by vision errors, machine breakdowns and part pick-up failures. In contrast, the objective of the scheduling problem is to determine the detailed sequencing and timing of all assembly tasks for each individual board, so as to maximize the line's productivity, which may be defined in terms of throughput or the assembly schedule length (makespan) for a mix of board types. The limited intermediate buffers between stations result in a *blocking scheduling* problem, where a completed board may remain on a machine and block it until a downstream machine becomes available.[5] This prevents another board from being processed on the blocked machine.

Scheduling with machine blocking has received considerable attention in the study[6] where an m machine flowshop with finite capacity buffers between machines is considered. A variety of scheduling environments with blocking are discussed.[7] In practice, the scheduling decisions are often made based on various heuristic approaches which cannot guarantee the best utilization of system capabilities.[1,8–12]

This paper provides the reader with mixed integer programming formulations for scheduling flexible flow lines with finite capacity buffers. While integer programming formulations have been widely used to express the assembly line design and balancing problems,[13–15] there are few applications for the much more complex problem of scheduling flexible flow lines such as SMT lines.[16–18] For example, an integer programming model for scheduling a flowshop with no buffers is developed in Ref. 19 and Ref. 20 presents a mixed integer program for scheduling a flexible flow line with unlimited buffers; however, no computational results are reported.

The high complexity of an SMT line scheduling problem is mainly caused by:

- Limited buffers that result in machine blocking and require separate board completion and board release time variables to be introduced for each board, machine, and buffer;
- Parallel processors that require additional binary assignment variables to be introduced for each board, machine, and buffer;
- Simultaneous assembly of different board types; and
- Medium to high volume production that requires a significant number of boards to be included in the scheduling horizon.

The formulations presented in this paper can be solved by commercially available software for mixed integer programming and used to construct blocking schedules for real-world scenarios. This has been illustrated in the paper with numerical examples. The example problems have been modelled using the advanced algebraic modeling language AMPL and solved with CPLEX solver running on a Windows platform.

The paper is organized as follows. In the next section, various configurations of SMT lines are described, and their basic characteristics are compared. New mixed integer programming formulations for blocking scheduling of various types of SMT lines are presented in Sec. 3. The formulations are next applied in Sec. 4 to determine optimal schedules for several examples, including an SMT line with parallel stations, and a dual-conveyor line modeled after lines at our industrial partner's factories. The influence of process time variability and machine breakdowns on an SMT line's performance is discussed in Sec. 5, followed by some concluding remarks in the last section.

2. SMT Line Configurations

Printed wiring board (PWB) assembly is typically performed on an automated SMT line which includes three different processes in the following sequence: solder printing, component placement and solder reflow. For the process of solder printing and reflow soldering, one machine per line is needed. The number of machines for the placement process can vary and depends on the number and type of components on the boards to be assembled. Basically, these electronic components can be divided into two major groups: small chip parts and large fine pitch parts. It is assumed that an SMT line contains at least one machine capable of placing each component group. The components can be assembled on one or both sides of a PWB. The manufacturing line can be single- or double-sided, which means that the board may travel once or twice through the same line. In addition, each PWB can be transported and assembled as a set of boards in a panel. The following are the basic SMT line configurations found in electronics assembly factories, see Ref. 21:

1. SMT lines for single-sided boards
   1.1. SMT line with single stations
   1.2. SMT line with parallel stations
   1.3. SMT line with dual-conveyor
2. SMT lines for double-sided boards
   2.1. Single-pass SMT line
   2.2. Double-pass SMT line

2.1. *SMT lines for single-sided boards*

A simple SMT line with single stations is shown in Fig. 1. In this basic configuration, all machines in the SMT line are connected in series. The line consists of a PWB loader, a solder printer, a reflow oven and two placement machines (one for small and one for fine pitch components). The placement machines have to be adjusted to the product running by controlling the conveyor width, installing the proper feeders for components, as well as selecting the nozzle configuration to pick up the required components. Machines are separated by buffers and connected with conveyors. The assembly process is as follows: A tote of bare (pre-assembly) PWBs is brought to the beginning of the line, and a material loader loads each PWB separately on the conveyor. Each PWB is transported by the conveyor system through each machine in the line and is then stored again in a tote box. The loader and the tote box are used as the input and output buffers of the line. There are external buffers in front of and behind each placement machine, except the last one. In addition, every placement machine has its own internal input and output buffers of a fixed capacity. The internal and external buffers are shown in Fig. 1 in gray.

The SMT line with parallel stations in Fig. 2 consists of two parallel placement machines for small

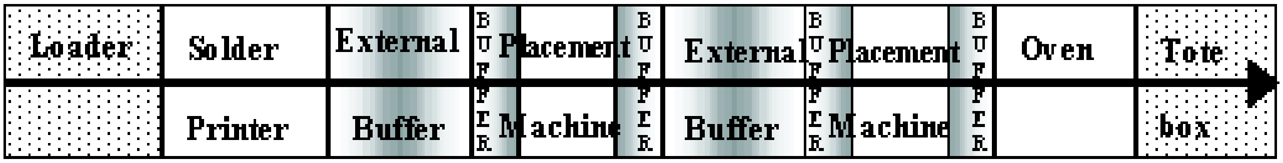

Fig. 1. A simple SMT line with single stations.

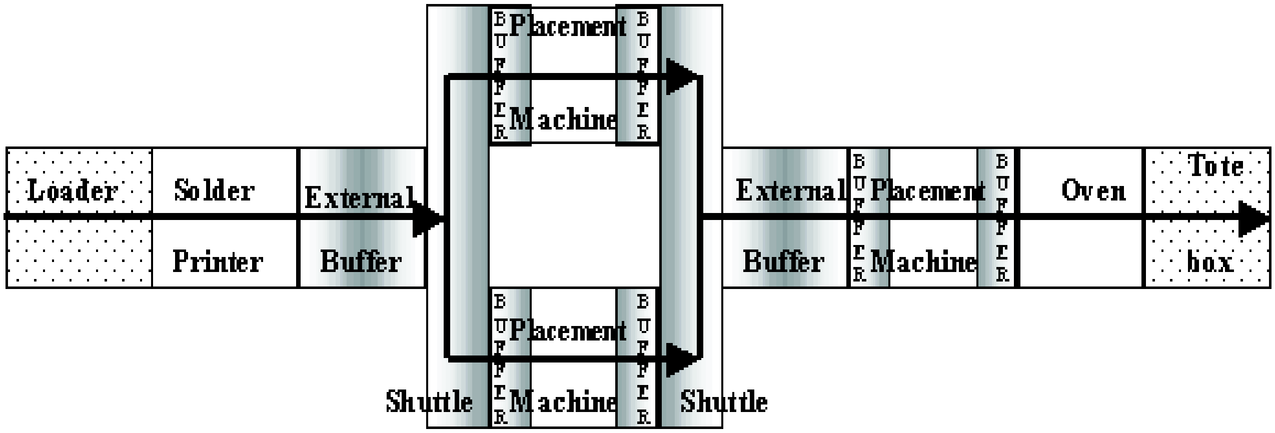

Fig. 2. SMT line with parallel stations.

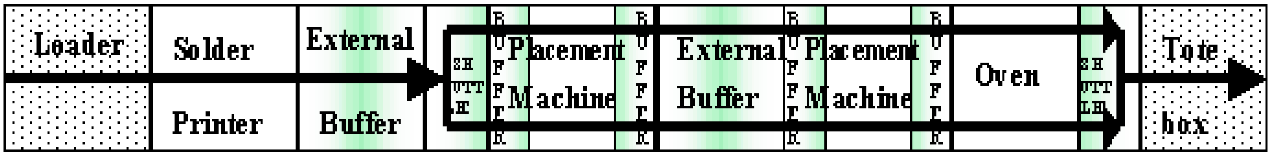

Fig. 3. Dual-conveyor SMT line.

components and two additional shuttles routing the PWB to the next available placement machine. A parallel station in Fig. 2 would assemble twice as many parts as a single station in Fig. 1. Because of that, the non-productive operations of board loading-unloading would represent a smaller fraction of the total assembly time at the station, and the average time per placement would decrease. Therefore, this so called "placement density effect" tends to increase throughput in this line configuration. A line with parallel stations also provides "redundancy" in the system. If a machine breakdown occurs, processing can continue because an alternate routing exists.

To further reduce the effects of load-unload times and achieve higher throughput at each station, a dual-conveyor SMT line (Fig. 3) has been introduced. Each placement machine is equipped with a dual conveyor system that can operate in either synchronous or asynchronous mode. In synchronous mode, two panels are loaded at the same time. Thus, the loading time per panel is halved, and the number of placements in the assembly program is doubled. In asynchronous mode, a second panel can be loaded or unloaded while the first panel is being assembled.

2.2. *SMT lines for double-sided boards*

Generally there are two ways to produce double-sided boards, i.e. using a single-pass (continuous) line or a double-pass (re-entrant) SMT line.

A single-pass SMT line (Fig. 4) consists of two lines linked together by a board flipping station. Each PWB is transported by the conveyor system through the complete line.

After the first side of the PWB is completed in a double-pass (re-entrant) SMT line (Fig. 5), the individual panels return to the front of the line, or the panels get collected in a cassette and then are returned as a batch to the front. At the beginning of the line, the PWB is flipped and inserted in the production flow again. During the second pass, the second side of the PWB is populated with components, and the finished products are collected in a tote and leave the line.

Table 1 gives an overview and comparison of the different SMT line configurations.

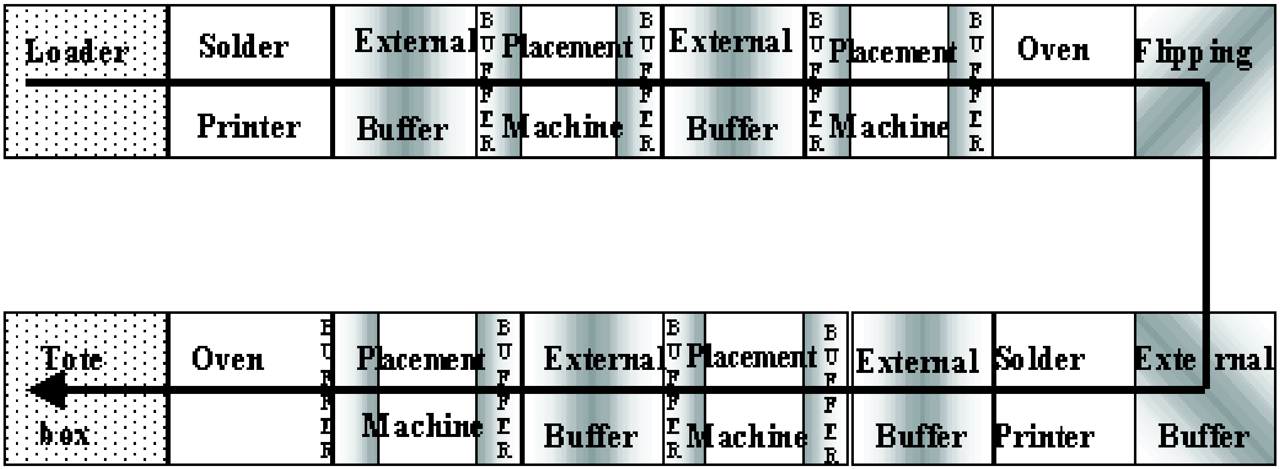

Fig. 4. Single-pass SMT line.

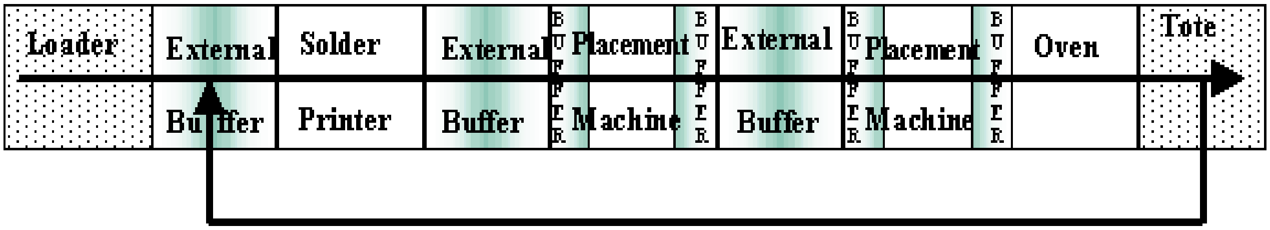

Fig. 5. Double-pass SMT line.

Table 1. Characteristics of various SMT line configurations.

Type of SMT Line	Functional Characteristics	Reliability	Cycle Time	Work in Process	Typical Applications
1.1. SMT line with single stations	Standard configuration	—	—	—	For production of any type of PWB
1.2. SMT line with parallel stations	Placement machines have a parallel configuration	Higher than 1.1	Lower than 1.1 due to placement density effect	Higher than 1.1	For medium volume production (24 hours a day, 7 days a week), with high replenishment and setup times
1.3. SMT line with dual-conveyor	Placement machines are equipped with a dual conveyor system	—	Lower than 1.1 due to elimination of the non-productive transport time	Higher than 1.1	For high volume production (24 hours a day, 7 days a week)
2.1. Single-pass SMT line	A manufacturing line for each side of the PWB	—	—	—	For high volume production (24 hours a day, 7 days a week)
2.1. Double-pass SMT line	PWB has to re-enter the line	Higher than 2.1	Higher than 2.1	Lower than 2.1	For medium volume and high mix production

3. Mixed Integer Programs for Scheduling SMT Lines

In this section, blocking scheduling of a flexible flow line with limited intermediate buffers is formulated as a mixed integer program that addresses the two basic questions:

- What should be the sequence of boards entering the line?
- What should be the assignment of boards to parallel stations and 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. Refs. 6 and 16. 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, a board's 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 2, where both buffers and machines are referred to as processors.

The flexible flow line under study (see Fig. 6) consists of m processing stages in series. Each stage, i ($i = 1, \ldots, m$) consists of $n_i \geq 1$ identical

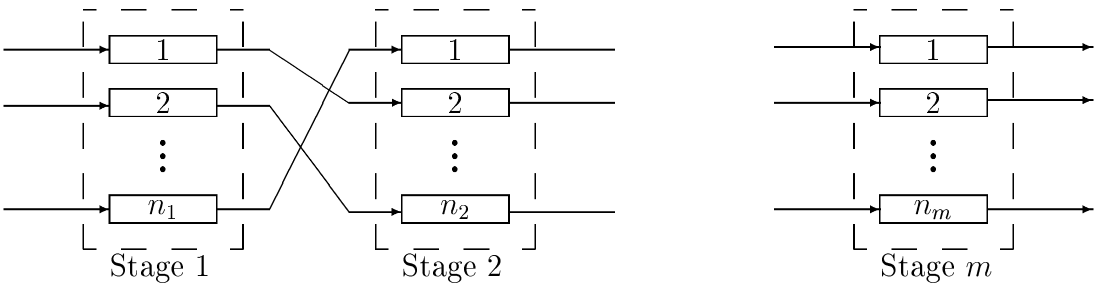

Fig. 6. A flexible flow line with no intermediate buffers.

Table 2. 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	=	number of parallel processors in stage i
p_{ik}	=	processing time for board k in stage i
v	=	number of boards
H_s	=	subset of processing stages $i \in I$ with parallel processors between two successive shuttles s and $s+1$
S	=	set of shuttles, $s \in S$
Q	=	a large number not less than the 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 $y_{kl} = 0$

parallel processors. The system produces v boards of various types. Each board must be processed without pre-emption 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 the input sequence in which the boards enter the line. Thus, a so-called permutation flowshop is considered.

Let $p_{ik} \geq 0$ be the processing time in stage i of board k, $(k = 1, \ldots, v)$. For every board k, let c_{ik} denote its completion time in each stage i, and d_{ik} its departure time from stage i. Processing without pre-emption indicates that board k completed in stage i at time c_{ik} starts its processing in that stage at time $c_{ik} - p_{ik}$. Board k completed in stage i at time c_{ik} departs at time $d_{ik} \geq c_{ik}$ 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 $i + 1$ (see Fig. 7).

The objective is to determine 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 generic scheduling model has the following structure:

Minimize **Maximum completion time**
subject to

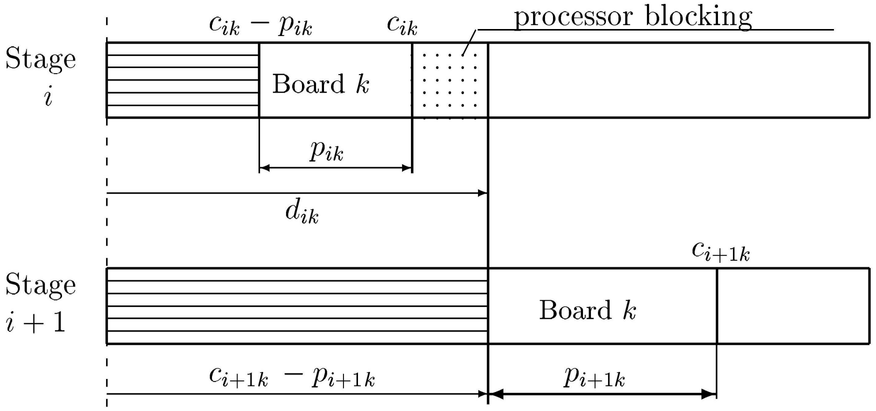

Fig. 7. A partial schedule for board k.

(1) **Assignment constraints for stages with parallel processors** to ensure that each board is assigned to exactly one processor and remains on the same conveyor until a shuttle stage. In addition, the workload assigned to each parallel processor is equalized.

(2) **Board completion constraints** to ensure that each board is processed at all stages.

(3) **Board non-interference constraints** to ensure that no two boards are processed by the same processor simultaneously.

(4) **No-store constraints** to ensure that processing of each board at every stage starts immediately after its departure from the previous stage.

(5) **Maximum completion time constraints** to ensure that each board leaves the line as soon as it is completed at the last stage. These constraints are also used to calculate the maximum completion time, i.e. the makespan of a given production schedule.

The mathematical formulation of the mixed integer program for scheduling SMT lines is presented below.

Model FF: *Scheduling of a flexible flow line*

Minimize

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

subject to

Assignment constraints for stages with parallel processors

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

$$\begin{gathered} x_{i_1,j_1,k} = x_{i_2,j_2,k}; \\ s \in S,\, i_1\, i_2 \in H_s,\, j_1 \in J_{i_1},\, j_2 \in J_{i_2},\, k \in K : \\ i_1 \neq i_2,\, j_1 = j_2 \end{gathered} \tag{3}$$

$$\begin{gathered} \sum_{k \in K} p_{ik} x_{ijk} \leq \sum_{k \in K} p_{ik}/n_i + \min_{k \in K} (p_{ik}); \\ i \in I,\, j \in J_i : n_i > 1 \end{gathered} \tag{4}$$

Board completion constraints

$$\begin{gathered} c_{1l} \geq p_{1l} + \sum_{k \in K: k<l} p_{1k} y_{kl} + \sum_{k \in K: k>l} p_{1k}(1 - y_{lk}); \\ l \in K : n_1 = 1 \end{gathered} \tag{5}$$

$$c_{1k} \geq p_{1k}; \quad k \in K : n_1 > 1 \tag{6}$$

$$c_{ik} - c_{i-1k} \geq p_{ik}; \quad i \in I,\, k \in K : i > 1 \tag{7}$$

Board non-interference constraints for stages with single processors

$$\begin{gathered} c_{ik} + Q y_{kl} \geq d_{il} + p_{ik}; \\ i \in I,\, k,\, l \in K : k < l \quad \text{and} \quad n_i = 1 \end{gathered} \tag{8}$$

$$\begin{gathered} c_{il} + Q(1 - y_{kl}) \geq d_{ik} + p_{il}; \\ i \in I,\, k,\, l \in K : k < l \quad \text{and} \quad n_i = 1 \end{gathered} \tag{9}$$

Board non-interference constraints for stages with parallel processors

$$\begin{gathered} c_{ik} + Q(2 + y_{kl} - x_{ijk} - x_{ijl}) \geq d_{il} + p_{ik}\,; \\ i \in I,\, j \in J_i,\, k,\, l \in K : k < l \quad \text{and} \quad n_i > 1 \end{gathered} \tag{10}$$

$$\begin{gathered} c_{il} + Q(3 - y_{kl} - x_{ijk} - x_{ijl}) \geq d_{ik} + p_{il}\,; \\ i \in I,\, j \in J_i,\, k,\, l \in K : k < l \quad \text{and} \quad n_i > 1 \end{gathered} \tag{11}$$

No-store constraints

$$c_{ik} = d_{i-1k} + p_{ik}\,; \quad i \in I,\, k \in K : i > 1 \tag{12}$$

Maximum completion time constraints

$$c_{mk} = d_{mk}\,; \quad k \in K \tag{13}$$

$$c_{mk} \le C_{\max}\,; \quad k \in K \tag{14}$$

$$C_{\max} \ge \sum_{k\in K} p_{ik}/n_i + \sum_{h\in I:h\neq i} \min_{k\in K}(p_{hk})\,;\ i \in I \tag{15}$$

Variable elimination constraints

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

Variable nonnegativity and integrality constraints

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

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

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

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

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. Constraint (3) ensures that each board remains on the same conveyor until a shuttle stage and (4) equalizes in every stage the workload assigned to each parallel processor. Constraint (5) or (6) ensures that each board is processed in the first stage, and (7) guarantees that it is also processed in all downstream stages. Constraints (8) and (9) are board non-interference constraints for single processors, and (10) and (11) apply for parallel processors. No two boards can be performed on the same processor simultaneously. For a given sequence of boards, either constraint (8) or (9) is active. Likewise, either (10) or (11) is active, and only if both boards k and l 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. Equation (13) ensures that each board leaves the line as soon as it is completed in the last stage. Finally (14) defines the maximum completion time, and (15) imposes a lower bound on it.

3.1. *Special cases*

Model **FF** for scheduling flexible flow line with limited intermediate buffers is a general formulation and includes various special cases. Some of them are presented below, see Refs. 16–18.

1. Flowshop with Single Processors

   If $n_i = 1, \forall\, i \in I$ model **FF** reduces to scheduling a flowshop with single processors, including buffers.
2. Flowshop with no In-process Buffers

   If $n_i = 1, \forall\, i \in I$ and $p_{ik} > 0, \forall\, i \in I,\, k \in K$ model **FF** reduces to scheduling flowshop with single processors and no in-process buffers.
3. Flexible Flow Line with no In-process Buffers

   If $p_{ik} > 0, \forall\, i \in I,\, k \in K$ model **FF** reduces to scheduling flexible flow line with no in-process buffers.
4. Double-Pass Re-entrant Line

   In order to enhance model **FF** for scheduling a double-pass re-entrant flow line, where a board visits the stages twice, 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. For each board $k+v$, $k = 1, \ldots, v$ additional board completion constraints are needed.

   Board completion constraints for the second pass

   $$c_{1k+v} \ge c_{mk} + p_{1k+v}\,; \quad k = 1, \ldots, v \tag{21}$$
5. Dual-Conveyor Line

   The model **FF** can be enhanced for scheduling a dual-conveyor line as follows. Each placement machine with a dual conveyor is modelled as a series of the following four processing stages:

   - Two parallel input buffers, i.e. one for each conveyor;
   - One “dummy buffer” within the machine;
   - One machine;
   - Two parallel output buffers, i.e. one for each conveyor.

   In a machine with dual conveyors, the conveyor, which does not contain the board being assembled, functions as a “dummy buffer”.

   For subset $I_2 \subset I$ of stages with a dual-conveyor and parallel processors, the following additional constraints are added to ensure that each board remains on the same conveyor until completion.

   Assignment constraints for stages with parallel processors and a dual-conveyor

$$x_{i_1 j_1 k} = x_{i_2 j_2 k}\,;$$
$$i_1,\, i_2 \in I_2,\, j_1 \in J_{i_1},\, j_2 \in J_{i_2},\, k \in K :$$
$$i_1 \neq i_2,\, n_{i_1} = n_{i_2} = 2,\, j_1 = j_2 \qquad (22)$$

3.2. *Scheduling modes*

In order to reduce the complexity of the **general scheduling** problem, where any sequence of boards is allowed, the following scheduling modes can also be considered:

- **Batch scheduling**, where boards of a given type are scheduled consecutively, where in addition:
  - (a) the sequence of board types is fixed and equal to the optimal sequence determined for a minimum set of boards (e.g. one board of each type) or
  - (b) the sequence of board types is not determined *a priori*, but is obtained with the optimal schedule for all boards.
- **Cyclic scheduling**, where different board types are scheduled alternately in a cyclic order of board types, where in addition:
  - (a) the cycle of board types is fixed and equal to the optimal sequence determined for a minimum set of boards (e.g. one board of each type) or
  - (b) the cycle of board types is not determined *a priori*, but is obtained with the optimal schedule for all boards.

4. Numerical Examples

In this section, four numerical examples are presented to illustrate application of the proposed model. In the examples, assembly schedules are determined by solving mixed integer programs representing some typical electronics assembly line configurations.

In the first two examples, the production mix consists of three board types. The assembly schedules were determined for the following two cases:

- Batch scheduling, where ten boards of each type are assembled, and all boards of a given type are scheduled consecutively. The optimal sequence of board types is obtained along with the optimal schedule for all boards.
- Cyclic scheduling, where ten 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.

4.1. *Example 1: SMT line with parallel stations*

The SMT line configuration for Example 1 is provided in Fig. 8. The line consists of $m = 12$ stages, where stage $i = 1$ is a screen printer, each stage $i = 5, 9$ represents two parallel machines for automatic placement of components, stage $i = 12$ is a vision inspection station, stages $i = 3, 7, 11$ are shuttles, stage $i = 2$ is a single external buffer, and each stage $i = 4, 8$ and $i = 6, 10$ represents two internal input and output buffers, respectively.

The processing times p_{ik} for the boards are shown in Table 3 (for the buffer and shuttle stages $i = 2, 3, 4, 6, 7, 8, 10, 11$ all processing times are equal to zero).

The optimal schedules obtained for batch and cyclic scheduling are shown in Fig. 9, where letters

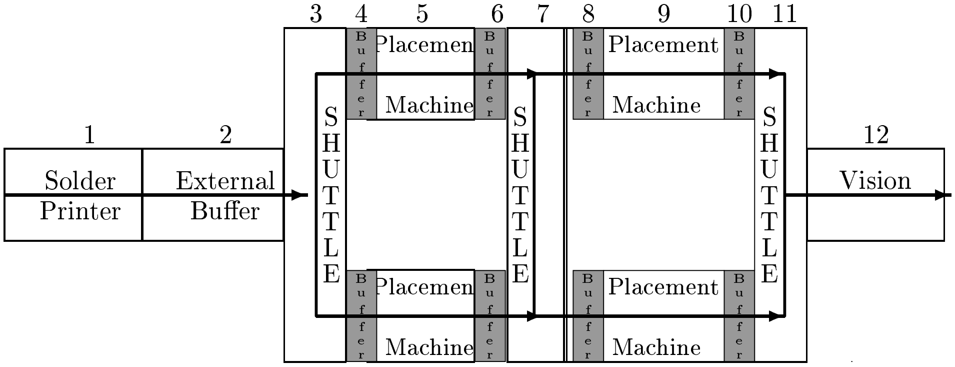

Fig. 8. SMT line with parallel stations.

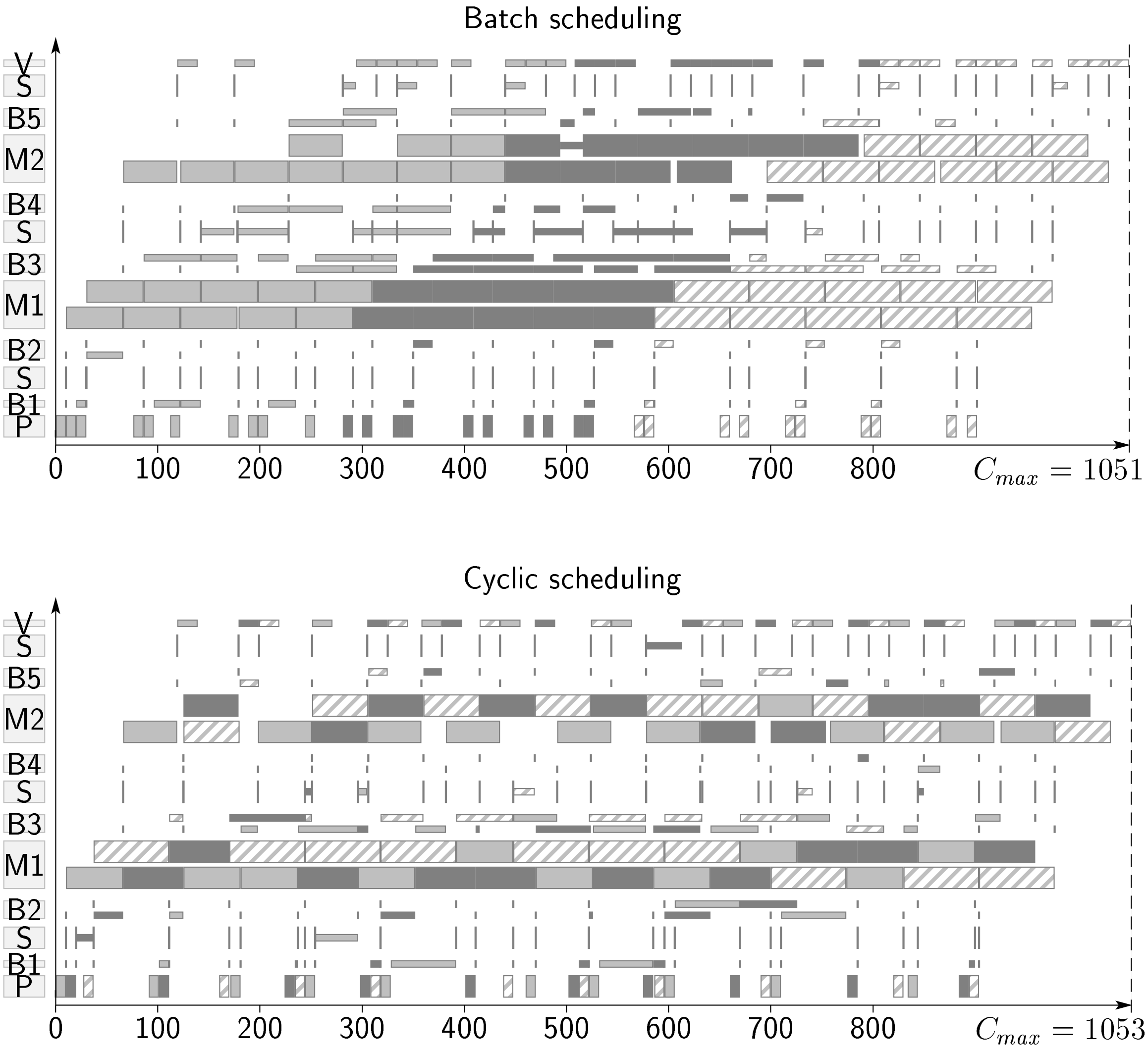

Fig. 9. Assembly schedules for SMT line with parallel stations.

Table 3. Processing times for SMT line with parallel stations.

Board Type	1	2	3
Screen printer P	10	10	10
Placement machine M1	56	59	74
Placement machine M2	53	54	55
Vision inspection machine V	20	20	20

B, M, P, S and V stand for Buffer, Machine for placement, Printer, Shuttle and Vision inspection machine, respectively. Boards of types 1 and 2 are indicated, respectively with gray and black shading, and boards type 3 with cross hatching. The optimal sequence of board types 1, 2, and 3 is identical for both batch and cyclic scheduling modes. The optimized makespans are as follows: $C_{\max} = 1051$ for batch scheduling, $C_{\max} = 1053$ for cyclic scheduling.

4.2. *Example 2: SMT line with a dual-conveyor*

In Example 2, optimal assembly schedules were determined for a dual conveyor line (see Fig. 3) with a screen printer, two placement machines and an oven. The line consists of $m = 14$ stages, where stage $i = 1$ is a screen printer, each stage $i = 6, 11$ represents a machine for automatic placement of components, stage $i = 13$ is a reflow oven, stages $i = 3, 14$ are shuttles, and stage $i = 2$ is a single external buffer. Each stage $i = 4, 7, 8, 9, 12$ represents 2 parallel buffers, one on each conveyor, and stages $i = 5, 10$ are single dummy buffers on the placement machines.

Table 4. Processing times for SMT line with a dual-conveyor.

Board Type	1	2	3
Screen printer P	20	20	20
Placement machine M1	112	117	147
Placement machine M2	120	102	113
Oven 0	40	40	40

The processing times p_{ik} for the boards are shown in Table 4. (For the buffer and shuttle stages, all processing times are equal to zero.)

The optimal schedules obtained for batch and cyclic scheduling are shown in Fig. 10, where letters B, BM, M, O, P and S stand for Buffer, Buffer on Machine, Machine for placement, Oven, Printer and Shuttle, respectively. Boards of types 1 and 2 are indicated with light gray and dark gray, respectively, and boards type 3 with cross hatching. The sequence of board types was not fixed *a priori*. The optimal sequence of board types for batch scheduling is $1, 3, 2$ and is the same as the optimal cycle of board types for cyclic scheduling. For Example 2, the same optimal makespan $C_{\max} = 3922$ was achieved for each scheduling mode.

The next two examples are modeled after real-world SMT lines at our industrial partner's factories, where production is typically scheduled by

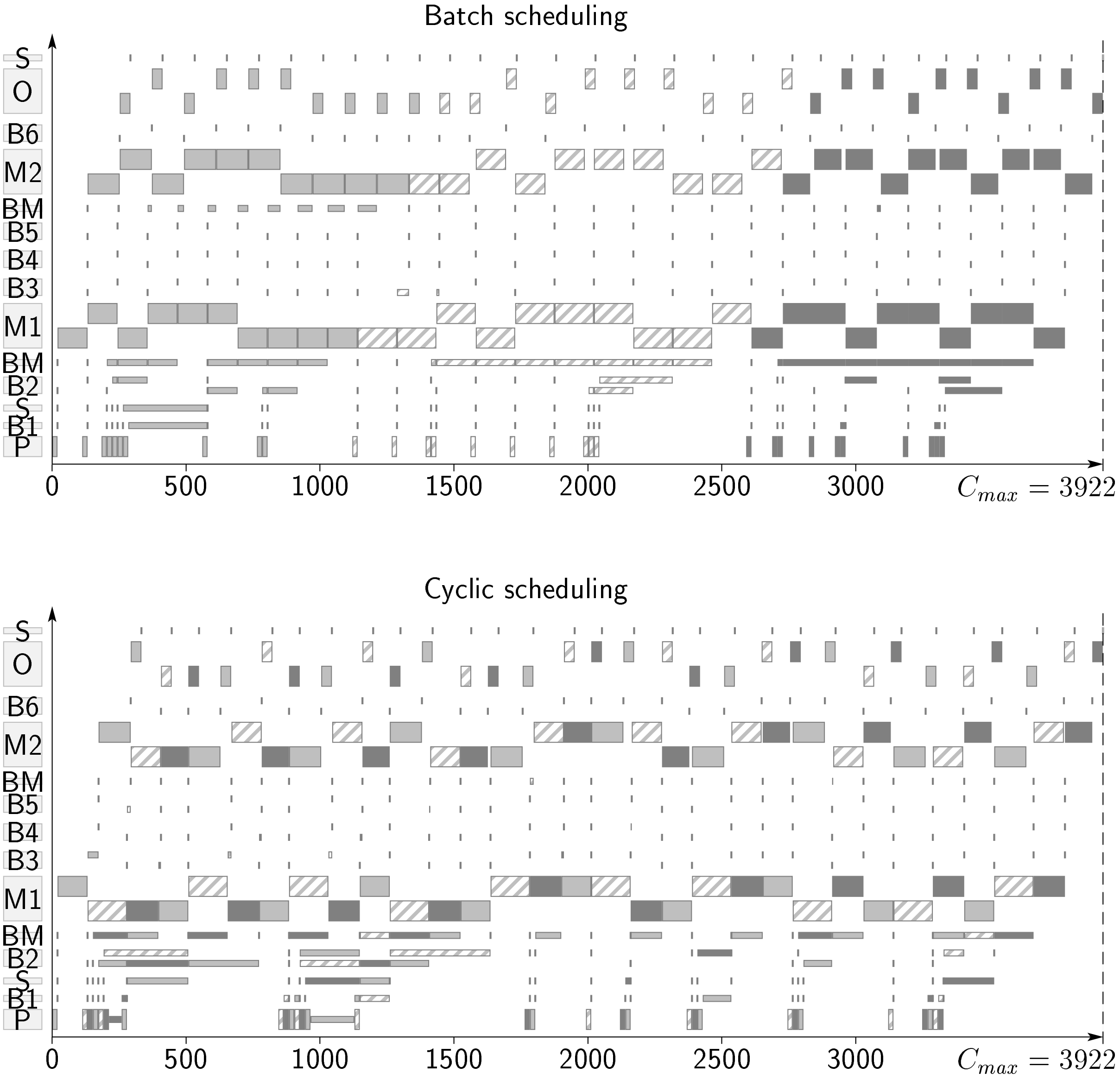

Fig. 10. Assembly schedules for SMT line with a dual-conveyor

Table 5. Description of SMT line with single stations.

Stage No.	Stage Type	Number of Processors	Description
1	Machine	1	Loader
2	Buffer	1	1 input buffer
3	Machine	1	Screen print
4	Buffer	5	1 output buffer, 3 extra buffers, 1 input buffer
5	Machine	1	Small part placement
6	Buffer	12	1 output buffer, 10 extra buffers, 1 input buffer
7	Machine	1	Small part placement
8	Buffer	6	1 output buffer, 4 extra buffers, 1 input buffer
9	Machine	1	Large part placement
10	Buffer	4	1 output buffer, 2 extra buffers, 1 input buffer
11	Machine	1	Large part placement
12	Buffer	4	1 output buffer, 2 extra buffers, 1 input buffer
13	Machine	1	Vision inspection

experienced factory managers, considering priority orders and available components.

4.3. *Example 3: Factory with single stations*

The SMT line configuration for Example 3 is described in Table 5. The line consists of loader, screen printer, 4 placement machines and vision inspection machine, in series separated by intermediate buffers.

The input data for Example 3 were prepared considering daily production of the line over a one month horizon. The line represents a typical low volume, medium variety production system, in which 13 different board types are assembled in batches between 1 and 265 boards. A daily production order consists of at most four different board types assembled in the line.

Table 6 presents input data for Example 3. The actual production order was reduced to obtain a clear Gantt chart representing the assembly schedule. The production requirements represent around 1/4th of a typical daily production order.

Table 6. Processing times and production volume for Example 3.

Board Type	1	2
Loader M1	20	20
Screen printer M2	25	25
Placement machine M3	34	66
Placement machine M4	78	28
Placement machine M5	92	34
Placement machine M6	55	0
Vision inspection machine M7	45	30
Production volume	15	15

The Gantt chart with the optimal batch schedule obtained for this example is shown in Fig. 11. The optimal input sequence of board types is $1, 2$, and the optimal makespan $C_{\max} = 2087$. For the actual production order which is made up of 57 boards of type 1 and 68 boards of type 2, the best input sequence for batch scheduling is the same, and the best makespan is $C_{\max} = 8661$.

4.4. *Example 4: Factory with parallel stations*

The SMT line configuration for Example 4 is described in Table 7. The line consists of screen printer, three sets of 2 parallel placement machines, vision inspection machine and a single placement machine, in series separated by intermediate buffers.

The input data for Example 4 were prepared considering daily production of the line over a one month horizon. The line represents a typical high volume, low variety production system, in which six different board types are produced in batches between 20 and 1080 boards. A daily production order consists of at most four different board types assembled in the line.

Table 8 presents input data for Example 4. The actual production order was reduced to obtain a clear Gantt chart representing the assembly schedule. The production requirements represent around 1/30th of a typical daily production order.

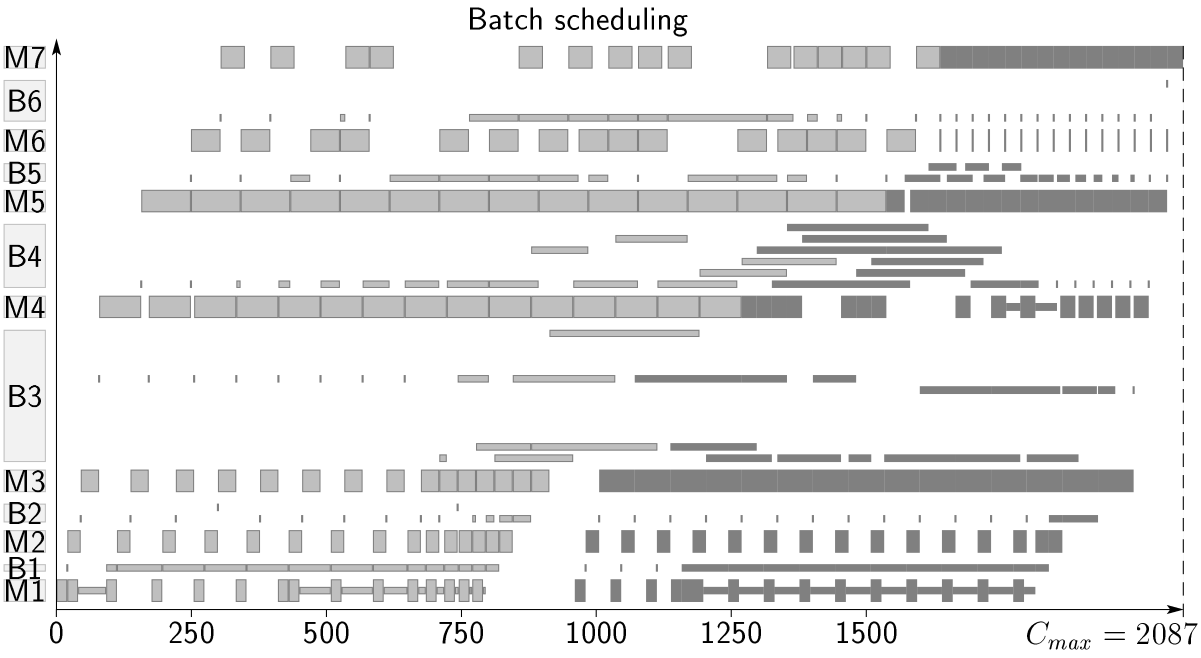

Fig. 11. Batch schedule for SMT line with single stations.

Table 7. Description of SMT line with parallel stations.

Stage No.	Stage Type	Number of Processors	Description
1	Machine	1	Screen print
2	Buffer	4	1 extra buffer, shuttle, 2 input buffers
3	Machine	2	Parallel machines for small part placement
4	Buffer	5	2 output buffers, shuttle, 2 input buffers
5	Machine	2	Parallel machines for small part placement
6	Buffer	5	2 output buffers, shuttle, 2 input buffers
7	Machine	2	Parallel machines for small part placement
8	Buffer	4	2 output buffers, shuttle, 1 input buffers
9	Machine	1	Large part placement
10	Buffer	3	1 output buffer, 2 extra buffers
11	Machine	1	Vision inspection
12	Buffer	2	1 extra buffer, 1 input buffer
13	Machine	1	Large part placement

The Gantt chart with the best batch schedule obtained for this example is shown in Fig. 12. The obtained input sequence of board types is $1, 2, 4$ and 3, and the resulting makespan $C_{\max} = 4124$. For the actual production order which is made up of 80, 100, 240, and 480 boards of type $1, 2, 3$ and 4, respectively, the input sequence of board types for batch scheduling is the same, and the best makespan is $C_{\max} = 99882$.

Both of the industrial examples clearly demonstrate that buffers are important to maximize throughput of an SMT line. In particular, when large-size batches of boards are scheduled, the buffers ahead of bottleneck stages (large part placement machines in Example 3 and small part placement machines in Example 4) are occupied most of the time.

The characteristics of mixed integer programs (MIP) for the example problems and the solution results are summarized in Table 9. The size of the mixed integer programming models for the example problems is represented by the total number of variables, *Var.*, number of binary variables, *Bin.*,

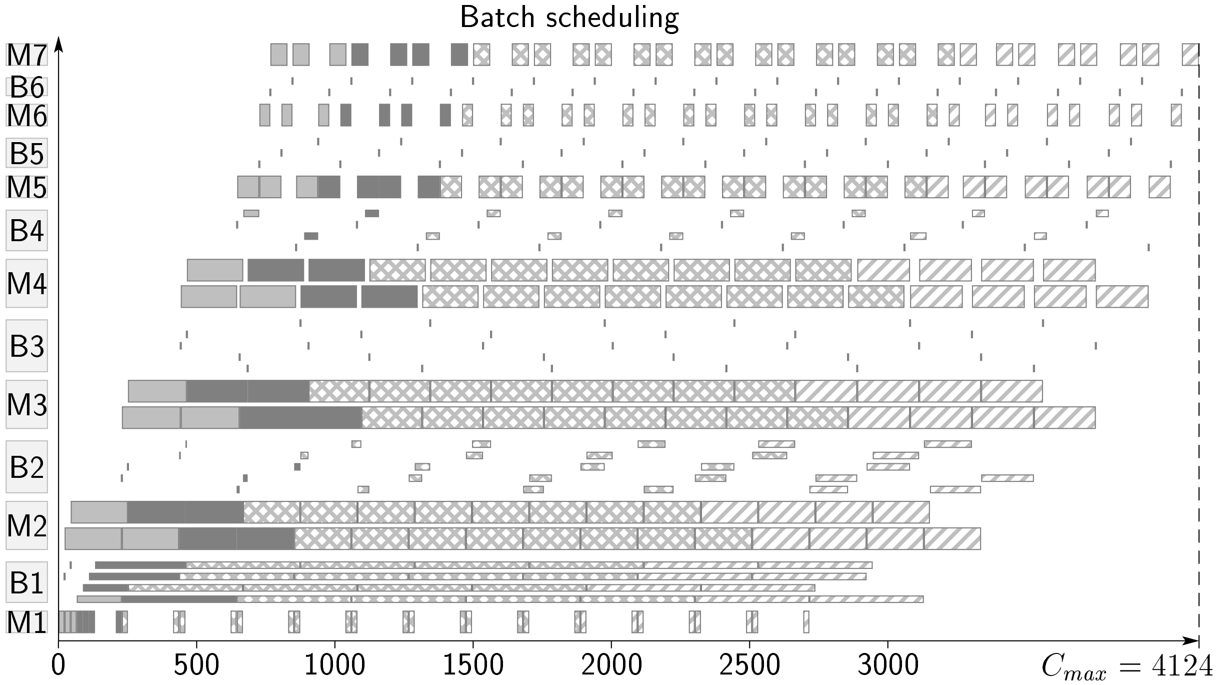

Fig. 12. Batch schedule for SMT line with parallel stations.

Table 8. Processing times and production volume for Example 4.

Board Type	1	2	3	4
Screen printer M1	22	22	22	22
Placement machine M2	207	208	207	207
Placement machine M3	213	220	224	220
Placement machine M4	204	204	191	204
Placement machine M5	80	80	80	80
Vision inspection machine M6	40	40	40	40
Placement machine M7	62	62	62	62
Production volume	3	4	8	16

number of constraints, *Cons.*, and number of nonzero coefficients, *Nonz.*, in the constraint matrix. The last four columns of Table 9 present the lower bound *LB* (15) on the makespan, the makespan $C_{\max}$, the number of nodes in the branch-and-bound tree, and CPU time in seconds required to find the best solution (proven optimal for Examples 2 and 3). The examples were solved on a Compaq Presario 1830 laptop with a Pentium III, 450 MHz processor using AMPL and the CPLEX v.6.5.2 solver. When CPLEX was given an additional four hours to run, no better solutions were found for Examples 1 and 4.

Experiments with various features of CPLEX solver to speed up the solution process have indicated that the best results are obtained with a nearly depth-first branch and bound strategy for node selection where limited backtracking is allowed.

5. Effects of Randomness on SMT Line Performance

In this section, the influence of process time variability and machine breakdowns on an SMT line's performance is discussed. In the proposed mathematical programming approach, processing times are assumed to be known with certainty in advance of scheduling. Furthermore, machines and other resources are assumed to be continuously available. In contrast, the real-life scheduling environment is dynamic and stochastic.[22–24]

One of the sources of randomness in an SMT line is the natural variability caused by differences in vision inspection times and in loading and unloading times. The former are affected by variation in component quality, key dimensions of the component and robustness of the vision algorithms. The latter are determined by the motors that run the conveyors. There is also some variability from machine breakdowns, which can require additional machine operations and/or actions by human operators. An example of the first type of breakdown is the picking of components from feeders. If a pickup is not

Table 9. MIP characteristics and solution results for example problems.

Problem	Var.	Bin.	Cons.	Nonz.	*LB*	$C_{\max}$	Nodes	CPU [sec]
Ex. 1/Batch	1381	660	17607	67584	1028	1051	11937	1330
Ex. 1/Cyclic	1327	606	17337	64872	1028	1053	3191	625
Ex. 2/Batch	1501	660	20651	75144	3922	3922	60	19
Ex. 2/Cyclic	1447	606	20381	72336	3922	3922	0	28
Ex. 3	1936	1155	35129	142798	2087	2087	530	420
Ex. 4	2014	1207	32114	139410	4018	4124	13650	2700

successful, the mispicked part has to be dropped into a trash box, and the complete pickup process will be repeated. Depending on how many of these mispicks occur during a machine cycle, the process time can increase up to 10% or more. This results in a randomized actual process time with a minimum raw process time for all SMT placement machines.

The solder printing process is a typical example of the second type of breakdown. The solder stencil has to be cleaned by a human operator (or automatically in newer machines) after approximately every 20 to 30 panels. This results in frequent breaks in the output of the machine itself. Another example is the replenishment of components on the SMT machine by the operator to keep the machine running. There are also frequent short stops in the SMT line which can be described by the Mean Time To Assist (MTTA) and Mean Time Between Assists (MTBA). Likewise, all machines have a characteristic breakdown behaviour known as Mean Time Between Failures (MTBF) and Mean Time To Repair (MTTR). The MTBF is assumed to be significantly longer than the makespan of a typical production schedule. Random disturbances that may occur in an SMT line and their influence on the line performance are described in Table 10.

An effective schedule in practice should be robust in the presence of uncertainties described above. Robustness of a schedule in PWB assembly is equivalent to robustness of input sequence of boards entering an SMT line and robustness of the assignment of boards to parallel machines or conveyors in the presence of varying processing times and intermittent machine availability.

Table 10. Random disturbances in SMT lines.

Random Effects	Natural Randomness	Randomness caused by breakdowns			Randomness Caused by Changeovers
		Pickup Errors	Operator Required Breakdowns	Machine Breakdowns	
Variability	Frequent	Frequent	Frequent (MTBA)	Infrequent (MTBF)	Infrequent
	Very small	Small	Medium (MTTA)	Large (MTTR)	Very large
Impact on Throughput	Not measurable	Cycle time increases on the bottleneck machine	Higher variability of the line flow	Complete stop of the production	Planned, organized stop of the complete line
Statistical Distribution	Normal	Pearson 6	Gamma	Gamma	—
Examples	Vision time, loading/ unloading time	Vacuum leakage, bent nozzles	Stencil cleaning, component replenishment	Broken part	Maintenance, feeder changeover

In practice, the required robustness of an assembly schedule can be achieved by keeping an SMT line balanced and by frequent re-scheduling using fast heuristics, e.g. Ref. 12, in response to perturbations introduced in the line or by applying various dispatching rules for dynamic scheduling, e.g. Ref. 9.

6. Concluding Remarks

This paper has proposed the use of mixed integer programming for scheduling SMT lines. It is a general approach which can be applied to a variety of different assembly line configurations with only small modifications to the constraint formulations and/or input data definitions. The most important scheduling decisions for PWB assembly include the input sequence of boards entering an SMT line and the assignment of boards to parallel machines or conveyors. The detailed timing of start and finish events for each station is of secondary importance. The computational effort to find optimal schedules for realistic problems in the electronics industry can be reduced by introducing specific scheduling modes, such as batching or cyclic. On the other hand, the proposed mixed integer programs can be applied for performance evaluation of fast heuristics used to solve large scheduling problems.

Batch sizes and cyclic schedules are two types of "environmental controls" that can be imposed on a production system to improve its performance and reduce the complexity of the associated optimization problems. Work-in-process (WIP) limits, e.g. kanban, have been proposed as effective countermeasures against variability in process times, e.g. Ref. 25. Future work should include an investigation of other environmental factors, such as the buffer sizes and locations. Future work should also focus on the application of large-scale mixed integer programs for simultaneous loading and scheduling of SMT lines, e.g. Ref. 26. This technique might be very useful both for off-line decision making as well as for developing fast heuristics for real-time re-balancing and re-scheduling, which is necessary when the different types of randomness discussed in this paper cause disruptions in the line.

Acknowledgments

This work has been partially supported by research grants of AGH #10.10.200.81 and KBN #7 T11F 007 20 (Poland) and by the Motorola Advanced Technology Center (USA).

References

1. T. Sawik, *Production Planning and Scheduling in Flexible Assembly Systems*, Springer-Verlag, Berlin, 1999.
2. S. C. Graves, H. C. Meal, D. Stefek and A. H. Zeghmi, "Scheduling of re-entrant flow shops", *J. Operations Management* **3** (1983) 197–207.
3. T. M. Tirpak, "Design-to-manufacturing information management for electronics assembly", *Int. J. Flexible Manufacturing Systems* **12**(2) (2000) 147–162.
4. W. Wang, P. C. Nelson and T. M. Tirpak, "Optimization of high-speed multistation SMT placement machines using evolutionary algorithms", *IEEE Transactions on Electronics Packaging Manufacturing* **22**(2) (2000) 137–146.
5. S. M. Gupta and A. Kavusturucu, "Production systems with interruptions, arbitrary topology and finite buffers", *Annals of Operations Research* **93** (2000) 145–176.
6. S. T. McCormick, M. L. Pinedo, S. Shenker and B. Wolf, "Sequencing in an assembly line with blocking to minimize cycle time", *Operations Research* **37** (1989) 925–936.
7. N. G. Hall and C. Sriskandarajah, "A survey of machine scheduling problems with blocking and no-wait in process", *Operations Research* **44** (1996) 510–525.
8. R. J. Wittrock, "An adaptable scheduling algorithm for flexible flow lines", *Operations Research* **36** (1988) 445–453.
9. R. Rachamadugu and K. E. Stecke, "Classification and review of FMS scheduling procedures", *Production Planning and Control* **5** (1994) 2–20.
10. I. N. K. Abadi, N. G. Hall and C. Sriskandarajah, "Minimizing cycle time in a blocking flowshop", *Operations Research* **48** (2000) 177–180.
11. T. A. Feo, J. F. Bard and S. D. Holland, "Facility-wide planning and scheduling of printed wiring board assembly", *Operations Research* **43** (1995) 219–230.
12. T. Sawik, "A scheduling algorithm for flexible flow lines with limited intermediate buffers", *Applied Stochastic Models and Data Analysis*, special issue on Manufacturing Systems **9** (1993) 127–138.
13. S. Ghosh and R. J. Gagnon, "A comprehensive literature review and analysis of the design, balancing and scheduling of assembly systems", *Int. J. Production Res.* **27** (1989) 637–670.
14. G. L. Nemhauser and L. A. Wolsey, *Integer and Combinatorial Optimization*, John Wiley, New York, 1988.
15. T. Sawik, "Integer programming models for the design and balancing of flexible assembly systems", *Mathematical and Computer Modelling* **21**(4) (1995) 1–12.
16. T. Sawik, "Mixed integer programming for scheduling flexible flow lines with limited intermediate buffers", *Mathematical and Computer Modelling* **31** (2000) 39–52.
17. T. Sawik, "Discrete programming model for scheduling SMT lines", *Zeszyty Naukowe Politechniki Śląskiej, Automatyka*, no. 129 (2000) 343–352.

18. T. Sawik, "Mixed integer programming for scheduling surface mount technology lines", *Int. J. Production Res.* **39** (2001) 3219–3235.
19. S. Karabati and P. Kouvelis, "Cycle scheduling in flow lines: Modeling observations, effective heuristics and a cycle time minimization procedure", *Naval Research Logistics* **43** (1996) 211–231.
20. A. G. P. Guinet and A. A. Solomon, "Scheduling hybrid flowshops to minimize maximum tardiness or maximum completion time", *Int. J. Production Res.* **34** (1996) 1643–1654.
21. T. Sawik, A. Schaller and T. M. Tirpak, "Issues in loading and scheduling of SMT lines", *Zeszyty Naukowe Politechniki Śląskiej, Automatyka*, no. 129 (2000) 331–341.
22. R. Shafaei and P. Brunn, "Workshop scheduling using practical (inaccurate) data – Part 1: The performance of heuristic scheduling rules in a dynamic job shop environment using a rolling time horizon approach", *Int. J. Production Res.* **37** (1999) 3913–3925.
23. R. Shafaei and P. Brunn, "Workshop scheduling using practical (inaccurate) data – Part 2: An investigation of the robustness of scheduling rules in a dynamic and stochastic environment", *Int. J. Production Res.* **37** (1999) 4105–4117.
24. R. Shafaei and P. Brunn, "Workshop scheduling using practical (inaccurate) data – Part 3: A framework to integrate job releasing, routing and scheduling functions to create a robust predictive schedule", *Int. J. Production Res.* **38** (2000) 85–99.
25. W. Hopp and M. Spearman, *Factory Physics: Foundations of Manufacturing Management*, McGraw-Hill, 1996.
26. T. Sawik, "Simultaneous versus sequential loading and scheduling of flexible assembly systems", *Int. J. Production Res.* **38** (2000) 3267–3282.