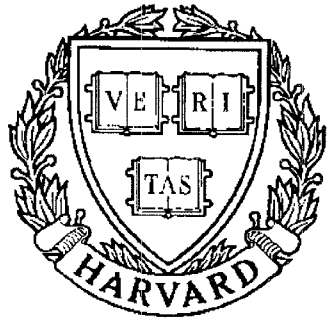
# TECHNICAL RESEARCH REPORT


SYSTEMS RESEARCH CENTER




*Supported by the National Science Foundation Engineering Research Center Program (NSFD CD 8803012), the University of Maryland, Harvard University, and Industry*

# Annealing Based Experiment in Scheduling

*by J.S. Dhingra, K.L. Musser, G.L. Blankenship and L. Ferguson*

TR 91-15

# Annealing Based Experiment in Scheduling*

J.S. Dhingra, K.L. Musser, G.L. Blankenship, L. Ferguson

February 11, 1991

**Abstract**

A scheduling software was developed for general job shop scheduling. It was implemented at a Texas Instruments electronic circuit board assembly facility. The assembly line scheduling problem is presented as an optimization problem. A variation of simulated annealing technique was used to find a solution to the optimization problem. A detailed description of the software is presented and various assembly line features which were incorporated in the software are discussed. Finally, an onsite evaluation is presented.

## 1 INTRODUCTION

Computer integrated manufacturing has played an important role in the automation, flexibility and integration of manufacturing systems. Numerically controlled machines have led to reduced human operator involvement in actual production. Supervisory control and scheduling systems lead to an efficient utilization of machines by reducing setup and process flow times. The cumulative effect of this automation and integration in manufacturing systems is improved productivity and resource utilization.

Job shop scheduling is an important aspect of flexible manufacturing systems. In a flexible manufacturing system, there are several machines capable of performing a variety of operations on different sets of parts. The parts are transported from one machine to another using robots, automated guided vehicles, conveyors or trolleys. The flexibility of the manufacturing environment allows for the capability of different machines to perform a particular operation. These multiple processing alternatives along with machine failure and varying demand, makes the scheduling environment dynamic and leads to a complex scheduling problem. Optimal schedule generation is almost impossible because of the complexity of the problem and the large number of possible solutions. In recent studies, many different approaches to job shop scheduling have been suggested. Among these are expert

*This work was supported by NSF Engineering Research Centers Program NSFD CDR 88003012, and by a grant from Texas Instruments, Inc.

systems methods [2], use of neural networks [3], heuristics for finding good schedules in specialized problems [4, 5], and other tools for combinatorial optimization.

The scheduling system *ABES Annealing Based Experiment in Scheduling* was developed using a variation on one of the promising methods of general combinatorial optimization, namely simulated annealing. We have adapted it specifically to due-date oriented scheduling. Although simulated annealing methods have been applied to the job shop scheduling problem [1], our approach is technically different and more general, and ours appears to be the first attempt to make use of the concept in a real manufacturing environment. The scheduling system *ABES* has been installed at a Texas Instruments PC board assembly facility in Johnson City, TN and is currently being used to schedule the surface mount machines of the Custom Manufacturing Business.

# 2 FACTORY DESCRIPTION

The Texas Instruments PC board assembly facility at Johnson City, TN assembles circuit boards for external customers and also for their Industrial automation division. The Custom Manufacturing Business consists of three main units: surface mount, through hole, and quality control. Both single and double sided boards are manufactured at this facility. A typical process flow for a double sided board is shown in Figure 1. For a single sided board the second surface mount operation is not needed. The surface mount unit is fully automated and has a lower operational cost than the through hole and quality control units which are semi-automated. The installed capacity of the surface mount unit is less than the other two units and is generally the bottleneck region in board process flow. Therefore, our effort was directed to optimal scheduling of the surface mount unit.

The surface mount unit consists of six machine cells called SMT lines. Of these six lines, lines 1, 2, 3, and 4 are dedicated to top-side processing, while the remaining two (5 and 6) are exclusively used for bottom side processing. The layout of the surface mount unit is shown in Figure 2. The top side processing lines, each consist of a screen-printer machine and a pair of component placement machines, while each of the remaining two lines have a glue-machine and a component placement machine followed by an Ultraviolet-Infrared Cure machine. The four SMT lines, 1 through 4, are connected through a conveyor system to the vapor phase and solvent clean machines. The boards, in general, are processed top side first on one of the four SMT lines and after being cleaned, are transported, via trolleys, to lines 5 or 6 for bottom side processing. The processing of the boards is continuous in each line, but the topside-processed and solvent cleaned boards are transported for bottom side processing in batches. Thus, we have a mix of continuous and batch flow processing, which makes the scheduling job even tougher.

Each board has a principal process flow and a set of alternate process flows, which are determined by the characteristics of the parts being used in production. For example, if many fine pitch parts are being used on a particular board, then it can be assembled only

on one of the four SMT lines for topside and similarly can be processed only on one of the two lines for bottom side processing. Hence, we only have a principal process flow and no alternate flows for this particular board. On the other extreme, there are certain boards which can be manufactured on any of the machines. Thus, a double sided board which can be run on any of the 4 lines for topside processing and either of the two lines (5 an 6) for bottom side processing has 8 possible process flow configurations. We assign any one of those process flows to be the principal one and the rest are alternate flows.

The Custom Manufacturing Business produces 50 different boards. Depending on the customer demand and parts availability, it handles 30 different orders, or batches, per week on an average. The number of operations to be scheduled for each batch varies between 3 and 6 depending on the number of sides to be processed. Therefore, in principle we have about 100 operations to be scheduled. The concept of "board families" plays an important role in schedule generation. If two batches of two different boards, belonging to the same board family, are scheduled successively on the same line, the setup time for the succeeding operation is reduced to a fraction of the original setup time. This occurs when the two boards have similar sets of parts to be used on that particular machine and so only a partial teardown/setup is required. These reduced setup times lead to increased machine utilization and higher productivity.

Prior to the development of $\mathcal{ABES}$, the surface mount machines were scheduled manually. Parts availability and customer demand date were the main criteria used for scheduling. The different operations for various batches were then scheduled appropriately, on the 6 lines, so as to meet the due-date requirements. Approximate board processing times and machine setup times were used to determine the production run for each batch. This process of schedule generation was time consuming and tedious. The schedule so generated was heavily dependent on the experience and expertise of the scheduler.

# 3 SCHEDULE GENERATION

The factory is modeled as a set of virtual machines. A virtual machine consists of a set of physical machines which process parts in tandem. The processing parameters of the bottleneck physical machine in the set are used as the processing parameters for the virtual machine. To represent processing at a single physical machine we model that particular machine as a virtual machine containing only one physical machine. Therefore no distinction is made between a physical and a virtual machine henceforth. The factory produces a set of *devices* and the process flow for each device is defined as a sequence of *device operations*. These device operations represent the processing of the device at a particular machine. For each such operation, two processing parameters namely, part processing time and batch setup time, have to be defined. As mentioned earlier, the part processing time and setup time for the virtual machine are defined respectively as the maximum of the part processing times and setup times of the constituent physical machines. Each device has one such *principal process flow*. For each device an alternate process flow, which represents the

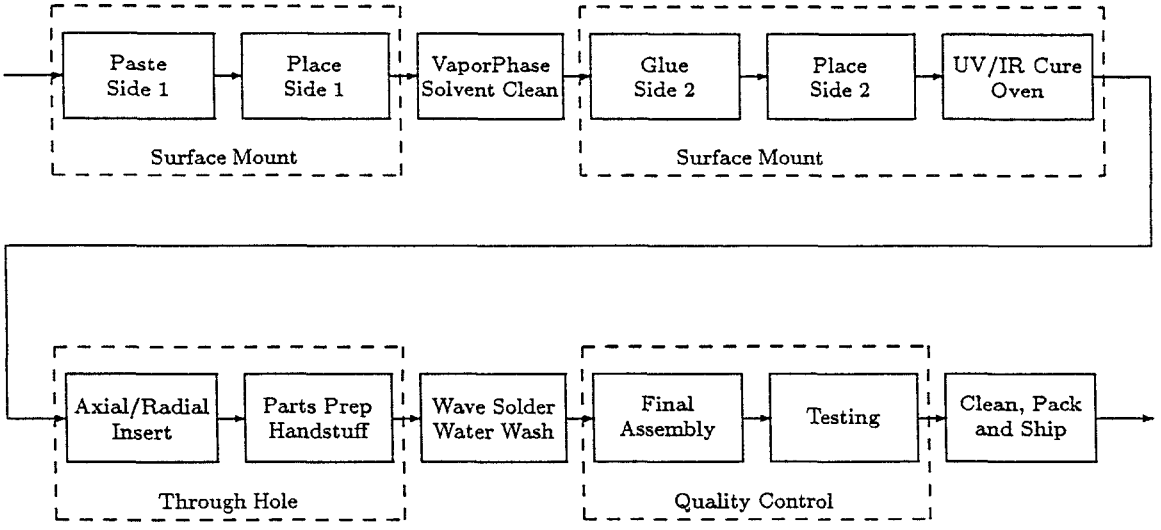



Figure 1: A typical process flow

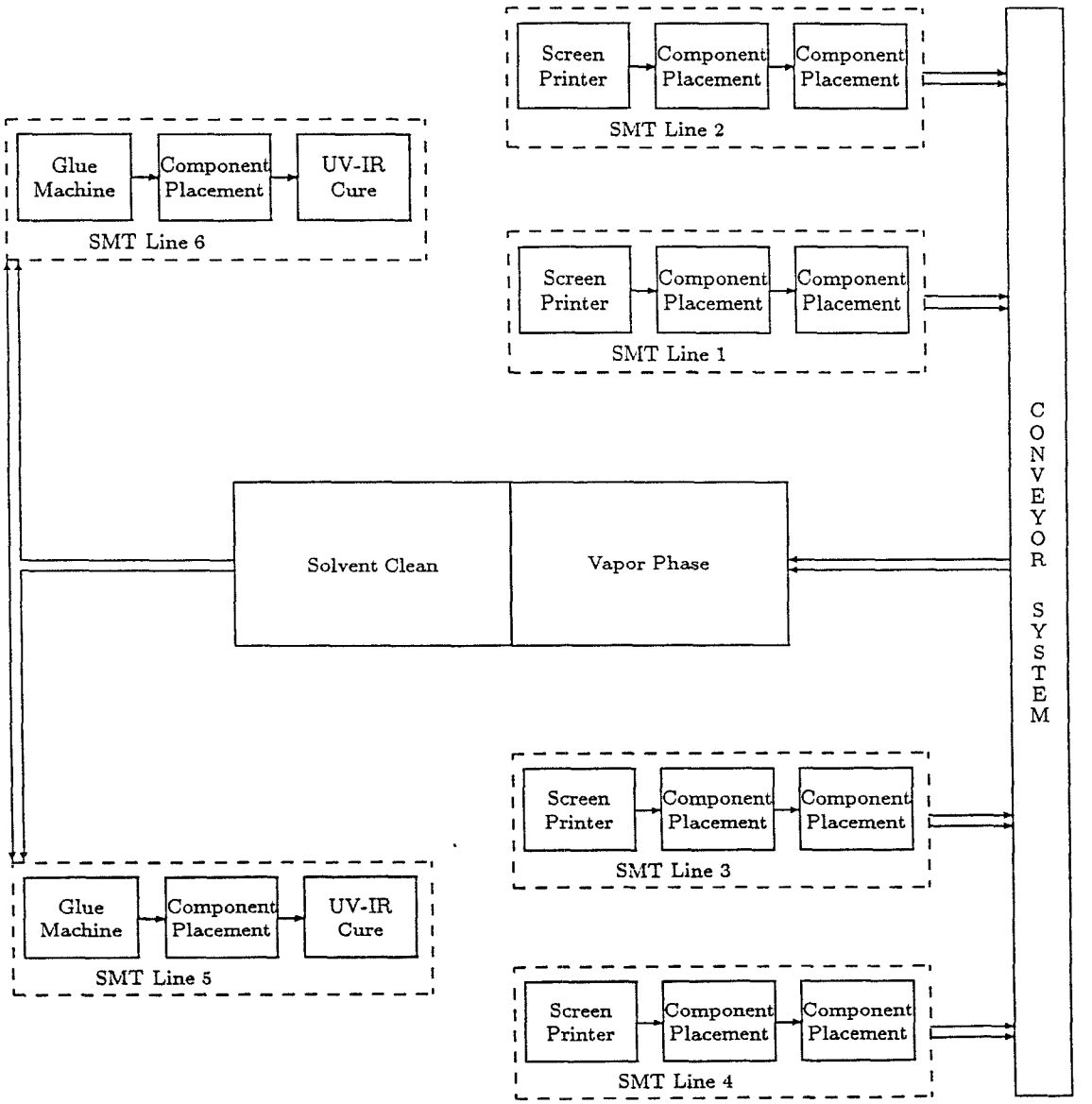



Figure 2: Surface Mount Unit

possibility of a set of *device operations* to be performed on a different set of virtual machines, can also be defined.

The main operational entity of $\mathcal{ABES}$ is a *schedule*. A *schedule* is generated by assigning start and finish times to the various device operations of different batches. A *batch* is a particular shipment or an order of a given device and is characterized by a batch size, a earliest start time and a due date. Therefore a batch can be represented as a sequence of operations with preassigned processing and setup times. A batch is part of a schedule if the constituent operations have been assigned start and finish times. These start and finish times are calculated on the basis of the two batch attributes namely, earliest start time and the due date, as well as machine availability times. Hence, a schedule can also be defined as a chronological ordering of operations on each machine in the factory setup. A detailed mathematical representation of the various data structures used is given in [12].

A given schedule may be evaluated according to its effectiveness at satisfying a number of objectives. Objectives of interest to factory managers include meeting customer commit dates, keeping work in process low, and maintaining a low finished goods inventory. Sometimes these objectives conflict with one another. For example, work in process and finished goods inventories can be used as a buffer against missing customer due dates. As a result, a schedule optimized only with respect to due dates will likely produce large inventories. $\mathcal{ABES}$ supports optimization with respect to a combination of objectives, where the user selects the relative weights of each. Each schedule proposed is then evaluated according to the composite cost function, and the schedule with the lowest cost is retained for actual use.

There are two main tasks a scheduler must perform. First, it must decide how to handle information about new orders for products. New operations must be inserted into an existing schedule. One way to handle this situation is to reschedule from the start, disregarding the schedule which has been in use up to the present. The alternative is to actually make use of the existing schedule, considering the new information as simply a modification. We prefer the second approach, though $\mathcal{ABES}$ also supports the first.

The second principal task of a scheduler is to react when the actual shop deviates from the predicted schedule. Deviations may be in the form of machine breakdowns or simply processing times that differ from those anticipated. In response we expect the scheduling algorithm to find a modified schedule. This process is called "reactive scheduling," to differentiate it from classical, or "predictive" scheduling.

## 3.1 Inserting New Batches into an Existing Schedule

Consider the schedule with which the new batch must be integrated. It consists of a sequence of operations which must be performed in order. It can be represented by an acyclic directed graph like that shown in Figure 3. An arrow from operation $A$ to operation $B$ indicates that operation $A$ must be completed before operation $B$ may begin. This precedence relationship

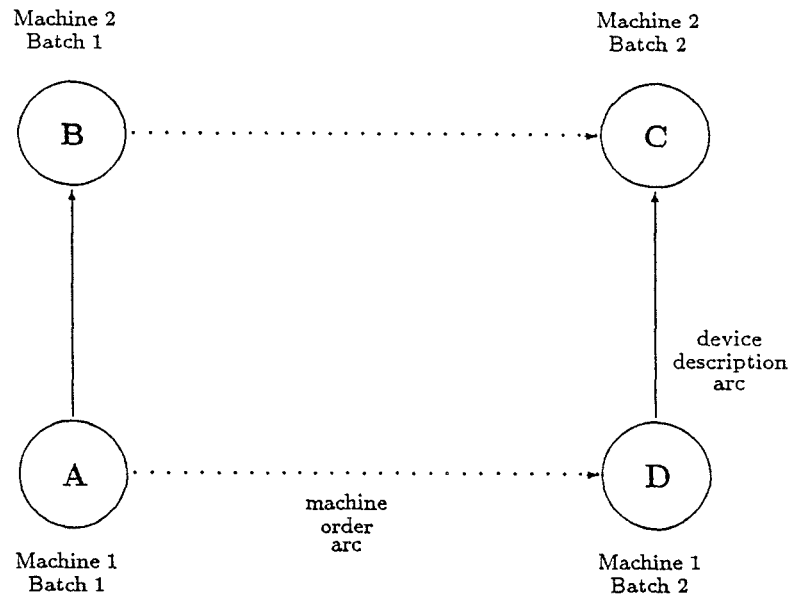



Figure 3: Graph showing *device description arcs* and *machine sequence arcs*.

can arise from two sources: operations $A$ and $B$ may require virtual machines with a certain physical machine in common and, hence, cannot be executed simultaneously, or the two operations may be successive steps in manufacturing a single item. For example, a PC board must be etched (operation $A$) before parts are placed on it (operation $B$). In the figure, precedence arcs of the first type are labeled "machine order precedence," and those of the second type are labeled "device description precedence." Clearly, the graph must be acyclic; no operation in a precedence cycle can ever be executed.

Device description precedence arcs are fixed by the description of a device. However, the machine order precedence arcs may be assigned freely, and any given assignment corresponds to a schedule. To insert new batches into a schedule, then one must reassign the machine order precedence arcs to include all operations in the new batch.

When inserting new batches, it is desirable not to increase the cost of currently scheduled batches. This motivates the following heuristic algorithm, which is used to perform the insertion:

```
algorithm INSERT_NEW_BATCH

    while (some operation in new batch is not in schedule)

       OP = next unscheduled operation in new batch;

       if (OP can be inserted without
                increasing the schedule cost)
          put OP in schedule at zero cost;
       else
          put OP in schedule at minimum cost;
       endif

    end_while

end_algorithm INSERT_NEW_BATCH
```

## 3.2 The Optimization Procedure

As mentioned above, we view schedule optimization essentially as a reactive function. That is, the schedule is held constant until a change takes place. Changes are either new batch arrivals or schedule deviations. When new batches arrive, the schedule is perturbed in a suboptimal manner by the "insert" algorithm described above. When schedule deviations take place, the schedule is perturbed by modifying the projected start and complete times for affected operations. The perturbed schedule does not alter the order of operations scheduled at the machines; it modifies only the projected times. Hence, the goal of the

optimization function is to move a schedule which is already "close" to optimal toward a truly "optimal" schedule.

We must devise a method for improving a schedule. To this end we propose the use of an algorithm called "simulated annealing." This algorithm, which has received a great deal of attention in the research literature [8, 9], is a general purpose method for combinatorial optimization. We present only a brief description of the simulated annealing algorithm, referring the interested reader to the literature cited for its analysis.

We wish to find one of a large but finite number of schedules which has lowest cost. There may be more than one schedule with this cost, and in that case any of them will do. Simulated annealing treats the schedules as states in a discrete time finite state Markov chain. The transition probabilities of the Markov chain are selected so that the state drifts toward the lower cost states.

The simulated annealing algorithm works as follows. An initial state, $S_0$, (the initial schedule) is chosen at random. Next, a nearby "trial" state, $S_1^{trial}$, is chosen by some probabilistic rule, and the two are compared. If the trial state has lower cost, it is accepted, with probability 1, as the new state of the Markov chain. Otherwise it is accepted with a lower probability determined by the relative costs. More precisely,

$$\begin{aligned} Pr\{S_{k+1} = S_{k+1}^{trial} | S_{k+1}^{trial}, S_k\} &= \begin{cases} 1 & \text{if } C(S_k^{trial}) < C(S_k) \\ e^{-[C(S_{k+1}^{trial}) - C(S_k)]/T_k} & \text{otherwise} \end{cases} \\ Pr\{S_{k+1} = S_k | S_{k+1}^{trial}, S_k\} &= 1 - Pr\{S_{k+1} = S_{k+1}^{trial} | S_{k+1}^{trial}, S_k\} \end{aligned}$$

Here the value $T_k$ is called the temperature, and the sequence $\{T_k\}_{k=0}^{\infty}$ is called the temperature schedule. If the trial state is only accepted when it has lower cost, the algorithm will remain in any local minimum it may find. This is the case when the temperature is zero. A nonzero temperature, however, gives the Markov chain the opportunity of escaping from local minima. In fact, the temperature schedule can be selected in such a way that the state of the Markov chain converges in probability to the collection of states with globally minimum cost [9].

Application of simulated annealing is the art of creating the random variables, $S_{k+1}^{trial}$ and $S_0$, and selecting the temperature schedule so that convergence to the global minimum is as rapid as possible. There are no systematic methods for selecting these items. Typically, one selects them on the basis of specific knowledge of the application, and of simple experimentation.

In our application, selecting the initial state is easy. As discussed earlier, the optimization algorithm is invoked to bring a good schedule closer to the true optimum. Hence, the existing schedule is the obvious initial state for the Markov chain.

The selection of the trial schedule is not as immediate. We seek a schedule, $S_{k+1}^{trial}$, which is close to the current schedule, $S_k$. If there is only one machine, the schedule is the

sequence of operations on that machine. The simplest adjacent sequences would be those which are identical except that two adjacent operations are interchanged. With multiple machines, we may choose to interchange two of the operations on a given machine. We may also choose to select a particular batch, and interchange each operation in that batch with the operation immediately following it. Other selections are also possible, but these are the ones we have chosen because of their simplicity.

In ABES, there are actually two such Markov processes evolving in a nested fashion. The high level process proposes as a trial state a change in some process flow assignment. The low level process then searches for a fixed number of iterations for the best operation sequence given that flow assignment. Finally, the high level process decides whether to keep the proposed change, depending on whether the cost improved or got worse during the low level iterations.

Finally, the temperature schedule must be selected. We have chosen a simple adaptive temperature scheduling method. It makes changes in the temperature sequence based on the current progress of the optimization. The magnitudes of the cost differentials, $c(S_{k+1}^{trial}) - c(S_k)$, from the previous iterations are averaged. The temperature is then set to a fixed fraction of this average. The user is able to change the magnitude of the fraction.

The evolution of the cost of the schedule sequence, $c(S_k)$, is graphed on the computer screen while the computation is taking place. Because the best schedule obtained so far is always retained, the user is encouraged to experiment with the temperature control. This enables him to feel comfortable with progress of the search process. At any time during the optimization, the user can stop the iterative procedure and use, view, or manipulate the best schedule found up to that point.

From a practical viewpoint, one of the attractive features of this type of algorithm is that it continually improves the schedule. Although a global optimum is only expected in an asymptotic sense, the schedule after a finite number of annealing steps is likely to be better than the original one. Hence, even when limited computer time is available, one can use all the available time and benefit from the limited improvement that results.

# 4 ABES FEATURES

In section II, we presented a brief description of the factory, which was used as a model for the development of the scheduling software $\mathcal{ABES}$. The goal was to incorporate the different real life aspects of job shop scheduling into $\mathcal{ABES}$ and then by applying an existing combinatorial optimization technique, generate an optimal schedule. In section III, we presented the annealing based optimization algorithm which was used in $\mathcal{ABES}$. In this section we give a detailed description of $\mathcal{ABES}$ features.

Since this is not an online real-time scheduler, each generated schedule has an *active*

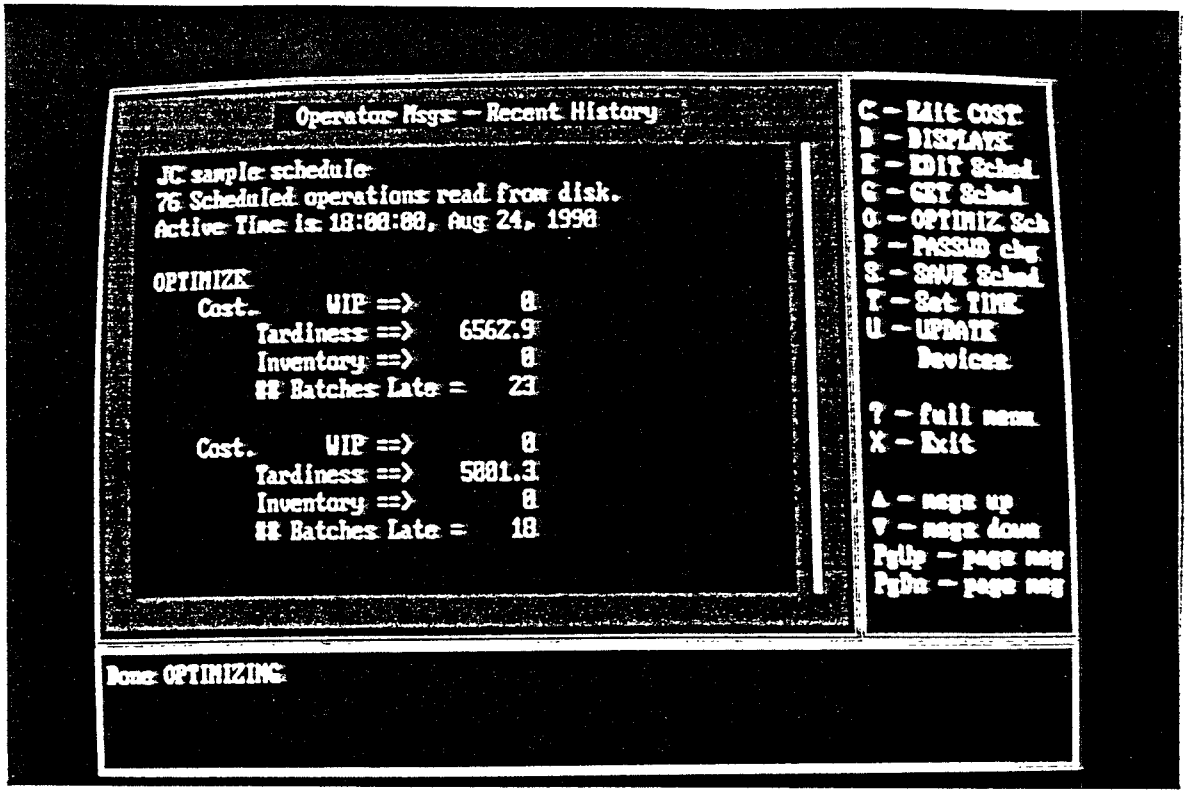



Figure 4: ABES Main Menu

*time* associated to it. This active time represents the time origin for operation scheduling and can be specified at the onset of a new schedule generation. The current clock time is used as the default active time. On generation of an optimal schedule, the factory status can be updated at regular time intervals by incrementing the active time and updating the operation status on all machines. The optimization procedure is carried out after each update.

There are five main modes of operation of $\mathcal{ABES}$, namely Cost configuration, Schedule edit, Update factory, Optimization and Display. A menu driven user interface, shown in Figure 4, allows the user to selct one of the above mentioned modes. In the following subsections we give a brief description and menu selections for each of these modes. The other options available to the user include: Get new schedule from or save present schedule to a binary file, set active time and change password. The password feature is added for security, so as to allow only certain users to be able to save a schedule.

## 4.1 Cost Configuration

The optimization criteria used in $\mathcal{ABES}$ is the *schedule cost*. Each batch in schedule has a cost associated to it. The schedule cost is obtained by adding the individual costs of constituent batches. The batch cost has three additive components, namely Work in process (WIP), Tardiness (TARD) and the Inventory (INV) costs. These components are calculated using the following expressions:

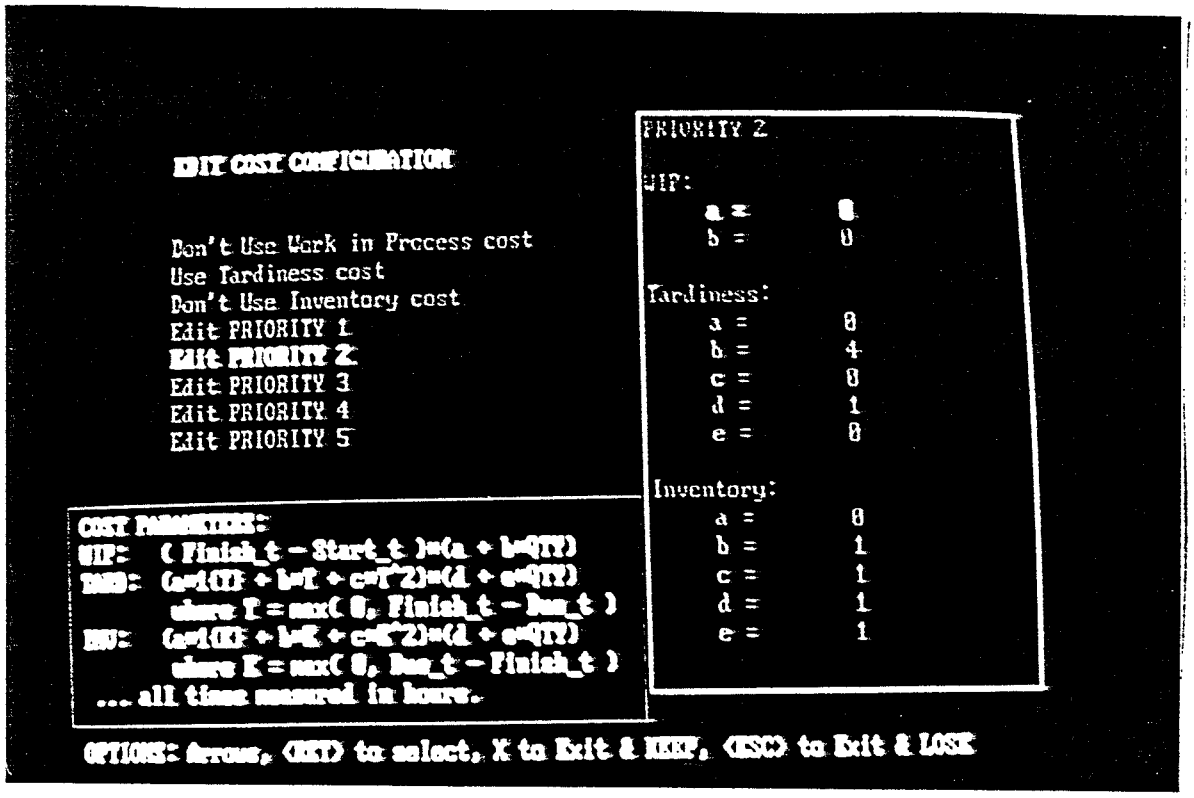



Figure 5: Edit Cost Configuration

(WIP) (FinishTime - StartTime)$*$(a + b$*$BatchQty)

(TARD) $(a + b*T+c*T^2) * (d + e*BatchQty)$
Here T = max(0, FinishTime - DueTime)

(INV) $(a + b*E+c*E^2) * (d + e*BatchQty)$
Here E = max(0, DueTime - FinishTime)

There are five different priority levels available which can be defined by assigning different set of values to the parameters a, b, c, d and e for the three cost components. Figure 5 depicts the edit menu, which allows the user to change the above mentioned parameters for each priority setup. This feature of prioritized batches in a schedule is a common occurrence in real life manufacturing environment. Global enabling or disabling of the three cost components is another useful option which is available to the user. This option is available in a toggle mode and only the selectively enabled components are used for purpose of schedule cost calculation (Figure 5).

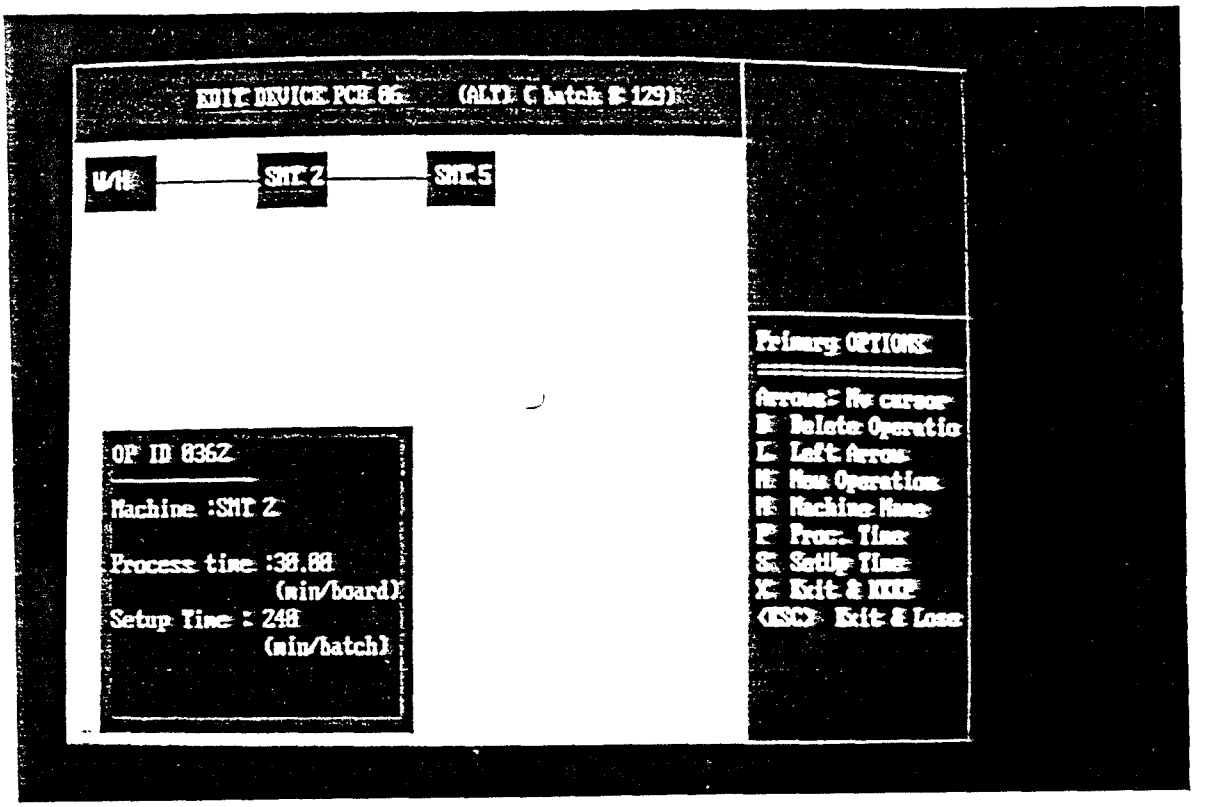



Figure 6: Device Process flow (Alternate) Definition

## 4.2 Update Factory

Figure 7 depicts the Update Factory mode graphical interface. This mode allows the user to initially define the factory and later make changes in the factory setup. For the initial factory setup, the first step is to name all the machines in the factory. The next step is to define the various devices being produced at that factory. The *principal process flow* and set of *alternate flows* along with the processing times (per unit), setup times (per batch) for each constituent operation, device priority and *board family* identification completes the definition of a device. These processing and setup times are used to calculate the batch operation timings for a schedule generation. The process flows are defined graphically on the screen as shown in Figure 6. The user can define new devices, remove devices from the setup and edit process flows or processing parameters during a session. All these changes can then be incorporated into the existing schedule. In Figure 7, devices with alternate process flows are marked with an asterisk.

## 4.3 Edit Schedule

In this mode, a menu driven user interface, shown in Figure 8, allows the user to make a new schedule or edit the present schedule, which already exists in the system. The user can add batches to or remove batches from the schedule. In case of addition of new batches, the user is prompted for the batch size and the due date for that particular batch. The processing

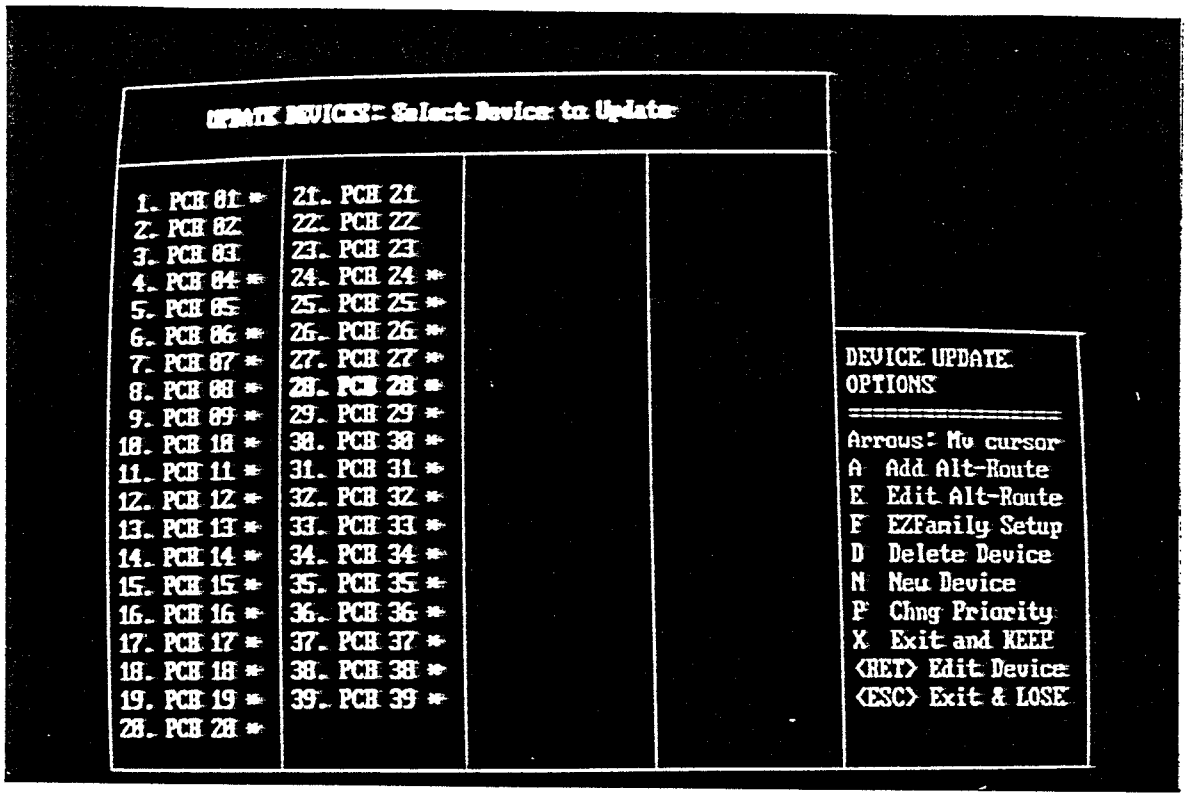



Figure 7: Update Factory Mode

parameters for the various operations are calculated using the parameter definitions for the particular device in the Update Factory mode. A batch is removed from the schedule either if all the operations are done or that particular order is no longer needed.

The user can also edit certain other parameters of different batches, namely earliest start time, priority, batch size, due date, process flow, processing times and operation status. In certain cases, for various reasons, the process flow or processing times for a particular batch are different from the ones defined for that device. Also in a reactive scheduling environment, we need to update the status of different operations so as to be able to reoptimize the schedule and reschedule the tardy operations. Thus the last three options are important to take into account the vagaries of a real life manufacturing environment. The other options are self explanatory and form an integral part of manufacturing. The available batch edit options are shown in Figure 9.

## 4.4 Optimization

The initial schedule is generated by inserting *batch operations* in the machine-idle slots in the schedule for an earliest possible finish. This generally does not lead to an optimal schedule as the generated schedule depends on the order of addition of new batches. Also if changes have been made in the schedule in the *Edit Schedule mode* the generated schedule may not be optimal. In such cases the optimization procedure is invoked so as to generate an optimal schedule. The user can define the time window in which to optimize the schedule.

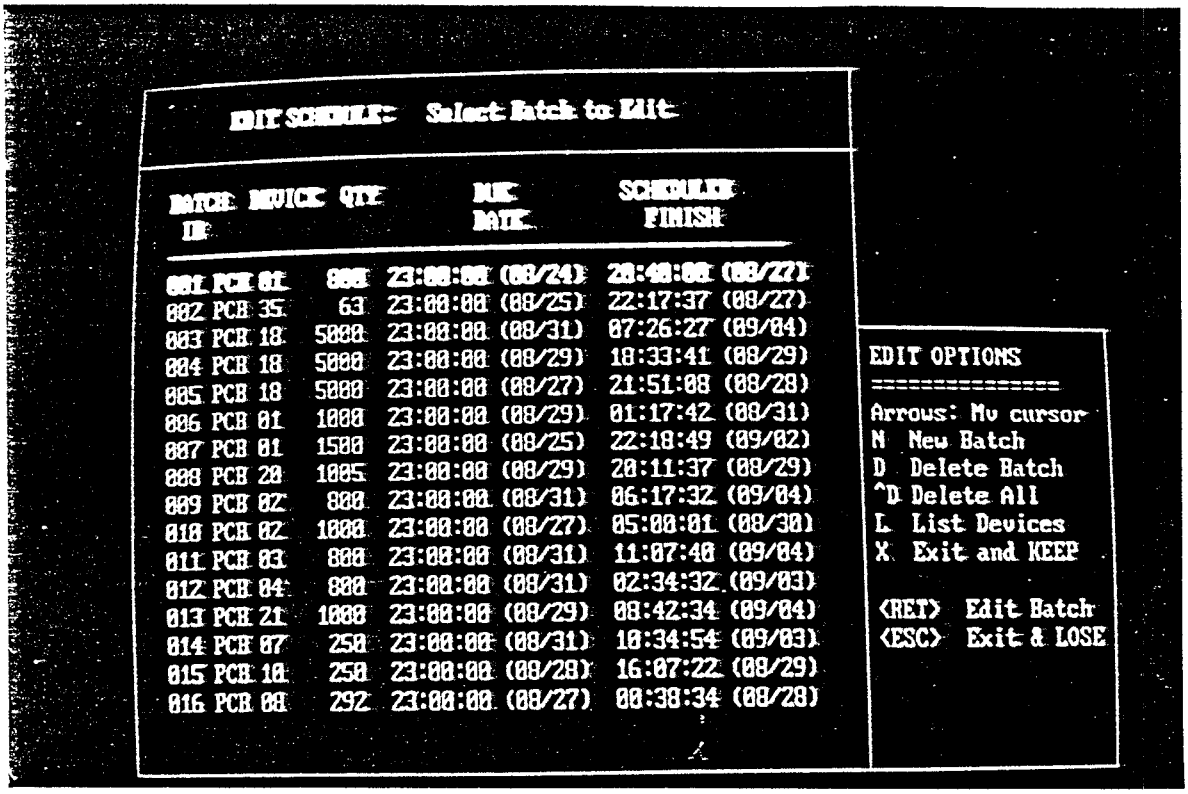



Figure 8: Edit Schedule Mode

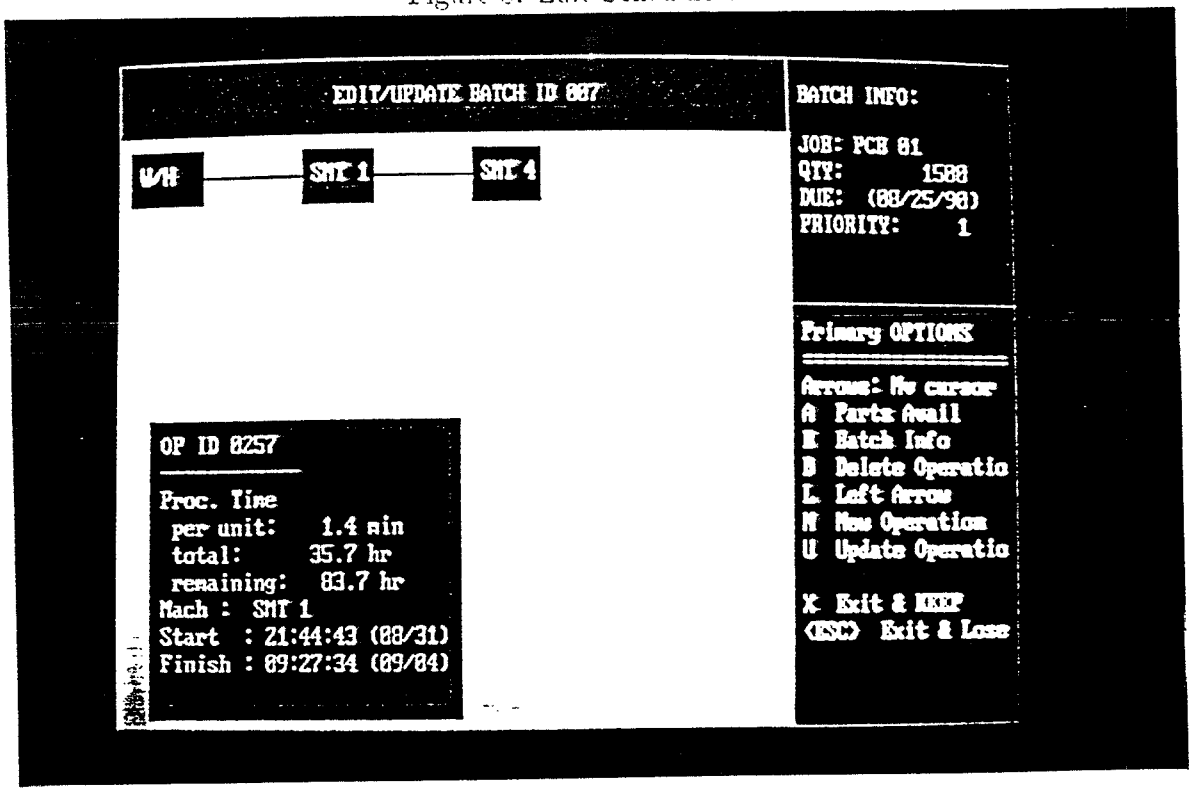



Figure 9: Edit Batch Parameters

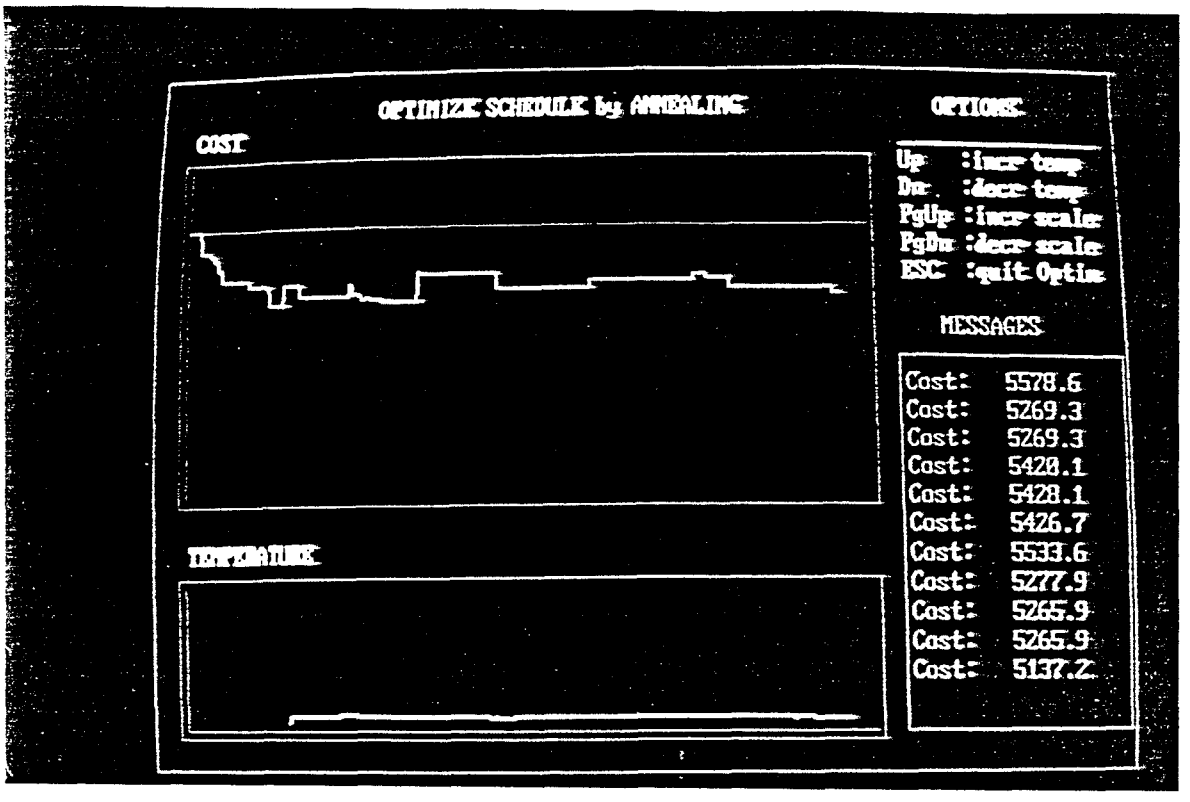



Figure 10: Optimization Process

This allows for constrained rescheduling of a partial set of operations. Obviously the user can define the time window starting at the active time and extending to the infinite future. Figure 10 depicts the optimization process. The user can increase or decrease the annealing temperature to directly control the optimization procedure.

## 4.5 Displays

Once an optimal schedule has been generated, different display options are available to the user. The user can opt for a GANTT chart (Figure 11) depicting the machine utilization or a Work-in-process display for different devices. The other options are to display the schedule by devices or schedule by machine. Figures 12 and 13 depict the two latter options respectively. Figure 14 shows another option to view the different statistics of the present schedule.

Other than these main modes of $\mathcal{ABES}$ operation there are certain other added features. The factory description along with the schedule can be saved in a binary format on a computer disk and can be retrieved later. Thus in principle, $\mathcal{ABES}$ can be used for scheduling different factory setups or can handle different schedules for the same setup at any given time. The feature of *active time* allows the user to make schedules in the future, past or the present. Also if by incrementing the active time, the schedule is updated frequently, this will lead to a reactive real time scheduler.

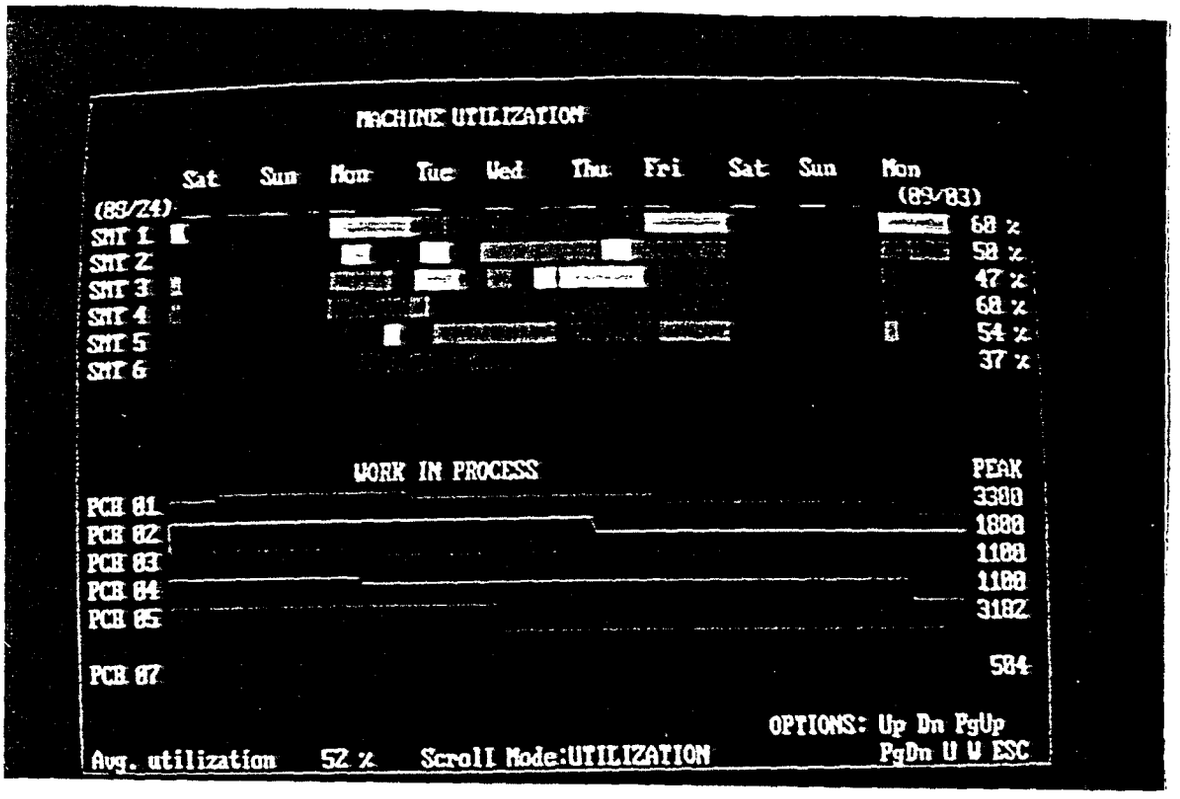



Figure 11: GANTT chart and WIP

# 5 ONSITE EVALUATION

The *ABES* was developed as an experimental scheduling system. The aim was to incorporate the vagaries of real-life manufacturing environment into the system. The issues of shifts, 5/6/7 day work weeks, board families, continuous flow representation while maintaining batch flow among lines, optimal schedule generation, changes in the factory status, addition/removal of batches from schedule etc. were incorporated in the software.

Our prior method of scheduling our surface mount lines required gathering the current production status, future requirements from several scheduling groups, and any new additions or changes to process flow for each device. The new scheduling system has turned a complex, manual task into a more organized automated approach. By loading the attributes, by device, we have eliminated many of the scheduling errors associated with timing and flow. The new system does an excellent job of searching out the optimum schedule to help us increase our throughput and minimize any delinquencies.

The software was developed using the programming language C and runs on a desktop computer. The user friendly graphical interfaces lead the user, step by step, through the different modes of *ABES*. In terms of the time involved for an optimal schedule generation, the initial factory definition, for the Custom Manufacturing Business, took some time, but once that was done it took a trained user about 10 mins to generate an initial suboptimal schedule. For a typical work load of 30 batches which translated to about 75 scheduled operations, the optimization procedure took about 18 minutes for 5000 iterations.

```
**************************************************************************
************************** PCB 01  ***********************************
**************************************************************************
Operation  Machine  Setup  Batch  Start                 Finish
   ID        ID     Time    Qty   Time                  Time
--------------------------------------------------------------------------
--------------------------------------------------------------------------
Batch ID: 001
 0380   SMT 4        24    800    18:00:00 (08/24)    20:40:00 (08/27)
                                              DUE:    23:00:00 (08/24)
Batch ID: 007
 0256      (parts)    0   1500                        18:00:00 (08/24)
 0257   SMT 1       150   1500    08:16:17 (08/29)    19:59:09 (08/30)
 0262   SMT 4        24   1500    20:18:49 (08/29)    22:18:49 (09/02)
                                              DUE:    23:00:00 (08/25)
Batch ID: 006
 0251      (parts)    0   1000                        10:00:00 (08/25)
 0252   SMT 1       150   1000    08:27:43 (08/28)    08:16:17 (08/29)
 0253   SMT 5        24   1000    15:57:42 (08/29)    01:17:42 (08/31)
                                              DUE:    23:00:00 (08/29)


**************************************************************************
************************** PCB 02  ***********************************
**************************************************************************
Operation  Machine  Setup  Batch  Start                 Finish
   ID        ID     Time    Qty   Time                  Time
--------------------------------------------------------------------------
--------------------------------------------------------------------------
Batch ID: 010
 0369      (parts)    0   1000                        18:00:00 (08/24)
 0373   SMT 5        78   1000    11:40:01 (08/28)    15:50:30 (08/29)
 0374   SMT 2       210   1000    19:40:01 (08/28)    05:00:01 (08/30)
                                              DUE:    23:00:00 (08/27)
Batch ID: 009
 0268      (parts)    0    800                        18:00:00 (08/24)
 0271   SMT 5        78    800    01:41:06 (08/31)    00:13:28 (09/03)
 0272   SMT 2       210    800    03:37:32 (09/03)    06:17:32 (09/04)
                                              DUE:    23:00:00 (08/31)
```

Figure 12: Schedule by Device

```
*************************************************************************
*************************** MACH: SMT 1     ***************************
*************************************************************************
Operation Job  Setup  Batch  Batch    Start              Finish
   ID          Time    ID     Qty     Time               Time
*************************************************************************
  0386  PCB 21 240     031     450    18:00:00 (08/24)   20:47:09 (08/27) L
        DUE:23:00:00 (08/24)
  0396  PCB 19  90     019     220    21:14:09 (08/27)   07:42:43 (08/28)
  0252  PCB 01 150     006    1000    08:27:43 (08/28)   08:16:17 (08/29)
  0257  PCB 01 150     007    1500    08:16:17 (08/29)   19:59:09 (08/30)
  0296  PCB 21 240     013    1000    21:11:09 (08/30)   08:42:34 (09/04) L
        DUE:23:00:00 (08/29)

*************************************************************************
*************************** MACH: SMT 2     ***************************
*************************************************************************
Operation Job  Setup  Batch  Batch    Start              Finish
   ID          Time    ID     Qty     Time               Time
*************************************************************************
  0377  PCB 07 120     029     254    18:00:00 (08/24)   21:40:44 (08/24)
  0314  PCB 09 180     017     422    00:40:44 (08/27)   07:17:37 (08/27)
  0392  PCB 35 360     002      63    13:17:37 (08/27)   22:17:37 (08/27) L
        DUE:23:00:00 (08/25)
  0337  PCB 13 240     022     200    02:17:37 (08/28)   07:12:51 (08/28)
  0374  PCB 02 210     010    1000    19:40:01 (08/28)   05:00:01 (08/30) L
        DUE:23:00:00 (08/27)
  0341  PCB 12 240     023      84    09:00:01 (08/30)   11:04:01 (08/30)
  0345  PCB 14 240     024     119    12:16:01 (08/30)   15:11:41 (08/30)
  0290  PCB 04 180     012     800    18:11:41 (08/30)   02:34:32 (09/03) L
        DUE:23:00:00 (08/31)
  0272  PCB 02 210     009     800    03:37:32 (09/03)   06:17:32 (09/04) L
        DUE:23:00:00 (08/31)
```

Figure 13: Machine Schedule

```
*********************************************************
*              SCHEDULE STATISTICS DISPLAY              *
*********************************************************
Schedule Active Time: 18:00:00, Aug 24, 1990

Number of batches...
           in schedule :     31
           tardy       :     18
Mean Tardiness          :  33.9 Hours

Average tardiness of tardy batches :     58.3 Hours

COST    WIP ==>             0.0
  Tardiness ==>          4994.1
  Inventory ==>             0.0


*********************************************************
```

Figure 14: Schedule Statistics

Therefore, it takes about 28 minutes to get a near optimal schedule. This system is under evaluation at the Texas Instruments assembly facility in Johnson City. It is difficult to quantify the system performance but the initial response to the system has been quite favorable.

**Acknowledgement:** We gratefully acknowledge the assistance and information provided by Texas Instruments, Inc. Special thanks to Gene Lamm, Andy Lobsenz, Barbara Papas, Richard Herrod, and Margaret Berbari.

# References

[1] P.J.M. van Laarhoven, E.H.L. Aarts, and J.K. Lenstra, "Job Shop Scheduling by Simulated Annealing," Philips Research Laboratories, Eindhoven, preprint, 1988.

[2] P.S. Ow, and S.F.Smith, "Viewing Scheduling as an Opportunistic Problem Solving Process," in R.G.Jeroslow(ed.) *Annals of Operations Research: Approaches to Intelligent Decision Support.* Baltzer Scientific Publishing Co., 1987.

[3] W. Jeffrey and R. Rosner, "Optimization Algorithms — Simulated Annealing and Neural Network Processing," *Astrophysical Journal, Part 1*, Vol. 310, 1 Nov 1986.

[4] M. Salimian, *A New Algorithm for the Three Machine Flow Shop Problem*, PhD Dissertation, *The University of Oklahoma*, 1988.

[5] M.J. Maddox, *Scheduling a Stochastic Job Shop to Minimize Tardiness Objectives*, PhD Dissertation, *The University of Michigan*, 1988.

[6] K.R. Baker, *Introduction to Sequencing and Scheduling*, J.Wiley & Sons, New York, 1974.

[7] Dempster M.A.H., J.K. Lenstra, and A.H.G. Rinnoy Kan, eds., *Deterministic and Stochastic Scheduling; Proceedings of the NATO Advanced Study and Research Institute on Theoretical Approaches to Scheduling Problems*, D. Reidel Publishing Co., 1982.

[8] B. Hajek, "A Tutorial Survey of Theory and Applications of Simulated Annealing," $24^{th}$ *IEEE Conference on Decision and Control*, Fort Lauderdale, FL, Dec. 1985.

[9] D. Mitra, F. Romeo, and A. Sangiovanni-Vincentelli, "Convergence and Finite-Time Behavior of Simulated Annealing," $24^{th}$ *IEEE Conference on Decision and Control*, Fort Lauderdale, FL, Dec. 1985.

[10] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, "Equations of State Calculations by Fast Computing Machines," *J. Chem. Phys.*, 21, pp. 1087-1091, 1953.

[11] J. Blazewicz, "Selected Topics in Scheduling Theory," *Annals of Discrete Mathematics*, 31, pp. 1-60, 1987.

[12] K.L. Musser, J.S. Dhingra, G.L. Blankenship, "Optimization Based Job Shop Scheduling", to appear.