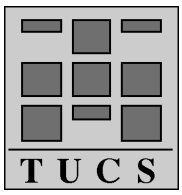
Supporting Production Planning by Production Process Simulation

Tommi Johtela
Jouni Smed
Mika Johnsson
Risto Lehtinen
Olli Nevalainen



Turku Centre for Computer Science
TUCS Technical Report No 86
January 1997

ISBN 951-650-931-2
ISSN 1239-1891

Abstract

An interactive production planning system for electronic industry is described. PCB component printing is used as a case study, but the method can be adapted to other similar environments of generalized flexible flow line. The system simulates the production from an initial situation to a given moment in the future. The input defines the product batches, their allocation and sequence for each machine and the due dates. The output includes summaries of the production period including the lateness of the batches and machine workload charts. The user can reconsider the allocation and sequencing of the batches and repeat the simulation and update operations to find a better balancing of the workload.

Keywords: production planning, scheduling, simulation, line balancing, PCB assembly, man-machine interaction

TUCS Research Group
Algorithmics

Introduction

In a typical production process, the products have to pass several distinct phases (or machine banks), and be processed in one of the machines. The task is to solve the routing of the jobs through phases and the sequencing within each machine in order to fulfill predefined criteria measuring the suitability of the schedule. In this paper we consider such an environment used in printed circuit board (PCB) assembling.

The scheduling problem is too complex to be solved optimally; even special cases of the problem are $\mathcal{NP}$-complete [1]. Although many different theoretical notions and solution methods for the problem have been put forward, practical solutions, which can be used in a real production plant, have for the time being been rare [2, 3, 4]. Theoretical notions tend to oversimplify—or sometimes even ignore—crucial factors of the actual production process.

Instead of analyzing the scheduling problem theoretically, the problem can be approached from a different point of view; we can simulate reality by building a model based on the actual production environment. The simulation model can then be used when solving the key elements of the problem [2, 5, 6]. This approach is usable when we want to construct a system which helps us to understand a specific case of the scheduling problem.

The advantages of simulation methodology for modeling flexible manufacturing systems (FMS) can be highlighted [5]:

- Simulation can reduce the risk of installing an FMS which may not provide sufficient flexibility.
- A simulation model can represent characteristics of an FMS more realistically. It may incorporate the complex interactions which may exist between various variables, for example, loading strategy at buffers and at workstations.
- Alternative FMS designs can be evaluated easily in a controlled environment.
- The ability of a computer simulation model to address directly the measures of performance typically used in FMS evaluation helps to calculate the same measures of system performance for hypothetical FMS configurations as used in judging the real systems.

In the system of the present paper we have chosen the simulation approach: we have analyzed the production process of an existing production

plant and built an interactive simulation system based on the knowledge achieved by the analysis. The system is a computer program which simulates the production process by using a built-in model of the operation principles of the factory. Furthermore, we have concentrated on creating a clear representation of the problem and developed an interactive graphical user interface.

There are few commercial packages currently available for scheduling the PCB component printing [7]. Our system differs from these in many ways. We are developing an easy-to-use interactive tool which helps the user to evaluate quickly the effects of the changes in the schedule. The system should work as a "think pad" for the planner. Much of the simple decisions are left to him, like the grouping of the products and component-to-machine mappings. The interface should resemble the familiar way of working with pen and paper.

Although the system was designed to be a tool for the production planning, it also serves as a testbench for the research of algorithms and it can be used for analyzing different theoretical scheduling algorithms. In other words, it is a natural meeting point for the two different approaches—mathematical programming and simulation—described above.

The structure of the work is as follows: A brief review of previous research on the subject is given in section 2. The actual production process and production planning are discussed in section 3. The production planning system is described in section 4. Section 5 summarizes our experiences with the system and directions for further development. Concluding remarks appear in section 6.

Flow-Shop Scheduling

Scheduling problems can be classified according to the type of production into two classes: in a *job shop* environment, machines performing similar operations are grouped together, and in a *flow shop* environment, the machines are organized according to a specific processing order [8]. In this paper we concentrate only on variations of the flow shop environment.

There are two problems to be considered in the flow shop environment, in which there is only one machine in each stage and each product is processed only once in each phase. In the *assembly line balancing problem* we must find a solution which divides the processing load between machines so that the chosen balancing criterion will be optimized. If several different products are manufactured in the same line their sequence has an effect on the throughput of the line because the time demands can vary considerably

between different schedules. In the *flow shop sequencing problem* we must find an optimal processing sequence based on some optimization criterion or criteria. A special case of flow-shop scheduling problem, called *permutation schedule flow-shop*, where the order of the products (batches) remains the same throughout the phases, have been given for example by Kim *et al.* [3]. *Flexible flow*, in which there is one machine per phase with a possible skip of some phases, has been discussed by Shmoys *et al.* [9].

Network flow-shop [10] and *hybrid flow-shop* [4, 11] are generalizations of flow-shop, where each phase may comprise several identical machines. Set-up times or lines connecting machines are not considered in this problem formulation.

The *flexible flow line* environment [12] comprises several machine banks or production phases. The machines in a single machine bank are identical with each other. A product can therefore be processed in any machine belonging to the machine bank, or it can skip over the phase ("flexibly") without being processed at all. Each product must pass the phases in a predefined order (thus the environment is called "flow line"). The machine set-up time between different products is short, and therefore ignored. Thus, the processing time is a function of the processed product and the phase. A flexible flow line is *non-preemptive*, i.e. the processing must be finished before the machine can switch to the next product. When the machine has processed the product, a transport mechanism takes it immediately to the buffer of the next machine.

In this paper we consider a production environment that is a generalization of the flexible flow line, called the *generalized flexible flow line* (GFFL) [13]. The GFFL environment also comprises successive machine banks, but the type of machines can vary even inside a particular machine bank (unlike in FFL). The machine type defines the speed of the machine, i.e. the average processing time of a given product in a machine. Therefore, the processing time in GFFL is a function of the product and the machine type. Several different products are manufactured, but they are grouped together in *batches* in order to minimize the set-ups between different types of products. Set-up times are also taken into account (unlike in FFL).

Expressed in the $\alpha|\beta|\gamma$ notation described by Lawler *et al.* [14] (see also [15]), the scheduling problem in the GFFL environment is of the type $FMPM|p_{ij}, d_i|\sum T_i^2$ which stands for a flow-shop with m machines, no preemption allowed, no precedence relations, all jobs ready for processing, processing demands differ, deadlines, batches. Optimization criterion is minimizing the total sum of squared tardiness of the batches.

The products are processed in a predefined order (*schedule*). A single product is processed at most once in the same phase. However, it can possibly

skip over one or more phases. The processing steps of the product are known in every phase. Similarly, we know for every machine how many processing steps it can do in a time unit on average. Although the precise processing time is stochastic, it can be estimated accurately enough on the basis of this knowledge.

When the batch in the machine changes, the machine must be set up for the next batch. Thus, the time period needed for the set-up operations depends on the machine, the processing order of batches (i.e. the sequence) and the state of the production process itself. It can be estimated by grouping the products in *families* so that the products belonging to the same family do not differ much from each other, and by defining beforehand the set-up times between the families [16].

The production is continuous in GFFL, and the production planning repeats itself periodically. The production is viewed as production (or planning) periods. In the beginning of each period we know the amount of processed products in the *buffers* and the need of finished products at the end of the period. The *production plan* contains information about the product, the amount to be manufactured and the due date (by which the processing must be finished).

The scheduling problem in a GFFL environment can be stated as follows: *We must organize a schedule for a given time period so that the due dates are met.* As a secondary criterion we want to *minimize the size of the buffers* and *the length of the makespan.* In addition to these, *the performance ratio must be balanced* in every phase in order to prevent starvation and blocking, which can slow down the production. Finally, we must consider the *set-up times.* The unnecessary set-ups between different products should be avoided, because they expose to errors and cause delays. Although many researchers have suggested makespan as an optimization criterion (cf. [3, 12]), it cannot be used in the GFFL environment because of the dynamic nature of the production process [17].

The problem of scheduling—even if the objective is just to minimize the makespan—is $\mathcal{NP}$-complete [1]. There are $n!$ possible unique permutations of the n jobs for a given machine; with m machines there are a total of $(n!)^m$ possible solutions to the problem in the flow shop model [18]. In the FFL and especially in the GFFL environment the problem is far more complex. For this reason fast heuristic algorithms that find usually good but not optimal schedules have been developed.

Wittrock [12] decomposes the FFL problem into three subproblems and solves them heuristically. The first subproblem, *machine allocation*, is to determine which products will visit a given machine of a machine bank. The second subproblem, *sequencing*, is to specify the order in which the products

should enter the line. The third subproblem, *timing*, is to decide the times at which the products should enter the line.

To summarize, GFFL has the following properties:

- set-up times important
- n phases and each phase has M_n possible different machines
- possible skip over phase(s)
- the size of the transportation unit can vary (if line then 1)
- processing time depends on machine and product
- no pre-emption allowed
- no precedence relations
- no permutation schedules
- dynamic arrival of jobs with different due dates
- batches
- optimizing not makespan but, for example, the sum of squared tardiness

The Production Environment

We study the GFFL scheduling problem of the printed circuit board assembly line of Nokia Display Products, Finland. We will concentrate our study on the automatic insertion phase of the production. The assembly line comprises robots which are specialized in mounting components to PCBs, and transporters between the robots. We describe the production process, and after that briefly the actual production planning process in use.

The Production Process

There are several different types of machines in the production plant. The main difference between them lies in the type of components mounted (figure 1). The automatic insertion consists of four successive *phases*, each comprising an individual machine bank. The machine banks are named according to the type of operations: 1) griplet insertion, 2) axial insertion, 3) radial insertion, and 4) surface mounted (SMD) onsertion.

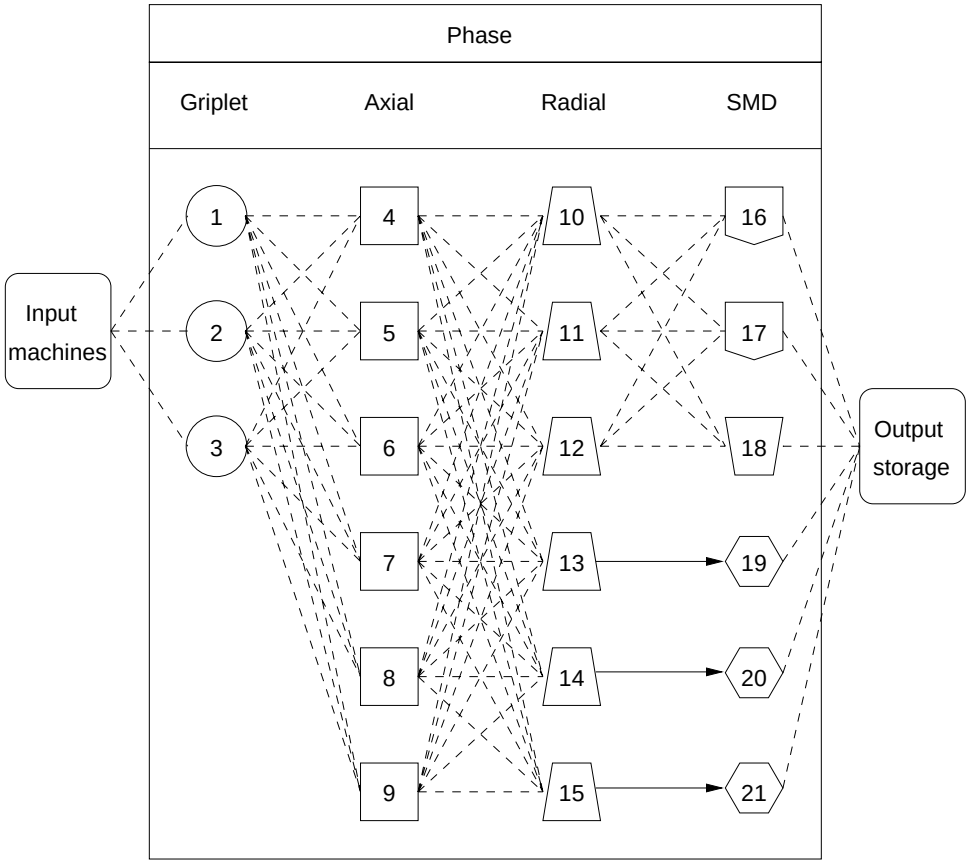


Figure 1: A sample machine configuration

Each machine type has an individual operation speed, which defines how many components the machine is capable of inserting in an hour on average (table 1). Machine types 4–6 are separated because of the physical restrictions of the machines to insert components, rather than different set-up or processing times. The machines receive the components usually from lockers or feederlines.

Machines can be on *line*; a line leads from a machine to the successive one in the next machine bank. Figure 1 shows three lines, all between radial insertion and SMD onsertion. Lines can be either *fixed* or *logical*. When there is a fixed line, there exists a conveyor belt or some other kind of transporter mechanism, which transports products automatically from one machine to another. A logical line does not presuppose such a physical transporter, but all products from the first machine are processed on the second one as if there were a fixed physical link between them.

Machine type	**Phase**	**Speed** (components/hour)	**Set-up time** (min)	
			small	large
1	Griplet	8000	2	10
2	Axial	11000	5	15
3	Radial	4500	5	30
4	SMD	12500	10	90
5	SMD	12500	10	90
6	SMD	12500	10	90

Table 1: Typical machine data

Same kind of PCBs are grouped together to form a production *batch*. In the production plan, the PCBs can be divided in batches of different sizes, but all PCBs belonging to a certain batch will be processed together in the same machine without preemption. Several machines in the same machine bank can process simultaneously same type of PCBs, but they must belong to different batches. When there are no PCBs to be processed in the machine, the machine is *idle*. The time interval during which a machine is idle between processing two batches is called *idle time*.

Before the processing can begin, we must define for every batch a *route*, i.e. the machines in which the batch will be processed (flow-line), and the timing or the processing period in relation to other batches. In addition, the batch has an individual due date by which it must be completed. The batch is sent to the next machine only after the previous batch on this machine has been completed.

Although the whole batch is allocated on a machine (i.e. to be processed in it), only one PCB at a time can actually be processed in the machine. Therefore all the other PCBs in the batch have to wait meanwhile in a buffer (which can also hold PCBs from other batches at the same time). The processed PCBs are put into a magazine, which after it has been filled is transported to the buffer of the next allocated machine. The same batch can be processed simultaneously on two or more successive machines belonging to different phases.

When the batch to be processed changes, the machine must go through *set-up* operations before it can start processing the next batch. The mechanical set-ups of the machine, the components in its feeder lockers, and the program of the machine must be changed to fit for the new batch. The time needed for the set-up operations depends on the *product family* of the processed and the next batch. If both of these belong to the same family, a *small set-up time* is needed, and if they belong to different families, a *large*

set-up time is needed instead (see table 1).

The batches are divided in families by the physical size of the PCB and by the type of the components (i.e. PCBs which mainly use the same components belong to the same family). In machine type 1 (griplet machines) and 2 (axial machines) the physical size of the PCB defines the family. In machine types 3–6 (radial and SMD machines) PCBs belong to the same family if the change does not require a new set-up for the feeder lockers. Thus, families are local to the machine types, i.e. the family of a PCB may vary from one machine type to another.

Each PCB type has its own insertion program for each phase. The program must be in the memory of the machine before the processing can start. The program is loaded from a disk unless it is already in the memory of the machine before the batch arrives.

Every machine is preceded by a buffer storage. It holds the PCBs which are ready to be processed in the machine. The size of the buffer is not limited, but because of costs and spacial limitations, they have to be kept as small as possible. In a physical line PCBs are transported straight after processing to the next machine on the line. Other machines are followed by magazines, in which the PCBs are put to wait for the transport to the next phase.

Processing time is defined by the machine, the PCBs in the batch, and the type, location and amount of the mounted components, and the presets of the feeder lockers. The processing time is often known from previous production periods, whereas in other cases the code generation system has given an approximation of it.

When a PCB has passed the production line, it is transported to a final storage where it waits for a delivery to manual setting and testing.

The Production Plan

The production is a continuous process, even though the interest is always focused on a certain time period and on the PCBs processed during it. The *production plan* is usually made one week in advance. It states the PCB types and their volumes to satisfy the demand. The demand is influenced by the present storage level and the estimated demand for different PCBs.

The production is commonly started with the most urgently needed PCBs. There are some mass products comprising thousands of PCBs which must be manufactured by the end of the period. In our sample factory, the mass products are mostly routed through fixed lines, and thus the same set of machines process the same type of PCB for several days or even the whole week.

The *production program* defines for each batch the amount and type of PCBs, routing including machines on each phase, and sequencing in each machine. Similarly, for each machine is defined the PCBs to be processed, the sequence of batches, and the size of batches to be processed.

Description of the Interactive Planning System

Before the introduction of the interactive planning system, the planner had to calculate on paper the machine allocation and the sequencing of the batches. The amount of information concerning this problem is large, and the decisions are hard to make, even if they are based on intuition and years of experience. In addition, planning the production without any computational support is slow and subject to human errors. The planner can use his experience and try to find a solution that fulfills the main criterion of scheduling, which is meeting the due dates.

The interactive planning system gives the production planner the kind of support needed in the evaluation and comparison of different solution alternatives. He can allocate and sequence the batches so that they meet the due dates, the idle periods are short, and the machine load is balanced at all time.

The system is based on a graphical user interface (GUI). Allocation and sequencing are represented by simple graphical components which can be easily operated with (see figure 4). The system supports decision making by giving feed-back of the solution (e.g., graphical charts). Thus, scheduling becomes an interactive process between the production planner and the system (see figure 2).

The system was developed with Borland Delphi, and it runs on a PC under Microsoft Windows. Delphi is an object orientated programming environment. The programming language is an extension of Pascal with object orientated features such as classes, inheritance and data hiding.

The production planning system consists of two independent parts: the *graphical user interface* (GUI) and the *simulation engine* ("core"). The simulation engine provides the system with common classes which can be used to construct the production environment. The GUI uses these "services" and interacts accordingly with the user. The structure of the system is depicted in figure 3.

The system we present here is the second version of the production planning system [13]. The first version did not correspond accurately enough to

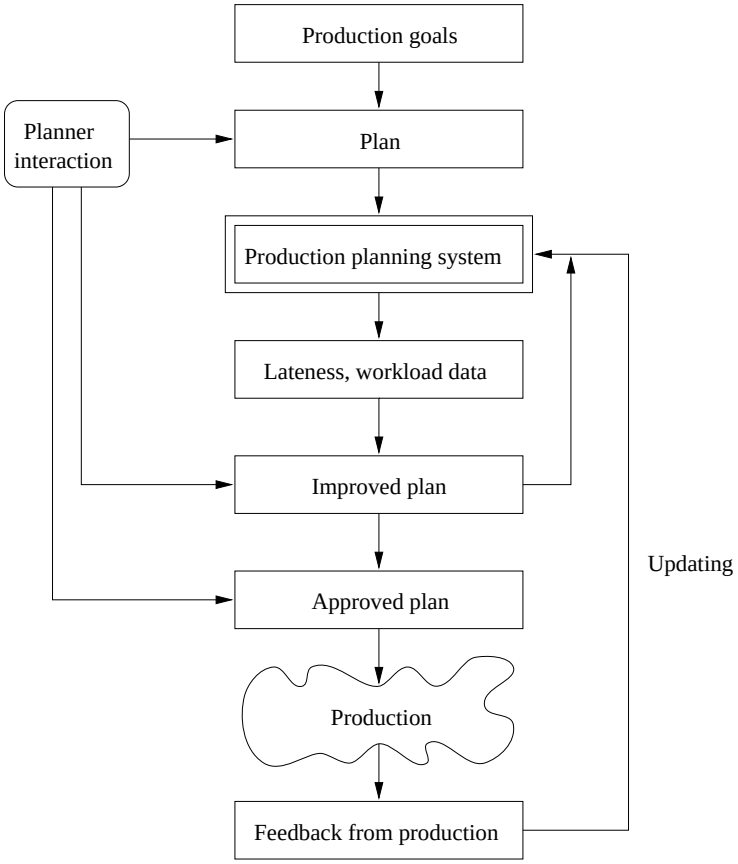


Figure 2: General view of the planning process

the continuous nature of the production and dynamic aspect of the scheduling. Although the production plan is made for a time period of one week, the production does not begin with an empty line, and neither does the line remain empty, when the last batch in the current plan is completed. In addition, usually we do not know the whole production program at the beginning of the planning period. Moreover, the planning is dynamic in the sense that new batches will be inserted gradually during the planning period, i.e. we have a *rolling schedule* [17]. In a rolling horizon framework, the simulation helps to solve the scheduling problem of the immediate decision period, and the problem is then updated and resolved one period later.

The new system is real-time, and thus, when new batches are inserted, the system can be updated instantly to correspond to the current situation. Features of the system include:

- updating the current situation (removing processed batches, changing

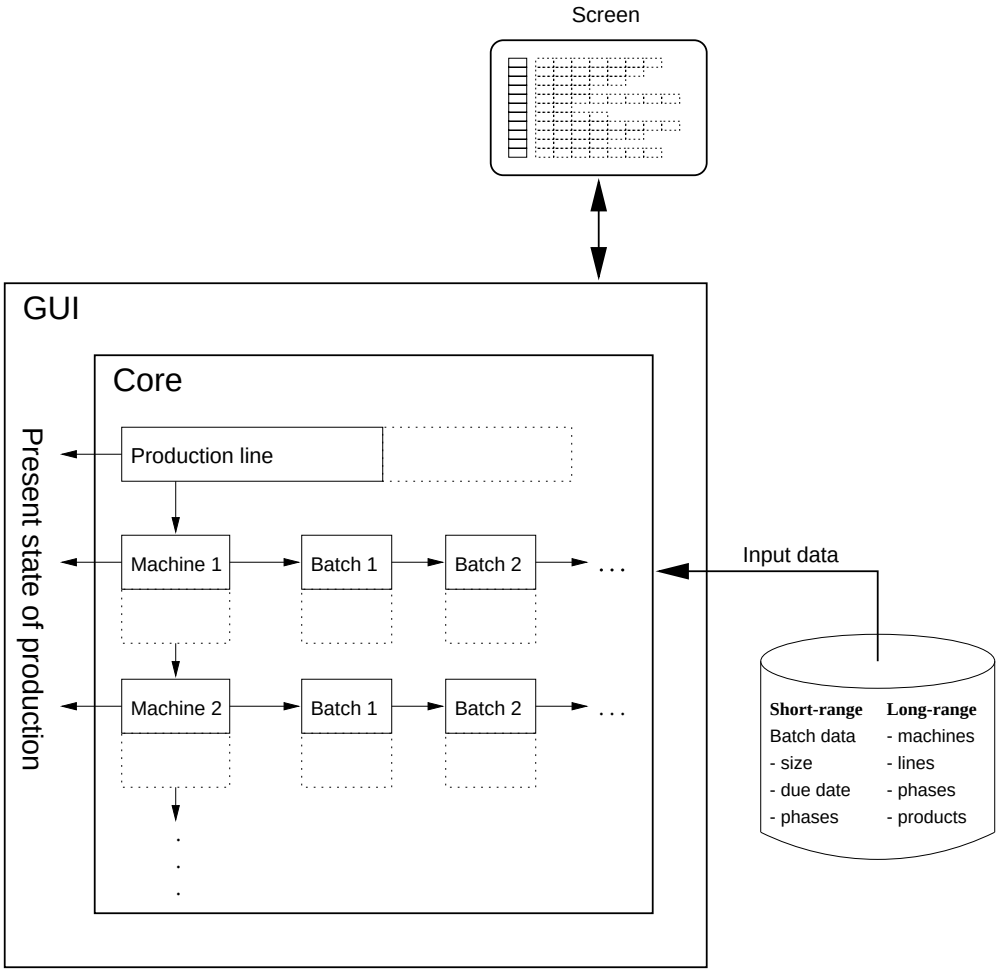


Figure 3: The structure of the system

batch sizes, editing the processing sequences)

- inserting new batches during the production period
- computing the finishing times based on the current situation, and tracking batches which do not meet due dates
- removing and inserting machines and products to the system
- coping with machine break-downs and other problems
- statistical data about the load of machines, idle times and the meeting of the due dates, which characterize the solution and reveals bottlenecks of the production

- testing different machine set-ups (e.g., the effect of additional machines on the production)
- interactive graphical user interface
- testing schedules produced algorithmically

Data Input and Edit

The usefulness of an interactive production planning system depends largely on the user interface. Here special care should be taken to the design and implementation of the editing operations. They should be as simple and clear as possible and comply with the guidelines for GUI design.

The situation of the production process is presented on the screen in the same manner as the production planner used to do it previously on paper in the manual system. The machines are lined vertically, and the batches appear as line segments in the order of their processing sequence. By this representation one can easily manipulate the queue of batches of each machine.

The basic operations of the system include moving batches inside a machine and inside a machine bank, inserting or removing a batch. Additional data like the size, the due date, the starting and finishing time of processing are hidden in default and shown if wanted. Several different forms for the data input are supported, and they can be launched from the main window.

The Main Window

The situation of the production is shown in the main window (figure 4). It represents for each machine the batches to be processed. The machine banks (i.e. phases) are indicated, and the user can edit information regarding the machines and machine banks. New batches can be inserted interactively (figure 5).

The batches can be moved freely inside a phase; a batch may be moved from one machine to another (allocation), or the order of batches can be changed (sequencing). The user can fix certain batches to machines. The editing of the fixed batches is restricted; they cannot be moved and no batches can be located before them. In addition, the physical lines make the editing somewhat more complicated: inserting a new batch to a line machine is restricted, and operations in the one end of the line are reflected on the other end, too.

Additional information is included also in the background and in the font color of the batch boxes: font color implies the family and background color

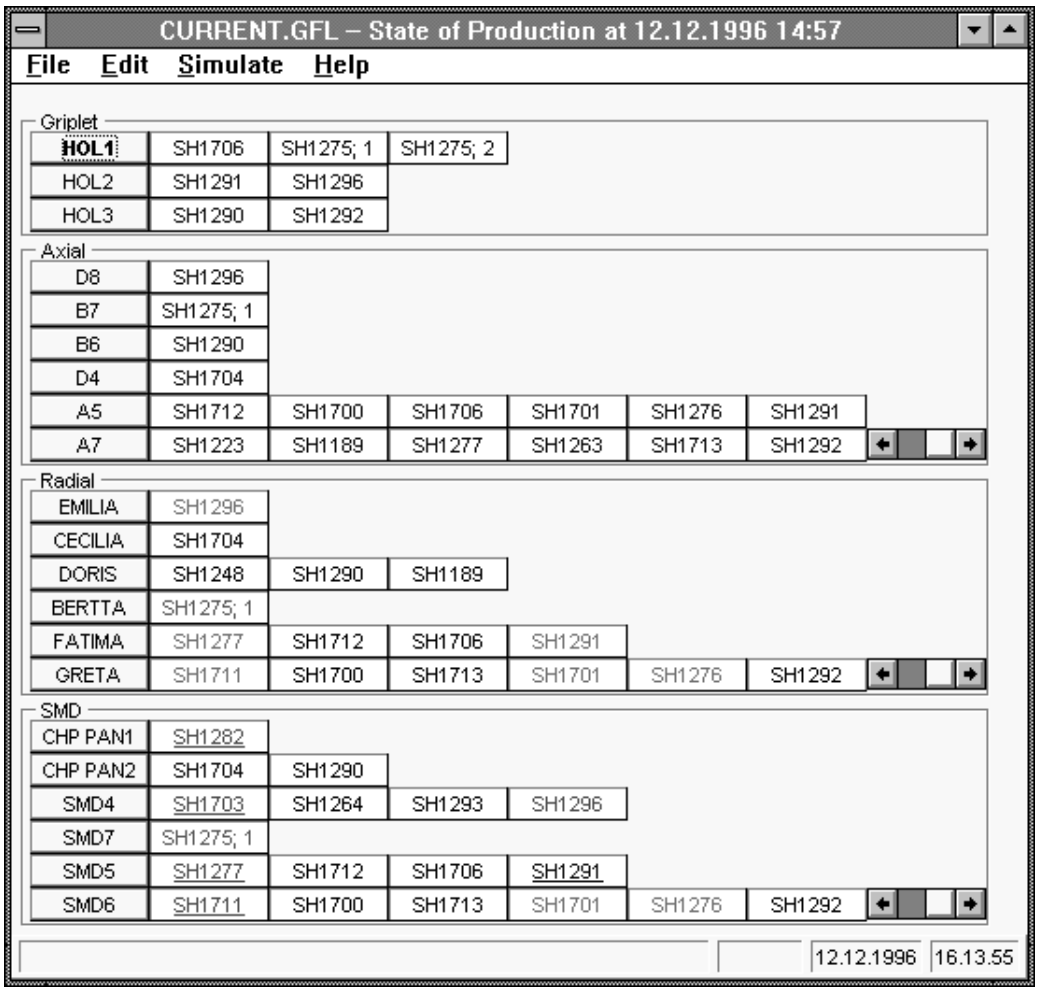


Figure 4: The main window

the state of the batch (free, fixed, late after computation, or ready after updating).

Computing the Finishing Times and Updating the Situation

The system computes the finishing time of each batch of the current situation. This reveals the batches being late. The processing time of a single PCB is calculated from the speed of the machine (components per hour) and the number of components in each phase or the time is given as a parameter obtained from previous production. Batches will be transported in a

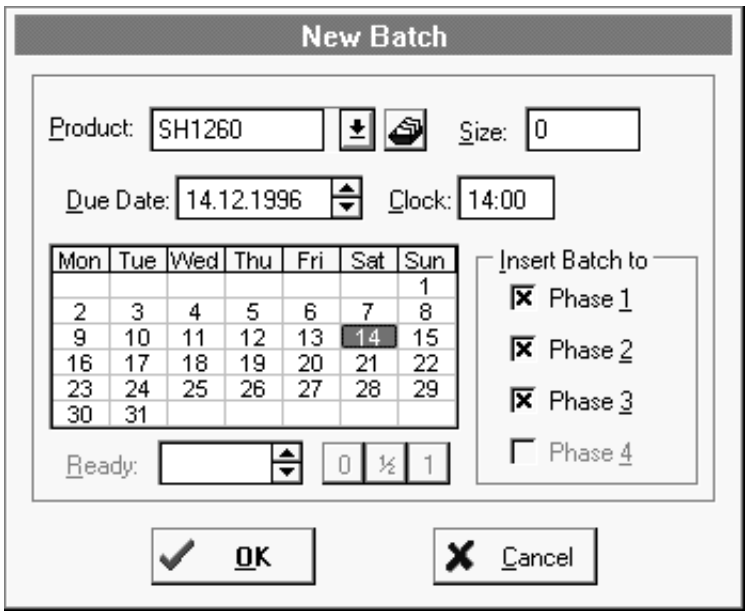


Figure 5: Inserting a new batch

magazine to the next machine (in zero time) as soon as a full magazine of PCBs has been produced on the previous machine. In the line machines the size of the magazine is one (i.e. the PCB is transported immediately after processing). Therefore, the speed of line machines depends on the slowest machine on line.

The starting time of a batch on a machine depends not only on the arrival time, but also on the family of the batch (cf. short and long set-up times, table 1). The line machines make an exception: the set-up of every machine on line must be ready before the processing can be started by the first machine of the line.

The system computes the starting and finishing times of processing for every batch in each phase (figure 6). The batches, which are late according to the simulation model, are highlighted (in red). The user can easily consider the starting and finishing times of the batches, and edit the situation on the screen.

The *update operation* uses the simulation model and computes the state of production process at given moments of time in the future. Updating works basically in the same manner as the computing described above. However, the events will be calculated only up to the given moment to which the situation will be updated.

If the situation suggested by the update operation does not correspond

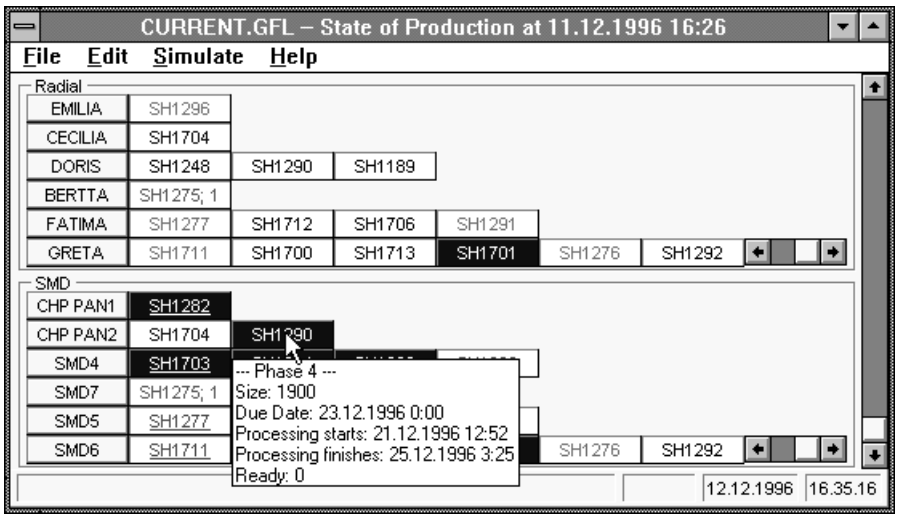


Figure 6: Part of the main window after computing

to reality (i.e. a batch is actually not yet finished, or is finished but not in the system), the user can make changes to the situation. After that he can either accept the update or cancel it. If the update is accepted, the system removes all finished batches.

Charts

The user receives graphical feed-back as well as numerical data of the situation. The charts depict how the situation in the main window will affect the aspects of the production plant which interest the production planner.

The due date chart implies how well the batches meet their due dates (figure 7). The work load can be observed either by machine banks or by individual machines (figure 8). The usage of the machines can be observed either by a Gantt chart or by a summary of different time components. The system gives also a summary of the average and the maximum size of every buffer.

Using the System

The operation of the system is dynamic: after the initial situation has been inserted, the user needs only to remove batches which have been finished. Before adding a new batch, the situation is updated to correspond to the actual situation of the production.

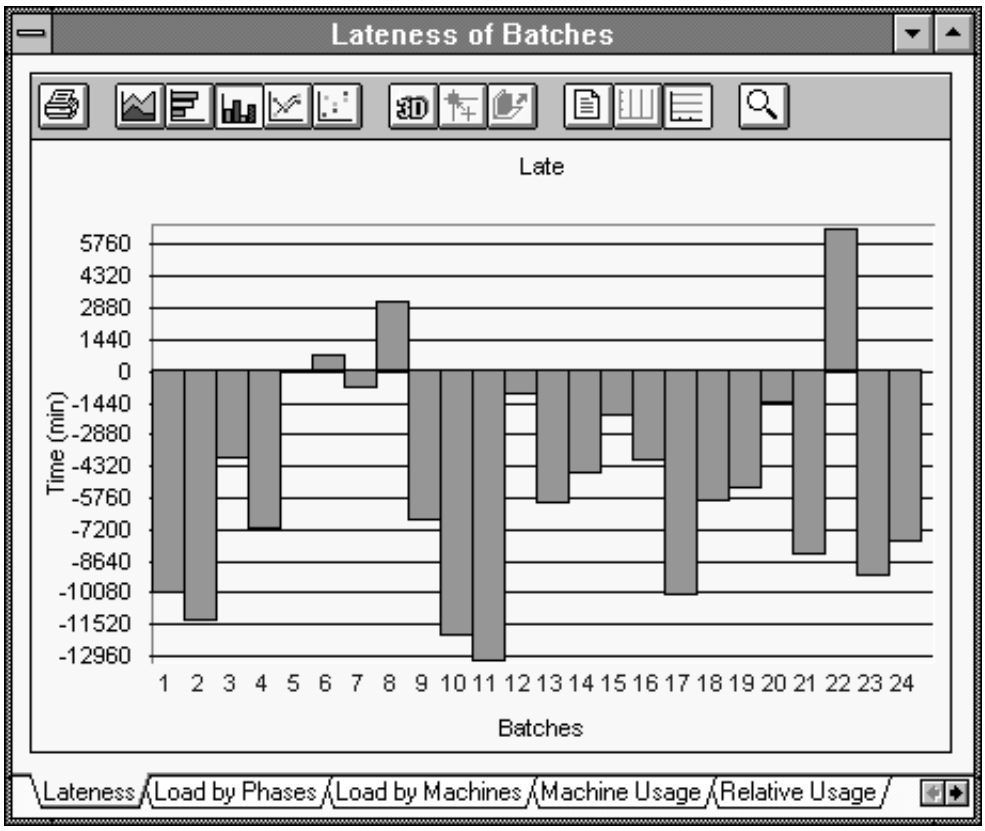


Figure 7: The due date chart. Bars above zero line stand for batches which are late.

The production planner uses the system as follows:

1. The previous situation is loaded and updated to correspond to the current state of the production. The system computes a tentative update, to which the planner can make changes.

2. New batches are added to the updated situation.

3. Different sequences and allocations of the batches can be experimented by observing their effect on the finishing times and machine statistics.

4. Problems in meeting the due dates are (possibly) removed by editing the situation.

5. Stages 3–4 are repeated until the solution is satisfactory. The plan is then printed and the situation is saved for the next update session.

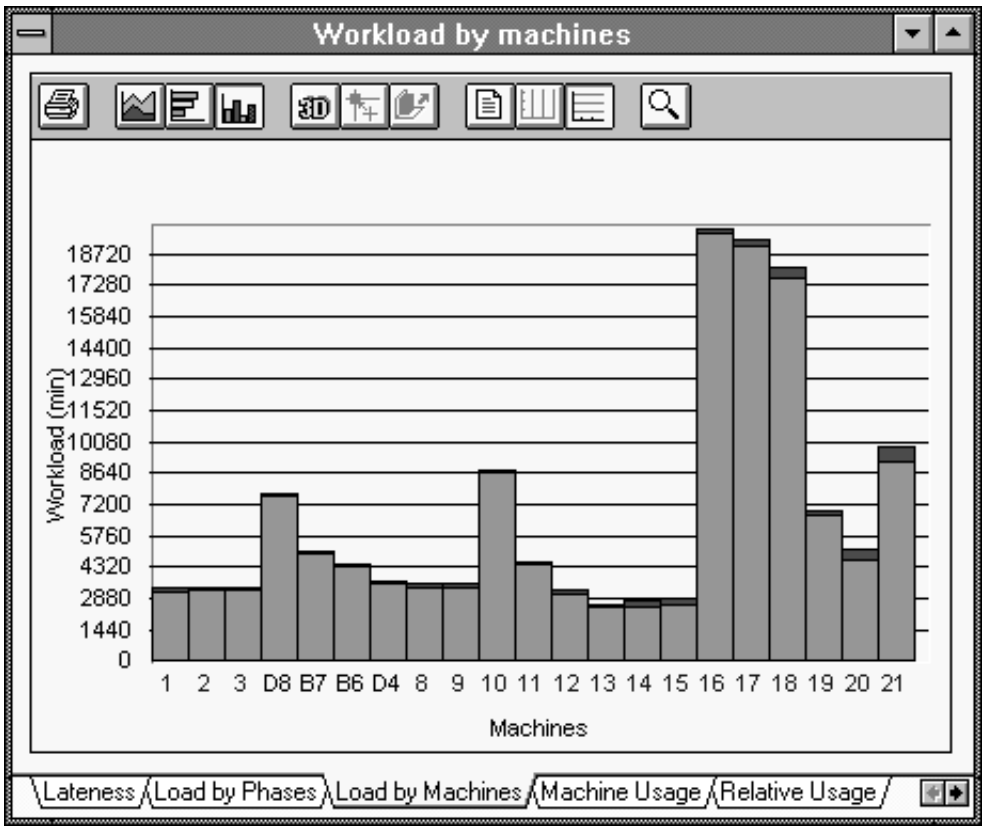


Figure 8: Work load of machines chart

Experiences

The system has been tested in the daily work of the production. The feedback from the user has convinced us that the simulation approach suits well for short range production planning. However, it has also revealed factors which were not considered carefully enough in the original designing process.

The possibility of batch splitting has turned out to be an important feature. In production, it is often necessary to allow to halt the batch being currently processed, because some other urgent batch must be processed before it. For this purpose, we have added an operation to the GUI, which splits a batch into two from a given point. After the split, the two batches can be handled independently.

It has turned out that the production planner sees a batch like a uniform unit and not a tightly-bound entity as we originally thought. 'Batch' is a much more flexible concept for him than it is in the present version of the

system. He wants to input the whole batch right in the beginning and divide it later (when needed) in smaller pieces, i.e. he cannot tell in advance what kind of batches the product must be divided into, but wants to do it as the production proceeds. In addition, the system requires also a counterpart for the split operation, namely batch merge.

The normal operation of a machine can be interrupted by a break-down, and the machine has then to be temporarily put off-line. Also the maintenance and preproduction series, which cause delays which are known beforehand, must be taken into consideration when planning the production. Therefore, the machines must have an option to take them off-line for either predetermined or indeterminated period of time. Paz and Leigh [20] give an extensive literature review of the research work done in the field of maintenance scheduling.

A conclusion drawn from these observations is that we cannot underestimate the dynamic nature of the production when designing a successful simulation model. The situation can have sudden and drastic changes, which the system must adapt to and give the planner quick response. The further development must concentrate on adapting even better to the indeterminate nature of the real production process.

Concluding Remarks

In this paper we have presented a dynamic system which can be used when performing production planning in a real production plant. The system simulates the production process and collects data of its operation.

We described the graphical user interface and the simulation model behind it. The simulation model is built by observing the actual production carefully and by analyzing its components and their interconnections. Object oriented programming made the system development easy and helped in adding new features into the system. The system can be used in studying the effects of various allocations and sequences of the production plan. The system renders the quick comparison of several different solutions possible. By using an interactive production planning system one can (at least) restrict a risky part of the production process: the planning process is no more based solely on the intuition and experience of one individual, but it can be studied, analyzed, and developed further.

The system supports the decision making but, for the time being, it does not solve the assembly line balancing problem or flow shop sequencing problem independently. However, our development project aims to add heuristic algorithms for both allocation and sequencing [21], and thus further support

the production planning. The work of improving the GUI is going on.

Our main interest was in the medium range planning of the production. We think that the system may be used also in the long-range (strategic) planning of the manufacturing process, see Fuh *et al.* [22]. With the system we can evaluate the effect of the changes of the machine configuration on the throughput and costs of the production plant. By using simulation one can balance resources, improve control logic, maximize throughput and eliminate bottlenecks of the production flow.

References

[1] Garey, M. R., Johnson, D. S., *Computers and Intractability: A Guide to the Theory of NP-completeness*, Freeman, 1979

[2] Chan, F. T. S., Smith, A. M., "Simulation Approach to Assembly Line Modification: A Case Study", *Journal of Manufacturing Systems* 12/3, pp. 239–45

[3] Kim, Y.-D., Lim, H.-G., Park, M.-W., "Search Heuristics for a Flowshop Scheduling Problem in a Printed Circuit Board Assembly Process", *European Journal of Operational Research* 91, pp. 124–43, 1996

[4] Lee, C.-Y., Vairaktarakis, G. L., "Minimizing Makespan in Hybrid Flowshops", *Operations Research Letters* 16, pp. 149–58, 1994

[5] Goyal, S. K., Mehta, K., Kodali, R., Deshmukh, S. G., "Simulation for Analysis of Scheduling Rules for a Flexible Manufacturing System", *Integrated Manufacturing Systems* 6/5, 1995

[6] Shevell, S. F., Buzacott, J. A., Magazine, M. J., "Simulation and Analysis of a Circuit Board Manufacturing Facility", *Procedings of the 1986 Winter Simulation Conference*, pp. 686–93, 1986

[7] Mitron Corporation, *Line Scheduler User Manual*, Version 3.2, 1995

[8] Scholl, A., *Balancing and Sequencing of Assembly Lines*, Physica-Verlag, 1995

[9] Shmoys, D. B., Stein, C., Wein, J., "Improved Approximation Algorithms for Shop Scheduling Problems", *Siam Journal on Computing* 23, pp. 617–32, 1994

[10] Kuriyan, K., Reklaitis, G. V., "Scheduling Network Flowshops So as to Minimize Makespan", *Computers & Chemical Engineering* 13/12, pp. 187–200, 1989

[11] Gupta, J. N. D., "Two Stage Hybrid Flowshop Scheduling Problem", *Journal of Operational Research Soc.* 39/4, pp. 359–64, 1988

[12] Wittrock, R. J., "An Adaptable Scheduling Algorithm for Flexible Flow Lines", *Operations Research* 36/3, pp. 445–53, 1988

[13] Johnsson, M., Peltonen, S., Leipälä, T., Nevalainen, O., "Work Load Balancing of a Generalized Flexible Flow Line in Printed Circuit Board Production", *TUCS Technical Reports* 59, Turku Centre for Computer Science, 1996

[14] Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H. G., Shmoys, D. B., "Sequencing and Scheduling: Algorithms and Complexity", *Handbook in Operations Research and Management Science, Volume 4: Logistics of Production and Inventory*, Graves, S. C., Rinnooy Kan, A. H. G., Zipkin, P. H. (eds.), North Holland, Amsterdam, pp. 445–522, 1993

[15] Brucker, P., *Scheduling Algorithms*, Springer-Verlag, 1995

[16] Kumar, K. R., Kusiak, A., Vanelli, A., "Grouping of Parts and Components in Flexible Manufacturing Systems", *European Journal of Operational Research* 24, pp. 387–97. 1986

[17] Venkataraman, R., Nathan, J., "Master Production Scheduling for a Process Industry Environment", *International Journal of Operations & Production Management* 14/10, 1994

[18] Caffrey, J., Hitchings, G., "Makespan Distribution in Flow Shop Scheduling", *International Journal of Operations & Production Management* 15/3, 1995

[19] Johnsson, M., Leipälä, T., Pulliainen, T., Nevalainen, O., "Integrated Program Generation System for a PC-board Assembly Line", *TUCS Technical Reports* 52, Turku Centre for Computer Science, 1996

[20] Paz, N. M., Leigh, W., "Maintenance Scheduling: Issues, Results and Research Needs", *International Journal of Operations & Production Management* 14/8, 1994

[21] Häyrinen, T., *Subsequenced Workphases and Parallel Machines—Scheduling Algorithms for Efficient Control of PC-board Assembly System* (in Finnish), University of Turku, Computer Science, M.Sc. Thesis, 1996

[22] Fuh, J. Y. H., Wong, Y. S., Yee, C. Y., Zhuang, L., Neo, K. S., "Modelling, Analysis and Simulation for the Design of a Robotic Assembly System", *Computer Integrated Manufacturing Systems* 9/1, pp. 19–31, 1996

Appendix A: Summary of the Windows

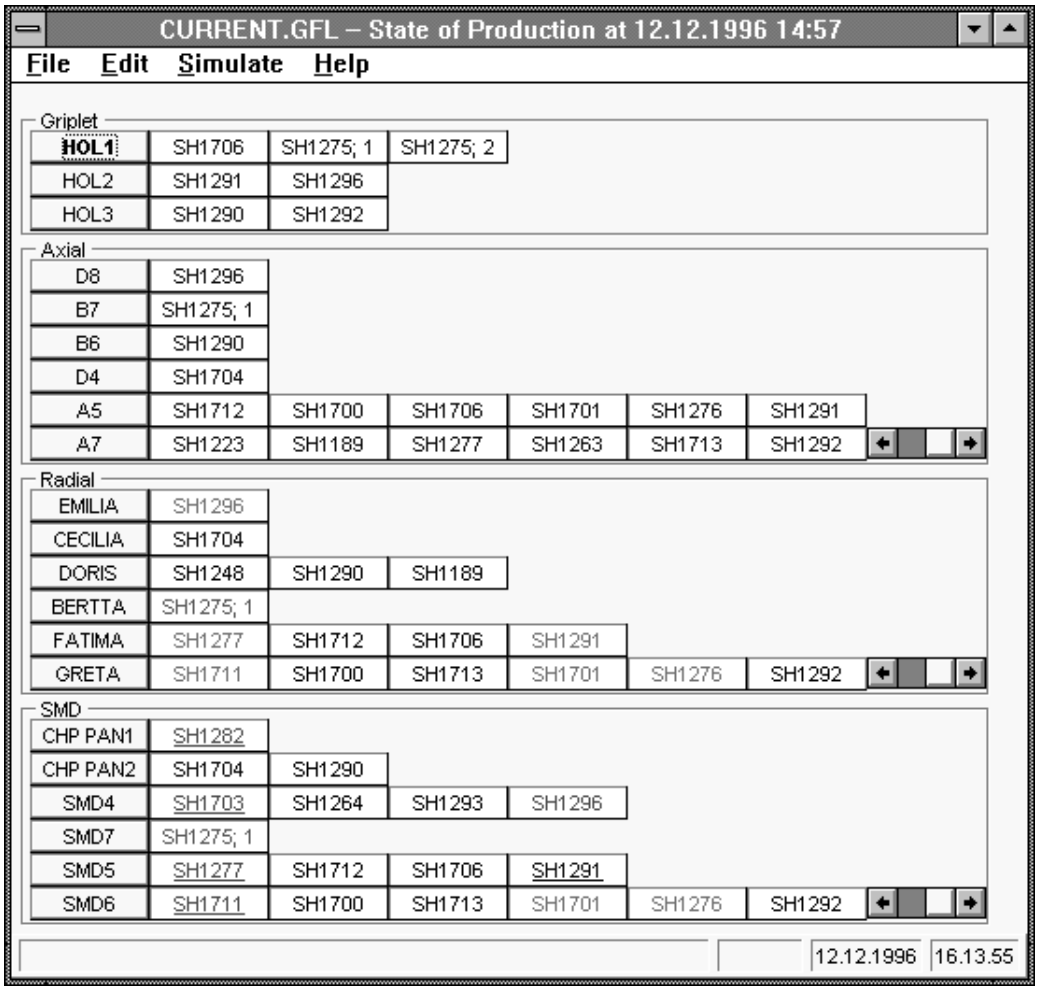


Figure 1: The main window

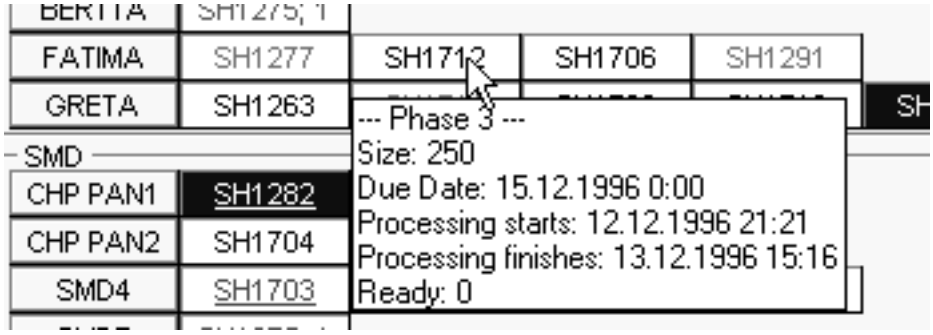


Figure 2: Batch box and batch information

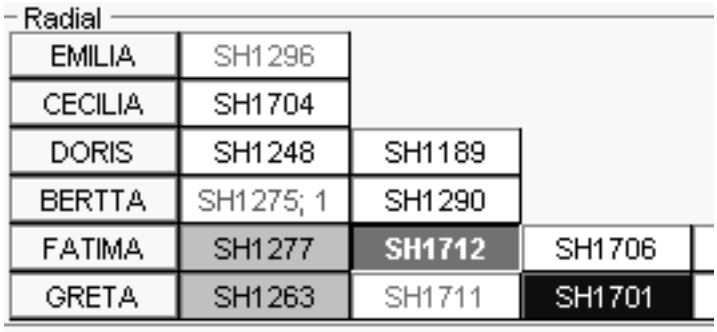


Figure 3: Batch boxes in various states

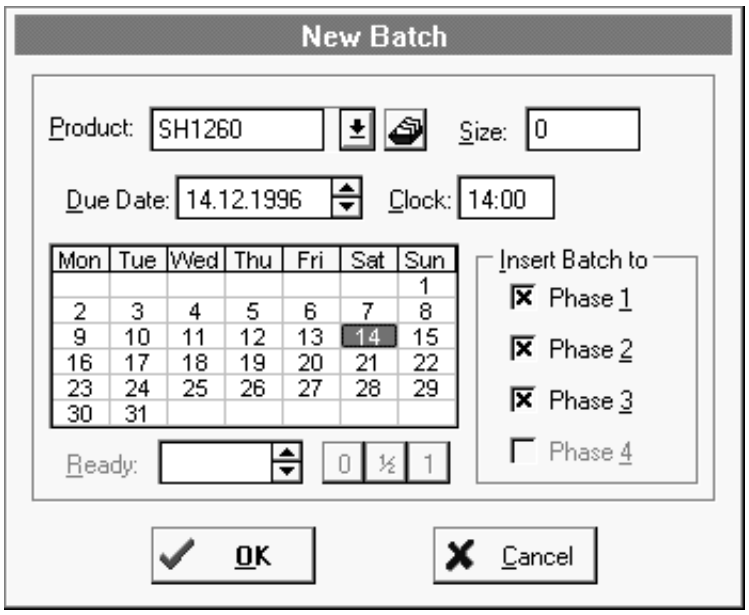


Figure 4: Inserting a new batch

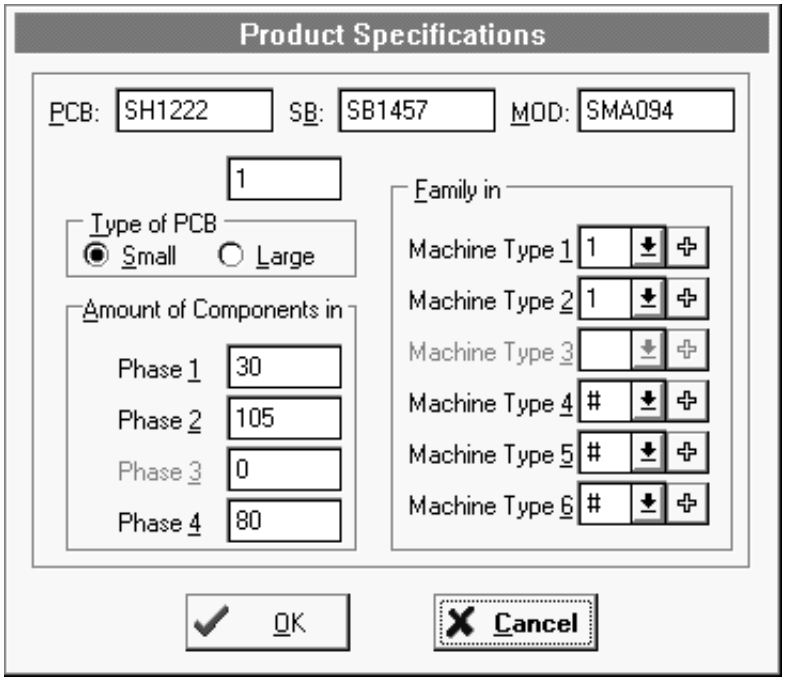


Figure 5: Editing the product information

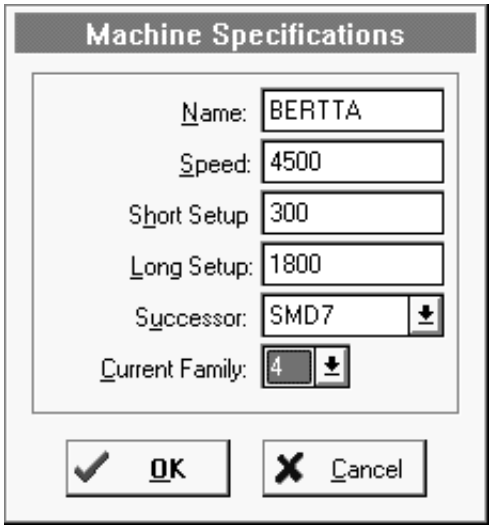


Figure 6: Machine information

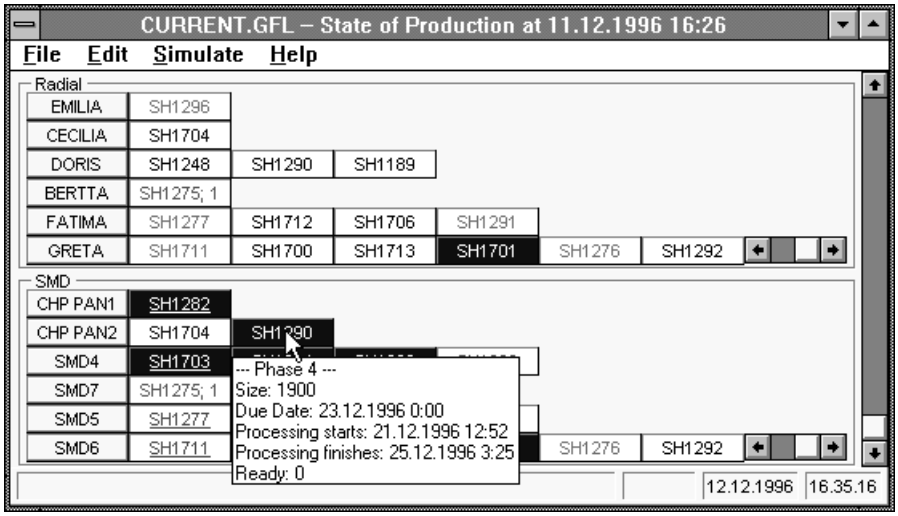


Figure 7: Part of the main window after computing

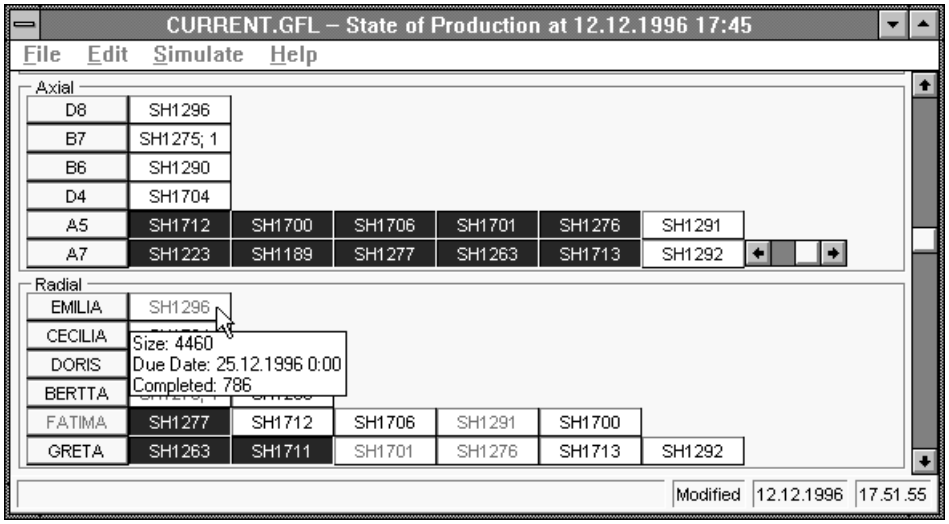


Figure 8: The main window after an update

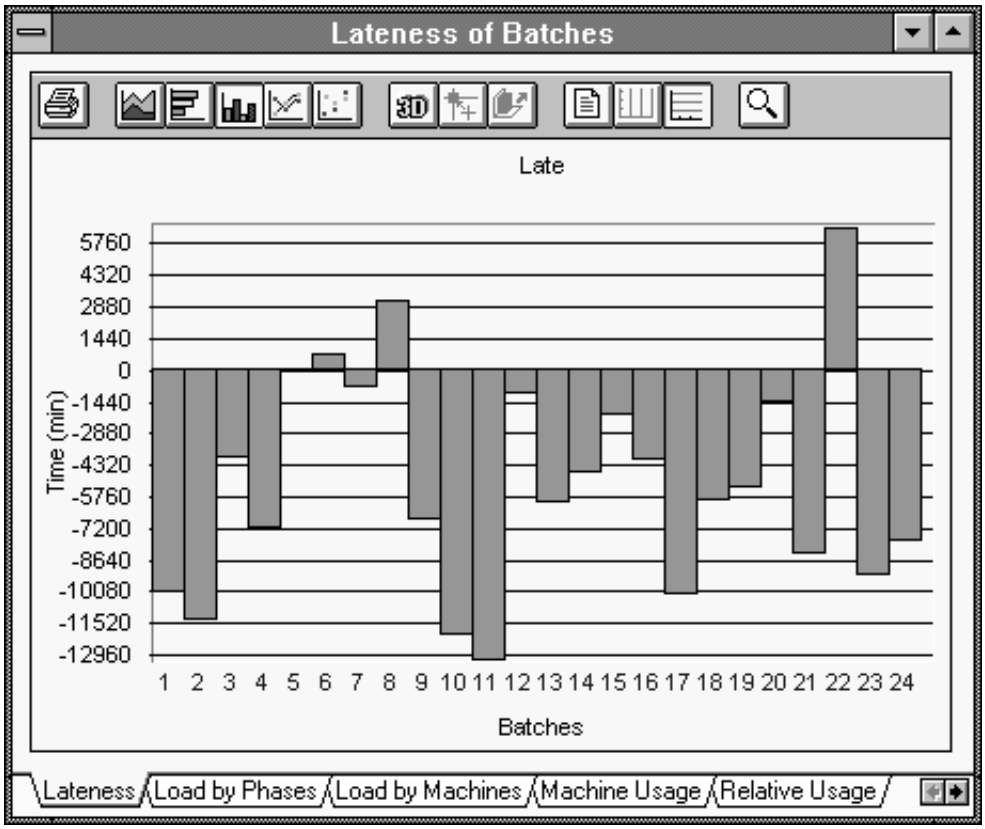


Figure 9: The due date chart. Bars above zero line stand for batches which are late.

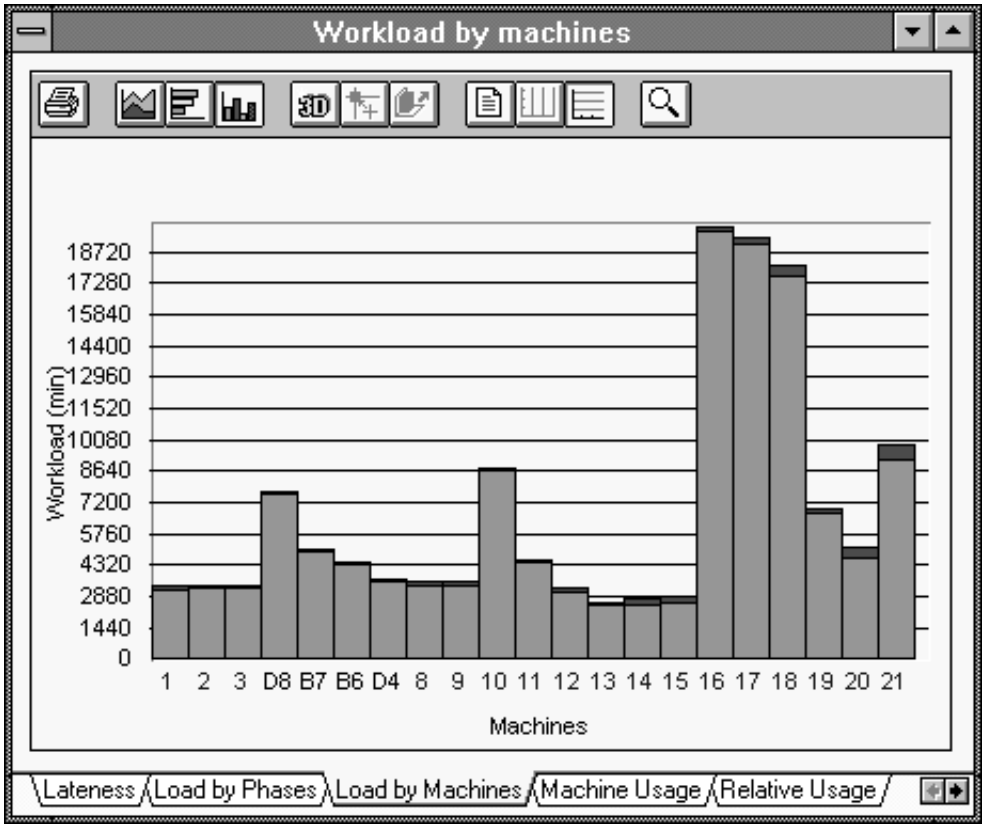


Figure 10: Work load of machines chart

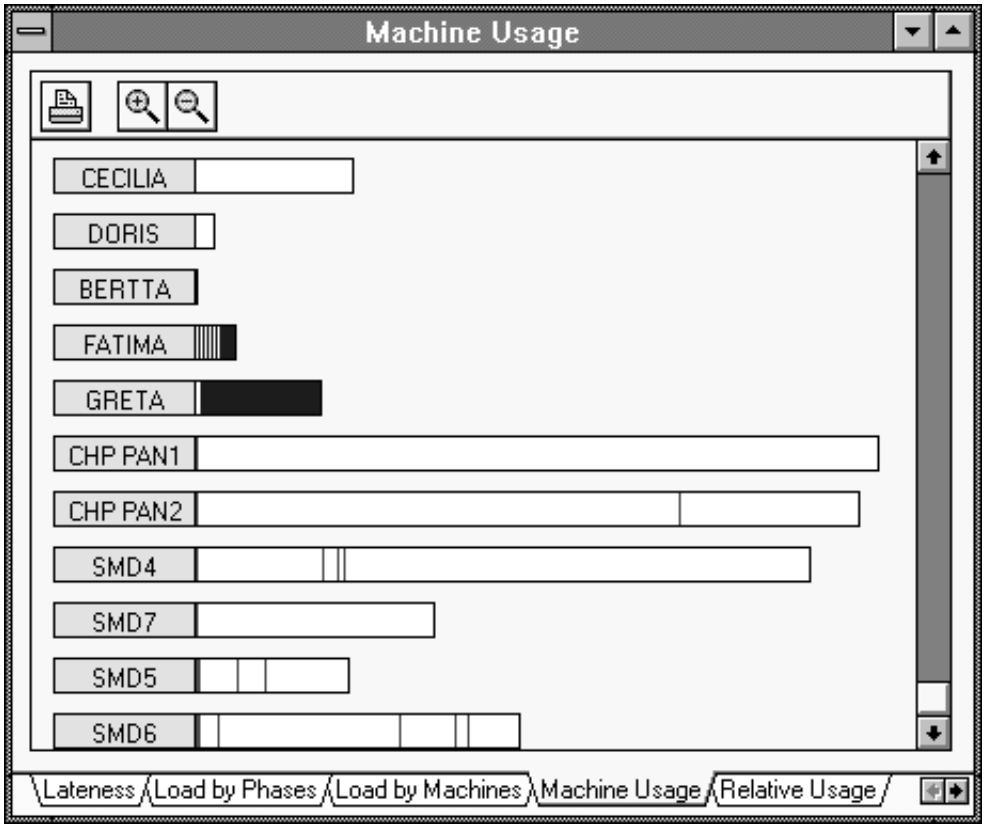


Figure 11: Machine usage chart

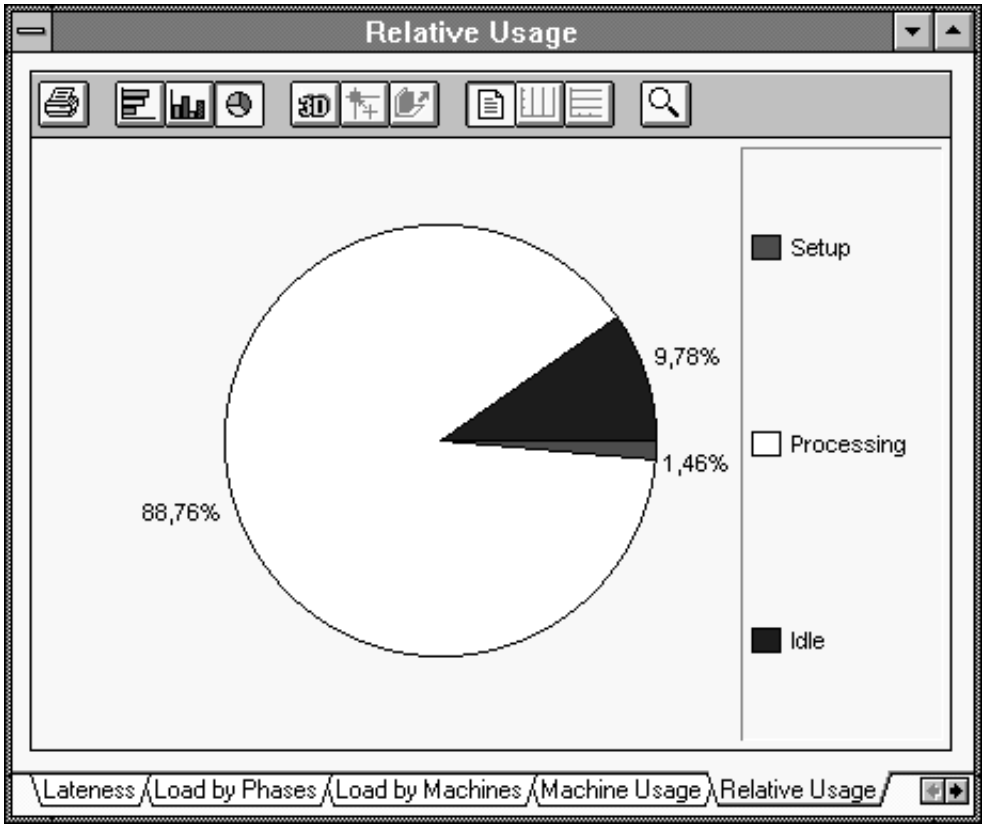


Figure 12: Relative usage chart

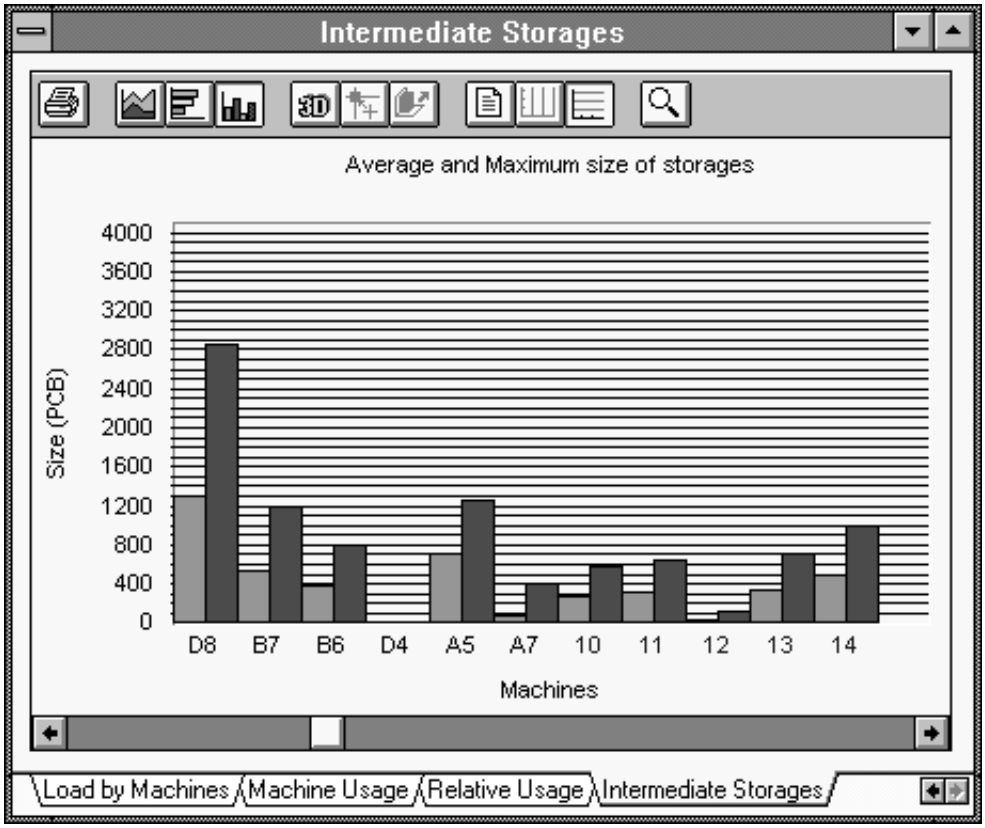
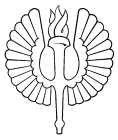


Figure 13: Average and maximum size of buffers chart

Turku Centre for Computer Science
Lemminkäisenkatu 14
FIN-20520 Turku
Finland

http://www.tucs.abo.fi



University of Turku
- **Department of Mathematical Sciences**

Åbo Akademi University
- **Department of Computer Science**
- **Institute for Advanced Management Systems Research**

Turku School of Economics and Business Administration
- **Institute of Information Systems Science**