Knowledge-Based Scheduling Systems in Industry and Medicine

Jürgen Sauer and Ralf Bruns, University of Oldenburg

A GENERIC FRAMEWORK FACILITATES THE CONSTRUCTION OF KNOWLEDGE-BASED SCHEDULING SYSTEMS. THE AUTHORS HAVE USED IT TO IMPLEMENT SCHEDULING SYSTEMS FOR DYE PRODUCTION, PIPELINE-FITTINGS PRODUCTION, AND HEART SURGERY.

PRODUCTION MANAGEMENT IS crucial for achieving the timely and cost-effective execution of industrial production processes. In recent years, interest has increased in the use of artificial intelligence technologies for production planning and scheduling.[1,2] However, scheduling research typically has been theoretical, has had a narrow focus, and has not covered adaptation to unforeseen events (see the "Scheduling problem" and "Previous scheduling research" sidebars).

Our objective has been to use computer-based scheduling systems to enhance the problem-solving capabilities of human domain experts. During our research, we have developed a generic framework for building practical scheduling systems. This framework fosters the reuse of algorithms and the integration of knowledge-based technology into the organizational environment. It also supports dynamic adaptation. We successfully applied our framework in the implementation of three scheduling systems—that is, they all share the same system architecture and use similar problem-solving methodologies. The first two systems deal with serious real-life problems in the manufacturing industry: the rarely investigated continuous-flow scheduling problem and the widely known job-shop problem. The third system shows how concepts and techniques developed for industry can be transferred successfully to a medical domain.

The framework

Our generic framework is based on two design principles:

- The combination of standard computer science components (for example, professional user interfaces and databases) with knowledge-based concepts—in particular, heuristic scheduling algorithms and declarative knowledge representation.
- The explicit and transparent representation of scheduling knowledge, allowing flexible reuse and adaptation of scheduling algorithms.

A common architecture. The first principle leads to a common architecture for all our scheduling systems, consisting of a *graphical user interface*, a *knowledge base*, an *interface component*, and a *knowledge-based algorithms component*. Each system supports effective production management by combining predictive, reactive, and interactive scheduling. Users can choose between fully automatic scheduling, fully interactive scheduling, or any combination of both.

The graphical user interface presents the information needed for monitoring the scheduling activities and supports interactive scheduling. It is window-oriented, and most of the functions are mouse-sensitive. It has these important features:

- It comprehensively presents, and provides easy access to, relevant information

0885-9000/97/$10.00 © 1997 IEEE

The scheduling problem

Scheduling is the temporal assignment of orders (for example, for manufacturing products) to resources (for example, machines), where a number of goals and conditions (for example, meeting the due dates or using only special machines) must be regarded. Scheduling includes creating a schedule of production processes (*predictive scheduling*) and adapting an existing schedule because of events in the scheduling environment (*reactive scheduling*).

A scheduling problem comprises

- a set of orders to manufacture products that are to be scheduled subject to several constraints;
- a set of products with information about process plans (routings), operations, machines, and so on;
- a set of resources with different functional capabilities (for example, machines and personnel);
- a set of hard constraints (for example, production requirements) that must be fulfilled; and
- a set of soft constraints (for example, meeting due dates) that should be fulfilled but may be relaxed.

Scheduling produces a schedule showing the temporal assignments of operations of orders to the resources to be used—that is, which resources should manufacture a particular product, and when they should be used.

A tough nut to crack. Apart from some theoretical cases of little practical importance, the optimal solution of a resource-scheduling problem is an NP-complete problem—that is, no deterministic algorithm is known yet for solving the problem in polynomial time. Traditionally, production-scheduling research has focused on methods for obtaining optimal solutions to simplified problems. To determine an optimal solution, different restrictions have been imposed on the problem domain (for example, on the number of orders or machines), which makes the application of the results to real-world scheduling problems very difficult or even impossible.

Besides combinatorial complexity, three other factors make real-world scheduling even more difficult:

- the requirements imposed by numerous details of the application domain (for example, alternative machines and cleaning times),
- the dynamic and uncertain nature of the manufacturing environment (for example, unpredictable setup times and machine breakdowns), and
- conflicting organizational goals (for example, minimizing work-in-process time and maximizing resource use).

Because of the problem domain's difficulty, the objective of a real-world scheduling environment should be to determine a good, feasible solution, not an optimal one. Very important for this is the (heuristic) knowledge of a human domain expert, who can use his or her experience to solve distinct scheduling problems and to judge the feasibility of schedules. Several knowledge-based approaches have been developed so far, using the experience of human experts and problem-specific knowledge of the application domain.[1,2]

Winning acceptance. In spite of a large number of developed scheduling methods, only a few practical applications have entered into everyday industrial use. Our experience with real-world scheduling problems has shown that several other features besides the applied scheduling methods significantly influence the acceptance of scheduling systems.[3] So, a scheduling system should meet these requirements:

- *Information presentation*. The system must present the information necessary for the scheduling task appropriately, showing specific information, such as capacities or alternative process plans, at a glance. Moreover, the user should be able to monitor all scheduling actions to see the immediate consequences of specific decisions and to maintain consistency.
- *Interaction*. Interaction should allow full manual control of the scheduling. The user should be able to make all decisions (for example, selecting orders, operations, or machines). However, the support of interactive scheduling merely by information presentation and consistency control seems to be insufficient because of the problem domain's combinatorial complexity. Purely automatic scheduling, on the other hand, is not realistic, because it neglects the important role of the human expert, who has the ultimate responsibility for all decisions. Thus, industrial scheduling systems should support the interactive as well as the automatic part of predictive and reactive scheduling.
- *Incorporation of scheduling expertise*. The main feature of a knowledge-based scheduling system is the identification and application of problem-specific knowledge to solve the addressed problem. Most previous knowledge-based approaches focused merely on predictive scheduling. However, in many applications, reactive scheduling is the more significant problem and must also be supported algorithmically. By using the problem knowledge, a system can employ heuristics for problem solving that are similar to those used by the human expert, who in turn can verify the solution's plausibility. However, merely mimicking a human expert's decision making is insufficient, because his or her decisions are often myopic, aimed at solving small subproblems immediately instead of global optimization.
- *Integration in the organizational environment*. Every modern industrial enterprise has a complex information technology infrastructure. Scheduling systems cannot be designed as stand-alone components, because they have to communicate, interact, access the same data, and share information with their organizational environment. Therefore, knowledge-based systems must be an integrated part of an existing information system, thus providing well-defined interfaces to standard application systems such as databases and networks.
- *Participation of the end user*. The end user's involvement in all phases of system design is crucial to final acceptance of the scheduling system. All other software-engineering principles should also be regarded.

References

1. S.F. Smith, "Knowledge-Based Production Management: Approaches, Results and Prospects," *Production Planning & Control*, Vol. 3, No. 4, 1992, pp. 350–380.
2. M. Zweben and M.S. Fox, eds., *Intelligent Scheduling*, Morgan Kaufmann, San Francisco, 1994.
3. K.G. Kempf et al., "Issues in the Design of AI-Based Schedulers: A Workshop Report," *AI Magazine*, Vol. 11, No. 5, Jan. 1991, pp. 37–46.

of the scheduling area. Such information includes the orders to be scheduled, the production requirements (alternative process plans and alternative machines, or the capacity load of resources), and a Gantt chart of the schedule.

- It visualizes the current state of the problem-solution process and lets users monitor the effects of scheduling decisions.
- It supports the interactive creation or repair of the schedule by selection, insertion, deletion, relocation, and substitution of resource allocations.
- The consistency control checks the feasibility of all interactions.
- Additional windows can be employed (for example, for administrative functions).

Previous scheduling research

Artificial intelligence and operations research has intensively investigated scheduling problems. Several knowledge-based approaches have emerged, using different paradigms—in particular, constraint-based search, heuristic search, and rule-based methods.[1] Most approaches address predictive scheduling problems. Only recently have an increasing number of algorithms for reactive scheduling appeared.[2]

Most predictive approaches can be characterized by the underlying perspective of the problem decomposition:

- *Order-based* approaches select an order from all unscheduled orders and completely schedule that order's operations before they select the next order.[3]
- *Resource-based* approaches select a resource, and determine and schedule a not-yet-scheduled operation on that resource.[4]
- *Operation-based* approaches select an operation from all unscheduled operations and schedule that operation on one of the resources.[5]

The algorithms differ essentially in the problem-specific heuristic knowledge they apply to determine the appropriate decisions during problem solving. This knowledge is often specifically tailored to just one special problem scenario.

Experimental projects, rather than projects that consider real-world industrial applications, dominate scheduling research. Most papers focus only on a detailed description of sophisticated algorithms, neglecting the integration of and complex interaction between different system components (for example, predictive-reactive-interactive scheduling). Stephen Smith, Monte Zweben and Mark Fox, and Jürgen Dorn and Karl Froeschl have provided comprehensive overviews of knowledge-based research for scheduling.[1,6,7]

References

1. S.F. Smith, "Knowledge-Based Production Management: Approaches, Results and Prospects," *Production Planning & Control*, Vol. 3, No. 4, 1992, pp. 350–380.
2. R. Kerr and E. Szelke, eds., *Artificial Intelligence in Reactive Scheduling*, Chapman & Hall, London, 1995.
3. M. Fox, *Constraint Directed Search: A Case Study of Job-Shop Scheduling*, Pitman Publishers, London, 1987.
4. S.F. Smith et al., "OPIS: An Opportunistic Factory Scheduling System," *Proc. Third Int'l Conf. Industrial & Engineering Applications of Artificial Intelligence & Expert Systems*, Gordon & Breach, New York, 1990, pp. 268–274.
5. N.P. Keng, D.Y. Yun, and M. Rossi, "Interaction Sensitive Planning System for Job-Shop Scheduling," in *Expert Systems and Intelligent Manufacturing*, M.D. Oliff, ed., Elsevier, Amsterdam, 1988, pp. 57–69.
6. M. Zweben and M.S. Fox, eds., *Intelligent Scheduling*, Morgan Kaufmann, San Francisco, 1994.
7. J. Dorn and K.A. Froeschl, eds., *Scheduling of Production Processes*, Ellis Horwood, New York, 1993.

Extensive use of colors makes the information easier to understand.

The knowledge base contains the relevant production-management knowledge describing the application domain, the addressed problem, and the solution. It represents the application and problem descriptions (for example, orders, products and their production requirements, resources, and the solution—the schedule) in a relational form. It represents the hard and soft constraints as rules. The declarative representation of the knowledge facilitates the modeling of additional requirements and constraints of the application as well as of the heuristic knowledge of the domain experts.

The interface component provides well-defined interfaces to standard software systems, such as commercial database systems and networks, shop-floor data-collection systems (important for industrial domains), and other programming languages. This enables the integration of the scheduling systems into an existing organizational environment.

The knowledge-based algorithms component provides sophisticated algorithms for predictive and reactive scheduling. To cope with the combinatorial complexity and to consider all requirements of a specific application, these scheduling algorithms incorporate problem-specific knowledge to guide the search.

The predictive scheduling algorithms aim at creating good production schedules using the expert scheduler's experience. The algorithms try to model the expert's behavior by using a specific strategy (how to create a schedule, step by step) and some of the heuristics applicable in guiding the search for a good solution.

The reactive scheduling algorithms enable appropriate reactions to unexpected events to reestablish a consistent state of the schedule. The events can be external (for example, short-term acceptance of a high-priority order) or internal (for example, a machine breakdown). The possible solution approaches range from simple integrated algorithms for the handling of specific events to complete leitstand systems specialized for rescheduling in the case of unforeseen circumstances. (A leitstand is a "computer-aided graphical decision-support system for short-term interactive and automatic production scheduling and control."[3])

Easy adaptation. The second principle of our framework allows the easy implementation of knowledge-based algorithms, because it advocates a reusable representation of scheduling knowledge. This approach circumvents a serious problem encountered in several scheduling projects: Advanced algorithms have been developed for special problem instances. However, it is often impossible to reuse any of the components in further systems or to transfer the basic algorithmic ideas to more general problem scenarios. So researchers often must design new systems from scratch, experimenting with algorithms similar to those used before.

Our framework separates predictive and reactive scheduling algorithms into the underlying scheduling strategy (for example, an order-based strategy) and the selection rules to be used in these strategies. The strategies for predictive as well as for reactive scheduling can be represented as scheduling skeletons, where different rules can be used to select appropriate orders, routings, alternative resources, and so on. The combination of different scheduling skeletons and selection rules leads to a large variety of different scheduling algorithms, which can be easily adapted, tested, and deployed for the given scheduling problems.[4,5]

Figures 1 and 2 show skeleton representations (in pseudocode) of order-based and resource-based predictive scheduling strate-

```
WHILE orders to schedule
   select order
   select routing for order
   WHILE operations to schedule
      select operation
      select resource
      select interval
      schedule operation or solve conflict
   END WHILE
END WHILE
```

Figure 1. An order-based scheduling strategy.

```
select routing for every order
find operations of orders
WHILE operations to schedule
   select resource
   select operation
   select interval
   schedule operation or solve conflict
END WHILE
```

Figure 2. A resource-based scheduling strategy.

gies. Both strategies decompose the problem into smaller subproblems (for example, the scheduling of one order after another). Reactive strategies can be expressed similarly. Each selection step (the `select` statements) must apply heuristic knowledge (for example, selecting critical products before others). This knowledge is represented separately as rules for the selection of orders, routings, operations, resources, and time intervals.

The scheduling strategies and the sets of rules are adopted from interviewing domain experts or analyzing existing systems, or come from the literature. Figure 3 gives examples of simple rules, known from operations research, and more complex rules. Weighted rules or weighted combinations of rules are also possible.

Scheduling applications

We'll now describe in detail the three scheduling systems that use our framework: *Protos*, which schedules production in the chemical industry; *PSY*, which schedules production in the metal industry; and *Medicus*, which schedules patients for heart surgery.

Protos. We developed this system as part of the Eureka Protos (Prolog Tools for Building Expert Systems) project. Three Protos subprojects developed tools for building expert systems, using Prolog. We implemented the Protos scheduling system in the fourth subproject, which concerned application development and tool evaluation. The evaluation of the tools occurred in the context of a real-world application problem: planning and control of the production of dyes. In one building, a continuous-flow production occurs from the top floor to the basement, using multipurpose apparatuses connected through pipes. These apparatuses allow a huge variety of alternative routings for every product. Therefore, selecting an appropriate routing is one of the main tasks in solving the scheduling problem.

The problem. A set of orders is given for one production period (half a year), with information about the amount, the release date, and the due date of the desired products.

The products are the dyes; every product is described by the routings that can be used to produce it, by the manufacturing steps that must occur in each routing, and by the apparatuses that can be used in each step. The resources are the apparatuses used for producing dyes.

The hard constraints describe the requirements of the production process, such as

- All orders must be scheduled.
- The production requirements (for example, durations of steps) must be met.
- The continuous-flow structure of the production process, defined by the given order and the relative start times of operations, must be upheld.
- Overlappings are not allowed—that is, an apparatus can perform only one step at a time.

For the selection of orders

- Earliest start time
- Shortest processing time
- Increasing number of alternatives (critical products first)
- Increasing slack intervals
- Increasing user priority

For the selection of routings

- Given order with stem first
- Inverse order (last come, first served)
- Critical routing first
- Simple routing first

For the selection of operations

- Increasing operation number (first come, first served)
- Decreasing operation number (last come, first served)
- Increasing number of alternative resources (critical operation first)
- Decreasing number of alternative resources (simple operation first)

For the selection of resources

- Given order with stem first
- Simple resource first
- Critical resource first

For the selection of intervals

- Forward from given release date
- Backward from due date.

Figure 3. A set of sets of selection rules.

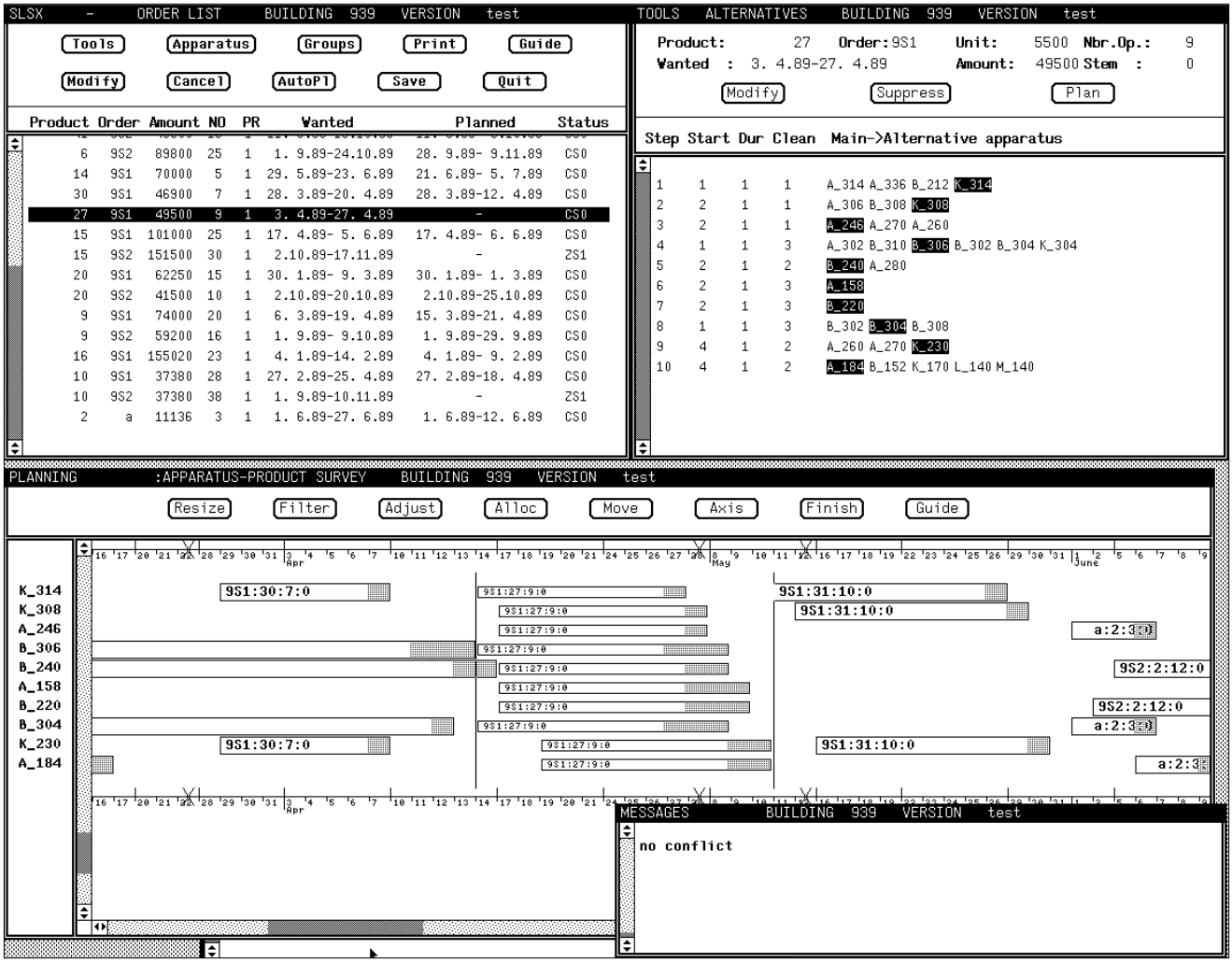

Figure 4. The Protos scheduling system's user interface.

The soft constraints describe the economical constraints, such as

- If possible, the Protos system should use the apparatuses and routings preferred by the domain expert.
- Due dates should be met.

The system. The Protos system supports interactive and automatic scheduling. Figure 4 shows the graphical interface. The user can select the orders to be scheduled in the **`Order List`** window, on the upper left; the routings and alternative apparatuses to produce the products in the **`Tools Alternatives`** window, on the upper right; and the time intervals for producing the products in the **`Planning`** window, in the center. The system automatically checks the hard constraints (for example, the preference constraints of the continuous-flow production). Conflicts are shown graphically and can be solved by interactively checking alternative routings, apparatuses, or time intervals. Additionally, the user can use the knowledge-based scheduling algorithm to propose a solution. We've developed a specific scheduling approach to meet the special requirements imposed by the continuous-flow production structure. The algorithmic solution upholds the hard and soft constraints and presents a consistent solution that can be changed by the user.

PSY. We developed PSY (Production Planning and Scheduling System) in cooperation with Siekmann Fittings, a company in the discrete-manufacturing metal industry that produces pipeline fittings. The company can produce more than 10,000 different products, ranging from standard fittings (for example, 90-degree arches for water mains) to special fittings for power plants. The lot sizes range from one for special fittings, to several thousands for standard fittings. PSY supports long- and short-term scheduling. In long-term scheduling (one year), the orders must be scheduled by weeks, taking into consideration the capacity of the machines. Short-term scheduling requires a detailed schedule of one week, showing the hours of production and consisting of about 100 orders, each with about six to 10 operations.

The problem. Long-term scheduling preschedules a set of orders, providing information about the products to be produced, the amount, the delivery week, and a given priority. This information provides the input for short-term scheduling.

The products are fittings for pipelines or mains. The resources are machines (for example, for cutting, heating, or forming pipes).

The hard constraints describe the production requirements, such as

- All orders must be scheduled.
- All operations of an order must be scheduled.

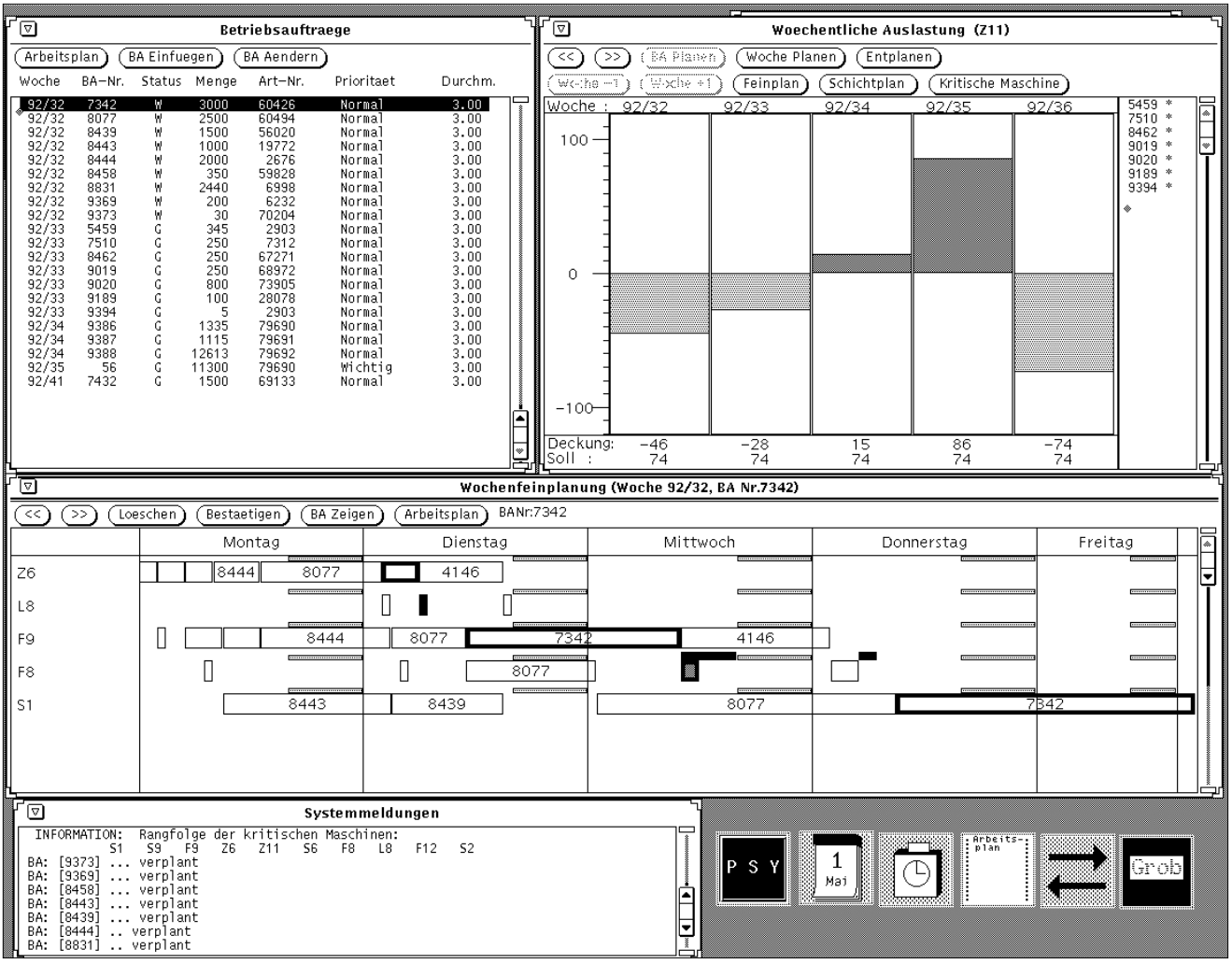

Figure 5. The PSY scheduling system's user interface.

- The preference constraints between operations must be met.
- Overlappings are not allowed.
- Orders must not be split.

The soft constraints are that

- Orders should be scheduled in the given week.
- Work-in-process and flow times should be minimal.

The system. Long-term scheduling involves mainly the upper two windows of the user interface (see Figure 5). Users can insert or change orders in the **Betriebsauftraege** (operating instructions) window, on the upper-left. The **Woechentliche Auslastung** (weekly load distribution) window, on the upper-right, shows the effect of these actions. It provides an overview of the weekly capacity information of all machines. The window presents the capacity diagram of one machine over five weeks at a glance. PSY derives capacity limits from calendars and shift plans. The system supports long-term scheduling by checking the capacities of the critical machines involved in the production of a desired product in the week it will be manufactured. If a schedule violates the given capacity limit, PSY proposes to shift the order to another week. The **Wochenfeinplanung** (weekly detailed plan) window, in the center, presents the detailed schedule.

For short-term scheduling, the user can choose from three strategies, which differ in the selection of operations, resources, and intervals. The basic algorithm fills the week from left to right (Monday to Friday), using the first-in, first-out principle and the shortest-processing-time rule. It schedules the operations for the earliest possible position, to optimize the flow times and to meet the due dates. The two other strategies first look at critical machines. These strategies select orders by priorities and schedule the operations using the critical machines first. The *first-fit* strategy schedules the operations for the earliest possible position, to minimize flow time. The *best-fit* strategy schedules the operations for the smallest possible position, to optimize machine use.

Using one of these strategies, the user can schedule one order or all the orders for a week. PSY presents the schedule as a Gantt chart; the user can alter the schedule interactively by deleting, substituting, or relocating complete orders or single operations.

Medicus. The Medicus (Medical Resource Scheduling System) project's objective was to develop a computer-based prototype system for scheduling hospital patients. We developed Medicus in collaboration with the department of heart surgery of the city hospital in Oldenburg. The department consists of four units: normal care, intensive care, intermediate care, and the operating

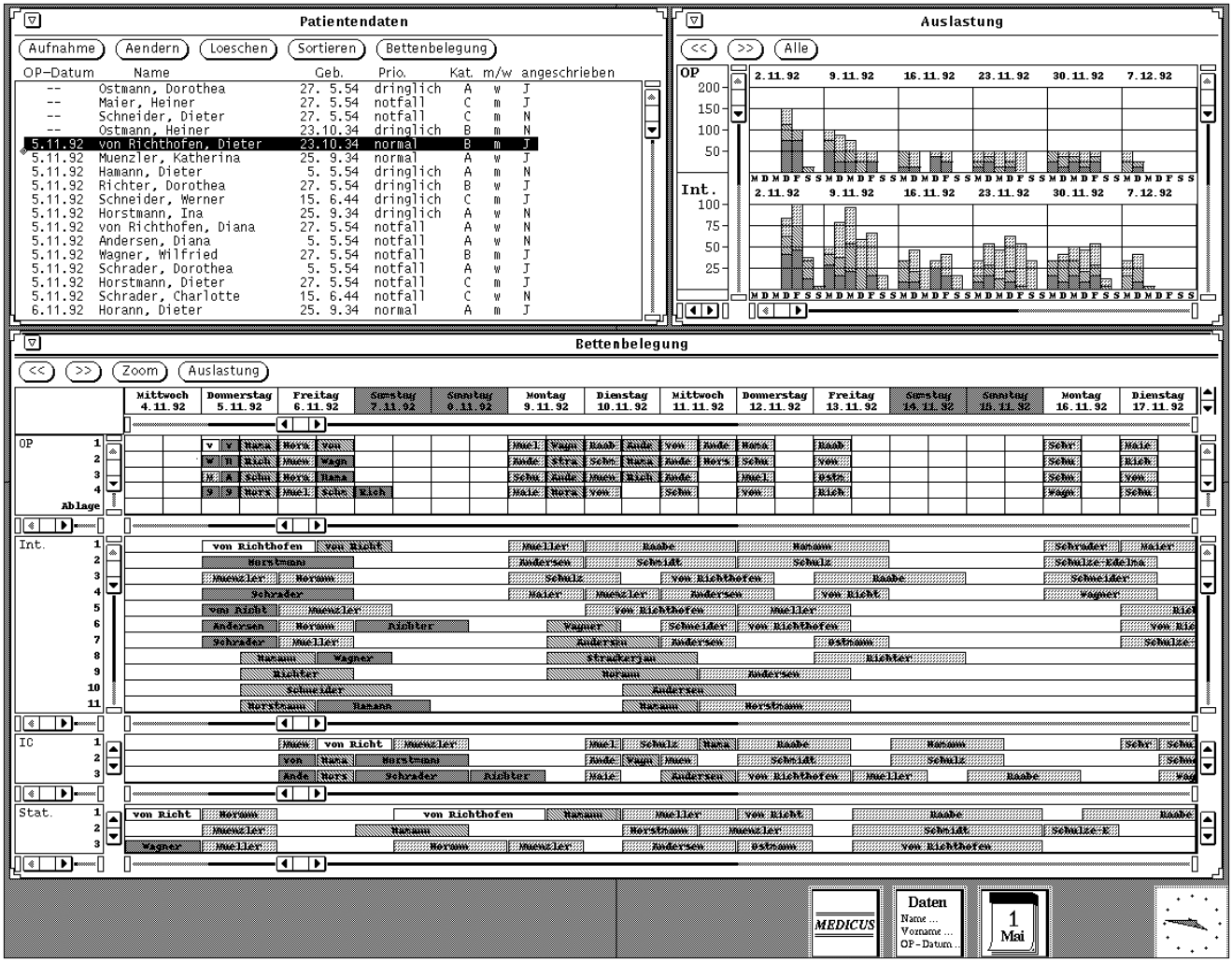

Figure 6 The Medicus scheduling system's user interface.

theaters. The most important feature of this medical application area is its highly dynamic and uncertain environment. New patients are admitted irregularly, and emergency patients must be treated immediately. The period of a patient's stay at a unit can vary because of differences in the course of illnesses.

The problem. The set of patients (which corresponds to the set of orders) is not known in advance. The products are the different kinds of medical treatment. The resources are the operating theaters and the hospital beds of the units.

The schedule must fulfill these hard constraints:

- A patient must be scheduled after his or her hospitalization or admittance.
- Emergency patients must be operated on no more than two days after hospitalization—urgent ones within one week.
- Each operating theater usually has two operations per day—at most four operations.

The schedule should meet these soft constraints:

- No operations are allowed on weekends, except for emergencies.
- Normal-priority patients should be operated on in the next six weeks.

The system. Medicus supports the hospital's domain expert in the long-term scheduling of normal-priority patients and the short-term scheduling of urgent and emergency patients. Figure 6 depicts the Medicus user interface. The **`Patientendaten`** (patients data) window, on the upper left, contains important information about all admitted patients. The **`Auslastung`** (load distribution) window, on the upper right, provides a six-week overview of each unit's daily occupancy. The **`Bettenbelegung`** (bed occupancy) window presents a Gantt chart depicting a two-week allocation of the operating theaters and hospital beds.

For long-term scheduling, the user consults the **`Auslastung`** (load distribution) window to determine a day where less than 50% of the capacity of the operating theaters has been allocated. The user schedules a patient by selecting a time unit of an operating theater in the **`Bettenbelegung`** window. Medicus automatically assigns the appropriate hospital beds if no constraints are violated.

To schedule an emergency patient, it is sometimes necessary to delay the operation date of an already scheduled patient with a lower priority. In this case, the user drags the lower-priority patient's entry onto a buffer, and schedules the emergency patient for the vacant operation date. Subsequently, the lower-priority patient must be rescheduled—however, this time with a higher pri-

ority to ensure that the operation date will be delayed only once.

In addition to interactive scheduling, the user can ask Medicus to propose an appropriate operation date for a selected patient. The system determines the next available operation date by using the same heuristics as the human expert, while considering all constraints. The user can confirm the proposal, alter it, or demand another one. If no appropriate operation date is available, Medicus determines the patient whose operation can be delayed with the fewest problems.

Application use and development. All three knowledge-based scheduling systems greatly reduce the time to create and maintain production schedules. For instance, at Sandoz, the application partner of Protos, the manual creation of a predictive schedule of the next half year lasted up to two weeks. With Protos, that task takes a couple of hours. Moreover, users can check alternative machines or routings the human expert never considered. Furthermore, they can detect inconsistencies and errors in the master schedule much earlier and report them to the corresponding logistics department. The PSY system also supports long-term predictive scheduling. Thus, the orders are prescheduled; consequently, fewer capacity problems occur when the detailed weekly schedule is created. With these applications, users can significantly improve the quality of the schedules, thereby improving capacity use and reducing production costs.

All these projects used rapid prototyping to achieve an incremental development process. A project team of university researchers and personnel from the application partners implemented, refined, and extended several prototypes until they produced prototype systems that satisfied the actual needs. We implemented all the systems in Prolog (Quintus Prolog and ProLog by BIM) on Sun SparcStations, and we developed the user interfaces using Prowindows, a Prolog-based user-interface-programming package. An interface to relational databases (ProDBi to Oracle) provided the connection to existing computing environments.

The Protos project lasted three years. However, we reduced the development time of PSY to one year, and that of Medicus to just three months, by exploiting our experience and reusing several concepts realized in the previous projects. We evaluated the final prototypes by field-testing them under realistic conditions in selected sample plants. Finally, we reimplemented the two industrial systems for commercial use. Protos has become PEPI (Production Expert Planning System). PEPI is not only in use on the shop floor in several of Sandoz's chemical multipurpose production plants, but is also available on the commercial market. The PSY system has been further extended and marketed by Competence Center Informatik, a software house not affiliated with the university.

WE REDUCED THE DEVELOPMENT TIME OF PSY TO ONE YEAR, AND THAT OF MEDICUS TO JUST THREE MONTHS, BY EXPLOITING OUR EXPERIENCE AND REUSING SEVERAL CONCEPTS REALIZED IN THE PREVIOUS PROJECTS.

BASED ON OUR EXPERIENCES IN developing these scheduling systems, we are developing a system to be a design assistant for building scheduling systems. This system, part of our Planner's Workbench project, implements our generic framework.[6] Its primary component is a library of scheduling knowledge, containing a set of predictive and reactive scheduling skeletons, as well as a set of sets of selection rules. The system's flexible reuse and easy adaptation of previous scheduling components enables the dynamic customization of new scheduling systems.

References

1. S.F. Smith, "Knowledge-Based Production Management: Approaches, Results and Prospects," *Production Planning & Control*, Vol. 3, No. 4, 1992, pp. 350–380.
2. M. Zweben and M.S. Fox, eds., *Intelligent Scheduling*, Morgan Kaufmann, San Francisco, 1994.
3. H. Adelsberger et al., "The Concept of a Knowledge-Based Leitstand—Summary of First Results and Achievements in ESPRIT Project 5161 (KBL)," *Proc. CIM-Europe Eighth Ann. Conf.*, Springer-Verlag, Brussels, 1992.
4. J. Sauer, "Wissensbasiertes Lösen von Ablaufplanungsproblemen durch explizite Heuristiken" ("Knowledge-Based Solutions to Scheduling Problems Using Explicit Heuristics"), *DISKI Band 37* (*Dissertations in Artificial Intelligence No. 37*), Infix Verlag, St. Augustin, Germany, 1993.
5. J. Sauer, "Meta-Scheduling Using Dynamic Scheduling Knowledge," in *Scheduling of Production Processes*, J. Dorn and K.A. Froeschl, eds., Ellis Horwood, New York, 1993, pp. 151–162.
6. J. Sauer, H.-J. Appelrath, and R. Bruns, "PWB: A Design Assistant for Scheduling Systems," tech. report, Dept. of Computer Science, Univ. of Oldenburg, Oldenburg, Germany, 1996.

Jürgen Sauer is a senior research scientist at the Department of Computer Science of the Carl von Ossietzky University of Oldenburg, Germany. His technical interests include knowledge-based scheduling, logic programming, and software engineering. He holds a Diplom in computer sciences from the University of Dortmund and a Dr. in computer science from the University of Oldenburg. He is a member of the AAAI and the German Computer Society. Contact him at Univ. Oldenburg, FB Informatik, Postfach 2503, D-26111 Oldenburg, Germany; sauer@informatik.uni-oldenburg.de.

Ralf Bruns is a senior software engineer at Software Design & Management GmbH & Co. His technical interests include evolutionary algorithms, constraint programming, knowledge-based scheduling, and software engineering. He received a Diplom and a Dr. in computer science from the University of Oldenburg. He is a member of the German Computer Society. Contact him at Software Design & Management GmbH & Co. KG, Thomas-Dehler-Str. 27, D-81737 München, Germany; ralf.bruns@sdm.de.