Constraint-Oriented Cooperative Scheduling for Aircraft Manufacturing

Patrick Esquirol and Pierre Lopez, LAAS–CNRS
Luc Haudot and Marc Sicard, Dassault Aviation

FOR SOLVING SCHEDULING PROBLEMS IN AN AIRCRAFT MANUFACTURING WORKSHOP, THESE AUTHORS PROPOSE A HUMAN-MACHINE COOPERATIVE FRAMEWORK WHERE COMPUTERS CHECK THE CONSISTENCY OF THOUSANDS OF CONSTRAINTS AND HUMANS SELECT RELEVANT CHOICES.

THE FLANGED-ELEMENT MANUfacturing workshop at Dassault Aviation's plant in Argenteuil, France, uses a number of short-run manufacturing cycles to produce small quantities of numerous aircraft structural parts. The many manual finishing operations, components, and manufacturing processes involved place a premium on management flexibility and quick reactions for meeting ever-changing production demands. Human operators, therefore, must assume a pivotal role in the workshop's decision-making processes.

This case study reports on our efforts to converge a top-down approach to functional design and a bottom-up approach to cognitive design of human-computer interactions to increase efficiency at the workshop. The functional design effort relies on a constraint-reasoning process to provide a set of decision-making solutions, while the cognitive design effort comprises a knowledge-acquisition phase. We combined these concepts to produce a highly interactive mock-up system, programmed with a constraint logic programming language, that lets users easily modify previous choices.

The final system this article describes is still under specification, as part of the ongoing reorganization of the shop. The mock-up design, supported by the French Ministry of Research, has been developed to investigate new technological possibilities in the domain of cooperative human-machine scheduling, but not to be integrated in a Dassault industrial site. If integrated, this mock-up would enable operators to react immediately to perturbations. This work could also migrate to other manufacturing operations, such as cutting operations in the clothing industry, which present similar constraints.

The need for cooperation

We developed this mock-up system as part of the Scoop (Cooperative System for Production Scheduling) project, for the Laboratoire d'Analyse et d'Architecture des Systèmes, French National Center of Scientific Research (LAAS-CNRS), Dassault Aviation, and the European Institute of Cognitive Sciences and Engineering (Eurisco), under funding from the French Ministry of Research. The main goal of the Scoop project was to demonstrate which and how potential technological capabilities (constraint-oriented reasoning as well as graphical environments) can be exploited to design an interactive problem solver in the domain of scheduling. In an ascendant approach, we studied the industrial site of Dassault Argenteuil to build a practical application. We also investigated a descendant approach to list and evaluate the technical possibilities for interacting during a cooperative solution process.

Research under the Scoop project has focused on cooperative scheduling systems

0885-9000/97/$10.00 © 1997 IEEE

operating in small, automated, short- or medium-run workshops. Humans play a major role in such scenarios, first by carrying out specific operations, such as manual alterations, or "finishings," and second by reacting quickly to disturbances, emergencies, and machine breakdowns.

Management decisions, such as schedule making, result from a complex process that relies on various sources of knowledge. Representing these decisions as regularly repeated solutions to an optimization problem is therefore difficult. The example we describe here, from which Scoop began, highlights a scheduling problem that cannot be globally solved by a deterministic program.

Scheduling decisions must simultaneously account for managerial data—due dates, processing times, and machine capacities, for example—as well as specific technological constraints, such as parts pooling constraints on special machines and human preferences when processing manual operations. The problem statement used in classical approaches, such as mathematical optimization, simulation with priority rules, or expert systems, is not adequate here.[1,2] In our case, human operators must react quickly to numerous disturbances, or changes in production demand. Because human expert know-how cannot be embedded in the models these approaches usually propose, the operators must control the problem solving, or at least understand and validate solutions. Yet, provided a major constraint is representable and tractable, it must be considered for the solution.

Our case study assumes that the decision power that humans need to quickly elaborate or adapt solutions for real-time disturbances implies a substantial level of autonomy, particularly when computer programs cannot easily represent the human operator know-how involved. Hence, an approach based on decision support seems better suited for designing such a scheduling system. This study therefore proposes a human-machine cooperative framework for solving scheduling problems that takes advantage both of computers in checking the consistency of thousands of constraints and of humans in selecting relevant choices.[3]

For example, in our case study, the existing batch-oriented tool computes solutions each night, which the user systematically modifies to account for real conditions on the shop floor. The number of daily decisions (around one hundred) is not large compared to the number of constraints. Thus, we have realistically specified a system that interactively integrates the decisions of the operator. It avoids the drawbacks of batch-oriented systems, while advising users about feasible decisions and letting users renege on any of them if the current state no longer fits the user's criteria or is not feasible from the constraint-satisfaction viewpoint.

Background

In a *scheduling problem*, the onset and control of operations should be assured during all processing steps.[1,2] Operations research organizations usually attempt to solve these combinatorial problems by designing exact but costly methods or specific heuristics.[4] A schedule results that represents a forecasted plan for accomplishing the various tasks. Theoretical scheduling problems, which are concerned with searching for optimal schedules subject to a limited number of constraints, suffer from excessive combinatorial complexity. Although more highly constrained and subject to antagonistic criteria, practical problems are even more complex because of the uncertainty of scheduling parameters in the shop floor's dynamic environment, where unexpected events continually occur. As a result, the forecasted plan can no longer be used. In optimization approaches, the objective is often to bypass or simulate human decisions, not to support them.

RATHER THAN CAPTURING THE KNOW-HOW OF HUMAN DECISION-MAKERS THROUGH A RESTRICTIVE, STATIC RULE-BASED MODEL, WE DECIDED TO INTEGRATE HUMANS, RATHER THAN SIMULATE THEM, INTO THE SOLUTION PROCESS.

Many expert systems have attempted to avoid these shortcomings by using other artificial intelligence techniques.[5] These tools emphasize the importance of combining both analytical and empirical knowledge. However, while they are developed in close connection with experts, future users are rarely consulted. Consequently, an interaction model results that is poorly adapted to the users' cognitive model. Moreover, the structuring and management of large knowledge bases remains an open issue.

Rather than trying to capture the know-how of human decision-makers through a very restrictive and static rule-based model, as in expert systems, we decided to integrate humans, rather than simulate them, into the solution process. The case problem that follows shows that an automated decision system is inappropriate. The matter was less to imitate humans when making the best decisions than to identify what pieces of information they need to make their decisions, which is a more realistic and attainable goal.

Case problem. The manufacturing process at the flanged-element manufacturing workshop we studied includes part routings in a cutting-out center, various heat treatments, flanging on a hydraulic press, and manual finishing. Aircraft workers cut parts out of thin sheet metal, then form them with a hydraulic press. Routings can differ for each product, so the workshop operates as a *job shop*. The workshop produces about 1,000 parts each week from 3,000 types, structured in 40 families or classes, with 12 different operating sequences. Each part is identified by its reference number, production routing, class, and delivery date.

Cutting machine operations involve determining the sets of parts that must be pooled on each metal sheet. The *built sets* must satisfy various requirements relating to the class, geometric overlapping, filling rate, and time constraints associated with the part delivery dates. This *processing-order release* phase governs the release of the parts in production and strongly influences the scheduling of the next operations. In this article, we limit our scope to the cutting workstation and therefore to this release function. (In parallel to the Scoop project, another effort led to the specification and prototyping of the MADE schedule system for this workshop.[6] The MADE project aims to build a solution generator, rather than a cooperative system, as in our work. MADE stands for *M*odule d'*AiD*e à l'*E*ngagement, which is French for *release support system*.)

At present, a batch process automatically creates the parts sets each night; the operator modifies or validates the sets the follow-

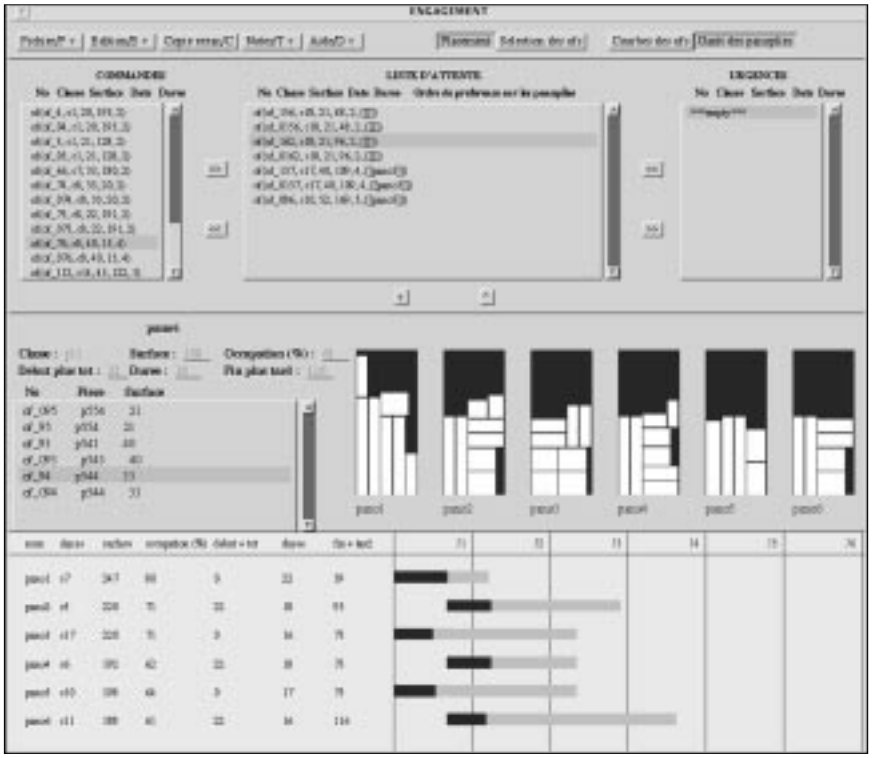

Figure 1. Main screen of the mock-up. This screen dump shows browsers that enable users to select and sort domains (left is the "normal list," right is the "urgent list," and the central one focuses attention on the orders that the user has finally retained, then sorted according to his or her criteria). The center of the screen represents the current contents of each of six metal sheets; this is graphically displayed to show how each sheet has been filled. The bottom of the screen is associated to a temporal representation of sheet-cutting operations.

ing morning. A new batch process must run for each modification. This system suffers many drawbacks, including

- *No indicators.* The system does not display indicators about such factors as workshop or processing-order conditions, resource breakdowns or overloadings, and delivery dates distribution, which would allow the operator to anticipate problems.
- *No assistance.* The system does not help users sort processing orders or choose candidate sheets for the best adapted solution.
- *No flexibility or interactivity.* The system allows neither interactive modification of sheets nor a fast computation and display of processing order placement or sheet duration and sequence. Quick reactions depend on such information.

These drawbacks arise because of the large number of part categories, products, and routings; dissimilar constraints (geometric, temporal, and grouping, for example); the nature of the workshop (sequential operations executed by automated and manual processes); and a number of critical disturbances (machine breakdowns, absent operators, and urgencies) that demand quick reaction times.

This batch-processing approach would lead to a model whose rigidity, combined with the requirement for an all-embracing solution, would make the corresponding tool of little use.

Cooperative scheduling. For these general reasons, and taking into account the real needs at the Dassault workshop, we developed an approach based on cooperative scheduling. In other application domains, the need for human-computer cooperation has become increasingly important, particularly with the growing development of group decision-support systems, or *groupwares*.[7] For our purposes, we consider a cooperative system as an organization in which the human and the computer are two interacting cognitive poles engaged in jointly carrying out a task. This definition suggests the sharing of goals by cooperative entities, which offer complementary knowledge and skills. Cooperation, a relatively new concept in scheduling, might look provocative in the context of production systems for which automation has long been the means of choice for improving efficiency.

Nonetheless, two systems drew our attention. Ordo, a software program for real-time scheduling developed by France's Cabinet Villaumié S.A., offers a family of schedules, which appear as a totally ordered sequence of groups of exchangeable tasks.[8] (Ordo stands for *Ordo*nnancement, or scheduling in French.) Forty French industrial sites use it for various application domains. Second, the Scheplan software program provides a cooperative environment for creating production plans in the steel industry.[9] Constraint-resolution procedures and expert rules let users directly modify the plan through an interactive interface. In creating the system, Scheplan's designers thoroughly considered user-interaction modes. Its architecture is defined around a constraint solver.

Ordo and Scheplan are rare examples of software that includes constraint-propagation mechanisms and enables limited, but not negligible cooperation modes. Ordo calculates a particular subset of schedules that satisfy all constraints and that can be represented as sequences of groups of permutable tasks. The user must choose from this set. In Scoop, the user is not restricted to particular forms of solutions; the main criteria is to not violate any constraints, and if so, to backtrack to the user's previous decision. In Scheplan, solution modifications are enabled but always on the basis of an initial solution that is always built by the system. This is not the case in Scoop, where the user entirely designs the solution.

Establishing cooperation between a human and a system requires that the degrees of freedom associated with the initial plan be made clear. As we discuss later, that is the objective of *constraint-based analysis*. Furthermore, to better understand the cognitive processes involved in problem solving, part of this work deals with knowledge acquisition from the users of current systems. The "Supporting the cooperation" sidebar discusses knowledge acquisition and constraint-based analysis.

Application

The knowledge-acquisition results, along with the implementation of the techniques discussed in the sidebar, enabled us to build an order-release cooperative system mock-up. (Figure 1 shows its main screen.) It allows us to handle all the processing orders originating from a production release office, as well as urgent purchasing orders that are bound to occur when demands on the shop change.

The resolution strategy lies in furnishing a

Supporting the cooperation

Development of the cooperation processes used in this case study involved both knowledge-acquisition procedures and constraint-based analysis.

Knowledge acquisition

The methodology we used grew out of a convergence of two approaches:

- Kelly's approach, which proposes the *personal construct* methodology,[1] and
- Warfield's methodology, which suggests the use of *brainwriting* techniques to elicit expert knowledge.[2]

To develop the knowledge elicitation in our approach, we adapted the original Warfield methodology by using a grid support on which each column has precise semantics (idea number, reference number, opinion, or idea). This methodology allows the collation of knowledge. In this approach, everyone gives his or her opinion, free from the influence of anybody else.[3]

To design cooperative systems, the collection phase must be conducted not only with the future user, but also with a broader population of experts, potential users, designers, ergonomists, and interested others. Indeed, such a collection phase gives designers valuable information, ideas for innovations, and various viewpoints for use in building flexible, reusable generic systems. Moreover, for cooperative development, such an approach exhibits many decision-support mechanisms that an approach by an individual would not identify.

During each knowledge-elicitation session (four in our case, each lasting two hours), we collected 10 grids per participant; each grid contains approximately 30 general ideas, yielding more than 600 elementary ideas. We then classified the collected knowledge and structured it in sets of trees: display modes (graphs, arrays, curves, bar graphs, and charts, for instance), actions (for example, add, delete, modify, and allocate), and so forth.

Next, we analyzed this information. For this, we modified the work of G. Kelly, which John Boose and Jeffrey Bradshaw used in Aquinas.[4] The analysis is built by crossing ideas to determine the links between concepts.[5] Mildred Shaw and Brian Gaines have identified four relevant cases (consensus and three classes of disagreement points).[6] In our study, we merge all disagreement points for which new knowledge-acquisition campaigns are restarted. Many disagreement points are associated with the vocabulary, so we built a consensual ontology in collaboration with all participants, which lets us remove most of these differences.

This processing phase generates cooperative system design directives for three important fields: flexibility, support, and interactivity.

Constraint-based analysis

The major role of the constraints of the problem at hand, along with our desire to characterize a set of admissible solutions, led us to use a constraint logic programming (CLP) tool that includes efficient constraint-propagation mechanisms.[7] Such languages aim primarily at proposing a solution to a problem; therefore, if you try to decrease the execution time, the propagation procedures will be incomplete. (Except for the case of rational constraints, most CLP languages do not ensure the completeness of their constraint-propagation procedures on the grounds of execution time savings: sometimes adding a new constraint to a set does not create an inconsistency, although the new set of constraints cannot be satisfied. To eliminate such situations as much as possible, we can program additional mechanisms, dedicated to a specific application domain.)

To enforce decision-problem consistency, we chose *constraint-based analysis* (CBA) principles.[8,9] This analysis is not designed to generate a solution relative to a criterion but to prepare the generation phase. It is based on a set of deductive rules that reveals some essential properties for all solutions. Several criteria permit us to exploit complementary aspects between CLP and CBA for cooperative scheduling, leading to a *constraint-oriented approach*.[10]

CBA applied to parts pooling. We use CBA primarily to represent and operate the time constraints that affect the pooling of parts required on a metal sheet. Because there is only one cutting machine, a *disjunction constraint* exists between any sheet couple (paired metal sheets), which can result in narrowing sheet time windows through basic CBA rules. By applying these rules, users can obtain two types of results:

- *Detection and processing of inconsistencies.* During the placement attempt, failure will occur if the current sheet window cannot accommodate the duration of the operation associated with placement, or if one of the disjunction constraints between two sheets cannot be satisfied.
- *Characterizing the feasibility of the regrouping decisions.* Whenever a new placement occurs, the time characteristics of the sheet concerned are modified. In certain cases, this may cause sequencing with other sheets through disjunctive constraints.

The temporal problem. Aircraft workers place parts on sheets with respect to three types of temporal constraints: individual time windows (one for each part); parts-aggregation constraints (one for each sheet); and disjunctive constraints on the cutting machine.

Constraint type 1 associates an initial time window `[earliest start-time, latest finish-time]` to realize the cutting of each part. It is derived from the time window `[release date, due date]` associated with the corresponding order and the knowledge of the durations of the operations following the cutting.

Constraint type 2 comes from some technological features of the cutting machine. For each metal sheet, cutting operations are chained; no part can be available before the sheet's whole cutting process finishes. This implies that

- The aggregated cutting operation on one sheet cannot start before the maximum earliest start time of its components.
- The aggregated cutting process for one sheet cannot finish after the minimum latest finish time of its components.

Consequently, a time window associated with each sheet represents the temporal intersection of individual time windows associated to elementary cutting operations of its parts. This time window must be large enough to include the sum of the cutting durations.

Finally, disjunctive constraints (type 3) come from the use of a single cutting machine, which implies that aggregated operations must be sequenced.

Time constraint propagation. Although this section discusses the principles of CBA rules used in our case problem, for brevity's sake we do not discuss the associated rules in detail. More precise information is available that provides an analytical formulation of rules as well as proofs.[10]

We first propose a rule that examines the consistency of each sequencing alternative of the disjunctive constraints. When one proves infeasible, taking into account the current time windows of the tasks (as illustrated at the top of Figure A), the opposite alternative must be posted. This leads by propagation to a possible adjustment of the time windows

of the pair of tasks thus sequenced (bottom of Figure A).

This mechanism instantiates what some call *arc-consistency*[11] and others call *immediate selection*,[12] and also lets us solve some disjunctive constraints and consequently tighten time windows.

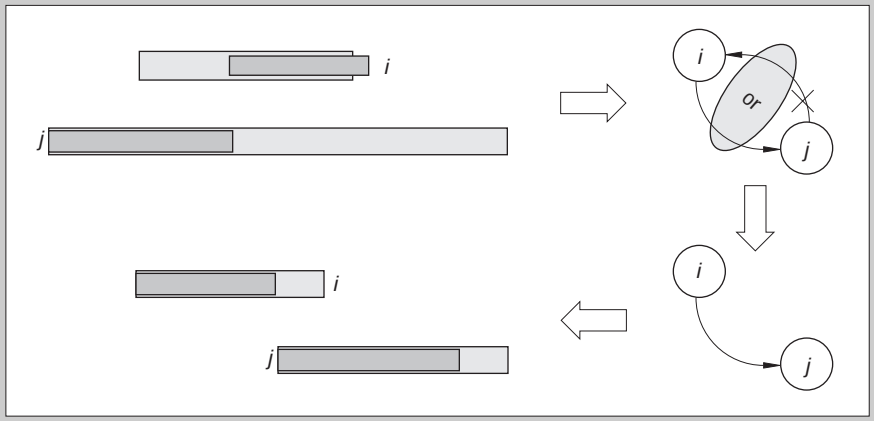

Figure A. Disjunction solving and updating.

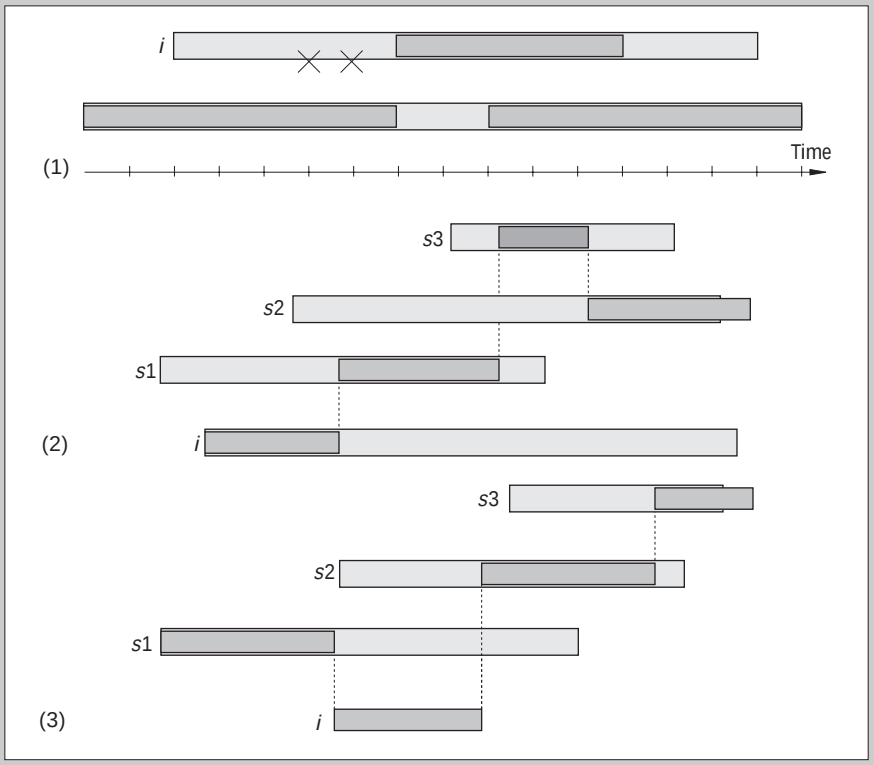

Figure B. Time windows: (1) inconsistent start times for *I*; (2) {*s*1, *s*2, *s*3} nonposterior to *I*; and (3) *i* noninsertable into {*s*1, *s*2, *s*3}.

Because this sole type of consistency is not complete, to actually achieve arc-consistency, an additional CBA rule detects intermediate task positions that violate both sequencing alternatives of a disjunctive constraint. This procedure also removes inconsistent dates, which could lead to the creation of holes in the domain associated with start times (see Figure B1). In practice, these two rules might be useless, given the very large time windows compared to processing times. It might then also be interesting to enlarge the study by considering extreme positions of a task relative to a group of other ones.[12,13]

Figure B2 identifies a subset $S = \{s1, s2, s3\}$ *nonposterior* to task *i*. At least one task of *S* must precede *i*. After such an identification of a nonposterior set, a following step might determine whether task *i* can be inserted into *S*. If not (as in Figure B3), *i* must be scheduled either before or after *S*. Moreover, if *S* has previously proved to be nonposterior to *i*, then *i* is necessarily scheduled after all tasks in *S*. Thus, we can deduce a lower bound for the start time of *i*, as well as an upper bound of certain finish times of *S*—values much stronger than under the sole nonposterior condition. An analogous reasoning symmetrically establishes *nonanterior* set conditions.

Cooperation processes

This constraint-oriented approach contributes to cooperative scheduling by showing infeasibilities as soon as the current set of decisions becomes inconsistent, and, by restricting the possibilities left to the parts not yet placed.

The knowledge-acquisition procedure also led us to derive information for the establishment of cooperation processes. The following points arose:

- The possibility of questioning the validity of already-made decisions, facing the dynamic context in the shop floor.
- The anticipation necessary for making advance decisions. Indeed, at each step of the decision-making process, the tool must exhibit the most robust choices, thereby limiting back-tracking.

Moreover, the decision sharing remains a preponderant factor: the machine (and specific heuristics) solves the problems, whether or not they are well modeled or highly com-

selection aid for those processing orders that must be processed as a priority. As soon as a user selects a processing order, it goes on the waiting list and a second decision aid, *the placement aid*, comes into play. Any selection or placement decisions carried out from the start of the resolution can be canceled in any order, which relaxes the related constraints. Figure 2 represents the architecture we have designed for a cooperative release system.

Selection aid. This process forms and makes explicit the subset of processing orders that must be placed in order of priority. The decision-maker views several presentation modes:

plex, while the human makes decisions for incompletely formalized problems.

To implement a cooperative tool for the scheduling problem, we devised the architecture shown in Figure C.

This kind of tool must provide several levels of cooperation:

- *Information exchange.* To build a schedule in cooperation, both the operator and the system exchange information; the first provides nonexplicit constraints coming from the real context; the second gives the current state of the solution process and also information about data.
- *Anticipation.* To help the user in a timely fashion, the system must eliminate inconsistent remaining decisions.
- *Explanation.* When a failure occurs, the system must provide some explanation as soon and as simply as possible.
- *Sharable decisions.* The distribution of decisional functions between both interactive poles (operator and system) must be possible dynamically, or decided statically.
- *Decision support.* The system's decision support comes from combining the analysis before the decision (by selection and classification of the most appropriated decisions) and the analysis after the decision (by inferring the consequences and eliminating wrong values for the remaining variables).

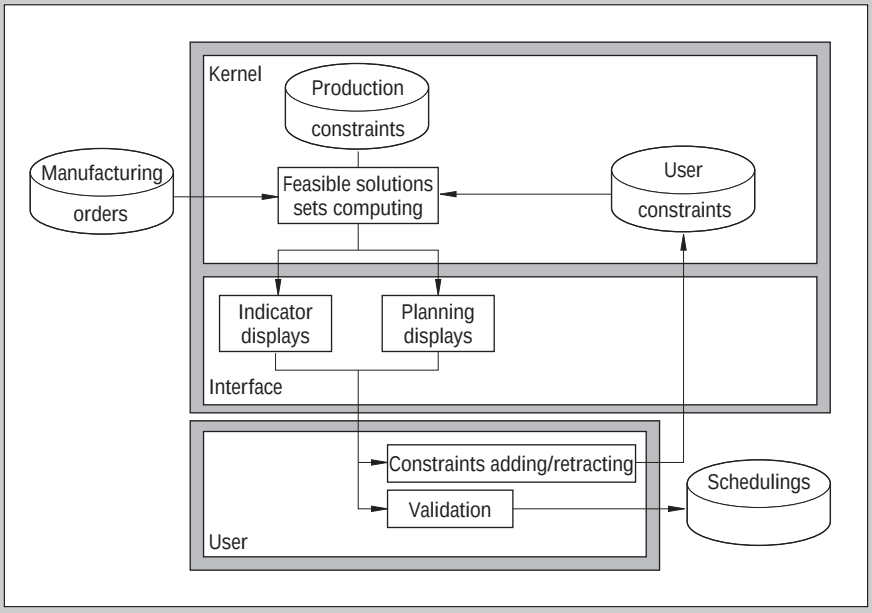

Figure C. Functional principles for cooperative scheduling.

References

1. G. Kelly, *The Psychology of Personal Constructs*, Norton, New York, 1995.
2. J.N. Warfield, *Societal Systems: Planning, Policy and Complexity*, John Wiley & Sons, New York, 1971.
3. L. Haudot et al., "A Knowledge Acquisition Approach for the Ergonomical Design of Cooperative Systems: The Scoop Project Experiment," *Proc. Fifth Int'l Conf. Human-Machine Interaction and AI in AeroSpace*, Eurisco, Toulouse, France, 1995.
4. J.H. Boose and J.M. Bradshaw, "Expertise Transfer and Complex Problems: Using Aquinas as a Knowledge Acquisition Workbench for Knowledge-Based Systems," *Int'l J. Man-Machine Studies*, Vol. 26, No. 1, 1987, pp. 3–28.
5. J.M. Bradshaw et al., "Beyond the Repertory Grid: New Approaches to Constructivist Knowledge Acquisition Tool Development," in *Knowledge Acquisition as Modeling*, K.M. Ford and J.M. Bradshaw, eds., John Wiley & Sons, 1993, pp. 287–333.
6. M.L.G. Shaw and B.R. Gaines, "A Methodology for Recognizing Consensus, Correspondence, Conflict, and Contrast in Knowledge Acquisition Systems," *Proc. Third AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop*, AAAI Press, Menlo Park, Calif., 1988, pp. 30.4–30.79.
7. J. Jaffar and M.J. Maher, "Constraint Logic Programming: A Survey," *J. Logic Programming*, Vols. 19–20, 1994, pp. 503–581.
8. J. Erschler, F. Roubellat, and J.-P. Vernhes, "Finding Some Essential Characteristics of the Feasible Solutions for a Scheduling Problem," *Operations Research*, Vol. 24, 1976, No. 4, Jul.–Aug., pp. 774–783.
9. J. Erschler and P. Esquirol, "Decision Aid in Job Shop Scheduling," *Proc. IEEE Int'l Conf. Robotics and Automation*, IEEE Press, Piscataway, N.J., 1986, pp.1651–1656.
10. P. Lopez et al., "Constraint-Based Approach to Design a DSS for Scheduling," *Proc. Third Int'l Conf. Practical Application of Prolog*, The Practical Application Co., Ltd., Blackpool, UK, 1995, pp. 405–422.
11. A.K. Mackworth, "Consistency in Networks of Relations," *Artificial Intelligence*, Vol. 8, No. 1, 1977, pp. 99–118.
12. J. Carlier and E. Pinson, "A Practical Use of Jackson's Preemptive Schedule for Solving the Job-Shop Problem," *Ann. Operations Research*, Vol. 26, 1990, pp. 269–287.
13. M.-L. Levy, P. Lopez, and B. Pradin, "A Decomposition Approach for the Single Machine Scheduling Problem," *J. Decision Systems*, Vol. 5, Nos. 1–2, June 1996, pp. 73–94.

- *Graphical representation of processing orders.* Two representations permit the detection of load peaks at certain dates and for certain types of processing orders. First, the system displays the proportion of processing orders for each class for the list of processing orders sent to the production release office (see Figure 3a). Second, for each class, a curve shows the time distribution of the processing orders according to their delivery date (see Figure 3b).
- *Filtering and processing order classification.* The system can filter the list of processing orders according to class,

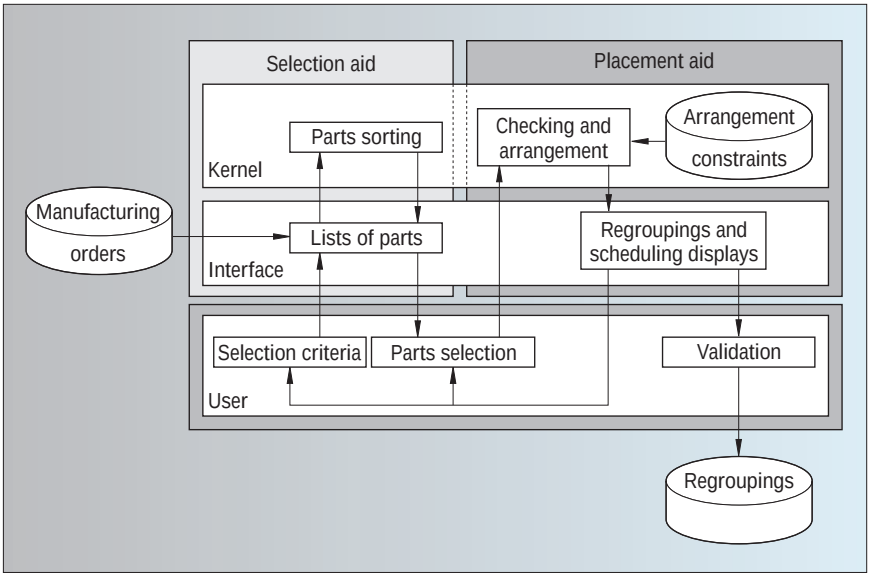

Figure 2. Human-computer cooperation for release.

production route, surfaces, or delivery dates. For waiting lists, these orders can appear in ascending or descending order of surfaces or delivery dates.

Placement aid. Users also receive assistance from two types of placement aids:

- *Feasibility analysis and constraint propagation.* At any time, the operator can try to assign one processing order to a sheet. If placement fails, the screen displays the cause of failure—insufficient amount of space left, date later than the delivery date, or incompatible classes, for example. If placement succeeds, the sheet data evolve, and certain characteristics of the processing orders still in the waiting list are adjusted.
- *Sheet display.* The screen permanently displays geometric solutions for placing the parts on the sheets (see Figure 4a). Synthetic information, or textual description, is also available for each sheet (class, occupied surface, current filling rate, time window, and duration). A Gantt chart helps users visualize the positioning of the sheets at a certain moment in time, showing marginal losses caused by regroupings (see Figure 4b). These two representation modes (in space and time) dynamically are updated when a decision is made or upon cancellation.

Implementation and first results. We implemented our system mock-up on a Sun Sparcstation 2 in the Chip constraint logic programming (CLP) language. To validate the mock-up, we have selected a realistic example, consisting of 300 parts to be manufactured, 70 processing orders, a one-week production horizon, and six sheets to be completed each day. In this example, the expected waiting time caused by constraint propagation is negligible, so users become highly interactive with the system and can easily modify their previous choices.

WHILE AN ERGONOMIC EVALUation of this mock-up has pointed out the merits of our system for this kind of application, it still needs validation by users. This validation phase belongs to the normal incremental cycle of prototyping, and will obviously involve some adjustments and innovative improvements. Further work will focus on a more systematic validation of the mock-up with potential users to enhance the system interactivity and develop better functionalities.

The Scoop project was completed in 1995. Unstable market forces have caused critical changes inside the Dassault organization, so the Scoop project has not been a high priority and the mock-up has not evolved further. Nonetheless, this project demonstrates the validity of the idea that computers must support human decision-makers, rather than imitate or replace them, particularly when enforcing the decision consistency needed to check a huge amount of constraints.

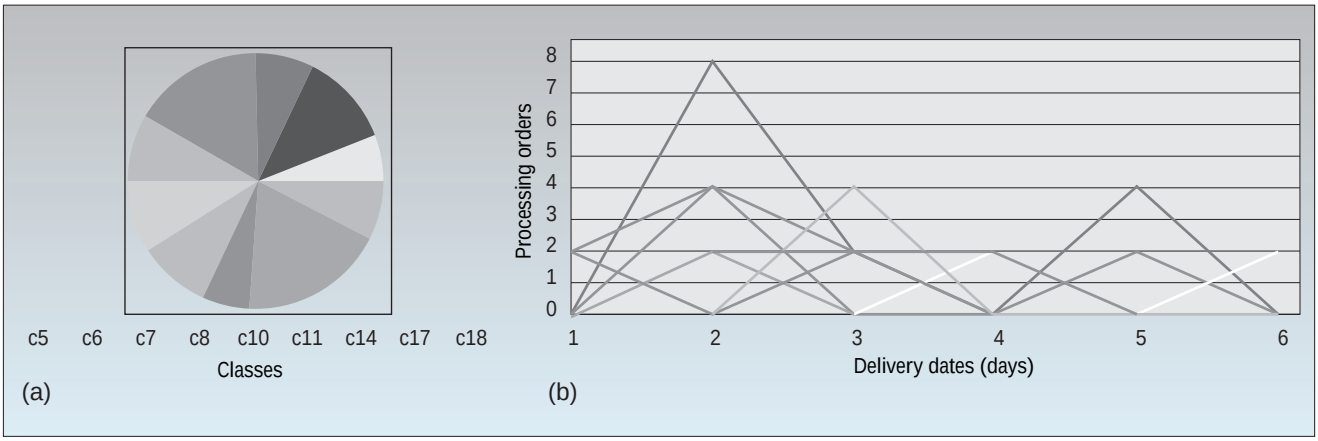

Figure 3. Processing order representations: (a) the proportion of processing orders for each class for the list of processing orders sent to the production release office; (b) for each class, the time distribution of the processing orders according to their delivery date.

Acknowledgments

This research received partial support from the French Ministry of Research under Contracts 92–P–236/237/238 (Scoop project) with LAAS-CNRS, Dassault Aviation, and the European Institute of Cognitive Sciences and Engineering (Eurisco). We are indebted to Guy Boy (Eurisco) and Jeffrey Bradshaw (Boeing Computer Services) for their assistance on knowledge-acquisition procedures.

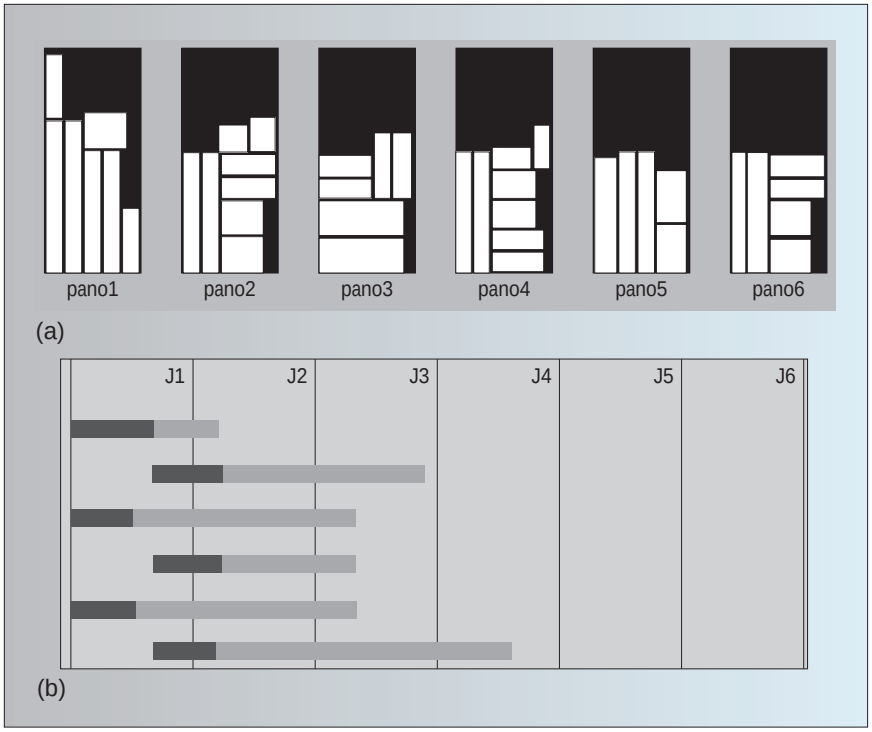

Figure 4. Placement aids: (a) placement area; (b) Gantt chart.

References

1. K.R. Baker, *Introduction to Sequencing and Scheduling*, John Wiley & Sons, New York, 1974.

2. M. Pinedo, *Scheduling: Theory, Algorithms and Systems*, Prentice Hall, Englewood Cliffs, N.J., 1995.

3. J. Grudin, "Computer-Supported Cooperative Work: History and Focus," *Computer*, 1994, Vol. 27, No. 5, May, 1994, pp. 19–26.

4. B.L. McCarthy and J. Liu, "Addressing the Gap in Scheduling Research: A Review of Optimization and Heuristic Methods in Production Scheduling," *Int'l J. Production Research*, Vol. 31, No. 1, 1993, pp. 59–79.

5. S.J. Noronha and V.V.S. Sarma, "Knowledge-Based Approaches for Scheduling Problems: A Survey," *IEEE Trans. Knowledge and Data Engineering*, Vol. 3, No. 2, June, 1991, pp. 160–171.

6. J. Bellone, A. Chamard, and A. Fischler, "Constraint Logic Programming Decision Support Systems for Planning and Scheduling Aircraft Manufacturing at Dassault Aviation," *Proc. Third Int'l Conf. Practical Application of Prolog*, The Practical Applications Co., Ltd., Blackpool, UK, 1995, pp. 63–68.

7. C. Ellis, S. Gibbs, and G. Rein, "Groupware: Some Issues and Experiences," *Comm. ACM*, Vol. 34, No. 1, Jan. 1991, pp. 38–58.

8. J.-C. Billaut and F. Roubellat, "A New Method for Workshop Real-Time Scheduling," *Int'l J. Production Research*, Vol. 34, No. 6, 1996, pp. 1555–1579.

9. M. Numao and S. Morishita, "Cooperative Scheduling and its Application to Steelmaking Processes," *IEEE Trans. Industrial Electronics*, Vol. 38, No. 2, 1991, pp. 150–155.

Patrick Esquirol is an associate professor at the French National Institute of Applied Sciences of Toulouse (Insat), where he teaches algorithms and programming, logic programming, AI techniques, and expert systems. His research activities, carried on at the Laboratory for Analysis and Architecture of Systems (LAAS) of the French National Organization of Scientific Research (CNRS), concern the design and application of combined AI and operations research techniques to production management in manufacturing systems, such as constraint-based reasoning in scheduling or distributed-decision architectures for cooperation in planning and scheduling. He received an MS in control engineering and a PhD in computer science and industrial engineering from Paul Sabatier University, Toulouse. He can be reached at LAAS-CNRS, 7 avenue du Colonel Roche, 31077 Toulouse Cedex 4, France; esquirol@laas.fr.

Pierre Lopez is a researcher at LAAS-CNRS. His research interests include scheduling problems, temporal decomposition, constraint-based analysis, constraint logic programming, and human-machine cooperation. He also teaches production scheduling, graph theory, and discrete-event simulation. He received an MS and a PhD in control engineering from Paul Sabatier University. He can be reached at LAAS-CNRS, 7 avenue du Colonel Roche, 31077 Toulouse Cedex 4, France; lopez@laas.fr.

Luc Haudot has held a doctoral position in collaboration with Dassault Aviation, LAAS-CNRS, and the European Institute of Cognitive Sciences and Engineering (Eurisco). He is a research associate at LAAS-CNRS. His research interests include scheduling problems, constraint logic programming, knowledge acquisition, and human-computer interaction. He received an MS and a PhD in computer science from Paul Sabatier University and Insat, respectively. He can be reached at LAAS-CNRS, 7 avenue du Colonel Roche, 31077 Toulouse Cedex 4, France; haudot@laas.fr.

Marc Sicard is a computer scientist at the AI department of Dassault Aviation. His research interests include scheduling problems, constraint logic programming, and cognitive sciences. He received an MS in AI and robotics from Paul Sabatier University. He can be reached at Dassault Aviation, DGT/DTN/EL, 78 Quai Marcel Dassault, 92214 St. Cloud, Cedex, France; sicard @dassault-avion.fr.