Two Years On-line: a Dynamic Scheduler for a Hot Steel Mill.

Virginio Chiodini

Gensym Corporation
125 Cambridge Park Drive
Cambridge, MA 02140

Abstract

This paper describes the development and installation of the 5 Mill Scheduler, a dynamic scheduling application for a hot steel mill. The main objective of the scheduler is to dynamically generate and revise schedules, and make effective use of the resources of the plant, while satisfying a complex set of constraints dealing with the dynamic behavior of the resources and the material processing specifications. The design of the system evolved from a knowledge-based approach, as suggested by the user, to a constraint-based reasoning approach embedding user supplied heuristics to prune the search space. The constraint-based approach proved to significantly outperform a pure knowledge-based approach. The goal of the scheduler was to operate on-line, closely linked to monitoring devices. The installation of the scheduler required some refinements of the design to cope with imprecise data and with the need to provide greater flexibility to the operators in dealing with resource constraints. The scheduler was installed in February 1992 and has been continuously operating on-line since that date.

1. The application domain.

Number 5 Mill is a hot rolling mill. It heats billets and ingots to required temperatures, according to specific heating cycles, and then runs them through rolls to reduce the cross-section areas of the material to customer specification.

The Number 5 Mill is a multi-faceted operation with a wide product range:

50 different heating cycles covering 350 grades of steel, with processing temperatures ranging from 1500°F to 2425°F.

16 furnaces of four different types, with a variety of dynamic characteristics and throughput rates, feed two hot rolling mills. Some of the furnaces have a very high efficiency, capable of dynamically switching their operating modes between continuous, intermittent, and batch. The layout of the plant is shown in Fig.1.

Each piece of material has to arrive at the mill at a precise time to ensure proper material composition of the steel and to keep the rolling mill fully utilized. A sophisticated SCADA system monitors each individual piece, from the moment it is loaded into one of the furnaces to the moment it exits the rolling mill. At Number 5 Mill, scheduling the furnaces is a mission-critical task. The complexity of the problem is defined by the number of constraints and the frequency of re-scheduling. Many times a day, new orders are entered and the current schedules must be revised within the constraints imposed by the ongoing implementation of the current schedules.

2. The Design of the 5 Mill Scheduler.

The scheduling problem involves scheduling a set of jobs $J=(J_1,\ldots,J_n)$ on a set of resources $RES=(R_1,\ldots,R_m)$. Each job J_i, originated by an order, consists of a set of tasks $TK=(TK_1,\ldots,TK_w)$, one for each operation in the process plan of the ordered product.

Two alternative problem solving approaches were considered: a heuristic dispatching (knowledge-based) approach and a constraint-directed search approach.

The knowledge-based approach solves the scheduling problem by iterating over each resource and deciding which tasks among those still to be scheduled should be processed next by the resource itself. The decision is based either on heuristic knowledge, usually encoded in the form of rules, derived from domain experts or from Operations Research heuristics. The scheduling process is monotonic and is able to produce schedules fairly quickly. However, its absolute reliance on heuristics and its inability to evaluate alternative

decisions may generate poor schedules. Furthermore, heuristic knowledge is context dependent and may require extensive revision whenever the structure and dynamics of the production process change.

The Constraint-directed search approach to scheduling is described in [1, 2, 3, 4, 5, 6, 8, 9]. Using this approach, a task is described as a vector of the following variables:

- The set of different resources $\{R=(R_1,\ldots,R_k)\}$ requested to perform the operation associated with the task.
- The set of time-intervals $\{T=(T_1,\ldots,T_k)\}$ during which each one of the required resources is demanded.

Each variable may assume a finite (discrete or continuous) domain of values. $R_i:\{r_{k1},\ldots,r_{in}\}$, $T_j:\{T_{j1} — T_{j2}\}$.

A declarative language based on consistency techniques offers substantial advantages over a heuristic approach:

- Constraints can be formulated in symbolic manner, enabling a more intuitive and natural formulation of the problem.
- Applications are easily modified and extended, due to the separation of the definition of constraints from the way they are applied.
- Control of knowledge can be stated in a declarative form, enabling a fast tailoring of scheduling algorithms to specific problems.

Because the scheduling problem is an NP-complete problem, a constraint-directed search could take exponential time in the worst case. However, both experimental and empirical studies indicate that, on the average, the amount of searching required to find a solution can be significantly reduced by judiciously selecting the order in which variables are processed (variable ordering heuristics), the values that are assigned to the variables (value ordering heuristics), and the conflict-resolution method (repair method) to be used whenever the search process encounters a dead-end.

Variable ordering heuristics always focus on the variables that are the most difficult to process, to avoid building partial solutions that cannot be completed later on. The goal of the value ordering heuristics is to select for a specific variable a value that leaves the largest number of solutions open to the variables still to be processed. Conflict resolution strategies are activated whenever the constraint-directed search is unable to find a value to assign to a specific variable. Conflict resolution strategies perform an intelligent backtracking, changing the sequence in which variables are processed and/or the values assigned to the variables

We preferred a constraint-search approach versus a heuristic dispatching approach, because we estimated that a constraint-directed search process could generate better schedules and that user supplied heuristics could effectively be used to guide the search-process.

The scheduling engine performs the constraint-directed search by iterating over the following cycle:

1. Select a task to be scheduled applying variable ordering heuristics.

2. Apply backward consistency enforcing procedures.

3. If no reservation is available for any of the requested resources then

   3.1 relax constraints and/or select a conflict resolution method.
   3.2 go to 1.

4. Select the reservations for the task (resource and time interval) applying value ordering heuristics.

5. Apply forward consistency enforcing procedures.

6. If a dead-end is detected then

   6.1 relax constraints and/or select a conflict resolution method.
   6.2 go to 1.

7. Create a new search state by adding the new reservation assignment to the current partial schedule.

Backward consistency checks the availability of values to be assigned to a variable, checking the constraints against the variables already processed. Forward consistency checks the availability of values for the variables still to be processed,

checking the constraints against the value selected for the variable being processed.

Constraints relaxation simply reformulates part of the scheduling problem, increasing the size of the solution space by relaxing soft constraints. Constraints are relaxed according to a priority that takes into account their cost and the probability of making the search process converge toward a satisfactory solution. For example in the 5 Mill application possible constraint relaxations included:

- Increasing the nominal capacity of some resources

- Extending the conditions under which tasks are allowed to be batched together.

The 5 Mill Scheduler applies two conflict resolution methods:

- Resource Permutation.

- Push Forward

The Resource Permutation method saves the current status of the search process and iteratively identifies conflicting set of tasks, selects a task in the conflict set for which alternative resource selections are possible and changes the resource assignment. The iterating process terminates when the original dead-end is removed, or a user defined time period assigned to the conflict resolution method expires, or an infeasibility condition is identified.

The use of the Resource Permutation method was prompted by the diversity in the structure and dynamics of the resources available for an operation. A change in the resource selected to perform an operation may dramatically affect the search process, improving the global utilization of the resources beyond the capability of the value ordering heuristics.

When the Resource Permutation fails to remove the dead-end situation, the Push Forward method is invoked. The Push Forward method iteratively shifts the schedule of conflicting sets of tasks, and possibly the schedule of their downstream tasks, forward in time. The possible outcome of this method is a set of delays forced in the requested flow of the mill, and therefore a lower quality schedule.

3. The Modeling Scheme.

The Process Model describes the structure, behavior, products, and goals of the manufacturing system to be scheduled. The Process Model is defined by selecting and extending a predefined library of object classes. Following the representation scheme described in [7], the main predefined object classes are:

States: defining the final products of the manufacturing process (for example: Finished-Steel) and the intermediate stages through which the process must proceed to reach a final product (for example: Hot-steel).

Operations: defining the activities required to perform transitions between two states. For example the Rolling operation transforms Hot-steel into Finished-steel. The graph of operations required to perform all the state transitions necessary to reach a product is called a "process plan".

Resources: defining the physical entities required to perform an operation. Examples include machines, manpower, and tools. Each operation may require multiple resources and each resource may be requested by operations belonging to different process plans.

Orders: requesting the creation of instances of specific states (order's final state). Orders may directly request the instantiation of the final state of a process plan or of any one of its intermediate states. Orders may specify a due-time, a release-time (the earliest time at which operations requested for the order may start) and a priority.

Constraints: providing a set of declarative assertions defining the dynamic behavior and constraints of the process to be scheduled . Constraints are logically divided into unary constraints that unconditionally restrict the domain that a scheduling variable may assume (for example: the set of furnaces allowed to process a certain grade), and binary constraints expressing mutual dependencies between the domains of two variables (for example the set-up time between the rolling of two different grades of steel). Some of the constraints used by the 5 Mill Scheduler are shown in Fig. 2.

Task Focus Guidelines: defining the variable ordering heuristics. Guidelines are called at the beginning of each scheduling cycle to select the tasks to be scheduled during the cycle and the sequence in which they should be scheduled. Guidelines specify also whether tasks should be scheduled Just in Time or As Soon As Possible.

In the 5 Mill Scheduler tasks are scheduled just in time in a sequence defined by a fixed set of variable ordering heuristics:

1. Tasks with an available number of resources below a defined threshold.

2. Tasks with an available slack below a defined threshold.

All the other tasks are scheduled in ascending order of late-finish-time.

Resource Preference Guidelines: defining the value ordering heuristics for the resources to be assigned to perform a task.

Both constraints and scheduling guidelines may include action procedures, allowing the user to express complex evaluation functions. In the 5 Mill Scheduler the definition of action procedures was facilitated by a natural language interactive editor and by a variety of graphic tools available in G2®, the software platform used for the development.

In the 5 Mill Scheduler, the Process Model is defined and updated in a special Model Editing Session and compiled into efficient run-time control structures.

4. The Installation and Operation of the 5 Mill Scheduler.

The 5 Mill Scheduler was installed in February 1992 and has been continuously operating on-line since that date. The scheduling function is automatically activated when a set of orders is released. When a schedule is completed, it is revised by the operator. The operator may firm or change the scheduling decisions made. When changes are applied, the operator resubmits the set of orders to be scheduled. In this case, the firmed schedule decisions and the applied changes are used by the 5 Mill Scheduler as additional constraints to be satisfied in the search for the schedule. On average, almost 5% of the schedules are manually edited by the operators.

The installation and the operation of the scheduler confirmed the validity of the constraint-directed search approach for the specific domain.

1) A declarative constraint language enabled a more intuitive and natural formulation of the problem, simplifying the knowledge acquisition phase. Once the constraint-based approach was decided, the dynamic behavior of the manufacturing system and, in particular, the complex dynamic behavior of some resources could be accurately defined, without the simplifying descriptions and assumptions required by the heuristic approach.

2) An accurate constraints definition increased the domain size of some variables. Actually, it became evident that in many cases the number of alternatives available to the operator before the installation of the 5 Mill Scheduler was limited by the need to manage the complexity of constraints, rather than by the structure and dynamics of the manufacturing process. The Scheduling Engine, enforcing the actual material and resource constraints, was able to generate a larger variety of scheduling solutions, without deteriorating the quality of the schedules.

3) The variable and value ordering heuristics proved to be very effective. In almost 60% of the cases, the Scheduling Engine is capable of finding a feasible solution without backtracking. In 20% of the cases, constraints relaxation and the Resource Permutation method are able to improve the schedule over the value ordering heuristics within the time limit allowed.

During the initial phase of operation, two minor changes in the design of the application became necessary.

The first change was made to support the schedule editing capability. Although schedule editing capabilities were provided by the 5 Mill Scheduler, an existing interface not interacting with the scheduler was preferred to minimize changes to the operator's environment. Schedule changes are therefore applied outside the control of the Scheduling Engine, at times violating capacity constraints. Constraints violations are detected during the rescheduling phase. In this phase, the imperative nature of the schedule changes prevents any explorations of alternative solutions, causing irrecoverable dead-ends in the search process. While some manual schedule changes are consequences of errors, other are forced by the occurrence of abnormal situations. To overcome the problem, an "override" attribute was added to the constraint object definition. When an override action procedure is defined and the Scheduling Engine encounters a constraint violation caused by the imperative decision of an operator, a warning message is issued, and the confirmation of the intention to violate the constraint is requested. Supporting this functionality required minor

changes to the Scheduling Engine and to the Process Model.

The second change was applied to provide greater flexibility and autonomy in the selection of resources by the operators. "Constraints Filters", interacting with the action procedures of some constraints, enabled the operator to graphically modify the resource selection criteria based on product attributes like grade and temperature. The Constraints Filters enabled the operators to dynamically change the selection criteria, without requiring any changes to the model.

The 5 Mill Scheduler has been developed on top of Gensym Corporation's G2® and is operating on a VAX™ 3800. It served as a prototype for DSP™, a G2-based dynamic scheduling product currently under development at Gensym.

References.

[1] Chiodini, V. "SCORE: An Integrated System for Dynamic Scheduling and Control of High-Volume Manufacturing". Proc. 5th IEEE Conf. on Artificial Intelligence Application (Miami), March 1989, pages 271-278.

[2] Fox, M. S., and Smith, S.F., "ISIS: A Knowledge-Based System for Factory Scheduling", Expert Systems, 1 (1), July 1984, pages 25-49.

[3] Fox, M. S., Sadeh, N., Baykan, C., "Constrained Heuristic Search" Proceeding of the Eleventh International Joint Conference on Artificial Intelligence, 1989, pages 309-315.

[4] Ow P.S., Smith, S. F., and Thiriez, A., "Reactive Plan Revision", Proceedings AAAI-88, St. Paul, MN, 1988.

[5] Sadeh, N., Fox, M. S., "Variable and Value Ordering Heuristics for Activity-based Job-shop Scheduling". Proceeding of the Fourth International Conference on Expert Systems in Production and Operations Management, Hilton Head Island, S.C., March 1990, pages 134-144.

[6] Sadeh, N., "Look-ahead Techniques for Micro-opportunistic Job Shop Scheduling". PhD thesis, School of Computer Science, Carnegie Mellon University, March 1991.

[7] Sathi, A., Fox, M. S., Greenberg, M., "Representation of Activity Knowledge for Project Management." IEEE Transaction on Pattern Analysis and Machine Intelligence, September, 1985.

[8] Smith, S. F., Ow, P. S., Muscettola, N., Potvin, J. V., and Matthys, D., "An Integrated Framework for Generating and Revising Factory Schedules", Journal of the Operational Research Society,41(6), June 1990.

[9] Zweben, M., Davis, E., and Deale, M., "Iterative repair for Scheduling and Rescheduling". Technical Report, NASA Ames Research Center, MS 244-17, Moffett Field, CA 94035, 1991.

††† VAX is a trade-mark of Digital Equipment Corporation.

G2 is a registered trade-mark of Gensym Corporation.

DSP is a trade-mark of Gensym Corporation.

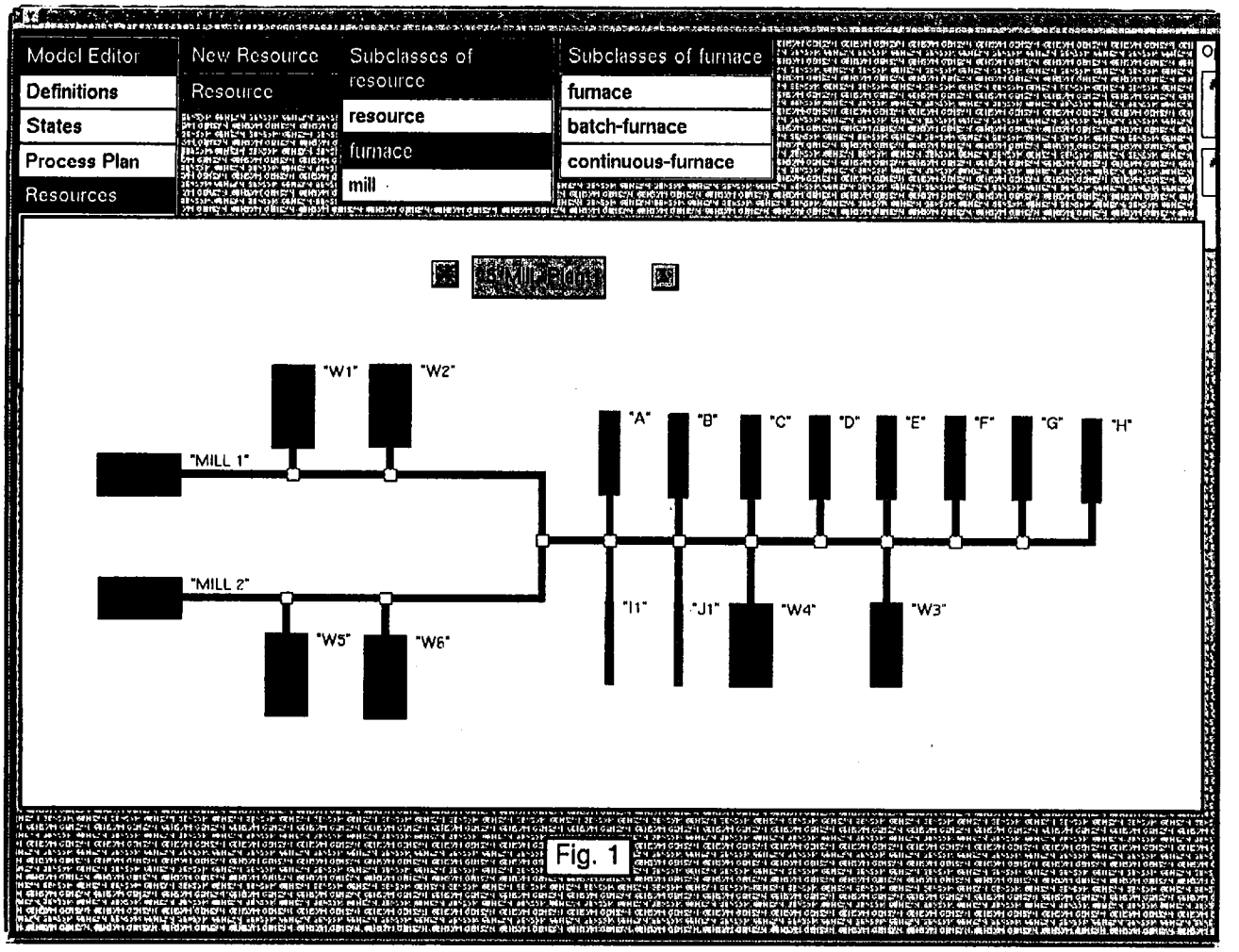
Model Editor
Definitions
States
Process Plan
Resources
New Resource
Resource
Subclasses of resource
resource
furnace
mill
Subclasses of furnace
furnace
batch-furnace
continuous-furnace
"W1"
"W2"
"MILL 1"
"MILL 2"
"W5"
"W6"
"A"
"B"
"C"
"D"
"E"
"F"
"G"
"H"
"I1"
"J1"
"W4"
"W3"

Fig. 1

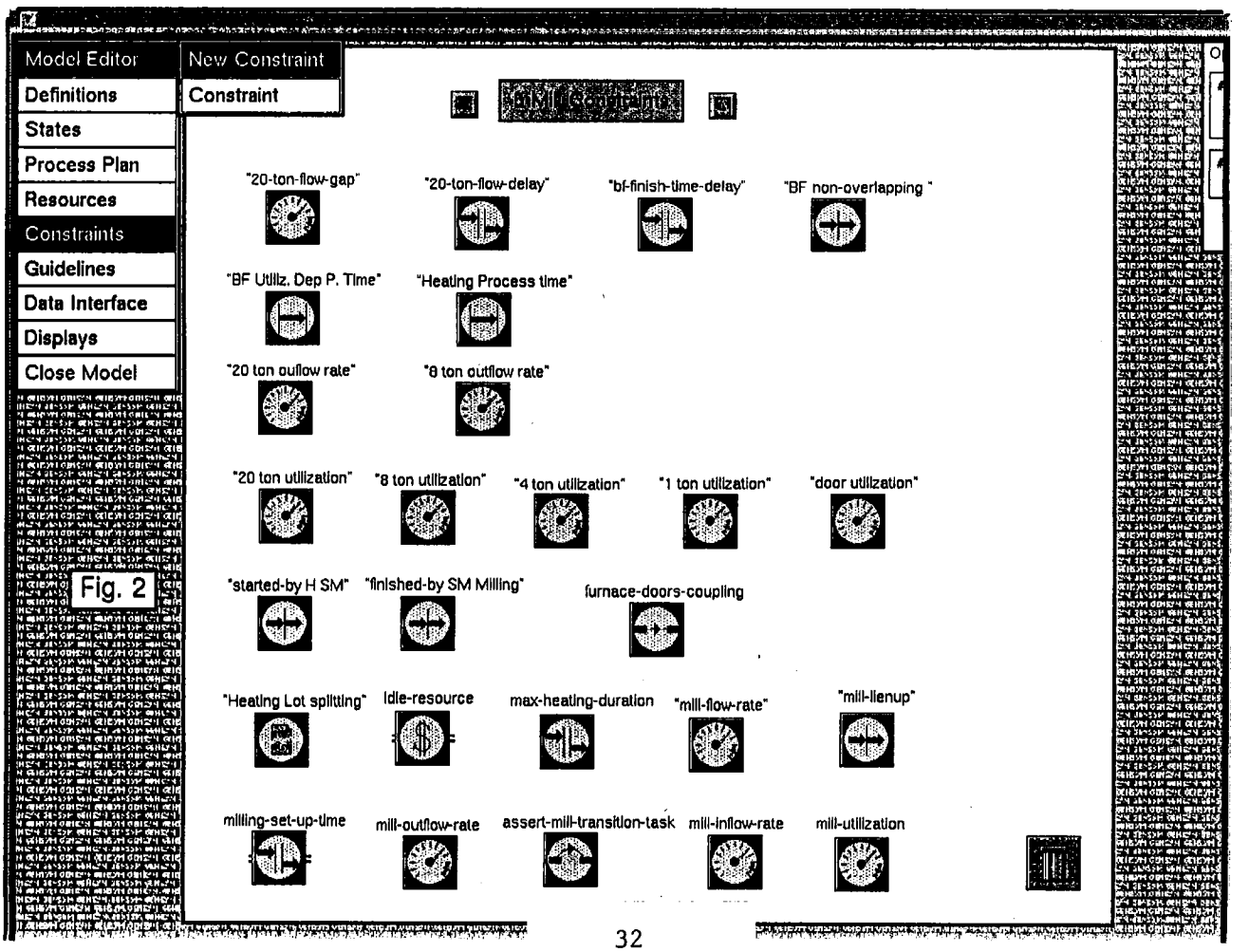
Model Editor
Definitions
States
Process Plan
Resources
Constraints
Guidelines
Data Interface
Displays
Close Model
New Constraint
Constraint
"20-ton-flow-gap"
"20-ton-flow-delay"
"bf-finish-time-delay"
"BF non-overlapping "
"BF Utiliz. Dep P. Time"
"Heating Process time"
"20 ton outflow rate"
"8 ton outflow rate"
"20 ton utilization"
"8 ton utilization"
"4 ton utilization"
"1 ton utilization"
"door utilization"
"started-by H SM"
"finished-by SM Milling"
furnace-doors-coupling
"Heating Lot splitting"
idle-resource
max-heating-duration
"mill-flow-rate"
"mill-lienup"
milling-set-up-time
mill-outflow-rate
assert-mill-transition-task
mill-inflow-rate
mill-utilization

Fig. 2