International Journal of Production Economics, 36, pp. 221-227, 1994.

Development of a Rule-Based Finite Capacity Scheduling System

This paper discusses the use of the finite capacity planning model as a basis for a job-shop scheduling system. The suitability of this approach for constructing short term schedules with very short lead times is examined. This paper introduces a rule-based scheduling system whose operation, unlike previous monolithic schedulers, is based around distinct though interlinked processes. Scheduling herein is defined as the selection of a set of orders for manufacture, and the allocation of processing time to each on an acyclic network of processing resources. The objectives are, firstly, compliance with all customer deadlines and, secondly, the efficient utilisation of available machine time and prevailing machine setups.

Key words : Finite capacity scheduling, rule-based systems.

INTRODUCTION

Production scheduling can be seen as forming a link between the areas of Computer Aided Design (CAD) and Computer Aided Manufacture (CAM), whose objectives are the implementation of all managerial directives relating to the manufacturing domain. Taking a more algorithmic approach, scheduling may be thought of as the problem of constructing a master production schedule (MPS) from a set of pending orders and a set of resource allocations. Early scheduling systems used priority dispatch rules[1], forming a deterministic method whereby orders are not scheduled until certain constraints have been fulfilled. Later heuristic systems (e.g. XCON & ISIS[2]) were intended more for medium or long term planning, rather than short term scheduling. Much work has also gone into the examination of Flexible Manufacturing Systems (FMS) - domains with multiple machines connected to an automated transportation system. This, however, only covers a fraction of manufacturing domains.

This paper describes the development of a scheduling system for use in a manufacturing domain which is viewed as a directed graph, with nodes representing processing machines and edges between nodes representing the movement of material between these machines. Each order has a predetermined "route" which is a list of nodes to be visited by that order, the object of the scheduling system being to produce a schedule at very short notice (within 5 minutes) to suit the current factory status. The most important feature of this system is that orders which fall into a category called "urgent orders" *must* be manufactured promptly. Other aims are the efficient use of machine setups and maximum utilisation of the available resources.

SCHEDULING

Many different systems have been developed to solve a variety of scheduling problems. The solution described in this paper differs from existing systems in many ways, and most of these differences are due to the need for reactive[3] scheduling.

The much referenced XCON, which proved the applicability of expert system techniques to solving real world problems, would not prove a good model on which to base this project. XCON configured the

PDP, and later the VAX, range of computers. However, it was part of a larger project with a natural language front end, XCALIBUR[4], and an "expert" sales assistant, XSEL. Each of these components were fully integrated, XCALIBUR coordinating user interaction with XCON and XSEL. The dimensions of the XCON project were far greater than the limited resources available to the project at hand. Furthermore, the continuing development of XCON was a trait thought undesirable in this project.

As a further example, the excellent ISIS[5] scheduler utilised a complex order storage format (including alternative delivery dates, etc.) which enabled extremely efficient schedules to be produced. This complex representation format, however, is not required for the domain under investigation because of the need for reactive scheduling. Also, the computing power needed to drive ISIS was not available to the company in question.

The main features required of our scheduler are that it creates schedules based on Finite Capacity Planning (FCP[6]) techniques, that it performs the scheduling task in as quick a time a possible, and that it provide over-ride facilities to the production controller (the reason for this will be explained later). A clean and simple user interface was of central importance to the success of the project overall.

THE PROBLEM

The scheduling problem arose in a company specialising in the manufacture of corrugated cases (cardboard boxes). A diverse range of products occurs, with variations among box styles, sizes, composite materials and order quantities. Manufacturing involves two specific operations: *corrugation* and *conversion*. The first of these is the process of forming corrugations on one sheet of paper before binding inner and outer walls to this layer - all these operations being carried out by a single machine. The boards are produced to each customer's needs, with the required paper types and sheet dimensions being used. The second operation involves the conversion of these cardboard sheets into the finished cardboard boxes, after being cut and folded to the required dimensions and possibly having the customer company's logo printed on the outside.

The scheduling process begins with the selection of a "board grade", or type of cardboard for production. Orders are grouped according to grade, and the grade is chosen for production according to the following heuristics (listed in order of priority):

- the most urgent order.
- the most due orders.
- avail of current corrugator setups.
- maintain steady flow of material through machines.
- grade which best avails of converter machine setups.

These constraints describe the grade selection process, though the system of catering for conflicts is not explicitly stated. The corrugator is loaded with the required paper types and the *flute* type is set, dictating the density of corrugations of the middle layer of paper of the cardboard. Because there is only one corrugator machine, orders are allocated processing time in sequential order and, to avail of the machine setup, all orders

utilising this type of board are scheduled together. Under the condition of an extreme accumulation of orders, all similar orders may not be scheduled together in an effort to meet the "schedule urgent orders" constraint.

There currently exists an optimisation system on the factory floor which minimises materials wastage on this machine. Occasionally, an order may be rejected from a production plan by the production controller on the grounds that its inclusion adversely affects materials wastage (i.e. sufficient other orders involving the same board type do not exist). Such rejecting of orders is a factor which had some effect on solution design, whereby orders included in an otherwise optimal schedule may subsequently be removed from that schedule.

Once corrugation is complete, resource allocations must be found on the *converters* for these orders. Conversion of an order generally involves finding a sequence of free processing time on several machines. Each of the conversion machines performs an operation on the cardboard sheets such as cutting, folding, printing, etc.; the operations required are determined by the customer's requirements and are specified as part of the details of each order. Conversion scheduling takes the list of orders scheduled on the corrugator and, for each order in turn, allocates time to each operation on the rest of that order's route. If sufficient time cannot be found for every operation on an order's route, then this order cannot be scheduled. Because urgent orders are given priority, if processing time is available, it is allocated to them first.

MANUFACTURING DOMAIN REPRESENTATION

The structure of the shop floor is shown in schematic outline in FIG-1. The production domain can be considered as a network of processing machines, with orders always starting with machine 0 (*corrugator*) then undergoing further processing or going directly to dispatch. Each order follows a predetermined path on this diagram corresponding to an order's *route* parameter. Machine 0 produces a continuous stream of cardboard from large rolls of paper, the paper type being dictated on the order. All orders which are to be manufactured from this paper type are scheduled together. This cardboard stream is chopped into the sizes according to customer requirements. Occasionally these plain boards are sent to dispatch (machine 6), but normally undergo conversion into cardboard boxes. The particular conversion machines (machines 1…5 in FIG-1) used by an order depend on the final form of box required by the customer. Conversion machines often require difficult setups (up to 30 minutes) for different box styles; thus, orders requiring similar setups are scheduled together when possible. When manufacturing is complete, the boxes are dispatched (machine 6) to the customer.

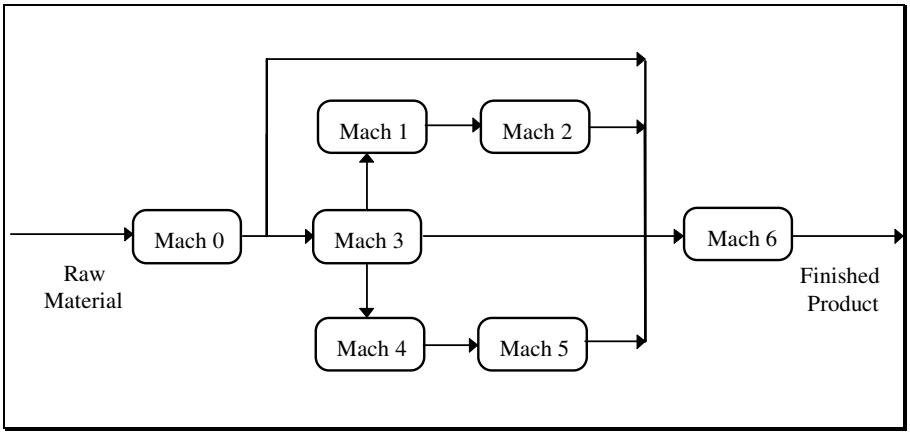

FIG-1: Shop Floor Configuration

THE OBJECTIVES

From the company's perspective, an optimal scheduling system should meet the following considerations:

- *Due Date requirements:* All customer due dates must be met; this includes urgent orders (orders which have a very short lead time before production).
- *Group Orders:* Orders of a similar grade should be manufactured together, thus minimising the number of setup changes required.
- *Machine Utilisation:* To maximise return from resources, all machines should be kept operating whenever possible.
- *Minimise Work In Progress:* Minimise the value of Work in Progress by ensuring a steady flow of goods through the processing area.
- *Machine Setups:* Orders requiring similar machine setups on the converters are scheduled together, minimising setup times.

These requirements frequently conflict with one another, and the way in which compromises are made depends on the company's operational policies.

DATA REPRESENTATION

The scheduling problem can be described by three separate sets of data: *order* data, describing pending orders; *resource* data, detailing the free and allocated processing time; and *priority* data, defining each order's priority. In addition, the priority ranking associated with each order will have a critical bearing on the outcome of scheduling. Resource data is maintained under the following format:

Resource(Machine-id, Time interval, Activity)

This allows the specification of any machine activity, from the processing of an order *(Activity = Order-id)*, to maintenance, or free time (available to orders). This is used in a manner similar to a Gantt chart as discussed by Rowe[7] (FIG-2), with "-" and "•" representing free time and maintenance time, and numbers representing the *order-id* of orders allocated processing time. The scheduling-relevant data maintained on orders is as follows:

Order(Order-id, Quantity, Type, Resources, Size, Due-date)

(In fact, this is a simplification, as some of these parameters are composite data elements). This information describes all information needed on an order. Further information is needed before the nature of the manufacturing environment is properly understood. To increase factory efficiency, orders requiring similar machine setups are grouped together, and each group is maintained in the global database. Instead of scheduling individual orders, *order groups* are scheduled to increase machine efficiency. Of course, this has to be done within the boundaries of resource availability and customer satisfaction.

More information in the form of priority data is maintained on orders, with three categories of priority available: high, medium, and suspended. The latter is, for reasons of efficiency, implemented by moving the order from the *order database* to a *suspension database*. This priority information is then used, in conjunction with a point allocation system, to determine the order-group which should be scheduled first.

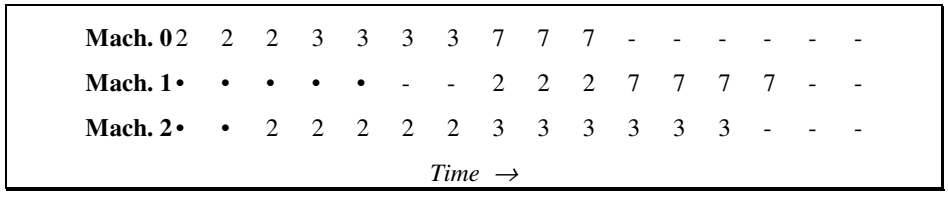

FIG-2: Machine Allocations

THE SOLUTION

Prior to development of the new system, schedules were compiled by a small team of domain experts who used various rules of thumb to build up a schedule which impugned as few of the objectives as possible. The new system is based on this expertise which, when codified as rules, performs the same task - but more efficiently. Due to the large throughput of orders at the plant, scheduling is based on the Finite Capacity Planning (FCP) methodology. This takes the available processing time as the main consideration during scheduling, allocating orders to these times in a sequence dictated by a ranking system.

FCP can be thought of as a sequence of operations (FIG- 3) which combine to result in the construction of a suitable schedule. The operations are:

- Define the relative priority of the orders, usually requiring orders to be placed in priority classes such as *urgent*, *medium* priority and *suspended*.

- Perform Rough Cut Capacity Planning[8] (RCCP) which selects a sequence of orders to be considered by the Resource Allocation Routine (RAR). This sequence is a list of orders which would constitute the optimal schedule, catering for the maximum number of urgent orders and utilising resource setups optimally. Sufficient orders must be selected to ensure that all resources have an adequate loading value.
- The Resource Allocation Routine (RAR) uses the sequence of orders selected by the RCCP and, in turn, allocates sufficient resource time to each order - and to each operation required by that order. The RAR then creates the Master Production Schedule (MPS) which is used on the shop floor. If sufficient time cannot be found for any operation of any order in the group, then no resource allocations are made. Instead, the offending order is extracted from the group and the *sequence* recalculated, before scheduling resumes with the order group of highest priority (which may or may not be a different group).

The two databases, *order* and *resource*, are used to calculate other scheduling parameters, such as the amount of processing time required by each order on each machine. Values are calculated only once, speeding up the actual scheduling process. All calculated values are then stored in the database for later retrieval. These values are recalculated after new orders are added to the order database.

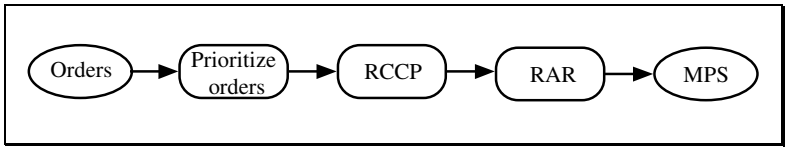

FIG-3: Finite Capacity Planning Operations

FINITE CAPACITY PLANING AND RULES

A prototype scheduling system was created, and written in Prolog (for a discussion on the use of Prolog[9] for implementing Expert Systems, see[10]). An example of a scheduling rule in Prolog (from the very early stages of RCCP) is:

```
update_priority_database :-
        order (No, [Machine|Route], Quantity, Due_date, _),        (1)
        due_order (Due_date),                                      (2)
        assert (medium_priority(No)),                              (3)
        fail.                                                      (4)
```

This Prolog rule may be interpreted in the following fashion. The order database (see FIG - 4) is examined and the variables *No...Due_date* are instantiated to the values of the first entry therein. Line 2 passes the *Due_date* to a predicate which, if the order is overdue, will succeed - and control will pass to line 3. However,

if line 2 failed, then either the order database would be reexamined and the variables reinstantiated to the values of the second order, or execution of this predicate would cease. When control passes to line 3, the number of the order is added to the database of *medium_priority* orders. Line 4 causes control to be passed back to line 1, and testing of the next order is performed.

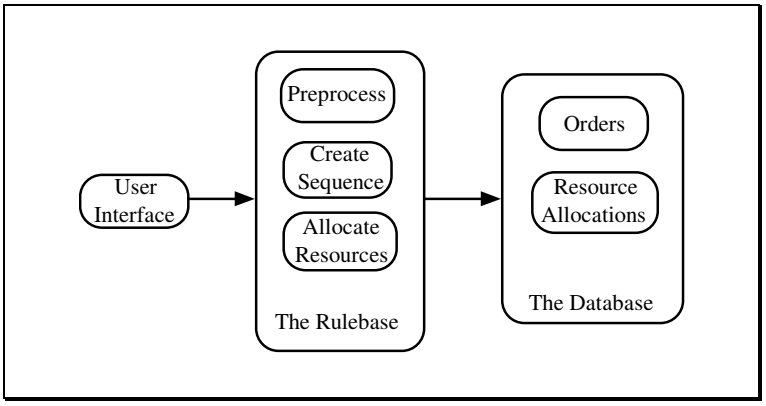

FIG-4: System Architecture

The resource allocation routine consists of two components - the first checks for resource availability and the second formally allocates the processing time to that order.

```
find_free_time ( _, 101, _, _).
find_free_time (Wo, [Machine\Route]) :-
        operation_duration (Wo, Machine,Time),
        job (Machine, Start, Finish, free, _),
        Finish >= Start + Time,
        find_free_time (Wo, Route),
        update_job (Wo, Mach, Start, Finish, Time),
        !.
```

The *find_free_time* rule is given an order number for which sufficent free time is sought on all machines along its route. This is done by finding the machine duration required on the first machine (which has been calculated previously and stored). Then a sufficient period of free time is sought on that machine. This process is continued until all operations have been found to have space on the existing schedule. This order (or group of orders) is then included in the schedule, producing a new more complete schedule. The rule terminates when either all operations along an order's route have been scheduled, or when it is discovered that sufficient time for one operation cannot be found. The latter condition results in no orders of that type being scheduled, and the production controller being notified of the reason for this.

The operation of the rulebase is controlled by the user through a special menu- type user interface. This provides the user with a list of options from which the required operation is chosen. This can be seen as another layer atop the Rule-base, as in FIG-4. This menu provides all the facilities required for the scheduling

expert to examine all data held in the database - thus enabling the user to manually construct schedules with this system. Functions for reactive plant monitoring were also provided, though they are not discussed here.

INSTALLATION

The scheduler was initially run in parallel with the existing semi-automated methods, but the production controller had the ability to override any decisions made by either the existing optimiser or new scheduler. After a few initial problems which were solved by minor alterations to the rulebase, the only orders eliminated from a production plan were those incorrectly inserted by the existing optimiser. These were easily catered for by the new scheduler.

An initial post-installation study revealed a significant decrease in the value of Work In Progress, greater than 5% - measured by the quantity of board on the shop floor at one time. This has been attributed to the previous scheduling methods effectively ignoring the scheduling of conversion machines, so that a significant improvement was expected. There was a similar, though smaller, increase in machine utilisation - measured as a percentage of busy time over machine idle time. These values are the result of a preliminary study taken over one working week. Final figures are expected to be broadly similar, but will have to be taken over a longer time period - this is due to the seasonal nature of orders in the problem domain.

CONCLUSION

This paper dealt with the development of a short-term scheduling system for a complex network of manufacturing machines. The objective of this system was to meet all customer deadlines while utilising machine setups as fully as possible. A rule-based system was written in Prolog which employed the Finite Capacity Planning methodology, this proving ideal for short term scheduling within a complex manufacturing environment. Initial results are very encouraging, with still better results expected from the integration of the existing optimisation system and the scheduler.

REFERENCES:

[1] C. Moodie, J. Walter, “A Scheduling Procedure for Parallel Processors in the Paper Industry”, Technical Papers - 21st Institute Conference and Convention AIIE, 339-345 (1970).

[2] S. Smith, M. Fox, B. Allen, G. Strom, F. Wimberly, “ISIS : A Constraint Directed Reasoning Approach to job-shop Scheduling”, Proc. Trends & Applications Automating Intelligent Behaviour”, 76-81, (1983).

[3] K. Barber, D. Osterfield, “Expert simulation for On-line Scheduling”, Proc. Winter Simulation Conf., 930-5, (1989).

[4] J. Carbonell, “The XCALIBUR Project”, Proc. 8th Int. Conf. on A. I. (1983).

[5] M. S. Fox, “Constraint-directed search : a case study in of job shop scheduling”, PhD Thesis Carnegie-Mellon University, (1983).

[6] B. Guise, J. Lischefska, “Finite Scheduling Software Yields Multiple Benefits”, Control Syst., 39, 3, 30-2, (1992).

[7] A.J. Rowe, J.R. Jackson, “Research Problems in Production Routing and Scheduling”, Jour. Industrial Engnr., 7, 116-121, (1956).

[8] S. McCarthy, K. Barber, “Medium to Short Term Finite Capacity Scheduling : A Planning Methodoloy for Capacity Constrained Workshops”, Engineering Costs and Production Economics, 19, 189-199, (1990).

[9] Clocksin, Mellish, “Programming in Prolog”, 2nd Ed., Springer-Verlag, (1984).

[10] G. Rossi, “Uses of Prolog in the Implementation of Expert Systems”, New Generation Computing (Japan), 4, 3, 321-329, (1986).