

Short Term Liquid Metal Scheduling

Mike Pegman, Vine Solutions Ltd, mp@vinesolutions.co.uk

This paper discusses a constraint based application for **scheduling and reactive re-scheduling** of a **liquid metal manufacturing process**. The scheduler was designed, and a demonstration system developed, during the later half of 1996, by Vine Solutions Ltd for Broner Group Ltd, a premier vendor of scheduling systems to the global metals manufacturing industry. **The demonstrator system had successful early trials at a customer site.** The application is to be developed further before **installation in an on-line control role.**

A **batch of liquid metal** is delivered to a series of processes, the last of which is casting, in which slabs of metal are created in a continuous process. A primary objective is to **ensure that the casting machine keeps running.** The scheduler ensures that liquid metal is available whenever the casting machine requires. To achieve this, the up-stream processes must be scheduled subject to constraints including the following.

- **valid product routes** A range of alternative routes for each product are modelled, leading to a resource allocation process.
- **machine process capabilities and processing time** The characteristics of different machines are represented and taken into account as the resource allocation and scheduling process occurs.
- **balanced material flow through the process as a whole** As well as the objective to keep the casters running, the system works to manage the overall liquid metal flow through the plant.
- **transport and waiting times between machines** The movement of a container of liquid metal between machines is subject to timing constraints.
- **maximum total process time** The process as a whole is subject to an overall limit (due to cooling of the metal).
- **automatic insertion of a re-heat operation if necessary** If necessary the total process time can be extended automatically by the scheduler inserting a re-heat task into the product routing.
- **machine down-times** Machine maintenance is frequent and a variety of types of down time are modelled.
- **a complex set-up time - based on a sequence of batches through a machine** Inter-batch set-up time can be a function of the machine and also the previous *sequence of batches* processed on that machine.

The context of operation is that of **on-line reactive scheduling** of the manufacturing process over a period several hours from the current time. The system produces schedules based on a desired sequence of caster operations. Each caster operation identifies a works order and hence possible routings and time windows for the up-stream processes. The **system performs resource allocation**, choosing a route for each works order **and schedules the operations** on each machine.

Due to **variability in process times**, or unforeseen problems in the plant, the duration of each operation, or even the machine on which an operation is undertaken, may be different from the plan. In addition, **the user may over-ride** the automatically generated route or timings, or **limit the variability** of timings of operations, and of the machines allowed for a particular batch of material. The re-scheduler takes such information into account, imposing these as constraints on the output schedule. **As time advances**, it also **automatically considers those works orders which are committed** to machines allocated in a previous run of the scheduler.

The current version of the system is running in an off-line mode, being tested at the customer site, prior to installation into an on-line environment in the liquid metal processing plant. The system employs ILOG Solver and Scheduler to model all aspects of the plant, routings and works order operations. A visualisation model and interactive user interface is constructed primarily around the ILOG Views Gantt Chart.

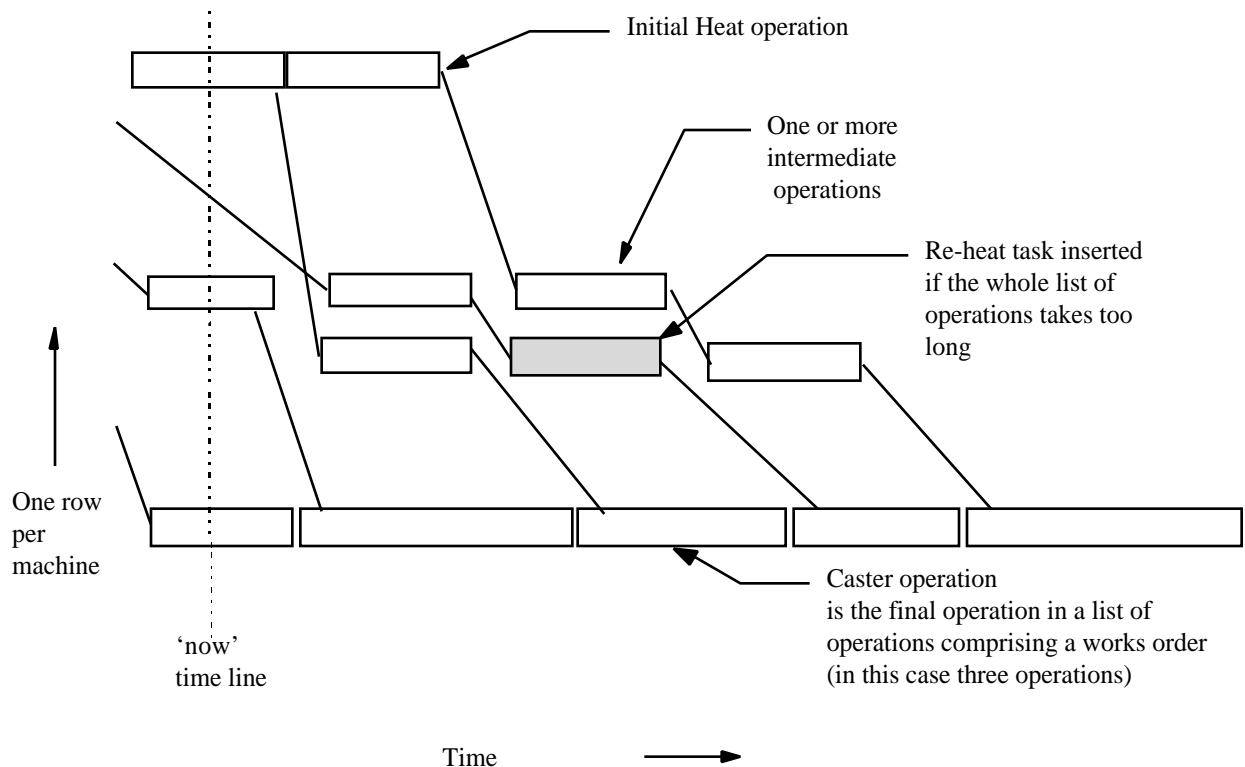
The current paper expands on the summary given above. It discusses the application domain in more depth, gives information on each of the listed constraints, and discusses some aspects of the implementation of the scheduling model and the search strategy. A generally applicable three layer top level object model and application control loop are also presented.

This paper discusses a constraint based application for scheduling and reactive re-scheduling of a **liquid metal manufacturing process**. The scheduler was designed, and a prototype developed, during the later half of 1996, by Vine Solutions Ltd for *Broner Group Ltd*, a premier vendor of scheduling systems to the global metals manufacturing industry.

Liquid Metal Manufacture

The overall requirement is to manufacture slabs of (solid) metal which meet certain product specifications. These specifications can be produced by passing the material through a sequence of operations, comprising a manufacturing route.. The manufacturing process is in part batch mode and in part continuous. A batch of liquid metal is delivered to a series of operations. The first of these is an initial heating operation. All routes include this operation. The last operation is casting, in which slabs of (solid) metal are created in a continuous process.

A works order is a request for a batch of a specified product to be produced on a specific caster at a certain point in a sequence of works orders. The main input data to the liquid metal scheduler is a sequence of works orders specified for each caster.



A primary objective is to ensure that the casting machine keeps running. (Stopping and restarting a caster is a time consuming process.) The planned schedule must ensure that liquid metal is available whenever the casting machine requires. To achieve this, the up-stream processes must be scheduled subject to various constraints. The scheduler can also vary, within certain limits, the duration of casting of a works order.

It is possible that the same specification of a product can be produced from different routes (i.e. by processing material on different machines). For a given works order there are several possible routes along which the batch of material may pass in order to fulfil the requirements of that product. For instance there may be two alternative routes: <Initial Heat> <operation O><Caster> OR <Initial Heat> <operation P><operation Q><Caster>. Note that these two routes have different lengths, so depending upon the choice of route made by the scheduler, the visualisation on the interface would have either one or two intermediate operations.

A given operation may be able to be performed on one of a group of machines. For instance <operation O> may be performed on machines M1, M3; whilst <operation P> may be performed on machines M1, M2.

Short Term Scheduling and Reactive Re-scheduling

The system discussed here is a scheduler for the liquid metal manufacturing process. Decisions concerning both manufacturing route (i.e. resource allocation) and timing of operations (i.e. scheduling) must be performed.

The context of operation is that of on-line reactive scheduling of the manufacturing process over a period several hours from the current time. Due to variability in process times, or unforeseen problems in the plant, the duration of each operation, or even the machine on which an operation is undertaken, the actual schedule may differ from the plan.

The scheduler is required to produce a plan for several hours into the future and to allow re-scheduling of appropriate parts of a schedule as the dynamics of the real process deviate from the planned schedule. The final system will have an ON-line data feed. Data from the real process will be fed into the scheduler by lower-level systems. For example, this information will effect the start, duration or end time of operations, and the duration or existence of machine down times.

In a similar fashion, input from the user via direct manipulation of the gantt chart visualisation of the schedule may be used to limit the range of variability of possible routes or machines, the duration or start and end times of operations, or a wide range of other factors. Certain aspects of a given plan may be locked, leaving others to be included in the scope of variability considered by the scheduler.

As well as these two data feeds the scheduler considers the advance of the current time. A works order of which at least one operation overlaps or precedes the current time is said to be 'in-process'. Clearly it cannot be reallocated or rescheduled by the system. Other constraints, generally concerning routing options, are also automatically imposed on such works orders. It is highly undesirable that the scheduler should change the plan for a certain period of time *after* the current time, since instructions to plant operations will already have been issued.

Whether this dynamic information comes from a lower level plant monitoring system, or from the graphical user interface, or is a result of the advance of the current time, the scheduler must consider them all in an incremental scheduling process.

An Overview of the Current System

The current version of the system is running in an OFF-line mode, being tested at the customer site, prior to installation into an on-line environment in the liquid metal processing plant. Those data required by the ON-line version of the scheduler are represented in the model and in the user interface so that the full model can be tested. It is expected that the model will require minimal change when installed in an on-line form - some of the data feed will then come from the lower level monitoring systems, not from the user.

The system employs ILOG Solver and Scheduler to model all aspects of the plant, routings and works order operations, as well as to implement custom constraints and the search strategy. A visualisation model and interactive user interface is constructed primarily around the ILOG Views Gantt Chart, in which each operation a specific machine is represented on a row, in the manner of the diagram above.

The system produces schedules based on a desired sequence of caster operations. Each caster operation identifies a works order and hence possible routings and time windows for the up-stream processes. The system performs resource allocation, choosing a route for each works order and schedules the operations on each machine. In the process of resource allocation and scheduling a set-up constraint, based on sequences of operations on specific machines, ensures that complex application specific set-ups are planned appropriately.

The user can specify the variety of constraints described above, several types of down time requirement can be specified, and the system takes into account the advance of 'current time', automatically handling 'in-process' works orders specially. The re-scheduler takes such information into account, imposing these as constraints on the output schedule.

The performance of the system is within the acceptable limits for the application, producing schedules in the usual conditions in under a minute. In complex or over-constrained cases run times are longer, and after a user determined time-out a separate fall-back scheduler is run to produce a viable very short term schedule (see Discussion of Search Strategy, below).

Discussion of Constraints

This section presents an overview of the types of constraint represented in the current system. For some of these, information is given on the modelling constructs employed in the implementation. The bulk of constraints were represented in constructs provided by the Ilog tools. Custom constraints were implemented

Caster Operation is Continuous

A primary objective is to ensure that the casting machine keeps running. The scheduler is given a sequence of works orders on each caster. It ensures that liquid metal is available whenever the casting machine requires. To achieve this, the up-stream processes must be scheduled such that all of their required operations are completed, including transportation timings between operations, before the operation on the caster. In addition the scheduler can vary the duration of the caster operations within certain limits which facilitates adjustments to the schedule.

Although the overall objective is to keep the caster running, it is sometimes necessary to shut the machine down. The user can specify a down time, either at a fixed time or more likely within a range of time. In this case the scheduler ensures that the operation is continuous before and after the down time. (See below for more information on machine down times.)

In the simplest case, where there is not a downtime with a time range, the implementation of this constraint is a straightforward. Since the input has a sequence of operations on each caster the system would simply apply the temporal constraint 'startsAtEnd' down the sequence of operations. However, in the case of a down time which may occur anywhere within a range, this down time may 'interrupt' the sequence. In other words, there is a requirement to *decide* where the down time occurs. 'startsAfterEnd' constraints are applied and the search code begins with a ranking to determine the order of the casting operations *and* the down time(s). After the ranking is determined, it is *then* possible to ensure the operations are continuous, posting the equivalent of 'startsAtEnd'.

Valid Product Routes

A range of alternative routes for each product are modelled. Route is a sequence of operations, beginning with the initial heating operation and ending with the casting operation. Each operation may have a choice of machine which could undertake that operation. An object representing a route for each works order is constructed, containing a range of alternative machines for each step in a given manufacturing route.

The user may remove one or more routes from the list of routes for a given works order. The user may lock a given operation to a specific machine (independently of the timing of that operation). These modifications are represented in the route object for that works order. (More details of the object model are given below in the section on Ensuring a Declarative Model.)

Machine Capabilities

The characteristics of different machines are represented and taken into account as the resource allocation and scheduling process occurs.

processing time Depending upon which machine processes an operation, the operation have a different duration. This duration may also be dependent upon the product being produced. This is implemented as a demon on the resource allocation decision.

per works order limitations It is possible to remove one or more machines from the list of possible machines for a given works order. Such machines will not then be candidates during the resource allocation process (for implementation see Ensuring a Declarative Model).

machine down times the user can specify various types of down time.

Balanced Material Flow

As well as the objective to keep the casters running, the system works to manage the overall liquid metal flow through the plant.

This is achieved by a user determined parameter controlling the rate at which the initial heating operations are required to occur. This essentially introduces a 'push' element into the scheduling process. Material is forced to enter the schedule at this rate, and it must be processed by the intermediate operations to meet the caster sequence. In implementation terms this 'push' is implemented as the calculation of the duration of the initial heating operations, and, if necessary to further decrease the 'push', a delay between initial heating operations.

The primary input, i.e. the sequence on the casters, provides the basis for a 'pull' element. The main variation which the scheduler can manipulate on the caster operations is their duration. Decreasing the duration of several caster tasks leads to the requirement for more material to be processed by the up-stream operations, including the initial heating operation. Correspondingly, increasing the duration decreases the number of up-stream operations, including initial heating operations.

The plant as a whole has no long term storage potential, so balancing the push and pull elements of the scheduling problem in the search leads (if a solution exists) to a schedule in which material flow has been controlled.

Inter-machine Timings

Transport times

The movement of a container of liquid metal between machines is subject to timing constraints. It takes a certain minimum time to move the container. This is a function of which machines are selected in the resource allocation process.

It is represented in the model as a constraint that states '*following operation*' starts after '*preceding operation*' by a minimum transportation delay. The transportation delay is not known until a) the resource allocation process is undertaken, and b) the operations in question have been determined to have a non-zero duration. For both operations the model posts a demon on the choice of machine and on the duration variable, and when both machines have been decided and both have non-zero duration the demon posts the above constraint.

Waiting times

It is necessary for some machines to ensure that there is a certain minimum and maximum waiting time imposed upon the batch of liquid metal after transport and *before* the operation begins on a machine. This is a function of the machine at which the batch is waiting.

Waiting times are represented by an additional `IlcIntervalActivity` which shares its end variable with the start variable of the main activity. Together these *two* `Ilog` activity objects represent the core of the scheduler class which represents one operation. The wait activity creates the delay, but since it does not require any machine resource, does not utilise the machine. Of course the main activity requires one of a list of possible machines which can support that operation.

Re-heat Operations

After the initial heating operation, the process as a whole is subject to an overall limit due to cooling of the metal. If necessary an additional re-heating operation may be introduced into the route.

For instance the two alternative routes given above must more correctly be specified in terms of possible reheat operations thus:

`<Initial Heat> <Reheat>.<operation O> <Reheat>.<Caster> or`
`<Initial Heat> <Reheat>.<operation P> <Reheat>.<operation Q> <Reheat>.<Caster>.`

At most one of these re-heating operations can take place.

This is modelled in terms of extra activities in the route, all may have duration of zero. If the route timing exceeds the threshold of time for a re-heating operation, their joint duration is constrained to be a non-zero value calculated as a function of the time from initial heating to casting operations. A counting constraint is posted to ensure that only one of the re-heat operations has a non-zero duration.

Machine Down-Times

Machine maintenance is frequent and a variety of types of down time are modelled. Users can specify down times of a machine giving precise figures for the timings. Alternatively a range of times can be specified for duration and / or start and end time. This allows the scheduling of a down time into the schedule, but its positioning and duration are decided by the scheduler.

These down times are implemented in with `IlcIntervalActivity` objects with appropriate bounds on the time variables, and a requirement constraint for the machine. The search code (and the resource constraint propagation mechanism) determine where, within the possible range of times, the down time is scheduled to occur.

Machine Set-up Times

The application requires a complex set-up time based on a sequence of batches through a machine. Inter-batch set-up time can be a function of the machine and also the *sequence of batches* processed on that machine. A set-up task is required at least once in every *n* tasks on a given machine. (This is effectively a special form of down time which is automatically scheduled depending upon the number and sequence of operations on a specific machine.)

The constraint which enforces the set-up time requirement, was implemented as a combination of a new constraint class and a series of demons, and is based upon the following information:

- Previously committed operations on the machine since the last set-up.
- The number of operations actually allocated to the machine.
- The allocation of down-time tasks by the user (which are defined to satisfy the requirements for the sequence based set-up constraint).
- The sequence of operations on the machine.

The set-up tasks are themselves modelled as activities which have a duration ranging from zero to the maximum set-up time. During the creation of the scheduler objects, as many of these are created for each machine as could possibly be required. (This is a calculation based on scanning all operations which *could possibly* use the machine and dividing by n, and is necessary to ensure resource closure.)

The constraint monitors resource allocation decisions to determine the number of set-up operations actually required on the machine (a demon constraints the duration of a set-up task to be an appropriate non-zero value). It also monitors scheduling decisions made in the search code in order a) to enforce the scheduling of a set-up task if a sequence of n-1 operations is detected, b) to verify during scheduling that the whole sequence satisfies the 1 of n requirement.

The constraint keeps a (reversible) list of the sequence of tasks (operations, down times and set-ups) on a machine. It maintains the list during resource allocation and scheduling. This list includes activities which are allocated to the machine (because of resource allocation decisions), activities of zero duration (for instance unused operations in a route) as well as the set-up tasks themselves. The core of the constraint scans this list imposing the sequence requirements.

Ensuring a Declarative Model

It is in general a good design principle to create a declarative model of the problem. All entities are created initially, with constraints which define their nature and the relationships between them. The search code is kept separate from this declarative model. The development of the current system has applied this principle.

Each works order is composed of a list of operations. Note that for any given works order there may be a choice of routes. These routes may have different lengths. The user may have removed route choices or machine choices for a given works order. Also, routes will usually have optional re-heat tasks. Therefore, a works order will usually have a variable number of operations / tasks depending upon which route is chosen and whether a re-heat task is necessary. All this information is maintained in a route object kept by each works order. Of course, the choice of the number and nature of the operations is made by the search code *after* the declarative model has been created.

To ensure that the model is declarative (and to ensure that all Scheduler resources can be 'closed') each works order is created with the maximum number of activities which any route could possibly use. This means that in a given solution one or more of the activities will have a duration of zero. This maximum number of activities are created and stored in a list in the scheduler's works order object.

Each operation requires one of a set of machines which may undertake that operation. Iterating over the list of activities, determining possible operations for each activity, derives a list of the machines which are possibly employed by each activity. For each activity, these machines are gathered into a list which is 'required by' the activity (i.e. an `IlcAlternativeResources` object is created for each activity where a choice of resource allocation exists).

Various constraints are required to ensure that the choice of one machine for a given operation propagates appropriately to the rest of the route. (For example: only one re-heat task is permitted. For example: if a machine is chosen which dictates that a route of length 4 operations is not possible, then the duration of the fourth activity is set to zero.)

For example, and omitting re-heat tasks for simplicity, consider two alternative routes for a works order.

Route A <Initial Heat> <operation O><Caster>
Route B <Initial Heat> <operation P><operation Q><Caster>.

Where:

<operation O> may be performed on machines M1, M3;

<operation P> may be performed on machines M1, M4.

<operation Q> may be performed on machine M2.

<Caster> must be performed on C1

Route A is restated : <Initial Heat> <operation O><null operation><Caster>
Where <null operation> is an activity of zero duration.

A list of activities length 4 is created and the possible machines for each activity is computed as above resulting in the possible machines for each entry in this list as follows (the index into the list is shown as a digit):
0[H1], 1[M1, M3, M4], 2[M2], 3[C1]

and constraints are posted, e.g.
if (M3 is selected from option list at index 1) then (activity at index 2 is null)

In this way the declarative semantics of the model are ensured. The very expensive alternative (of the dynamic creation and destruction of activities during the search) is avoided. Resources can be closed, enabling more propagation to occur in Scheduler's internal constraints.

Note that, as well as the above activities to represent operations / tasks in a works order, the model also creates a calculated maximum number of set-up tasks (calculated from all those activities which *could* use a specified machine) and the specified down time activities.

Discussion of Search Strategy

As is usual in projects representing complex industrial resource allocation and scheduling applications the control of the search process is an essential feature of the successful application.

In this system the constraints are quite strong. That is the following constraints place heavy limitations on the variability of the schedule: the continuous nature of the casting process, the heavy resource demands of the complex set-up constraints, and the fact that all products must pass through the initial heating operation. These 'strong' constraints are useful in controlling search - after all this is an example of the benefit of constraint programming! However much variability remains and this needs to be carefully controlled.

High Level Search Strategy

The sequencing on the caster (potentially including down times) is undertaken as described above to determine the continuous 'output' of the schedule. The duration of the initial heating operations, and possibly delays between initial heating operations is determined by the material flow parameter. (Note however that sequencing on the initial heating machines is *not* determined by sequence on the casting machines.)

As is conventional in combined resource allocation and scheduling problems, the resource allocation component of the problem is undertaken first in the search space. The positioning of the operations in time (i.e. scheduling) is then undertaken. The overall search is in these two stages: search for a resource allocation, search for a schedule.

The number of set-up tasks are dependent upon the number of operations allocated to a specific machine. The set-up task sequencing has a major impact on resource availability. The exact timing of the transport delays and of the reheat tasks are dependent upon machine and upon scheduling decisions. These facts lead to the possibility of there being no solution to the scheduling problem following a given resource allocation. The system thus must ensure that the search for a schedule will backtrack appropriately into the resource allocation stage of the search. Time spent in the scheduling search may be controlled so that, in the event that a schedule cannot be found for a given resource allocation, the search for an alternative resource allocation is continued.

Over Constrained Problems

Note that in this application the primary objective is to 'keep the plant running'. Thus those materials which are already 'in-process' in the plant - containers of molten metal - **MUST** be scheduled. In the case of such over-constrained problems, in which there is no solution or no solution can be found within acceptable time periods, an *additional* search procedure is applied.. This finds a schedule for only those materials which are already in-process, creating a (partial) schedule to meet the very short term requirement to instruct the operators concerning the immediate future. Of course, in generating this partial schedule, works orders which have yet to be started are ignored, and further (human) intervention is required to modify the longer term problem, making it tractable.

Application Control Loop

This section briefly presents the overall application control, as far as it applies to the integration of a user

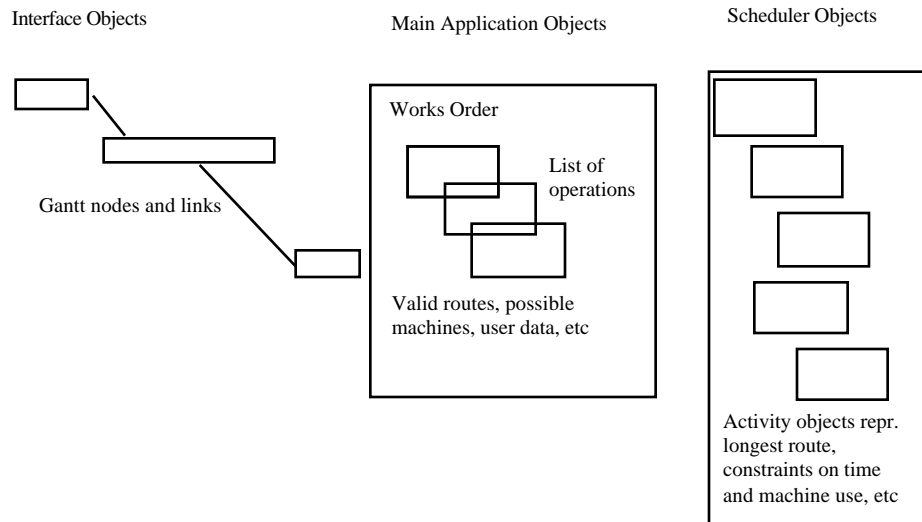
Short Term Scheduling for Liquid Metal Manufacture

Mike Pegman, mp@vinesolutions.co.uk

suggest a framework to solve the data and application control problems which occur in interactive rescheduling systems.

There are objects in the model which represent each logical entity in the manufacturing process. These objects represent the application in terms of instances of data and relationships. They are independent of both the visualisation objects and the internal scheduler's objects. The object model of the application is thus divided into three layers - main application object model, visualisation objects and scheduler objects.

At various times in the running application these main application objects will create temporary objects which are reflected in the user interface on one side and the scheduling system on another. (The interface objects are sub classes of Ilog Views gantt chart objects. The scheduling objects are application specific classes containing instances of Ilog Scheduler objects.) Data concerning operation timings, machine allocation, route options etc.



comes from the user. Data concerning the resulting plan comes from the scheduler. The main application objects represent and maintain the 'master' view of the schedule. (The main application objects will ultimately be updated by the low level plant data acquisition systems as well as by the user interface and the scheduler.)

Since the scheduler is free to create a route which, when compared with a previous plan, has a different length or uses different machines, interface objects may need to be created, destroyed or moved across resources. Similarly the user (or eventually, in the on-line system, the plant itself) may impose additional constraints on times or machines so that a subsequent run of the scheduler will need to create potentially different scheduler objects.

The above motivates an application structure comprising a loop:

1. create initial main application objects (potentially from data base)
2. create interface objects representing existing or new main objects, and allow user to edit these and to create new interface objects
3. user requests a (re-)schedule
4. update, and if necessary create, main application objects from user interface objects
5. create entirely new scheduler objects corresponding to main application objects
6. run scheduler, update main application objects, destroy scheduler objects
7. loop to 2.

Step 4 above includes the representation in the main application objects of constraints from the user interface / on-line monitoring system.

Step 4 also needs to consider the impact of the advance of the real time clock. These will include the automatic imposition of additional constraints on resource allocation, routings and timings of operations.

Step 5 & 6 are the only steps in the application control loop which require the use of Ilog Solver / Scheduler code and can conveniently be wrapped in calls to initialise and reset the libraries (IlogInit, IlogEnd). (Conceptually this code could run on a separate server in a client-server architecture; the server receiving and returning relevant main application objects, or equivalent database transactions.)

Step 2 may of course be impossible (due for instance to an over-constrained or infeasible problem). The segmentation of the code at step 2, perhaps creating objects in priority order, enables information flow back to the user to guide the correction / modification of the problem.

Short Term Scheduling for Liquid Metal Manufacture

Mike Pegman, mp@vinesolutions.co.uk

This structure has been employed in a number of interactive planning and scheduling systems which the author has designed, and it has been found to facilitate both their design and the control of data flow within them.

Summary

The demonstrator exercise upon which this paper is based successfully produced a reactive scheduler for a short term liquid metal manufacturing process plant. Complex constraints were represented in the system. The system successfully solves typical problems.

The demonstrator system has had successful early trials at a customer site. Trials are continuing and the application is being developed further and integrated into the rest of a suite of manufacturing systems before installation in an on-line control role.

This paper has presented an overview of a resource allocation and scheduling application and described several of the main constraints, giving comments on their implementation in Ilog tools and a brief description of aspects of search strategy. In addition a three layer object model has been presented which has more general applicability to similar interactive incremental scheduling systems. It is hoped that in these ways the paper will be of assistance to those developing schedulers in other application areas.

For more information please contact:

Mike Pegman,

Vine Solutions Ltd

Tel / Fax +44 (0) 191 4166389

9 The Chase, Washington, Tyne & Wear, NE38 9DX, UK

www.vinesolutions.co.uk

mp@vinesolutions.co.uk

Vine Solutions Ltd gratefully acknowledges the permission of Broner Group Ltd to publish this paper.