S21-63
N93-18220
p. 5

# Realization of High Quality Production Schedules : Structuring Quality Factors via Iteration of User Specification Processes

Takashi Hamazaki

Department of Artificial Intelligence, University of Edinburgh
80 South Bridge, Edinburgh EH1 1HN, United Kingdom
E-mail:T.Hamazaki@edinburgh.ac.uk

## Abstract

This paper describes an architecture for realizing the high quality production schedules.

Although quality is one of the most important aspects of production scheduling, it is difficult even for a user to specify precisely. However it is also true that the decision whether a schedule is good or bad can be taken only by a user.

This paper proposes;

- The quality of a schedule can be represented in the form of quality factors, i.e. constraints and objectives of the domain, and their structure.
- Quality factors and their structure can be used for decision making at local decision points during the scheduling process.
- They can be defined via iteration of user specification processes.

## 1 Introduction

Production scheduling is a *hard* problem in general because of the large search space, large number of factors lead to combinatorial explosion, and also its ill-structured (or ill-defined) nature. The primary concern of this paper is realizing high quality schedules, which is one of the major difficulties of production scheduling.

Since the schedule should be evaluated by several often *conflicting* aspects, it is a common approach to expect the user to specify a single evaluation function, i.e. satisfaction level of each aspect and priority among them.[6, 7, 13] However, it is difficult even for a user to define an evaluation criterion for a schedule precisely.[1] The methods that a system can use to decide an evaluation criterion, and to produce a schedule which optimizes that criterion, are important issues in this domain.

Although it is difficult for users to define an evaluation criterion, at the same time, it is also true that the decision whether a schedule is good or bad can be taken only by a user. (Please compare other aspects, e.g. the performance of a system can be evaluated by an absolute measure – i.e. time. ) It follows that the schedule should be evaluated on domain specific information relating to a definition of quality given by the user. Furthermore, the quality information should be used for search guidance during scheduling, since the primary goal of search is affected by the decision of what a good/bad schedule is.

Quality is determined by the combination of the extent to which constraints are satisfied and how well objectives are achieved. Since constraints and objectives can be regarded as atomic factors of the quality of a schedule, I call them *quality factors* in this paper.

This paper describes an architecture which can acquire quality information from the user and reflect the information on the resulting schedule via iteration of user specification processes. This work is currently in progress.

## 2 Analysis of quality factors

### 2.1 Structure of quality factors

Before attempting classification, this section will concentrate on the relationships among quality factors.

Since quality factors are defined by a user, they are often interrelated of each other. Some *include* other quality factors, and some *cause* another factor. For instance, a prohibition against changeover at the same time (due to a limitation on operators) can be divided into two levels below.

1. changeover itself - namely, a condition of changeover (defines this quality factor as QF1)
2. simultaneous occurrence of QF1 (QF2)

QF2 can be thought of as a meta-level quality factor.
This information relating to the relationships among quality factors is quite useful for scheduling, and they are defined as a *structure* of quality factors in this paper.

## 2.2 Classification of quality factors

Quality factors can be classified from several points of views; for examples the function in a real plant[12, 14], the influence on scheduling.[3] In this section I attempt to classify quality factors based on the relationship with the scheduling algorithm. This classification is more detailed than others in order to use it for acquiring additional information about quality factors from the user.

**hard – soft** The first dimension of classification is based on the strictness of satisfaction/violation ;

- *hard factor* : one which must be satisfied, i.e. cannot be relaxed any more.
- *soft factor* : one which is preferable to satisfy. As all quality factors are preferable to satisfy, *soft factor* can be defined as the complement of *hard factor* more strictly.

**Job and Resource** The next dimension of classification is based on parameters which a factor contains. The parameters of a factor are either/both of *Job* and *Resource*.
They can be broken down further according to the necessity for the reference to other objects during an evaluation of the quality factor as follows;

*Job* –

**inter-lot** need to refer to operations in other lots,

**intra-lot** need to refer to other operations in the same lot and

**no-interaction(no-int)** no need to refer to other operations, and

*Resource* –

**inter-machine** need to refer to other machine,

**intra-machine** no need to refer to other machines.

For instance, the parameters of a changeover(QF3) are Job and Resource and detailed class of Job part is *inter-lot* and that of Resource is *intra-machine*. Therefore, this quality factor, "a changeover"(QF3), can be classified as (*inter-lot Job* , *intra-machine Resource*) type.

**Global and local** The last dimension is based on the applicability at a local decision point. Intuitively, *global* factors are those which can be used to evaluate a full schedule, while *local* factors are those which can be used to evaluate a partial schedule. However, many quality factors can be applied to the evaluation of candidates at a local decision point (even if it looks like *global* one) by using an estimation of resulting value. For instance, although QF2 in previous examples cannot necessarily always be applied at local decision points as it is, it might be possible if the probability of changeover for each product type could be estimated.
It follows that the revised version of the definition is

**A global factor** is one for which a user cannot define an estimation function at all.

**A local factor** is the complement of global factor, i.e. those which a user can define so that they can be applied at local decision points.

# 3 Scheduling via quality factors

## 3.1 Iterative user specification processes

In the previous section, the concept of quality factors was introduced. This concept makes it possible to characterize the user's evaluation of a schedule, mentioned earlier , as follows.
Suppose as an example two schedules are compared.

1. apply hard quality factors to every item – presumably operations – of each schedule.

   **IF** violation has occurred in either of two schedules $\longrightarrow$ unacceptable schedule

   **ELSE** $\longrightarrow$ next step

2. apply most important soft quality factors to
   – every item of the schedule – if the quality factor is local
   – the whole schedule – if the quality factor is global

**IF** a sufficient difference between them is identified in any of the quality factors $\longrightarrow$ decide

**ELSE** $\longrightarrow$ next step

3. apply next level of soft-quality factors
   ... same as above.

If the importance of every QF could be categorized and exact values for a *sufficient* difference could be defined beforehand, this process could be done automatically and it might be possible (apart from realistic processing speed) to optimize a schedule. However, specification – especially the criterion for sufficiency – is difficult (or close to impossible) to define precisely in advance. Consequently, it will be indispensable to adopt some sort of trial and error process for deciding a good schedule. From this view, the following procedure should be an acceptable method for acquiring the information from a user.

1. user specifies each quality factor
   user specifies priority among quality factors in as much detail as possible.
2. system produces a schedule based on quality information acquired so far.
3. user analyzes the resulting schedule produced in the previous step.
   user judges whether the schedule is satisfactory or not.

   **IF** satisfactory $\longrightarrow$ end.

   **ELSE** $\longrightarrow$ specify which QF should be improved.
4. system re-structures quality information
   goto 2.

## 3.2 Search guidance by quality factors

Schedule production,i.e. step 2 in the procedure defined in the previous section, is accomplished by a repetition of target selection , i.e. operation, and a reservation for it, i.e. resource and start time. In the each repetition cycle, it is desired to rate candidates appropriately which results in a *good* overall schedule. The next section focuses on this rating step.

### 3.2.1 Rating by quality factors

When a schedule can be evaluated by a function – defines it as $E$ – two dimensions can be viewed as rating methods.

**local optimization(LO)** how good will the quality of a partial schedule be
— measures candidates by $E(Pn)$ : where $Pn$ is a partial schedule after adopting candidate-N

**predicted global optimization(PGO)** how good is the quality of the final schedule likely to be, in other words from the opposite perspective, how difficult will expected problems be.[1]
— measures candidates by $E'(Pn)$ : where $E'$ is a probabilistic function which expresses the value likely to be achieved.

As described in section 3.1 , $E$ is realized by a series of applications of quality factors and filtering out in each quality factor application. $E'$ is similar in general, since a *predicted* final schedule should be evaluated in the same manner. However a probability among quality factors should be also taken into account as well as the priority among them. For instance, if it is known that QF1 frequently identifies a bad value, it might be better to apply QF1 prior to other QFs, even though the priority of QF1 is not highest.

The application of quality factors is stopped when a sufficient *difference* among candidates is identified. This *difference* is described as a *threshold* in this paper.

## 3.3 Feedback from the result

In this section, we describe how the system restructures quality factors in reaction to the schedule produced,i.e. step 4 in the procedure defined in Section 3.1. This process can be accomplished both automatically and manually.

### 3.3.1 Manual feedback

As described earlier, the user should judge whether the resulting schedule is satisfactory or not. Generally speaking, a user can communicate with the system via quality factors and structures among them. That is, if the schedule is not satisfactory for a user, the reason why its schedule is not satisfactory is expressed by indicating which quality factors' values should be improved. This feedback from the user will influence structures of quality factors.

In order to support a user in detecting problems, the system provides information, as follows:

**verification of assumption** It is unrealistic to expect that a user can specify the structure of

[1]There are two heuristics for realizing this method; i.e. variable ordering and value ordering.[5]

quality factors precisely from the early stages of scheduling generation. Consequently, a system should assume some information about structure. The information assumed by a system should be verified at the end of scheduling generation process by the user. The user is informed of assumed structures at the feedback stage.

**evaluation by global factors** Resulting schedules can often be evaluated at a gross level by global factors, like overall utilization, although all quality factors should be involved for a precise evaluation. Furthermore, since global factors are considered only via causal factors during the scheduling generation process, verification is indispensable. Statistical information based on global factors is provided by the system automatically.

**evaluation by specific factors** It is quite usual that a user knows which quality factor is critical in the specific application/domain. Statistical information is also provided in response to the user.

### 3.3.2 Automatic feedback

When scheduling has not been completed, i.e. there remain unassigned operations, the system analyzes its reasons and restructures quality information based on some heuristics, which include

- If there are quality factors in which unassigned operation got the best value
  $\longrightarrow$ decrease threshold of those quality factors
- If there are quality factors in which unassigned operation was the next candidate
  $\longrightarrow$ increase threshold of those quality factors

## 4 System structure

The system consists of mainly four parts, namely; Scheduler, Generator, Analyzer and Data-Base manager and six system files, namely; Quality Data Base(QDB), Evaluation Procedures File(EPF), Scheduling results File(SF), Decision history file(DHF), Order file(OF) and Knowledge-Base(KB).

The general flow of this system is as follows (this can be thought of as a detailed version of the iterative procedure described in Section 3.1.);

1. user specifies initial information
   - order data (presumably from other system) $\longrightarrow$ OF
   - domain information, e.g. factory, machine $\longrightarrow$ KB
   - quality factor[2] $\longrightarrow$ KB
   - attribute of quality factors, e.g. global/local , hard/soft $\longrightarrow$ QDB
   - structure of quality factors (in as much detail as possible) $\longrightarrow$ QDB
2. Generator generates evaluation procedures, which can be used in the rating of candidates during scheduling process, based on QDB information and output $\longrightarrow$ EPF
3. Scheduler generates a schedule based on OF, KB and EPF and output
   - resulting schedule(including unassigned operations) $\longrightarrow$ SF
   - history of rating by quality factors at every local decision points $\longrightarrow$ DHF
4. Analyzer analyzes SF and DHF and queries the user if necessary and restructures QDB.
   goto 2 if not satisfactory

## 5 Future work

This system uses a traditional algorithm as its scheduling mechanism, since the scheduling algorithm itself is not the major concern. However, it is obvious from the analysis in Section 3.2 that quality information which is acquired from a user and scheduling algorithm have tight connection. It follows that the ideas proposed here are restricted by this algorithm. It is required to analyze validity on other algorithms, e.g. distributed scheduling system[4] , as well and extend these ideas.

This system is now being implemented and will be evaluated using real problems, although it is based on my experiences in developing practical production scheduling systems.[9, 8]

## 6 Conclusion

Although quality is one of the most important parts of production scheduling, it is difficult even for users to define precisely. The first step in realizing a high quality production schedule is to clarify what "high quality" means.

[2]The system requires a user to specify function which represents the goodness of the selected candidate for every quality factor. At the same time, the current system also requires a probabilistic function for each quality factor, although this should be eventually supported by the system.

This paper proposed;

- The quality of production schedules can ultimately be evaluated/measured only by a user, and his intention can be represented in the form of quality factors and their structures defined by him/her. (global evaluation)
- Quality factors and their structures can be used for decision making at local decision points during the scheduling process. (local evaluation)
- They can be refined via iteration of the user specification process. (iterative process)

## 7 Acknowledgement

This research has been supported by Hitachi Limited(Japan). The author of this paper wishes to acknowledge the support of the following people: Tim Duncan and Howard Beck of AIAI and Peter Ross of DAI.

## References

[1] Mostafa El Agizy. Design of systems for replenishing stocks of materials. In Haluk Bekiroglu, editor, *Computer Models for Production and Inventory Control*, pages 61–. The Society for Computer Simulation, California, 1988.

[2] H. Atabakhsh. A survey of constraint based scheduling systems using an artificial intelligence approach. *Artificial Intelligence in Engineering*, 6(2):58–73, April 1991.

[3] Pauline M. Berry. *A Predictive Model for Satisfying Conflicting Objectives in Scheduling Problems.* PhD thesis, University of Strathclyde, 1991.

[4] Iain Buchanan, Peter Burke, John Costello, and Patrick Prosser. A Distributed Asysnchronus Hierarchical Problem-Solving Architecture applied to Plant Scheduling. In G. Rzevski, editor, *Proceedings of the forth International Conference on the Applications of Artificial Intelligence in Engineering*, pages 107–114, Cambridge, UK, July 1989.

[5] Rina Dechter and Judea Pearl. Network-Based Heuristic for Constraint Satisfaction Problems. *Artificial Intelligence*, 34(1):1–30, 1988.

[6] Mark S. Fox and Katia P. Sycara. The CORTES Project:A Unified Framework for Planning Scheduling and Control. In *Proceedings of the 1990 DARPA Workshop on Innovative Approaches to Planning, Scheduling and Control*, pages 412–421, 1990.

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

[8] T. Hamazaki, T. Kameda, S. Okuide, M. Koroku, and K Kozuka. Approach to Planning and Scheduling Expert Systems. *The Hitachi Hyouron*, 72(11):41–46, November 1990. Japanese.

[9] N. Irisawa, T. Hamazaki, and T. Yamanaka. Production Scheduling by using Expert Systems. *Communication of the operations research society of Japan*, 36(3):141–145, 1991. Japanese.

[10] Richard E. Korf. Search:A Survey of Recent Result. In Howard E. Shrobe, editor, *Exploring Artificial Intelligence*, pages 197–237. Morgan Kaufman, California, 1988.

[11] P. A. Newman. Scheduling in CIM systems. In Andrew Kusiak, editor, *Artificial Intelligence:Implication for CIM*. Bedford IFS, 1988.

[12] BC Niew, BS Lim, and NC Ho. Knowledge Based Master Production Scheduler. In *First International Conference on Expert Planning Systems*, pages 88–93, London, 1990. IEE.

[13] F. Stephen Smith, Peng Si Ow, Nicola Muscettola, Jean-Yves Potvin, and Dirk C. Matthys. OPIS:An Integrated Framework for Generating and Revising Factory Schedules. In *Proceedings of the 1990 DARPA Workshop on Innovative Approaches to Planning, Scheduling and Control*, pages 497–507, 1990.

[14] S. Smith, M. Fox, and P.S. Ow. Constructing and Maintaining detailed Production Plans: Investigations into the Development of Knowledge-based Factory Scheduling Systems. *AI Magazine*, 7(4), 1986.

[15] S. F. Smith and P. S. Pw. The use of multiple problem decomposition in time constrained planning tasks. In *Proceedings of IJCAI 85*, pages 1013–1015, 1985.