ELSEVIER

Computers and Chemical Engineering 24 (2000) 345–351

Computers & Chemical Engineering

www.elsevier.com/locate/compchemeng

Autonomous decentralized scheduling system for just-in-time production

Tatsushi Nishi, Akihiro Sakata, Shinji Hasebe *, Iori Hashimoto

Department of Chemical Engineering, Kyoto University, Kyoto 606-8501, Japan

Abstract

In this paper, an autonomous decentralized scheduling system for just-in-time production is proposed. In the proposed system, each scheduling sub-system belonging to respective production stage derives a near optimal schedule by repeating the generation of the schedule of its own stage and data exchange among the other production stages. The objective function for each scheduling sub-system includes the storage costs for intermediate and final products in addition to the changeover costs and the due date penalties. When the storage costs are included in the objective function, the calculation of starting times of jobs becomes difficult even if the production sequence is given. In this study, an efficient algorithm is introduced to each scheduling sub-system to reduce the computational effort. It is shown that the performance of the proposed system is almost as good as that of a conventional scheduling system considering the entire plant. To cope with various unexpected events for the proposed system, a rescheduling procedure is developed and tested on a discrete-event simulator. The results show, if the proposed procedure is adopted, improved schedules, which minimize the effects of disturbances, can be generated in a significantly shorter computation time.

Keywords: Decentralized scheduling; Just-in-time production; Flowshop scheduling; Storage cost

1. Introduction

Just-in-time production has been regarded as a promising production system ensuring in time product delivery, low inventory, and short product lead time. However, in the daily manufacturing of most plants producing wide range of products, there are some inevitable changes caused by unexpected events such as equipment failure, product inferiority or sudden order of special products. Therefore, it is necessary to develop rescheduling methods, which can frequently and quickly modify a schedule in order to cope with those disturbances.

Most scheduling systems in the literature deal with regular measures, such as makespan or tardiness penalty, which are nondecreasing with respect to the completion times of jobs. In just-in-time production, earliness as well as tardiness should be incorporated in the objective function in order to reduce the storage costs for intermediate and final products (Baker & Scudder, 1990). In scheduling systems for just-in-time production, pull-mechanism is commonly used as a heuristic rule for generating a schedule. However, these systems are applicable only to the cases where changeover cost between two successively processed operations is negligible. Furthermore, such systems often become increasingly complex and large when new functions to handle the unforeseen are added. If schedules are created independently at each production stage, it is easy for the scheduling system to adjust the schedules to various requirement changes.

From that point of view, an autonomous decentralized scheduling system that has no supervisory system controlling the entire plant with regard to creating schedules was proposed by Hasebe, Kitajima, Shiren, Murakami and Hashimoto (1994). The outline of the autonomous decentralized scheduling system is shown in Fig. 1. The total scheduling system consists of a database for the entire plant and some scheduling sub-systems belonging to the respective production stages. Each scheduling sub-system derives a plausible schedule in the following steps: first, each scheduling sub-system contacts the database and obtains the demand data,

* Corresponding author. Fax: +81-75-7529639.
E-mail address: hasebe@cheme.kyoto-u.ac.jp (S. Hasebe)

0098-1354/00/$ - see front matter
PII: S0098-1354(00)00471-3

and independently generates a schedule of the stage without considering the schedules of the other production stages. However, the schedule obtained by simply combining the schedules of all production stages is impracticable in most of the cases. Therefore, at the second step, the scheduling sub-system contacts the sub-systems of the other production stages and obtains the schedule information of those stages, and generates a new schedule. The schedule generation at each stage and data exchange among the scheduling sub-systems are repeated until a feasible schedule for the entire plant is derived. The effectiveness of the system when applied to flowshop and jobshop problems is discussed in Hasebe et al. (1994).

In this paper, an autonomous decentralized scheduling system for just-in-time production is proposed. The scheduling algorithm, which aims at minimizing changeover costs and storage costs for both intermediate and final products, is developed for each scheduling sub-system. To verify the effectiveness of the proposed system, a conventional scheduling system looking at the entire plant is also developed. A variety of multi-stage flowshop problems are solved by both the systems.

In the proposed system, each sub-system is capable to start rescheduling when unexpected events occur. Taking this feature into account, a rescheduling procedure in which the influence of unexpected events can be localized, is proposed.

2. Problem definition

The multi-stage flowshop batch plant to be scheduled is divided into several production stages by taking into account the technical and/or managerial relationships in the plant. It is assumed that each production stage consists of several parallel units. It is also assumed that the plant satisfies the following conditions:

1. each job has its own due date, and standard processing time at each unit is given a priori. The earliness and tardiness penalties at the final stage are embedded in the objective function;
2. the changeover cost at each unit depends on the product types of jobs successively processed; and
3. the intermediate products can be stored temporarily. Thus, the production sequences of jobs at two successive production stages need not be the same. Intermediate storage cost is proportional to the storage period of the job between two production stages.

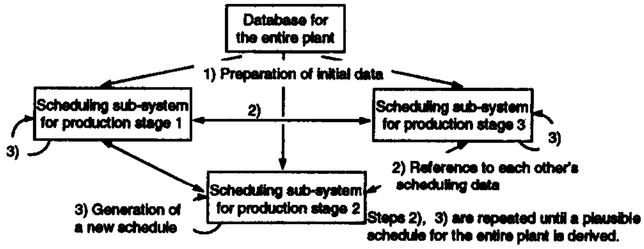

Fig. 1. Autonomous decentralized scheduling system.

The objective of the scheduling problem discussed in this research is to determine the production sequence of jobs and the starting times of jobs which minimize the weighted sum of the changeover costs, the intermediate storage costs, and the earliness and tardiness penalties.

3. Autonomous decentralized scheduling system

3.1. Scheduling algorithm

Each scheduling sub-system derives a near optimal schedule in the following steps.

3.1.1. Step 1: Preparation of initial data

Each scheduling sub-system contacts the database managing the entire plant data and obtains the demand data, such as product name, earliest starting time and due date for each job to be processed at the production stage.

3.1.2. Step 2: Generation of an initial schedule

Each scheduling sub-system independently generates a schedule of its own production stage without considering the schedules of other stages.

3.1.3. Step 3: Reference to scheduling data belonging to the other scheduling sub-systems

The scheduling sub-system belonging to production stage k contacts the other scheduling sub-systems and exchanges the following data:

1. the tentative earliest starting time (TEST) for each job i (e_i^k): the ending time of job i at the immediately preceding production stage; and
2. the tentative latest ending time (TLET) for each job i (f_i^k): the starting time of job i at the immediately following production stage.

TEST and TLET of job i at the production stage k are schematically shown in Fig. 2.

3.1.4. Step 4: Judgment on whether schedule generation should be executed or not

In some cases, same schedules are generated cyclically at Step 5. In order to avoid such a cyclic generation of the schedules, the scheduling sub-system skips the schedule generation step (Step 5) randomly with a certain probability.

3.1.5. Step 5: Schedule generation

Using the data obtained at Step 3, each scheduling sub-system improves the schedule of that production stage. In order to derive a near optimal schedule, the