Reactive Schedule Repair of Job Shops

Amritpal Singh Raheja and Velusamy Subramaniam

Abstract— **Disruptions to job shop schedules are tedious and difficult to incorporate after the schedule has been generated and implemented on the shop floor. In order to deal with such disruptions, a real time reactive scheduling strategy is essential. Reactive scheduling is the process of repairing the predictive schedule during online execution for internal disruptions (e.g. machine breakdowns) and external deviations (e.g. prepone or postpone of orders). Existing approaches for schedule repair in real time mainly utilize heuristics such as Right Shift Rescheduling (RSR), and Affected Operation Rescheduling (AOR). In the present form, both these approaches are only used for handling machine breakdowns in the shop floor, but are inept in accommodating new and unexpected job orders. These approaches also neglect specific issues related to urgent jobs, for instance multiple job routings during the repair of the schedule. In this paper the existing heuristics (RSR and AOR) have been modified to include urgent jobs. Also a modified AOR approach (*m*AOR) is proposed that considers urgent jobs with multiple job routings. An extensive simulation study has been conducted on a job shop simulation testbed for the efficiency and stability of the repaired schedule using the mAOR and RSR heuristics. The efficiency of the repaired schedule is a measure of the percentage change in the makespan after incorporating repairs whereas the stability of the schedule is a function of starting time deviations that indicate the degree by which it deviates from the original schedule. The results of the experiments indicate significant benefits of the modified AOR algorithm over the existing RSR schedule repair heuristic.**

Keywords— **Schedule Repair Heuristics and Job Shop Schedule.**

A.S. Raheja is a research student at the Department of Mechanical Engineering, National University of Singapore. E-mail: engp1059@nus.edu.sg
V. Subramaniam is an Asst. Professor at the Department of Mechanical Engineering, National University of Singapore. He is also a Fellow with the Singapore-MIT Alliance. E-mail: mpesubra@nus.edu.sg.

I. INTRODUCTION

R*eactive Scheduling* is a process of revising a given schedule in real time due to the occurrence of unexpected events during the execution of the schedule. The predominant scheduling activity in practice is that of reactive scheduling, which can be broadly defined as the continuous adaptation and improvement of pre-computed predictive schedules [1]. Reactive scheduling can be construed as similar to offline scheduling, albeit conducted "online" in which the previously accepted schedule, which has now been flawed due to an unexpected event, is repaired by techniques that can be essentially similar to those used to iteratively improve a predictive schedule.

When a schedule goes bad due to small disruptions, an intuitive approach would be to resolve the problem from scratch, i.e., rerun the predictive schedule and generate a new optimized schedule. This approach is not encouraged in the industry as the new schedule can differ considerably from the old one and this is not desirable since many other decisions like assignment of personnel, delivery of raw material and the subsequent processing of the jobs in other facilities may be severely disrupted. This phenomenon is commonly referred to as *Shop floor nervousness* [2]. If the disruptions are large and frequent there is no option but to totally reschedule the system. In case of minor disruptions on the shop floor a better approach would be to adapt the old schedule to the new situation. This is possible using a specialized repair mechanism in an iterative mode as illustrated in Figure 1.

Minor disruptions of the order of 8% of the makespan have been used to test repair algorithms [3]. Thus a significant portion of disruptions occurring on the shop floor can be handled using the repair techniques without triggering "Total Rescheduling". Schedule repair approaches that have been reported in literature do not consider all types of disruptions that occur as current research has concentrated on machine breakdowns and accommodating unexpected jobs. Other factors, such as multiple job routings have also not been addressed.

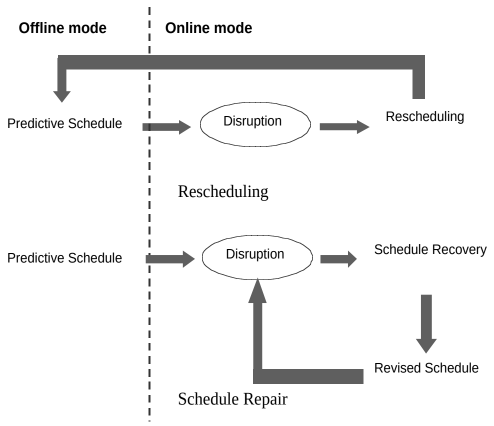

Fig. 1. Rescheduling vs. Schedule Repair

In the next section, the common schedule repair approaches reported in literature are presented. In Section 3, a new schedule repair heuristic based on AOR is described in detail. Subsequently, an extensive experimentation is reported and the paper concludes with a detailed discussion of the results.

II. LITERATURE SURVEY

The common reactive repair approaches are summarized in Table 1 and are discussed in this section.

Heuristic based approaches [3,4] largely consist of explicit algorithms for schedule repair and optimization. One of the common heuristics reported in the literature is the *Right - Shift Rescheduling* (RSR) [3,4]. This heuristic involves the global shifting of the job operations and expanding the schedule towards the right on the time axis in order to accommodate the disruptions. Another heuristic that is considered effective is the Affected Operation Rescheduling (AOR) [3], in which only the job operations that are affected by the disruption are rescheduled. The basic concept of AOR is to accommodate any disruption by pushing the starting times of job operations forward by the minimum amount possible so as to keep the technological constraints satisfied and to preserve the initial sequence of the operations on each machine.

Multi Agents in a Distributed Artificial Intelligence (DAI) environment has been widely reported in the literature [5-7]. In DAI based approaches, reactive schedule recovery is achieved using multi-agents. It allows the independent agents to coordinate their knowledge and solve sub problems while working toward a common goal. In this approach intelligent agents possess the knowledge pertaining to the schedule repair.

Fuzzy logic has also been reported in the literature for reactive schedule repair [8-9]. The fuzzy processing times are substituted for the crisp processing times used by a heuristic based scheduler. This approach has an advantage of a complete scan of the schedule for constraint violations every time a new event is integrated, and the schedule is optimized globally during the repair. Case based reasoning, [2, 10] has been used for reactive scheduling and the goal is to find a case that best suits the disrupted schedule. The case database is created by domain experts and is capable of specifying the correct reaction to schedule disruptions. The concept of Constraint based scheduling has been reported in the literature [11,12] for schedule repair of a prototype scheduling system (CABINS). The incremental accumulation and reuse of past experiences is achieved through case based reasoning, while constraint based scheduling has been used for the propagation and resolution of the effect of repair.

Artificial Intelligence techniques such as Neural Networks [13,14] and Genetic Algorithms [15] have also been used in schedule recovery and repair. The training of neural networks is usually performed in a single pass, and Genetic Algorithms mimic the natural selection process of genes to perform schedule repair. Crossover and mutation operators are used to generate successive population of better schedules [15] to accommodate the deviations in the original schedule. Genetic Algorithms are efficient and produce nearly optimum schedules but require high computational effort.

III. THE MODIFIED AOR HEURISTIC

In this paper, the authors have proposed a new schedule repair heuristic based on AOR. This new heuristic is referred to as the Modified Affected Operation Rescheduling or mAOR. The basic principle behind the concept of AOR is to accommodate a minor disruption on a particular machine in the initial schedule by pushing the start times of some job operations forward (delaying them) by the minimum amount required to: (1) keep the technological constraint satisfied and (2) preserve the initial sequence of operations on each machine. This guarantees that the robustness of the initial schedule is preserved