Distributed Integrated Scheduling in Automated Manufacturing Systems with Transient Machine Failures

H. F. Wedde, S. K. Taneja[1], S. Lehnhoff
School of Computer Science

M. ten Hompel, D. Liekenbrock, S. Libert
College of Mechanical Engineering

University of Dortmund
44221 Dortmund / Germany

Abstract

For a Just-in-Time production scheme in large manufacturing systems we describe a considerable methodological extension into an integrated automated manufacturing system that is supported by a distributed real-time computer system. This allows for higher flexibility in demand fluctuation as well as for a larger variety of products. At the same time better adaptability of transportation planning and transportation cost reduction are achieved through their integration. For this integrated scheduling we will present new bidding algorithms. Eventually, in an extension of the presented scheduling algorithms, novel fault tolerant strategies are included for overcoming or neutralizing the effect of transient machine failures. These are handled by cooperating local schedulers and managed to guarantee a minimal damage of schedules and its propagation, in the presence of failures. Through their integration into the production and transportation scheduling the advantages of the Just-in-Time approach (no storage costs) are preserved in principle while a near-optimal way for the affected jobs is found to meet their deadlines. The results of our extensive simulation experiments for a real production scenario are also discussed.

Keywords: safety-critical, real-time systems, embedded systems, distributed systems, multi-agent systems, electronic negotiations

1. Introduction

In the research landscape of Automated Manufacturing Systems much work has been done in automizing single production and transportation control, originally inspired by the demand-driven just-in-time concept, with its kanban communication between cells. Different from earlier hierarchical concepts (e.g. [JONE86]) we assume a high amount of autonomy both for cell control computers and AGV control. In this paper the novelty of distributed control goes hand in hand with a concept of complete integration of production and transportation control. While in such physical systems there are end-to-end deadlines which in turn are broken down into deadlines for all single production and transportation processes (in a coherently dynamic fashion) transient machine failures, as they cannot be avoided, require a partial replication of machines in cells, or a certain amount of multifunctionality, and a certain AGV redundancy. At the same time this provides for a high amount of scheduling and planning flexibility of the whole system. Thanks to new distributed bidding algorithms we achieve all objectives mentioned, through incremental construction. It is important to note that the time granularity of the production/transportation processes on the one hand, and of the automated control processes are orders of magnitude apart such that the computer scheduling overhead is nearly neglectable. This will be further discussed below.

Previous and related work. After work on cell scheduling under hierarchical control [JONE86] and a remarkable number of concepts for uniprocessor systems the idea of distributed architectures for manufacturing systems had been introduced [DILT91]. So far no in-depth treatment of this subject, e.g. through formally defining production scheduling algorithms and performing experimental analysis, has been reported. Although several control architectures for integrating cell planning (including fault tolerance measures) and control functions had been proposed [LRP01], corresponding distributed algorithms have not yet materialized. Representative examples for such architectures are [SKS+05, SMH+04]. In our case the concepts, and part of the results, in [TAN93] are a major basis for this paper. Also, the work described here is part of the research project COMTRANS which has received initial funding through the Fraunhofer Society (FhG-IML).

Organization of the paper. After formally defining the distributed manufacturing model for production scheduling our distributed bidding algorithm will be presented in section 2 (referred to as *Algorithm I)*. In section 3 its extension as to integrate AGV scheduling will be introduced (referred to as *Algorithm II)*. In section 4 both the manufacturing model and the algorithm will be again refined and extended such that transient machine failures can be handled in a completely distributed fashion *(Algorithm III)*. A comparative simulation study on the novel algorithms will be reported in

[1] Lucent Technologies Inc.

1-4244-0681-1/06/$20.00 ©2006 IEEE

section 5 from which conclusions will be drawn in section 6.

Pseudo-code of the 3 algorithms can be found online at: ***http://ls3-www.cs.uni-dortmund.de/ComTrans***

2. Scheduling of Cells

2.1 The Distributed Manufacturing Model

The manufacturing cells are interconnected with a local area network to form a loosely coupled distributed system. The transportation time from one cell to another is assumed to be negligible for the purpose of cell scheduling in this chapter. (For accommodating and integrating Automated-Guided-Vehicle (AGV) scheduling however, the transportation time will be taken into consideration in chapter 3.)

For a given diversity of products that could be manufactured, each product consists of *parts* which may undergo a series of operations (drilling, boring,..) and which will be assembled into components through *assembly operations*. The components themselves may undergo specific operations, and the resulting subproducts may again be assembled into larger components. In this way a final product is generated.

All operations are performed within *cells*. Each cell has a variety of different machines each of which could perform different operations on a specific set of subproducts or components. (As described in the introduction this provides the basis for both a certain production flexibility and fault tolerance.) The cell flexibility, in contrast, is constrained by functional precedences of operations.

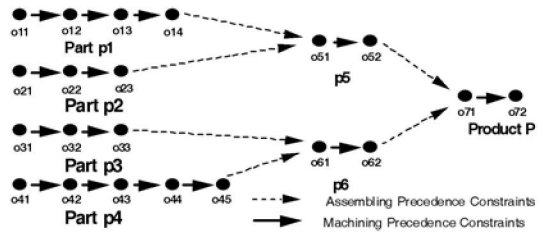

Figure 1: Assembling and Machining Structure of a Product

Under a given set of machining and assembling constraints, production schedules of each involved cell are to be developed in a distributed fashion by the cooperation of the cell controllers. Each cell controller is in charge for scheduling all operations that are performed on a subproduct in its cell. Furthermore we assume:

1. The products/sub-assemblies can be produced by selecting from alternative production sequences within the cells. The cell scheduler will select a sequence based on the current schedule of machines and the deadline of the job to be scheduled.

2. According to the standard manufacturing practice, operations are considered to be non-preemptive, i.e. an operation runs to completion once started.

3. While scheduling a new job, the cell scheduler follows the First-Come-First-Serve (FCFS) policy. Since scheduling within the cells is not a major focus of our study we do not make assumptions on job priorities.

4. We assume that manufacturing is consumer-driven. Thus the arrival times of the jobs are considered to be *aperiodic* or *unpredictable*. Jobs for manufacturing products arrive in the *Scheduling Department SD*.

5. Completion times for a given operation and its subproducts may vary between different capable machines.

6. If an operation on given subproducts could be performed within a cell, the corresponding cell scheduler will schedule it on a local basis. (Consequently the scheduler will not consider to cooperate with other cell schedulers to this extent even if an external machine would be faster. The idea is that this potential advantage would be more than outweighed by additional transportation costs and scheduling overhead.)

2.2 Formal Notations and Definitions

Given a real-time cellular manufacturing system with c cells $C_1, C_2, \ldots, C_C$. A cell C_i will have I machines $M_{i1}, M_{i2}, \ldots, M_{iI}$.

For a job J let $Q(J)$ and $D(J)$ be the quantity of the ordered product P_j and the job deadline, respectively.

The machines in the manufacturing system will produce p different known ***products.*** Each product is made up of a set of ***sub-assembly(ies)*** and/or ***part(s)*** called ***components***. A ***part*** is a component which can not be separated.

The component is assembled in an order given by the ***assembling precedence*** constraints. Each component is manufactured by a set of ***operations*** called the ***process-sequence***, performed in an order determined by the ***machining precedence*** constraints.

For example, the product 'P' in figure 1 is made up of sub-assemblies p_5 and p_6. Sub-assembly p_5 is made up of parts p_1 and p_2. Part p_1 is manufactured by the operations o_{11}, o_{12}, o_{13} and o_{14} by following the machining precedence constraints $o_{11} \rightarrow o_{12} \rightarrow o_{13} \rightarrow o_{14}$. This means o_{11} must be completed before o_{12} etc. Parts p_1 and p_2 must be completed before the operations on component p_5 can be initiated.

2.3 The Bidding Scheme

In our approach the task to meet the job deadlines is pursued through a distributed bidding mechanism. Upon arrival of a job, the Scheduling Department *SD* requests bids for completing the job by its deadline D(J), and selects one of them based on their submitted bids. Any cell, if requested to submit a bid for a delivery of a subproduct, either would be able to complete the needed operations completely, or it would itself request bids for the

missing operations or subproduct deliveries. In the latter case it would need the bids in order to compute its own bid. Cell C_i (or *SD*) requesting a bid from cell C_j is called the ***bidding predecessor cell*** of C_j. By similar reasoning, cell C_j is the ***bidding successor cell*** of C_i. On submitting a bid, a successor cell ***commits*** the offered operations, which are the requested sub-job. A sub-job is said to be ***scheduled*** if one of the *committed* cells has been selected and notified.

Bidding may lead to a chain of predecessor-successor cells in which each cell knows only about its bidding predecessor cell at the previous level, or successor cell at the next level.

Successor cell C_j submits its bid to the bidding predecessor cell C_i. The bidding predecessor cell C_i selects the cell with minimum makespan for commitment. C_i in turn submits its bid to its own bidding predecessor cell at the next level. Finally, a commitment path with minimum makespan is selected for scheduling. It should be noted that *the propagation of bidding is opposite to the flow of production operations (Backward propagation scheme).* Several feasible commitment paths are possible for the same job when searched by different cells. In this context, for an operation sequence $O_1 \rightarrow O_2$ (which means that operation O_1 must be completed before operation O_2), the cell C_2 *committing the operation* O_2 is called the *bidding predecessor of cell* C_1 *committing the operation* O_1. At this early stage the following assumptions are made for the algorithm:

1. There are no production delays due to machine, computer or network failures. (Machine failures are considered in chapter 4.)
2. A cell is said to be capable if it has the ability to process at least the last operation of the sub-job in the production flow with respect to the special properties of the component in work. For each product P the assembling and machining precedence structure (production structure) is uniquely defined. To facilitate the bidding procedure, it is assumed that every cell has prior knowledge of operation capabilities of all potential bidding successor cells. Therefore, a cell needs to request bids for a sub-job only from the capable cells.
3. Every message will be correctly delivered to its destination, within a well bounded amount of time.
4. Messages sent from one cell to another are received in the same order in which they are sent.
5. Each cell is always connected with all other cells, network failures are not considered.
6. A bidding time interval defines a deadline by which a requesting cell must receive the bids. After this deadline it is considered useless to start the requested production.

2.4 Algorithm Description (Algorithm I)

When a cell C_j receives the REQUEST-BID-CELL(C_i, P_i, D_i) message for the job from the bidding predecessor cell C_i , cell C_j commits the operations which could be performed on machines within the cell. It selects the machines, based on the latest start time to meet the deadline. It is conceivable that C_j receives two messages REQUEST-BID-CELL(C_i, P_i, D_i) and REQUEST-BID-CELL(C_i', P_i', D_i') for the subproducts P_i and P_i' but where the requests refer to the same job and subproducts $P_i = P_i'$. Each cell keeps track of the commitments by the unique customer job number, subassembly number and part number. Therefore, in the mentioned situation of overlapping requests for the same job and subproducts (but possibly different deadlines), time slots would be scheduled independently since at most one of the submitted bids would be selected.

Case i) If all operations of the part/sub-assembly P_j could be performed in C_j in due time, cell C_j computes the latest start time T_j for the operations P_j. It then multicasts a REQUEST-BID-CELL(C_j, P_j, D_j) message to the capable cells for each of the components P_j. Here D_j is the new deadline which is equal to T_j. If there are no components left to be requested, C_j submits a bid to its bidding predecessor cell C_j by sending a message SUBMIT-BID-CELL(C_j, P_j ,T_j) with the latest start time T_j.

Case ii) If not all operations of the part/sub-assembly could be performed in C_j in due time, cell C_j multicasts a REQUEST-BID-CELL(C_j, P_j, D_j) message to the capable cells for completing the operations on manufacturing P_j.

Case iii) If none of the operations of the job could be performed in C_j - this could happen only due to timing problems - then the cell C_j stops and submits no bid.

On receiving SUBMIT-BID-CELL(C_j, P_i, T_j) messages within the bidding time-interval, cell C_i selects the best bid based on the latest start time T_j. It sends a message RELEASE-CELL(C_i, P_i) to the unsuccessful bidders. Cell C_i will send a message SUBMIT-BID-CELL(C_i, P_i, T_j) to the bidding predecessor cell, which again repeats this step, until the bid is received by the SD.

If the cell C_i receives no SUBMIT-BID-CELL message, a time-out occurs. Cell C_i then undoes the machine commitments for the job. It submits no bid and stops.

SD finally selects the best bid based on the latest start time. SD sends a message SCHEDULE-CELL(SD, P_{sd}) to the cell whose bid was selected. A message RELEASE-CELL(SD, P_{sd}) is sent to the cells with the rejected bids. If SD receives no bid within the bidding time interval, scheduling terminates with failure.

Cell C_i, on receiving a message RELEASE-CELL, deletes the commitments for the corresponding sub-job. It then sends a message RELEASE-CELL to its successor cells. This contin-

ues, until there are no successors.

Cell C_i, on receiving a message SCHEDULE-CELL, schedules the machines for the corresponding sub-job. It then sends a message SCHEDULE-CELL to its successor cells. This continues, until there are no successors.

3. Distributed Cell/AGV-Scheduling: Algorithm II

3.1 Modification of the Model

In the previous section the manufacturing system consisted solely of flexible manufacturing cells. The assumption on the transportation subsystem was that the transportation was available at constant cost whenever needed. In this section it is assumed that a limited number of AGVs are responsible for transportation of parts or subassemblies from one cell to another. AGVs operate in an autonomous mode. Their control system and the manufacturing cells are inter-connected through a local area network to form a loosely coupled distributed system. *In this paper we assume that AGVs are always operational.*

Problem: Given a real-time cellular manufacturing system together with m AGVs $A_1, A_2, \ldots, A_m$. Through the cooperation of cell-controllers and autonomous AGV-controllers, a schedule for cells and AGVs is to be determined such that the deadlines of the arriving jobs will be met. Cooperation among the cells and AGVs is achieved by a bidding scheme as discussed below.

3.1 The Bidding Scheme

A bidding successor cell C_j receives a request for a bid from a bidding predecessor cell C_j (see section 2.3) for delivering a needed quantity of components. It attempts to determine the earliest start time for producing the requested components. For doing so, it first requests bids from all AGVs for transporting components from C_j to C_j. After selecting a bid from the AGVs, that allows the latest start time for C_j, C_j submits its own bid to C_i. If selected by C_i, C_j commits the requested operation in case that it can produce the needed component on its own. If it cannot commit the operation completely, it requests further bids for the missing machine operations. As a result of our particular method this structure contains exactly one path for each part, from its generation or production until the final product to which it belongs. This path allows for the latest start time to complete the part in time (**Just-In-Time principle**). This will lead to a bidding chain of commitments by cells and AGVs, constituting a feasible commitment path structure for the given job. Several feasible commitment paths are possible for the same job as searched by different capable cells. Finally, a commitment path with minimum makespan is selected for scheduling

3.1 Algorithm Description

In addition to the scheduling rules as described in section 2.4 for including AGV scheduling we formulate the following rules:

When cell C_j receives the message REQUEST-BID-CELL(C_i, P_i, D_i) for the job from the predecessor cell C_i, the cell C_j multicasts a REQUEST-BID-AGV(C_j, C_i, P_i, D_i) to the AGVs, requesting for transportation from the source cell C_j to the destination cell C_i, to be completed by the transportation deadline D_i.

When AGV v receives a message REQUEST-BID-AGV from cell C_j, v commits the transportation in a manner that meets the deadline D_i and its start time T_v is latest. The bid is submitted to the requesting cell C_j by a message SUBMIT-BID-AGV(A_v, P_i, T_v). If the AGV cannot commit any time slot, then it will not submit any bid. If the received bid request is for the transportation of a part, for which it had already been committed on behalf of another cell, then the previously committed time slot is ignored for calculation the new bid

Upon receiving the SUBMIT-BID-AGV messages from the AGVs within the bidding time-interval, cell C_j will select the bid with the latest start time T_l. The deadline D_j for completing the cell operation in C_j is set to T_l. A message RELEASE-AGV(C_j, P_i) is sent to each AGV whose bid was not approved.

If cell C_j does not receive any SUBMIT-BID-AGV message from AGVs before the timeout, then it stops and submits no bid.

Upon receiving a message RELEASE-CELL, cell C_j sends a message RELEASE-AGV(C_i, P_i) to the AGV that had committed to do the transportation for C_j.

Upon receiving the message SCHEDULE-CELL, cell C_i sends a message SCHEDULE-AGV(C_i, P_i) to the AGV that had committed the transportation for C_j.

Upon AGV receiving a message RELEASE-AGV(C_j, P_j), the addressed AGV will delete the committed time slot for the transportation job.

Upon AGV receiving a message SCHEDULE-AGV(C_j, P_j) the addressed AGV will schedule the committed time slot for the transportation.

4. Distributed Rescheduling: Algorithm III

In a manufacturing system one distinguishes between *faults* and *failures*. The main difference between a fault and a failure lies in their frequency of occurrence and repair time. Faults are caused, for example, by misfeeding of a part, breaking of a tool, an improper insertion of a tool etc. Failures in term are often caused by wear and tear, as in a motor bearing. In this paper we do not consider faults.

In this section, an algorithm is presented for dynamic rescheduling machines and AGVs, in the presence of machine failures so that the products

can still be manufactured in time. We will call it ***algorithm III.*** The major idea for this extension will be presented subsequently.

4.1 Problem Description and Solution Idea

For a system of loosely coupled cells and AGVs we consider a more realistic system where machine failures could occur. In case of a failure a machine needs repair. Machine failures and repair will occur in a stochastic fashion. The probability of more than one machine failing at the same time is assumed to be zero [AKEL90]. The repair time is assumed to last between 1 and 7 hours. Repair times of less than an hour are considered as part of normal machine maintenance. (Cell controllers, AGVs and the network are assumed to be always operational.) In case of a machine failure the production of parts or sub-assemblies may be discontinued or at least are in danger to miss their deadlines. In order to meet this challenge we introduced an additional time interval (slack) to each part/subassembly production time (about 5-15% of the scheduled production time). So if D_i was the deadline for a particular part production in cell C_i and P_i its production time, then the latest start time is S_i for the production in C_i - which will be transmitted as the deadline in bidding request - will be adjusted by a slack Δ_i such that $S_i=D_i-P_i-\Delta_i$.

In case of a machine failure in cell C_i, this cell tries to reschedule the involved operations by assigning them to idle machines. If that is not possible even under the new slack C_i requests bids for part or all of the machine operations in question from neighbor cells. If a timely schedule could not be found in this way C_i notifies its bidding predecessor C_k about the failure. C_k would in turn try to reinitiate the bidding procedure for the services it expected from C_i. If a chain of failure messages would build up until it reaches the capable cell for the final product (see definition 2.3), this cell would again try to reinitiate the production job. If unsuccessful it would discard the job. In terms of rules the rescheduling part of algorithm III is indicated in section 4.2.

4.2 Rule Description for Rescheduling (Configuration Example)

If a machine fails in cell C_j at a certain time, then cell C_j becomes responsible for rescheduling of the parts scheduled at this time.

Let operations (O1→O2→O3→O4) be scheduled in cell C_j. Here, O2 is the successor of O1, and so on. Deadline for completion of O4 is equal to the start time of the AGV for transportation of the part to the next cell. Let ***machine m fail at the time of scheduled operation O2***. Here, O1 has already been completed. The local scheduler starts rescheduling. For local rescheduling to be successful, the local scheduler must be able to reschedule O2, O3, and O4 within the cell to finish by its deadline. Cell C_j selects the machine which can start the operation O2 earliest. Similarly, machines for operations O3 and O4 are selected based on the earliest start time and after completion of the previous operation. If finish-time of O4 is before its deadline, then local rescheduling is successful. Otherwise, it starts global rescheduling by requesting bids from capable cells for the job, by multicasting the message REQUEST-BID-CELL(C_j, P_j, D_{jt}) to all the capable cells.

When rescheduling successor cell C_k receives the REQUEST-BID-CELL message for the operations (O2→O3→O4), cell C_k multicasts the REQUEST-BID-AGV(C_j, C_k, P_k) to all the AGVs requesting for transportation of part P_k, from the cell C_j to the cell C_k, by the earliest start time. A bid is submitted to the requesting cell C_k, by a message SUBMIT-BID-AGV(A_v, P_k, T_v), where T_v is the earliest start time of the AGV. Cell C_k commits operation O2 after time $T_v + TT_v$ (TT is the transportation time) such that the start time is earliest. It requests bids from other cells for the operations (O3→O4). After cell C_k receives all the bids, it selects the bid which can finish the operations earliest. It submits the selected bid to the rescheduling predecessor cell C_j.

Upon receiving bids from the cells, cell C_j selects the bid which can finish the operations earliest. If the selected bid meets the deadline then rescheduling is successful and cell C_j stops. Otherwise, cell C_j sends a message UNABLE-TO-SCHEDULE(C_j, P_i, T_f) to its bidding predecessor cell C_i where T_f is the new arrival time of the part P_i at the cell C_i.

Upon receiving UNABLE-TO-SCHEDULE(C_j, P_i, T_f) message, the bidding predecessor cell C_i first checks, whether the new arrival-time T_f can be accepted or not. If the additional time slack is taken into account, it is possible that T_f is early enough (T_f - D_{jt} < slack$_i$(P) for the component P in cell C_i) to recover from the failure by a local rescheduling in cell C_i. In this case, no further global rescheduling is necessary. If the duration is too large the bidding predecessor cell C_i performs local rescheduling as in step 2 to make up for the delayed arrival (T_f-D_{jt}) from the bidding successor cell C_j. If operations are rescheduled before deadline D_{jt}, then further rescheduling is not needed, otherwise it starts rescheduling. This continues until rescheduling is successful, or until there is no bidding predecessor cell.

If there is no bidding predecessor cell and bidding is not yet successful then the algorithm discards the job.

5. Simulation Study

Simulation experiments have been performed for evaluating the distributed algorithm described, in the simple practical context of furniture manufacturing. Three similar product types are chosen for scheduling in three different shop floor configurations. Three product types ensure some production

variety in the system, thus justifying the need of a flexible manufacturing system.

Component	Quantity	Operations for Production							
		Oper. A		Oper B.		Oper. C		Oper. D	
		Name	Time	Name	Time	Name	Time	Name	Time
Leg	2	Cut	45	Bend1	30	Bend2	35	Drill	152
Connect-Leg	2	Cut	45	Drill	152				
Arm	2	Cut	45	Bend1	25	Drill	152		
Connect-Arm	3	Cut	45	Drill	152				
Seat	1	Drill	152						
Support-back	1	Drill	152						
Sub-Assy-1	1	Screw	160	Weld	240	Grind	85		
Sub-Assy-2	1	Screw	160	Weld	200	Grind	75		
Sub-Assy-3	1	Screw	240	Weld	200	Grind	75	Paint	200
Steel Chair	1	Screw	320						

Table 1: Materials and Production Operations for Steel Chair

Some details of the production process as well as the model for the simulation experiments are provided.

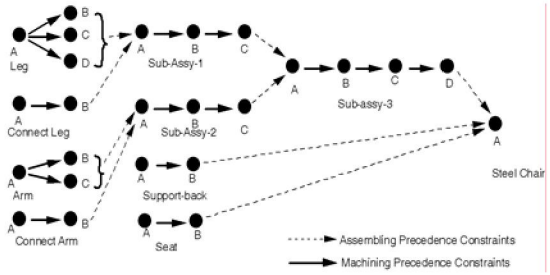

Figure 2: Materials and Production Operations for Steel Chair

5.1 Product Model Description

The following requirements were derived from the real furniture production process for the selection of three furniture or *product types*:

1. Each product is produced in three different sizes, called *product size.*
2. Each product is produced by machining and assembly processes in a discrete manufacturing environment.
3. The average batch size of each job is 40 pieces.
4. Each job is produced within a lead time of 2 to 4 weeks.
5. The total number of parts and sub-assemblies for each product type is less than 1.

The selected product types were ***steel table***, ***steel chair*** and ***bed frame***. Table 1 shows the bill of material and the operation times of operations for the steel chair production as an example. Figures 2 and 3 show the assembly and machining constraints for the steel chair.

5.2 Shop Floor Model Description

Three different shop floor configurations S_1, S_2, and S_3 of cells and machines were selected. They are shown in table 2. The following context was derived from shop floor models:

1. The assembly is either manual or done by robots.
2. There are at least two machines in the shop floor capable of doing any given operation.
3. The maximum number of machines in each cell is six.
4. The machines are running one shift of 8 hours/day.

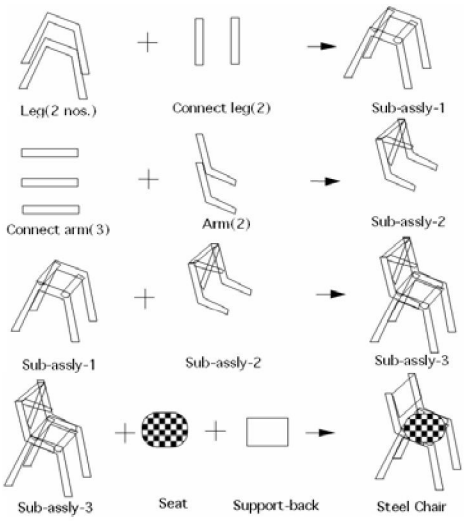

Figure 3: Steel Chair Assembly Sequence

5.3 Simulation Experiments

The set of jobs requested by the customers per day is defined as *job set*. Its quantity is called *job set quantity*. The subset of the *job set quantity* that could be successfully scheduled is called the *guaranteed job set quantity*. For each shop floor configuration, three different batch size patterns are selected as shown in table 3.

For example, in a simulation run with medium batch size variation, the batch size will be 20, 30, 40, 50, or 60. For the simulation study a *job generator* module generates five customer-requested jobs (as 1 *job-set)* per day. For each job, it generates the product type from the 3 different types, the product size from three sizes, and the batch size. The deadline value of the job is chosen between minimum and maximum time values, by uniformly distributed random number generation. The conventions in table 4 are used.

5.4 Results

Algorithm I. From the beginning of section 3 we recall that transportation for parts from one cell to another one is assumed to be available whenever needed, at constant costs.

9 different types of simulation runs were performed for accommodating the 3 shop configurations and 3 batch size variations. Each simulation is run over a 50-day time period. The performance of the algorithm is evaluated by looking at the pattern of the *guaranteed job set quantity,* relative to the *job set quantity* for each of the simulation runs. In figures 4, 5 and 6 the graphs show for each day the job set quantity and the guaranteed job set quantity as scheduled by the algorithm, each for one of

the three shop floor configurations and batch size variations. *(Note that the graphs do not show the quantity actually produced on any given day. They show only the quantity scheduled on the particular day.)*

Cell Name	Shop Name	Machine Name	Operations	Mean failure time
c1	S1, S2, S3	bc-1	bending, cutting	80
		bc-2	bending, cutting	140
		pr-1	press, rivet	90
		pr-2	press, rivet	110
		d-1	drill	80
		d-2	drill	160
c2	S1, S2, S3	g-1	grind	140
		g-2	grind	160
		s-1	screw	90
		s-2	screw	180
		w-1	weld	190
		w-2	weld	100
c3	S1, S2, S3	p-1	paint	100
		p-2	paint	170
c4	S2	bc-3	bending, cutting	140
		pr-3	press, rivet	110
		d-3	drill	130
		d-4	drill	120
c5	S3	g-3	grind	140
		g-4	grind	110
		s-3	screw	110
		s-4	screw	150
		w-3	weld	130
		w-4	weld	120
c6	S3	bc-4	bending, cutting	160
		bc-5	bending, cutting	90
		pr-3	press, rivet	120
		pr-4	press, rivet	80
		d-5	drill	140
		d-6	drill	130

Table 2: Shop Floor Configuration

The requested job quantity is equal to the guaranteed job set quantity for the first seven days. Hence the graphs coincide. This is because during the initial stages of the simulation, all the machines are free.

After seven days, the unavailability of time slots on machines causes quite a few jobs to fail.

Batch size Variation	Batch size Min	Batch size Max	Batch sizes permitted
Small	30	50	30, 40, 50
Medium	20	60	20,30,40,50,60
Large	10	70	10,20,30,40,50,60,70

Table 3: Batch Sizes of Jobs

Minium_time_of_deadline	= Current_time + (Lead_Time * Batch_Size)
Maximum_time_of_deadline	= Minimum_time_of_deadline + Maximum_Laxity
Lead_Time	= Average lead-time for the batch size = 1 (production time)
Maximum_Laxity	= 20 days (considered equal to the lead time of 20 days for a product with average batch size)

Table 4: Simulation Conventions

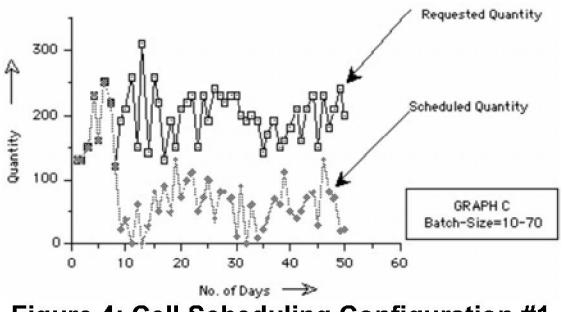

Figure 4: Cell Scheduling Configuration #1

It can also be seen that during simulation with medium batch size variation more jobs than with small batch size variation can be guaranteed. This is because the smaller size jobs could readily fit into the small free slots of the machine schedules. By the same token, the large batch size variation allows a still larger number of jobs to be scheduled in due time.

For the shop floor configurations S_2 and S_3, with their increasing number of cells and machines, the

requested job set quantity was equal to the guaranteed job set quantity even for the first eight to fifteen days, compared to seven days in cell configuration S_1 (see fig 5 and 6). Also otherwise the increasingly better basis for bidding results in a clearly better performance of Algorithm I. Using the data in figures 4, 5, and 6 the mean and standard deviation for each simulation run are calculated in table 5.

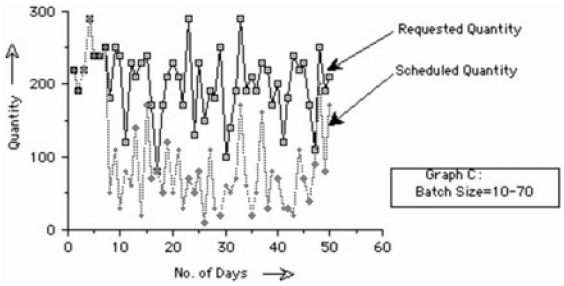

Figure 5: Scheduling Configuration #2

To compare the mean deviation of the simulation runs, a significance test was performed. Table 6 shows the computed test values given by the test. For all cases in table 6 the test statistic is greater than 1.67. Also the confidence interval does not include zero. Hence, we can say with 90% confidence that the mean quantity scheduled increases in shop configuration S_2 compared to S_1. Also, the mean quantity of scheduled jobs increases again in shop configuration S_3, compared to S_2.

Algorithm II. The performance of the algorithm was evaluated in the shop floor configuration S_2, with 1, 2 or 3 AGVs. (S_2 can be found in table 2. It consists of the cells c_1, c_2, c_3, c_4.) For every day the guaranteed job set quantity was compared to the job set of that day. The time of transportation between any pair of cells which includes loading and unloading, is assumed to be known (within the range of 12 to 15 minutes).

Shop-Config	Graph	Sample Size	Mean	Standard Deviation
S_1	A	40	51.8	39.5
	B	40	59.7	48.0
	C	40	59.3	34.1
S_2	A	40	76.3	40.9
	B	40	78.5	45.2
	C	40	75.3	48.5
S_3	A	40	136.0	50.3
	B	40	137.8	52.9
	C	40	143.3	58.9

Table 5: Mean and Standard Deviation

The medium batch size variation is assumed for the arriving jobs (see table 3). The simulation runs were performed over a 50 day time period. The quantity which could not be scheduled to meet the deadline was discarded.

Detailed plots of all simulation results can be found at http://ls3-www.cs.uni-dortmund.de/ComTrans. The results are discussed below.

With one AGV in the system, the requested quantity was equal to the guaranteed job quantity on the first day. This is so since during the initial stage of the simulation, all the time slots of the machines and (the only) AGV are free. With two AGVs in the system, the requested quantity was equal to the guaranteed job quantity for the first two days while with three AGVs in the System, this optimal scheduling result lasted even for the first five days. The reason is that the availability of additional AGVs in the system provided a growing number of empty time slots for transportation.

We computed the mean and standard deviation of the simulation runs as shown in table 7. The test statistic is 1.67 at the 10% significance level (or 90 % confidence interval). The test statistic in table 8 for two and three AGVs is less than 2.3 > 1.67. Also the confidence interval does not include zero. Hence, we can say with 90% confidence that the mean quantity scheduled increases with two AGVs relative to just one AGV.

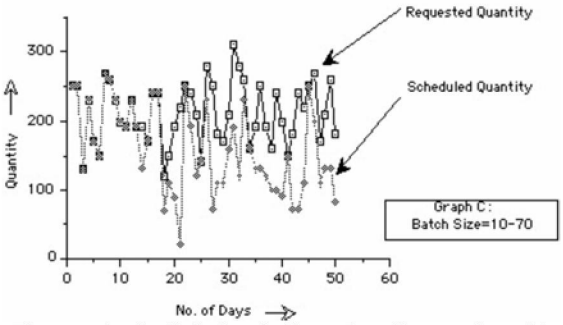

Figure 6: Cell Scheduling Configuration #3

With two AGVs, the higher transportation availability results in fewer deadline failures. With two and three AGVs, the test statistic is 1.0 < 1.67. Also the confidence interval includes zero. Hence, we can say with 90% confidence that the mean quantity scheduled does not increase with three AGVs relative to two AGVs. In this configuration the machines in the shop floor are not able to produce enough parts to keep all three AGVs busy.

Shopfloor (Graph)	Combined S.D.	Test Statistic	Confidence Interval
$S_{1(A)}$ and $S_{2(A)}$	40.2	2.7	24.5 ± 15.1
$S_{1(B)}$ and $S_{2(B)}$	46.6	1.8	18.8 ± 17.5
$S_{1(C)}$ and $S_{2(C)}$	42.1	1.7	16 ± 15.8
$S_{2(A)}$ and $S_{3(A)}$	45.8	5.76	59.7 ± 17.2
$S_{2(B)}$ and $S_{3(B)}$	49.2	5.32	59.3 ± 18.5
$S_{2(C)}$ and $S_{3(C)}$	54.0	5.56	65 ± 20.3

Table 6: Simulation at 90% Confidence Interval

Shop-Config.	# of AGVs	Sample Size	Mean	Standard Deviation
S_2	1	40	29.5	27.6
	2	40	48.0	43.0
	3	40	57.0	40.5

Table 7: Mean and Standard Deviations

# of AGVs	Combined S.D.	Test Statistic	Confidence Interval (90%)
1 and 2	36.1	2.3	18.5 ± 13.6
2 and 3	46.6	1.0	9.0 ± 15.7

Table 8: Simulation at 90% Confidence Interval

Algorithm III. A module for machine failure generation has been added. This module generates exponentially distributed random machine failure patterns in the system. The mean working time of the machines in the system is set to vary between 80 and 200 hours. The mean time for an operator to repair a machine which experiences a failure was between 1 and 7 hours. The module first generates the mean time of failure for a machine by the uniform distribution. This mean value was used to generate the exponentially distributed random repair time.

For higher flexibility of cell scheduling under machine failures a slack of 10% was added (see section 4.1). If no job was scheduled for production during a machine breakdown no production delay was to be experienced. Hence no rescheduling was done.

Cell configuration S_3 of table 2 with two AGVs was selected for the simulation, and the medium batch size variation was assumed for the arriving jobs. The simulation was run three times, each time for the duration of fifty days.

A different seed number was used in each simulation run so that different machine failure patterns were generated each time.

Figures that depict the deadline failures per day can be found at http://ls3-www.cs.uni-dortmund.de/ComTrans.

In figure 9, the total quantity which Algorithm III was able to reschedule successfully was plotted.

Using the data of the graphs in figure 9 the mean and standard deviation of the rescheduled quantities in each simulation run were calculated in table 9.

From table 10, using 68 degrees of freedom, the test statistic is given as 1.67 at the 10% significance level (or 90 % confidence interval). The test statistic in table 10 for all the graphs is less than 0.64. Also the confidence interval includes zero in all the cases. Hence, we can say with 90% confidence that the mean quantity rescheduled is the same in all simulation runs, thus confirming that the simulation runs are representatively significant.

5. Conclusion

In this paper we presented novel distributed algorithms for production scheduling in an automated manufacturing system. We incrementally extended a new distributed cell scheduling algorithm by integrating, through model refinement, Automated-Guided-Vehicle (AGV) scheduling and finally,

through adding slack times for cell operation and transportation deadlines, by adaptive distributed rescheduling features for handling transient machine failures, thus providing for fault tolerance capacity.

Even for the singular cell scheduling problem as dealt with in section 2, there had not been any distributed algorithm in the literature to handle both machining and sub-assembly operations. The further integration of AGV scheduling and adaptive rescheduling in case of transient machine failures are our original contributions.

We performed preliminary simulation studies with a real manufacturing example (furniture production) and realistic data. The results show how with increased replication of machines the real-time performance improves considerably.

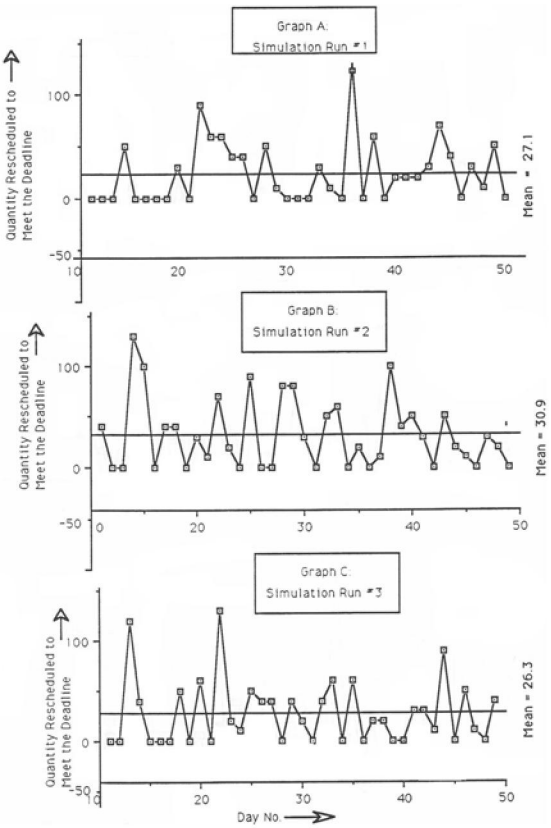

Figure 9: Rescheduled Quantity

Shop-Config.	Graph	Sample Size	Mean	Standard Deviation
S_3	A	35	27.1	30.8
	B	35	30.9	31.6
	C	35	26.3	30.1

Table 9: Mean and Standard Deviation

Graphs	Combined S.D.	Test Statistic	Confidence Interval (90%)
A and B	31.2	0.63	4.8 ± 12.6
B and C	30.9	0.61	4.6 ± 12.4
A and C	30.5	0.11	0.1 ± 12.3

Table 10: Simulation at 90% Confidence Interval

The *flexibility* of the underlying bidding technique is further demonstrated by the improved performance under higher variation of batch sizes since the algorithm obviously takes good advantage of smaller batches to be scheduled as smaller time slots will be immediately utilized. One of the open questions we are currently investigating is the impact of the frequency of variation of batch size over time, on the real-time behavior of the system.

In the experiments with Algorithm II, we found a direct performance dependency on the number of AGVs involved until the cell throughput arrived at a limit while the available AGV capacity was not exhausted. When comparing Algorithm III with its rescheduling features to Algorithm II under different failure injection patterns, the performance was throughout found highly superior, due to the rescheduling techniques used in Algorithm III.

In our current research we are working on the problem how to minimize penalties for missed job deadlines, a conceptual refinement of the deadline studies which has been given considerable attention, however, in case of centralized control only. As a problem of a very high practical impact, a major body of work is now being devoted to adaptively tailoring the degree of replication of machines or cells, according to ***local*** needs and costs. This is beyond the scope of this paper.

References

[AKEL90] Akela R., Krogh B.H., and Singh M.R., "Efficient Computation of Coordinating Controls in Hierarchical Structures for Failure-Prone Multi-Cell Flexible Assembly Systems", *IEEE Transactions on Robotics and Automation,* vol. 6, No. 6, Dec 1990.

[DILT91] Dilts D.M., Boyd N.P. and Whorms H.H., "The Evolution of Control Architectures for Automated Manufacturing Systems", *Journal of Manufacturing Systems,* vol. 10, no. 1, pp. 79-93, 1991.

[JONE86] Jones A. T., and McLean C.R., "A Proposed Hierarchical Control Model for Automated Manufacturing Systems", *Journal of Manufacturing Systems,* vol. 5, no. 1, pp. 15-25, 1986.

[LRP01] Leitao, P.. ; Restivo, F. ; Putnik, G. : "A Multi-Agent Based Cell Controller". In *Proceedings of the 8th IEEE Int'l Conference on Emerging Technologies and Factory Automation*, Antibes-Juan les Pins, France, October 15-18, 2001.

[SKS+05] Shankaran, N. ; Koutsoukos, X. ; Schmidt, D. ; Gokhale, A. : "Evaluating Adaptive Resource Management for Distributed Real-Time Embedded Systems". In *Proc. of the Real-time and Embedded Systems Workshop*, Arlington, Virginia 2005.

[SMH+04] W.D. Scott, D.A. Maxwell, S.J. Hsieh, J.S. Smith, and C.O. Malave, "A flexible control system for flexible manufacturing systems", *Thesis and Dissertations*, Texas A&M University, 30-Sep-2004.

[TAN93] Taneja S.K., "Distributed Computing in Computer-Integrated-Manufacturing Systems", *Ph. D. Dissertation*, Wayne State University, Detroit, Michigan, 1993.