Proceedings of the 2007 Winter Simulation Conference
S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, eds.

DDDAS-BASED MULTI-FIDELITY SIMULATION FOR ONLINE PREVENTIVE MAINTENANCE SCHEDULING IN SEMICONDUCTOR SUPPLY CHAIN

Nurcin Koyuncu
Seungho Lee
Karthik K. Vasudevan
Young-Jun Son

Department of Systems and Industrial Engineering
The University of Arizona
Tucson, AZ 85721, U.S.A.

Parag Sarfare

Department of Computer Science
The University of Arizona
Tucson, AZ 85721, U.S.A.

ABSTRACT

This research intends to augment the validity of simulation models in the most economic way using the DDDAS (Dynamic Data Driven Application Systems) paradigm. Implementation of DDDAS requires automated switching of model fidelity and incorporating selective, dynamic data into the executing simulation model. Comprehensive system architecture and methodologies are proposed, where the components include 1) RT (Real Time) DDDAS simulation, 2) grid computing modules, 3) Web Service communication server, 4) database, 5) various sensors, and 6) real system. Four algorithms are developed to facilitate integration of the various components in the DDDAS system. They are 1) data filtering algorithm using control charts, 2) preliminary fidelity selection algorithm using Bayesian belief network, 3) fidelity assignment algorithm using integer programming and 4) simulation model reconstruction algorithm using multiple linear regression. A prototype DDDAS simulation was successfully implemented for preventive maintenance scheduling in a semiconductor supply chain. The initial results look quite promising.

1 INTRODUCTION

In today's global markets, supply chain systems are both large scale and dynamic in nature (Fox, Barbuceanu, and Teigen 2000). In these large scale, dynamic supply chains, data overload across partners is a common problem owing to the various factors affecting system performance and lack of understanding of the relationship between them. Discrete event simulation has become one of the most used analysis tools for large scale systems as it can take randomness into account and address very detailed models. Given the enormous amount of data that exist in today's supply chains, data needs to be used and updated wisely in the model (simulation model in this research) in order to prevent unnecessary usage of computing resources, which are severely constrained across the supply chain. This concern can be resolved by implementing multiple models, having different fidelities for different echelons of the supply chain, or even for different areas within same echelon of the supply chain. However, use of different models for different applications prove to be both cumbersome and infeasible from the viewpoint of enterprise as well as supply chain integration. There is considerable ongoing research regarding integration of supply chain models. One example is a conceptual framework proposed by Jain (2006). Another issue is within each of those multiple models, it is not completely practical to change the current fidelity level once it is set. Hence, when there is a need for detailed information (e.g. in case of an emergency), the level of fidelity cannot be increased. In the same way, when there is no need for in depth data regarding the supply chain the fidelity level cannot be decreased to save from computational resource consumption. Adaptive simulations, on the other hand, are able to represent the actual process of real supply chain system in the most accurate manner while enabling 1) efficient usage of computational resource 2) access to all of the necessary data regarding the supply chain and 3) superior enterprise integration in terms of communication and information synchronization.

Several researchers have investigated the impact of simulation modeling fidelity on performance of supply chain systems. Venkateswaran and Son (2004) investigated the effect of modeling fidelity on the analysis of supply chain dynamics. It was found that the approximations used in modeling, such as delays and capacity, have more impact on the outcome of supply chain analysis than end customer demand. Similarly, Towill (1991) concluded that the most effective strategy that can be applied to the simulation models is improving the flow of new information at all levels throughout the chain.

Prior research conducted in this field served as a motivation for this research in this paper. To solve the above elucidated problems, this paper employs a novel methodology, DDDAS to enable integration of real time dynamic single simulation model through facility and within multiple distributed multi-fidelity simulations through supply chain as a whole. In this research, we implemented multi fidelity simulation by changing the number of data and data acquisition frequency. Also we developed a web service to communicate between each echelon of supply chain and DDDAS simulation.

Many researchers from diverse domains have spent considerable effort in a collaborative work environment in order to successfully propose and develop the above discussed DDDAS simulation for supply chain systems. Anjos et al. (2007) has successfully revealed the optimized preventive maintenance scheduling in a single facility with the incorporation of DDDAS architecture. This research extends the work by Anjos et al. (2007) in the sense that we aim to optimize preventive maintenance scheduling in the whole supply chain within a distributed simulation environment (see Section 3.1 for details). To the best of our knowledge, our research is the first effort to 1) construct a formal comprehensive infrastructure for DDDAS in supply chain systems, 2) handle the dynamicity issue of the system by selectively incorporating up-to-date information into the distributed simulation, 3) introduce adaptive multi-fidelity simulations which are capable of adjusting the fidelity level according to the altering conditions of supply chain in the most economic way. Moreover, this study makes use of simulations for making instantaneous decisions in addition to the analysis purposes. The broad gamut of contributions of this work is demonstrated through the implementation of DDDAS for optimization of preventive maintenance (PM) scheduling within a semiconductor supply chain. The proposed DDDAS methodologies are extensible not only to other areas of the supply chain system but also to a multitude of domains where there is a need to handle a large volume of highly dynamic data.

The rest of the paper is organized keeping in mind readability and easy understanding of the various methodologies proposed as a part of DDDAS. Section 2 describes the prototype supply chain under consideration in this paper to help understand and implement the methods proposed in Sections 3 and 4 which describe the detailed system architecture along with all the techniques and methods proposed to implement the DDDAS. Section 5 describes the experiments and results that prove the usefulness of our methods. Finally, Section 6 concludes the paper by providing a summary of the contributions made through this paper.

2 CONSIDERED SEMICONDUCTOR MANUFACTURING SUPPLY CHAIN

Supply chain systems are defined as a collection of business units that interact with one another to transform the raw materials into finished goods and distribute the fin-data into the simulation to manage and optimize available resources through automatic fidelity switching within a ished goods to the customers (Ganeshan and Harrison 1995; Venkateswaran and Son 2004). Supply chain systems can be of two types: 1) collaborative or 2) competitive. Collaborative supply chain systems are usually comprised of echelons which belong to the same company, and a lot of information is shared among the echelons. On the other hand, the competitive supply chain systems are composed of echelons which in general belong to different companies. In collaborative supply chain systems simulation, as the level of the fidelity increases, the sensor data accessed from the other echelons of the supply chain increases as well without any limitation on access to data (Figures 1a and 1b) whereas in the competitive one, access to the required sensor data is restricted by the regulations of the other competitors (Figure 2). Another significant difference is, in collaborative supply chain systems the computational resources from the partners are pooled and then by help of grid computing distributed among the elements as needed, where as in the competitive supply chain computational resource is being shared only within particular parties. Hence, in terms of managing the computational resources, a single resource manager is adequate for all of the partners of the collaborative supply chain while distinct resource managers are needed for each group of parties that share the resources in the competitive supply chain systems.

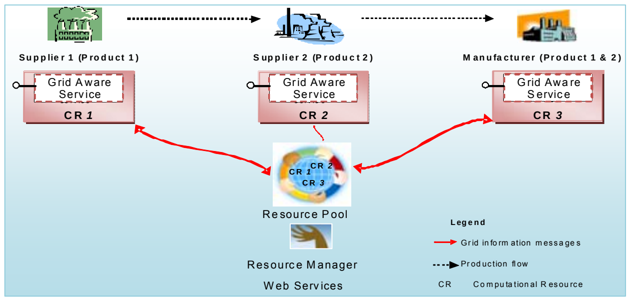


Figure 1a: Collaborative supply chain system 1 (Common resource pool)

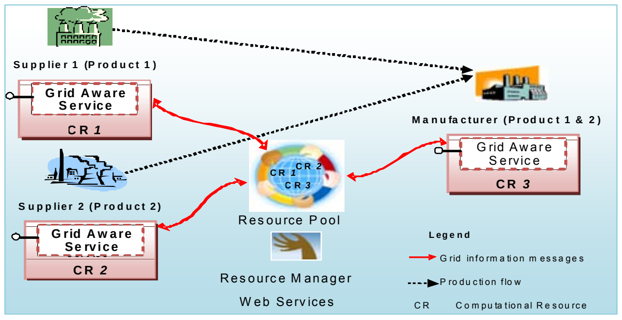


Figure 1b: Collaborative supply chain system 2 (Common resource pool)

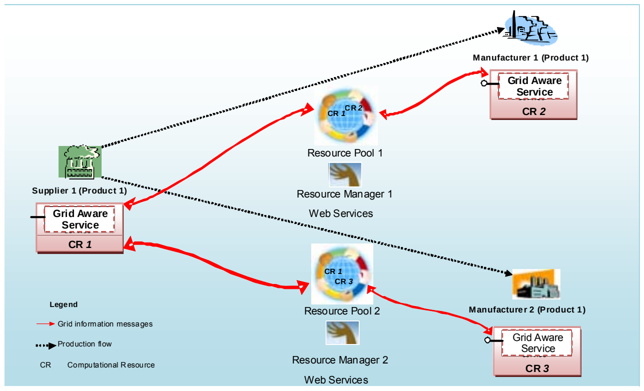


Figure 2: Competitive supply chain system (Multiple resource pools)

The applicability of DDDAS to a supply chain is demonstrated in this study via collaborative Semiconductor Supply Chain systems. However, it should be noted that the DDDAS methodologies such as techniques and algorithms developed here are applicable to any component or echelon of a supply chain (collaborative or competitive). The semiconductor manufacturing supply chain considered in this work begins at a wafer fabrication unit (fab.), where the raw silicon wafers are formed. The wafers are then sent to the semiconductor die fabs consisting of five major areas to develop multiple layers of circuits on the silicon wafer (diffusion, photo, etch, metals, probe). The silicon wafers with the circuits are cut into individual chips called dies. The dies are transferred to an assembly and packaging fab to package into Integrated Circuits.

PM schedules establish specific time frames for machine repairs in order to avoid unscheduled breakdowns. Unscheduled breakdowns (caused by machine failure) cost significantly more than scheduled machine down. An effective PM schedule can therefore greatly reduce operational costs of a facility. This way, PM is scheduled with the objective of minimizing maintenance and associated costs for all equipment over time. This interdependence between maintenance and costs leads to complex interactions between the other components in the supply chain and the machine states.

To demonstrate the proposed DDDAS system, the DDDAS based preventive maintenance scheme is built for each partner in the supply chain, where each partner is composed of 3 groups of machines and each machine utilizes 15 sensors of various types. Current PM schedules are built from the historical data and followed over prescribed periods of time (generally weeks). However in the dynamically changing manufacturing environment, static PM schedule derived from the historical data is not efficient and flexible. Thus, a dynamic generation of an optimized PM schedule that uses the most current data in real time is necessary. This would go a long way in contributing to better utilization of facility machinery and better fulfillment of customer orders with acceptable lead times.

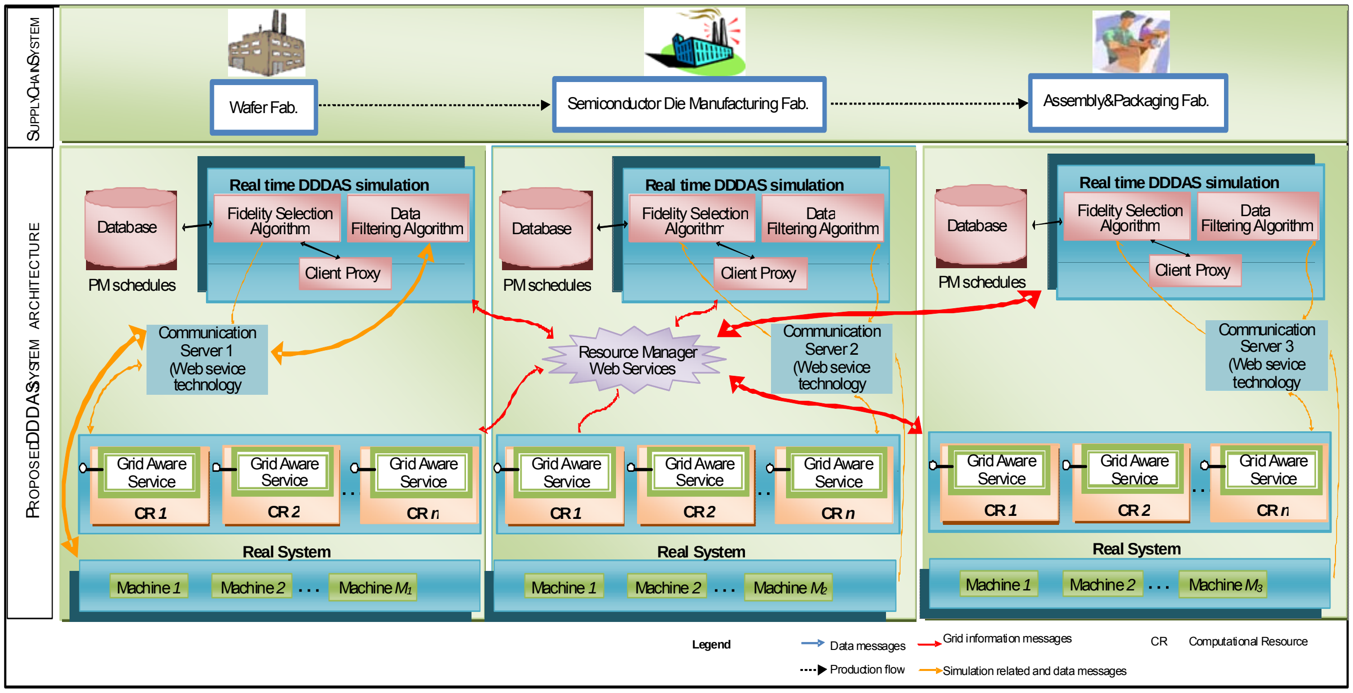

SUPPLY CHAIN SYSTEM
PROPOSED DDDAS SYSTEM ARCHITECTURE
Wafer Fab.
Semiconductor Die Manufacturing Fab.
Assembly&Packaging Fab.
Real time DDDAS simulation
Fidelity Selection Algorithm
Data Filtering Algorithm
Client Proxy
Database
PM schedules
Communication Server 1 (Web sevice technology)
Communication Server 2 (Web sevice technology)
Communication Server 3 (Web sevice technology)
Resource Manager Web Services
Grid Aware Service
CR 1
CR 2
CR n
Real System
Machine 1
Machine 2
Machine M1
Machine M2
Machine M3
Legend
Data messages
Grid information messages
CR Computational Resource
Production flow
Simulation related and data messages

Figure 3: Proposed DDDAS system architecture in semiconductor manufacturing supply chain

3 PROPOSED DDDAS SYSTEM ARCHITECTURE AND ALGORITHMS

3.1 Proposed System Architecture

The proposed architecture of the DDDAS system for supply chain systems is shown in Figure 3. It consists of real (physical) systems, Real Time DDDAS simulations (RT DDDAS), a grid computing, communication servers (Web service), and databases.

Sensor data from each machine of the real system are sent to RT DDDAS simulation via the communication server. Table 1 depicts the various sensor data which are believed to affect the quality of wafer (e.g., high temperatures can cause the over-baking of the die on a wafer), and thus relevant for maintenance scheduling. The algorithms in the RT DDDAS simulations then use these sensor data in combination with database information to generate the updated PM schedules. These updates are sent via the communication server back to both the real systems (for the implementation through automated control) and databases. This process is an ongoing, continuous loop while the simulation is running.

Table 1: Considered sensor data (maintenance focus)

Sensors from machine	Sensors from environment	Production schedule from archived data
temperature	temperature	type of process
pressure	sound	type of material
vibration	air quality	operator skill level
status	humidity	past/present/future product flow volumes
sound		time since last maintenance

The communication server in Figure 3 has been developed using web services technology (state-of-the-art distributed computing technology) which overcomes barriers of standard communication via the usage of W3C (`http://www.w3c.org`) standard protocols (e.g., XML, WSDL, and SOAP) and facilitates message passing between disparate and geographically distant components. The grid computing technology is also employed to manage the distributed computational resources (e.g. evaluation of the computational availability), where WSRF.net toolkit (following the Open Source Groupware Architecture standards) is used as a backbone grid computing architecture. The individual resource pools are managed by a resource management web service that acts as a proxy, single point resource manager. All simulations that are in need of computational resources use the resource management web service for querying their availability and consuming them. The whole collaborative supply chain environment considered in this work shares a single grid computing infrastructure, while a competitive supply chain environment might have a single or multiple distinct infrastructures per fabrication group depending on the willingness of the competitors to share their resources for potential, mutual performance improvement.

3.2 Enabling Algorithms in RT DDDAS Simulation

The goal of the DDDAS simulation is to achieve the effective synchronization of time and information between the simulation and real-system. This synchronization will then let the simulation run with the most up-to-date data (see Figure 4). In this study, in order to enable an effective time synchronization, DDDAS simulations are implemented via real-time simulations. A considerable amount of computational resource is consumed by simulation due to its execution as well as sensor data update to reach such an accurately synchronized system. Therefore, DDDAS simulations should dynamically adjust their levels of fidelity to assure information synchronization within a predetermined time interval (see δt in Figure 4) to save from computing resources without missing any significant measurement. It is noted that a level of fidelity affects both the simulation model execution time as well as the time taken to collect required sensory updates. At the beginning of each interval (δt), new measurements are requested for the machines, and updated data are to be obtained within δt.

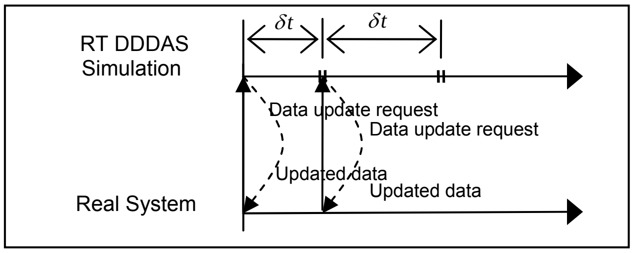


Figure 4: RT DDDAS simulation and real system

In this work, the four algorithms developed and embedded into the simulation to enable the DDDAS capability are represented in Figure 5 and discussed in the next sections.

3.2.1 Measurement Data Filtering (Algorithm 1) using Control Charts

Each time a new measurement is taken, this algorithm is executed to filter the sensor data. Its two major functions are to filter out noisy measurements and to identify machine malfunctions by means of X-bar and moving range control charts. The algorithm establishes control limits by constantly calculating a moving average of the recent data points, and is able to adapt to changing machine conditions (see Figure 6).

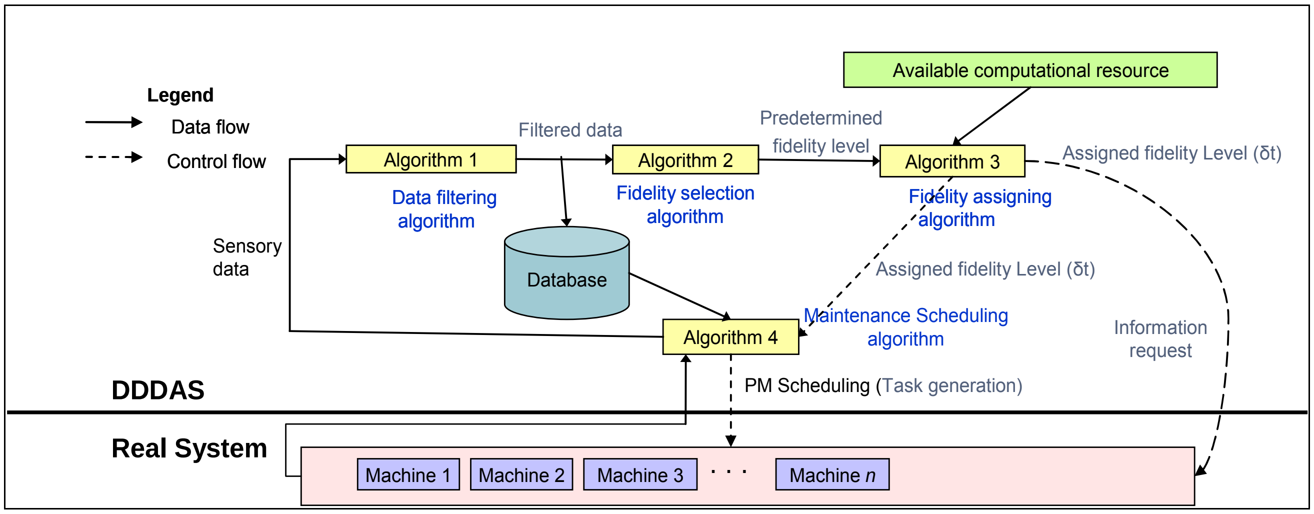

Legend
Data flow
Control flow
Available computational resource
Filtered data
Predetermined fidelity level
Algorithm 1
Algorithm 2
Algorithm 3
Assigned fidelity Level (δt)
Data filtering algorithm
Fidelity selection algorithm
Fidelity assigning algorithm
Sensory data
Database
Assigned fidelity Level (δt)
Maintenance Scheduling algorithm
Information request
Algorithm 4
DDDAS
PM Scheduling (Task generation)
Real System
Machine 1
Machine 2
Machine 3
. . .
Machine *n*

Figure 5: Overview of algorithms enabling DDDAS simulation capability

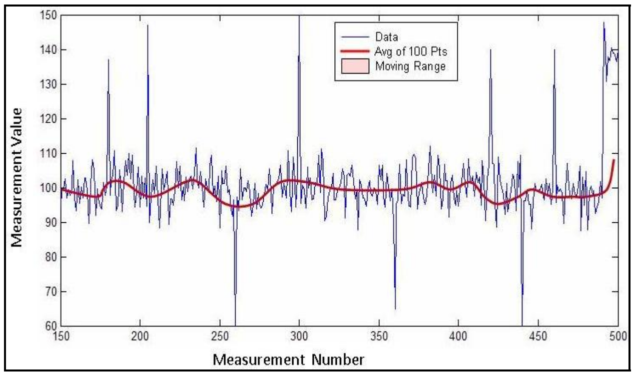


Figure 6: Algorithm 1 – moving average

Algorithm 1 sends an alarm message if a string of out-of-range values is received. Thus, Algorithm 1 is able to alert the machine operator of changes in machine conditions that could necessitate a machine shutdown, avoiding potential machine or product damages. However, the control limits used to assess incoming data are not calculated until at least one hundred data points are received, so a short training period is required to establish proper control limits. Up until that time, only basic checks against out-of-range values are performed on the input data. Algorithm 1 is called with a timestamp and data point, and returns a single filtered data value. It also stores past data points in the database, which is constantly updated at the rate of which new measurements are taken. This algorithm is written in C++ and plugged into the Arena model (simulation software used in this work) via Visual Basic Application (VBA).

3.2.2 Simulation Model Fidelity Selection (Algorithm 2) using Bayesian Belief Network

Algorithm 2 is designed to select the proper fidelity level of each machine in the simulation through the Bayesian Belief Network (BBN). The BBN is a virtuous technique to embody the parameters interactions via a directed acyclic graph. The nodes represent variables and the arcs represent statistical dependence relations of conditional probability distributions for these variables given the values of their parents based on the Bayes' theorem. The BBN provides valuable analysis, even when some data are uncertain, in which possible parental causes for an observed event can still be traced. For these reasons, the BBN is employed for fidelity selection. Algorithm 2 takes the filtered data as input from Algorithm 1 and outputs a matrix which captures the proper fidelity level for each machine. Considering a specific machine, if the measurement of a parameter goes beyond the threshold values, Algorithm 2 determines the most likely interactions which might have caused this result and accordingly selects the parameters needed to be measured in more detail.

If, for instance, the measurement of a temperature (Temp1) goes beyond the threshold value (assuming the current fidelity is the lowest), Algorithm 2 will infer, based on historical data (captured in BBN) which were the causes of such high values (Vib2, Temp2, Pres2) and determine a proper fidelity level involving a new set of data to be monitored (see Figure 7 and Table 2).

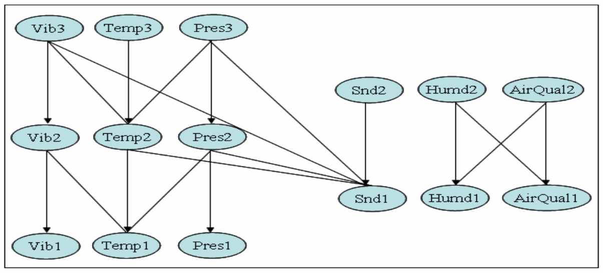
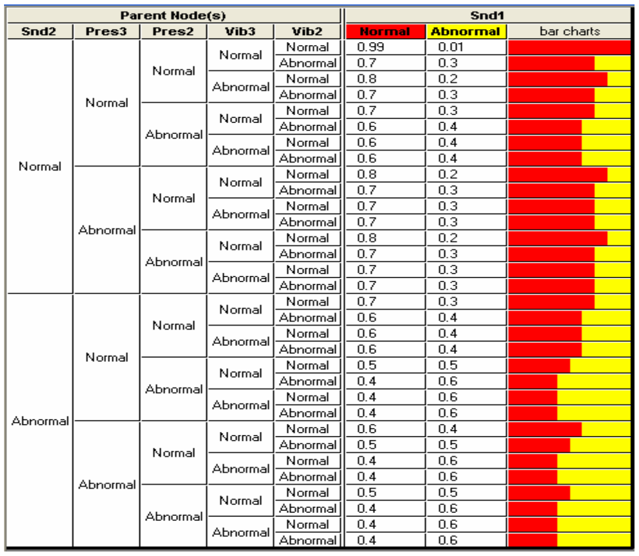


Figure 7: BBN for simulation fidelity selection

Table 2: Hierarchical structure for the conditional probabilities of "sound"



Parent Node(s)					Snd1		
Snd2	**Pres3**	**Pres2**	**Vib3**	**Vib2**	**Normal**	**Abnormal**	bar charts
Normal	Normal	Normal	Normal	Normal	0.99	0.01	
				Abnormal	0.7	0.3	
			Abnormal	Normal	0.8	0.2	
				Abnormal	0.7	0.3	
		Abnormal	Normal	Normal	0.7	0.3	
				Abnormal	0.6	0.4	
			Abnormal	Normal	0.6	0.4	
				Abnormal	0.6	0.4	
	Abnormal	Normal	Normal	Normal	0.8	0.2	
				Abnormal	0.7	0.3	
			Abnormal	Normal	0.7	0.3	
				Abnormal	0.7	0.3	
		Abnormal	Normal	Normal	0.8	0.2	
				Abnormal	0.7	0.3	
			Abnormal	Normal	0.7	0.3	
				Abnormal	0.7	0.3	
Abnormal	Normal	Normal	Normal	Normal	0.7	0.3	
				Abnormal	0.6	0.4	
			Abnormal	Normal	0.6	0.4	
				Abnormal	0.6	0.4	
		Abnormal	Normal	Normal	0.5	0.5	
				Abnormal	0.4	0.6	
			Abnormal	Normal	0.4	0.6	
				Abnormal	0.4	0.6	
	Abnormal	Normal	Normal	Normal	0.6	0.4	
				Abnormal	0.5	0.5	
			Abnormal	Normal	0.4	0.6	
				Abnormal	0.4	0.6	
		Abnormal	Normal	Normal	0.5	0.5	
				Abnormal	0.4	0.6	
			Abnormal	Normal	0.4	0.6	
				Abnormal	0.4	0.6	

In this implementation for PM, data for Fidelity Level 1, is collected from 6 sensors namely, Temp1, Vib1, Pres1, Snd1, Humd1, AirQual1. Each sensor data can be in either one of the two states – normal or abnormal. If data shows that the value of the sensor parameter is beyond the threshold value, the state of the sensor data is classified as abnormal. Once the state of any sensor changes, the probability of the abnormal state of a parent node will be changed by Bayes' theorem. If any of the sensors has a probability greater than 0.4 (abnormal), then the node is decided to be critical and hence needs more detailed observation. If said sensor is not included in the current fidelity level, then the fidelity level is increased. For instance, the probability that Pres2 data is abnormal when the sensor data Temp1, Vib1, Pres1, Snd1 are normal is 0.01, i.e., P(Pres2=Abnormal | Vib1 = Normal, Temp1 = Normal, Pres1 = Normal, Snd1 = Normal) = 0.01. However, if Pres1 is also abnormal, then the probability increases to 0.20, i.e., P(Pres2=Abnormal | Vib1 = Normal, Temp1 = Normal, Pres1 = Abnormal, Snd1 = Normal) = 0.20. Furthermore, if Snd1 is abnormal as well, then this probability increases to 0.78, i.e., P(Pres2=Abnormal | Vib1 = Normal, Temp1 = Normal, Pres1 = Abnormal, Snd1 = Abnormal) = 0.78. Consequently, the fidelity level for this machine is changed to Fidelity Level 2 since the probability is greater than 0.4 (see Table 2). Such a BBN and rule based system can be developed for any application by modifying the variables and parameters considered appropriately.

3.2.3 Simulation Model Fidelity Assignment (Algorithm 3) using Integer Programming

Algorithm 3 is designed to opt for the available fidelity level of each machine by taking the computational resource constraint into account. This algorithm obtains a matrix which encapsulates the proper (desirable) fidelity level of each machine from Algorithm 2 as well as the available computational resource capacity from Grid Computing as inputs and returns a new matrix which captures the assigned fidelity level for each machine as an output. This algorithm is based on the well-known Knapsack problem. Equation (1) represents the formulation of the problem.

$$\begin{aligned} \min \quad & \sum_{i\in S} C_i(R_i - RA_i) \\ \text{s.t.} \quad & \sum_{i\in S} RA_i \le TR \end{aligned} \qquad (1)$$

In this equation, S is the set of machines that require resource, C_i is the penalty cost if the resource requirement for machine i is not met, TR are the available resources, R_i is the resource requirement for machine i and RA_i (the decision variable) is the resource assigned to machine i. The problem minimizes the penalty cost resulting from not fulfilling the computational resource needs of each machine. Here, we are unable to increase the computational resource on hand. Therefore, to minimize the total cost we first assign the resources to the machines of those having greater penalty costs. To solve this problem, a widely known heuristic method for the knapsack problem, namely, the greedy algorithm (Martello and Toth, 1990) is used because it is more efficient in terms of execution time.

3.2.4 Dynamic Simulation Model Reconstruction (Algorithm 4) using Multiple Linear Regression

This algorithm is used to produce the optimal overall PM interval and, when combined with the known length of time since the last servicing, provides recommendations on when the next maintenance operation should take place. The DDDAS simulation uses a formula to calculate the PM interval from a linear combination of the sensor inputs with variable weights assigned to each inputs. The formula is given in Equation (2), where Y_i is the optimal PM interval for period i, X_{ij} is the sensor j measurement for maintenance period i, and W_j is the weight assigned to sensor j measurement.

$$Y_t = \sum_{j=1}^{n} W_j + \sum_{j=1}^{n} W_j X_{tj} \qquad (2)$$

As Figure 8 shows, Algorithm 4 accepts three inputs: 1) the time since the last machine maintenance, 2) the mean values for each sensor parameter during that period, and 3) feedback estimates by the maintenance team of the optimal preventive maintenance period. The algorithm uses multiple linear regressions to fit the formula to the given data set, and returns the weights of the terms in the PM interval formula that provide the best fit. Initially, the formula uses weights derived as best estimates from historical data. As the simulation evolves, more data measurements and PM feedback is used to produce a formula that provides predictions of the best possible PM interval for the current maintenance period. Using multiple linear regressions, the weights in the desired formula can be estimated by solving the equation $W= (X^TX)^{-1} (X^TY)$, where X is a matrix of the sensor measurements, Y is the optimal interval estimates, and W is the vector of formula weights. Once these weights are known, they can be used along with current sensor measurement data to predict the optimal PM interval for the current period. The predictions using this method of estimation are completely reliant on the accuracy of the feedback estimates from the maintenance personnel, and an assumption is those estimates have a zero mean difference from the true values.

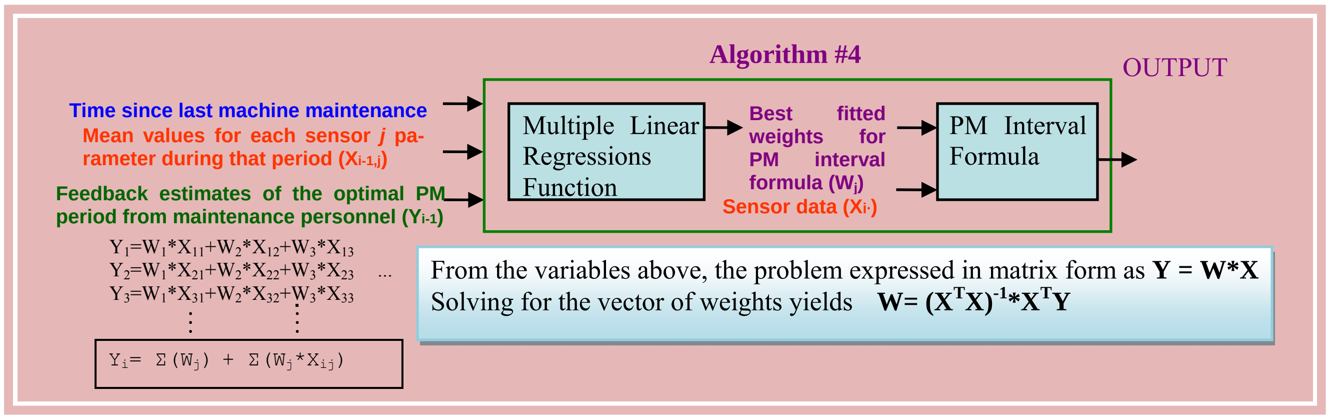
4 DDDAS SIMULATION IMPLEMENTATION

In this work, we employed Arena 10.0 to build a distributed simulation consisting of three RT DDDAS simulations. Each of the RT DDDAS was used to control the preventive maintenance scheduling in an echelon of the supply chain system. Although the RT DDDAS simulation can directly work on data from a real supply chain, for demonstration purposes each of the echelons (the real system) was simulated instead of these real systems. The communication server which was built using web services, was used to enable communication across this distributed system. The real systems of each of the simulations exchange messages with the order quantity data and expected lead times. This data is then transmitted to the RT DDDAS simulations. The order quantity is used to create entities in the create block of successive simulations to mimic product flow in the supply chain system.

The RT DDDAS simulation is split into two distinct sub-models. The algorithms sub model (ASM) and the process sub model (PSM). The ASM is a loop that reads in data from the real system based on an update time determined by the fidelity of each component of the real system. This is a looped sub model with a timing entity (agent) to execute each algorithm and a delay block having a delay time of δt as specified by the preceding algorithms. Once δt has elapsed, the agent repeats its operation with different parameters. The interaction between the real system and RT DDDAS is modeled in this ASM.

The output from the ASM (fidelity of each component) is used by the PSM to route entities into the correct fidelity level. The PSM is essentially composed of Process blocks, VBA blocks and a Decide block. The decide block is used to switch between the available fidelities. Within the PSM's process blocks we employ a formula to calculate the delay of a particular process based on sensor inputs. This is essential in order for the PSM to mimic the cycle times in the real system and in turn for the RT DDDAS simulation to synchronize with the real system.

The algorithms in the ASM loop are executed in sequence: Algorithm 1 populates the sensor machine-id array, which is used by Algorithm 2 & 3 which in turn populates the fidelity machine-id array and also δ*t* machine-id array. Algorithm 4 then uses these arrays to update the preventive maintenance schedule (in the generic case this will be any suitable parameter of choice). Algorithm 4 also

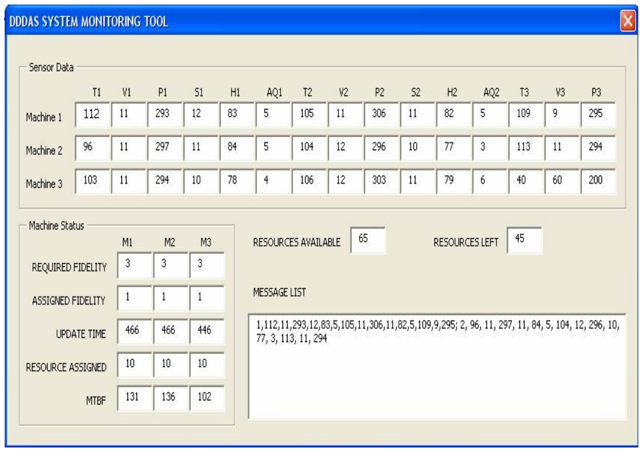

Algorithm #4
OUTPUT
Time since last machine maintenance
Mean values for each sensor j parameter during that period ($X_{i-1,j}$)
Feedback estimates of the optimal PM period from maintenance personnel (Y_{i-1})
Multiple Linear Regressions Function
Best fitted weights for PM interval formula (W_j)
Sensor data ($X_{i\cdot}$)
PM Interval Formula
$Y_1=W_1*X_{11}+W_2*X_{12}+W_3*X_{13}$
$Y_2=W_1*X_{21}+W_2*X_{22}+W_3*X_{23}$ ...
$Y_3=W_1*X_{31}+W_2*X_{32}+W_3*X_{33}$
$Y_i= \Sigma(W_j) + \Sigma(W_j*X_{ij})$

From the variables above, the problem expressed in matrix form as **Y = W*X**
Solving for the vector of weights yields **$W= (X^TX)^{-1}*X^TY$**

Figure 8: Details of algorithm 4 (dynamic simulation model construction algorithm for real-time scheduling)

requests the data from the real system's data sources for the next loop of the ASM. Once the new data is received, the loop is re-executed after a δ*t* time period.

In this implementation, we use a minimum of 6 sensors in the lowest fidelity and 15 sensors in the highest fidelity for each machine (e.g. temperature, pressure, air humidity). Fidelity affects both number of data acquired and the frequency of data collection. In this model, highest fidelity level requires collecting 15 data in every 10 seconds. In the real system, sensitive data from the sensors would affect processing times. This is modeled in the RT DDDAS simulation using a formula to calculate processing times (in real time). The number of sensory data also affects the ASM as it influences the number of data to process and the linear regression equation employed in Algorithm 4. CR are allocated to collect sensory data from each machine based on the number of requests made by RT DDDAS simulation. Each of the sensory data collected, CR used and MTBF calculated are reported through a monitoring tool shown in Figure 9.



DDDAS SYSTEM MONITORING TOOL

Sensor Data

	T1	V1	P1	S1	H1	AQ1	T2	V2	P2	S2	H2	AQ2	T3	V3	P3
Machine 1	112	11	293	12	83	5	105	11	306	11	82	5	109	9	295
Machine 2	96	11	297	11	84	5	104	12	296	10	77	3	113	11	294
Machine 3	103	11	294	10	78	4	106	12	303	11	79	6	40	60	200

Machine Status

	M1	M2	M3
REQUIRED FIDELITY	3	3	3
ASSIGNED FIDELITY	1	1	1
UPDATE TIME	466	466	446
RESOURCE ASSIGNED	10	10	10
MTBF	131	136	102

RESOURCES AVAILABLE 65

RESOURCES LEFT 45

MESSAGE LIST

```
1,112,11,293,12,83,5,105,11,306,11,82,5,109,9,295; 2, 96, 11, 297, 11, 84, 5, 104, 12, 296, 10,
77, 3, 113, 11, 294
```

Figure 9: An instance of the results of the algorithms in Arena model

5 EXPERIMENT AND RESULTS

The supply chain under consideration is modeled and simulated in two ways – 1) the conventional maintenance schedules (static) and 2) RT DDDAS maintenance schedules. A comparison of production lead times between the two systems is then developed. Because the RT DDDAS indicates the necessity of machine maintenance only when the sensors shows so, the PM schedule gets closer to a maximum machine utilization. This way, unexpected machine breakdowns, unnecessary maintenance, and downtime for problem identification are all avoided. Thus, the production lead time is minimized. Figure 10 shows the dynamic fidelity change of RT DDDAS simulation of wafer fab during its execution. The graph shown in Figure 11 depicts the CR usage of wafer fab in RT DDDAS simulation. Since we represent the collaborative supply chain system, each Fab is competing for CR coming from a single resource pool shared by all echelons. Note in the same figure that, the fidelities assigned are in direct variation with the computing resource availability in the system.

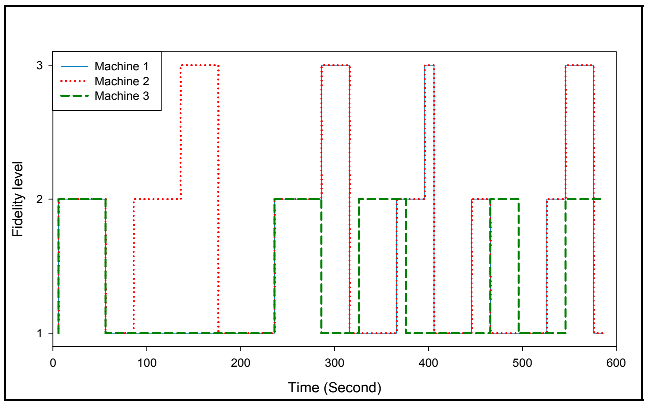


Figure 10: Dynamic change of fidelity level on each machine of wafer fab

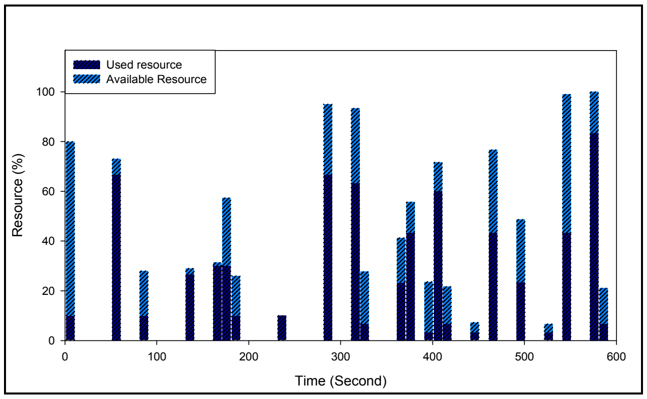
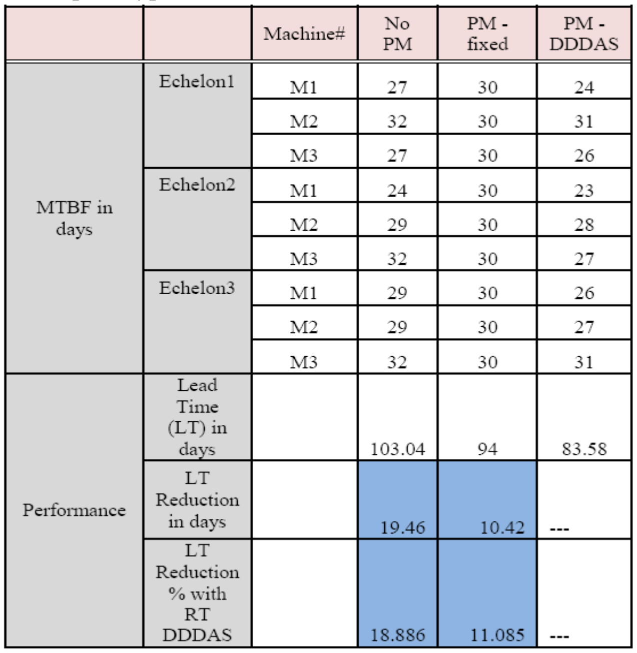


Figure 11: Dynamic change of computing resources and its usage in wafer fab

Table 3 depicts comparison of performance of the PM schedule generated by the RT DDDAS against that of the conventional PM schedule (static) and a system not involving any PM. Results show that the maintenance schedule generated by RT DDDAS allows a 11% reduction in product lead times compared to conventional scheduling and an 18% reduction compared to a system with no PM. Lead times are measured as the difference between the time that final products are shipped to the next station (customer) from the assembly and packaging fab (echelon 3) and the time production gets started in the wafer manufacturing fab (echelon 1). This reduction in lead time translates to gigantic monetary savings. The improvement achieved by DDDAS is attributed to the increased utilization achieved through better PM scheduling.

Table 3: Production lead time reduction via DDDAS system in prototype model



		Machine#	No PM	PM - fixed	PM - DDDAS
MTBF in days	Echelon1	M1	27	30	24
		M2	32	30	31
		M3	27	30	26
	Echelon2	M1	24	30	23
		M2	29	30	28
		M3	32	30	27
	Echelon3	M1	29	30	26
		M2	29	30	27
		M3	32	30	31
Performance	Lead Time (LT) in days		103.04	94	83.58
	LT Reduction in days		19.46	10.42	---
	LT Reduction % with RT DDDAS		18.886	11.085	---

6 DISCUSSION AND CONCLUSION

In this paper, the DDDAS methodology was applied to minimize lead times in a supply chain system by optimizing the PM schedules of individual echelons. The processes in a 3 echelon collaborative, semiconductor supply chain system was considered to demonstrate the proposed DDDAS system. Results relating to the automatic fidelity switching in each echelon and optimized resource allocations were observed and graphed. A lead time reduction of around 11% was achieved using the PM schedule generated by DDDAS. While we focused on the PM for a collaborative supply chain, the generality (and necessary modifications) of the proposed DDDAS system for the competitive supply chain has been discussed as well. Our ongoing research focuses on extending the proposed DDDAS system to various phases of supply chains such as production scheduling and transportation management.

REFERENCES

Anjos, D., D. Cohen, C. Hansen, N. Koyuncu, S. Lee, A. Shendarkar, K. Vasudevan, and Y. Son. 2007. DDDAS-based Multi-fidelity Simulation for Online PM Scheduling. *Proceedings of IERC 2007*.

Fox, M. S., M. Barbuceanu, and R. Teigen. 2000. Agent-oriented supply chain management. *The International Journal of Flexible Manufacturing Systems* 12:165-188.

Ganeshan, R., and T. P. Harrison. 1997. An introduction to supply chain management. Available via <http://lcm.csa.iisc.ernet.in/scm/supply_chain_intro.html> [accessed March 31, 2007].

Jain, S. 2006. A conceptual framework for supply chain modeling and simulation. *International Journal of Simulation and Process Modeling* 2(3/4):164-174.

Martello, S. and P. Toth. 1990. *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley & Sons.

Towill, D. R. 1991. Supply chain dynamics. *International Journal of Computer Integrated Manufacturing* 4(4):197-208.

Venkateswaran, J., K. K. Vasudevan and Y. Son. 2004. Impact of Modeling Approximations in Supply Chain Analysis - An Experimental Study. *International Journal of Production Research* 42(15):2971 - 2992.

AUTHOR BIOGRAPHIES

NURCIN KOYUNCU is a Ph.D. student in the department of Systems and Industrial Engineering at the University of Arizona. Her research focuses on architectural design and application of distributed dynamic data driven systems. She is a student member of IIE and SME. She can be reached by email at <koyuncu@email.arizona.edu>.

SEUNGHO LEE is a Ph.D. student in the department of Systems and Industrial Engineering at the University of Arizona. His research focuses on application of distributed simulation and simulation of human decision making. He is a student member of IIE and INFORMS. He can be reached by email at <mountlee@email.arizona.edu>.

KARTHIK VASUDEVAN is a Master's student in the department of Systems and Industrial Engineering at the University of Arizona. His research focuses on distributed multi-paradigm simulations and applications of virtual reality in manufacturing. He is a student member of IIE. He can be reached by email at <karthikv@email.arizona.edu>.

YOUNG JUN SON is an associate professor in the Department of Systems and Industrial Engineering at The University of Arizona. His research focuses primarily on the application of distributed and hybrid simulation to large scale dynamic systems. He is a recipient of 2004 SME Eugene Merchant Outstanding Young Manufacturing Engineer Award and 2005 IIE Outstanding Young Industrial Engineer Award. He can be reached by email at <son@sie.arizona.edu >.

PARAG SARFARE is a Master's student in the Department of Computer Science at the University of Arizona. His research interests include Operating Systems, Storage Systems and Distributed Computing. He can be reached by email at <parags@cs.arizona.edu>.