An Agent-based Learning Approach for Teaching the Relationship Between Lot Size and Cycle Time

Maged M. Dessouky *
Department of Industrial and Systems Engineering,
University of Southern California,
Los Angeles, CA 90089-0193
maged@usc.edu

Jeff Rickel
USC Information Sciences Institute
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292-6695
rickel@isi.edu

Narayanan Sadagopan
USC Information Sciences Institute
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292-6695
nara@isi.edu

* Corresponding Author

Abstract

Students in most industrial engineering programs are taught just-in-time and other production concepts aimed at reducing cycle time and work-in-process inventory. Much of the material stresses setup time and lot size reduction. However, if the setup times cannot be eliminated or reduced significantly, reducing the lot size in some cases may have an adverse effect on the cycle time because more setup time is incurred. Thus, it is important for students to understand the complex relationship between lot size and cycle time when setups are required. In this paper, we present an agent-based learning approach for teaching this relationship. The agent watches as students experiment with a virtual factory and intervenes opportunistically with questions and explanations.

Introduction

The thrust in today's manufacturing environment is to move towards a Just-in-Time (JIT) manufacturing environment. As outlined by numerous studies (e.g., Schonberger, 1986; Finch, 1993; Voss and Robinson, 1987; Im and Lee, 1989; Mehra and Inman, 1992; and Handfield, 1993), successful implementation of just-in-time principles may lead to reduced inventory costs, improved quality, and increased equipment utilization. Some of the basic tenets of JIT production concepts are setup time reduction or elimination and the running of small lot sizes.

Small lot sizes can reduce cycle time and work-in-process inventory (WIP) if the setup times are not much larger than the unit run times. A reduction in the cycle time allows the manufacturer to respond more quickly to new customer orders or any changes in demand, and increases the likelihood of meeting the demand on time. Small lot sizes tend to reduce the cycle time because a lot spends less time at a machining center, causing new arriving lots to wait less for the machines to become available. However, reducing the lot size too much can sometimes have the opposite effect by increasing the cycle time because machine utilization may increase significantly due to an increase in the total setup times per unit period. Figure 1 shows a sample relationship between the lot size and cycle time for a printed circuit board assembly line (Dessouky, 1998). As the figure shows, an initial increase in the lot size reduces the cycle time due to the reduction in the machine utilization because of reduced setups. However, increasing the lot size past 10 starts to increase the cycle time.

As the above example illustrates, the relationship between lot size and cycle time and the factors that affect this relationship can be complex and difficult for students to understand. In this paper, we present an agent-based learning approach for teaching this relationship. Although

there has been much research on determining the best or optimal lot sizes (e.g., Hill and Raturi, 1992; Karmarkar et al., 1992), there has been little work in developing instructional tools to teach the important relationship between cycle time and lot size. Our intention is not to develop an algorithm to determine the optimal lot size, but to create a learning environment for students to develop a fundamental understanding of the causal interdependences between all the factors (e.g., setup times, processing times, equipment utilization, queue time) that affect this relationship. Starting from a base case, students are expected to iteratively make changes to the lot size, with the agent using the students' experiments as a context for illustrating the various factors.

The primary instructional domain for which our approach is intended is the undergraduate production course commonly found in most industrial engineering curriculums. We note that our approach is applicable to other domains but we focus on the production course in the industrial engineering curriculum for illustrative purposes.

Our agent-based approach is integrated into the Virtual Factory Teaching System (VFTS), a collaborative learning environment that is accessible via the Internet's World Wide Web. At the core of the VFTS lies a Visual SLAM (Pritsker and O'Reilly, 1999) factory simulation model. The VFTS allows students, working alone or in teams, to build factories, forecast demand for products, plan production, establish release rules for new work into the factory, and set scheduling rules for workstations. An animated panel displays jobs progressing through their factory simulation, complete with queue counts, finished goods counts, graphs, and reporting functions (Dessouky et al., 1998, 2001).

The agent's role is to monitor students as they experiment with their virtual factory and

to intervene opportunistically with questions and explanations. The central pedagogical principle behind our approach is that the agent should teach the target relationships in the context of the student's own experiments (de Jong and van Joolingen, 1998). To support such a tutorial style, the agent must be able to recognize learning opportunities in a student's experiments, test the student's understanding, and relate particular simulation results to appropriate domain principles. The general approach of using a computer tutor to monitor a student's problem-solving activities and provide guidance has been shown effective in a wide variety of domains, including botany (Lester et al., 1997), ecology (Meyer et al., 1999), mathematics and programming (Anderson et al., 1995), and physics (Gertner and VanLehn, 2000). Our particular agent is an extension of a domain-independent Automated Lab Instructor (ALI) that has previously been applied to chemistry and biology simulations (D'Souza et al., 2001). Our application of ALI to the VFTS provides further evidence of ALI's generality.

The remainder of the paper is organized as follows. First, we describe our model of the relationship between lot size and cycle time in the context of a job shop manufacturing environment; this model serves as the target knowledge we want students to learn. Next, we describe the basic VFTS learning environment and how we integrated ALI into it, including the software architecture and ALI's representation of the target knowledge. We show how ALI uses its knowledge to help students learn as they experiment with the VFTS. Finally, we discuss our experience using the VFTS and ALI in an undergraduate production course at the University of Southern of California.

Relationship Between Cycle Time and Lot Size

We consider a job shop manufacturing environment that may consist of many different work centers and product types. A work center is a collection of one or more machines with similar capabilities. Each product type has a unique route through the work centers.

The total time to complete processing of a lot on a machine includes both a *setup time per lot* component and a *run time per unit* component for each unit in the lot. A setup is required whenever a changeover to a new product type is made. The setup time is incurred once at the beginning of processing a new lot of different type than the preceding lot. That is, a setup is incurred only if the product type of the arriving lot is different than the preceding lot type. The setup time is independent of the lot size. We assume the lot size for a given product type is the same at all the machines. The run time per unit is incurred for each unit in the lot. That is,

$$T_{ik} = S_{ik} + q_i P_{ik}$$

where:

T_{ik} is the total time to process a lot of type i on machine k

S_{ik} is the setup time of product type i on machine type k

P_{ik} is the processing time per unit of product type i on machine k

q_i is the lot size for product type i

Let the period's demand for product type i be D_i. Then, the number of lots, n_i, needed to meet the period's demand is:

$$n_i = (D_i / q_i)^+$$

The rate of arrival of the n_i lots is specified by the arrival rate, λ_i. A new lot of type i is released to the factory every $1/\lambda_i$ time units until n_i lots are released.

Figure 2 shows our model for describing the relationship between cycle time and lot size. The cycle time is counted from the time a lot is released to the factory until the time it finishes processing at the last work center on its route. The components of the cycle time are:

- Setup time – this is the total time the lot is at all the work centers on its route while the machines are being setup.
- Queue time – this is the total time a lot is waiting at the work centers for an available machine to begin processing.
- Lot Processing time – this is the total time a lot spends being processed by all the machines on its route.
- Transportation time – this is the total time the lot spends being transported from one work center to another. Since we assume the transportation time is independent of the lot size, there is no linkage in our model between transportation time and lot size.

The figure shows the directional linkage between related elements (nodes) in the diagram. For example, there is a positive link between queue time and cycle time since an increase in queue time will increase the cycle time. Table 1 summarizes the relationship between the elements that the agent uses to describe the causal effect.

Table 1. Relationship between the elements.

Element 1	Element 2	Direction of Relationship	Explanation
Lot Size	# of Lots	Negative	The number of lots has an inverse relationship with the

			lot size to be produced. Hence, a bigger lot size will generate fewer lots of the same type to produce.
# of Lots	# of Setups	Positive	If the sequence of lot arrivals to a work center alternates between several product types, it is more likely a setup is required at a work center for each new lot arrival since the number of setups is the number of times a machine has to be prepared to process a new product type. In this case, more lots will increase the number of setups required at a work center.
# of Setups	Machine Utilization	Positive	The utilization is the ratio of busy time to total time available for a particular work center. A machine at a work center is considered busy if it is either in the setup or processing state. Thus, increasing the number of setups at a machine will require more setup time causing higher machine utilization.
Machine Utilization	Queue Time	Positive	Queuing theory states that the higher the utilization of the machine the longer a lot will have to wait in the queue.
# of Lots	Prob. Of Requiring a Setup	Negative	When there are more lots to produce of the same product type, the chance that two lots of the same type are processed back to back at the same machine increases. In this case, no setup is required for the second arriving lot thus reducing its setup time to zero.
Prob. of Requiring a Setup	Setup Time	Positive	When there is more of a chance of requiring a setup, the total time the lots spent in setup increases.
Lot Size	Queue Time	Positive	A large lot size means that a lot waiting in the queue will have to wait longer for the lot being machined to finish processing, thus, increasing the queue time for the waiting lot.
Lot Size	Lot Processing Time	Positive	The total lot processing time is simply the unit processing time multiplied by the lot size. Since there are more units in the lot with increased lot size, the total lot processing time increases with increasing lot size.

From these element relationships, our model consists of four possible paths that explain the linkage between the cycle time and lot size. Table 2 summarizes the four paths and the direction of the relationship between the cycle time and lot size for the path. For example, the path consisting of nodes 1, 8, and 10 has an overall positive direction meaning that if the lot size

increases the cycle time will also increase when all the individual links in the path hold true. We also list the explanation that the agent provides if the path holds true. Note that more than one path may be true for a given change in the lot size.

Table 2. Description of each path that shows the linkage between lot size and cycle time

Path	Nodes	Overall Direction Of the Path	Explanation
1	1,2,6,7, 10	Positive	When cycle time increases as lot size increases, it often means the lot spent more time in the setup state due to the increased probability of requiring a setup with fewer lots.
2	1,2,3,4, 5,10	Negative	When the cycle time decreases as lot size increases, it often means the savings in queue time due to fewer setups dominates the direct effect of longer lot processing times.
3	1,5,10	Positive	When cycle time increases as lot size increases, it often means the increase in the queue time due to lots waiting longer for other lots to finish processing dominates any possible decrease in machine utilization due to less setups (since there are fewer lots to process).
4	1,8,10	Positive	When cycle time increases as lot size increases, it often means the increase in the lot processing time due to the increased lot size dominates any possible decrease in machine utilization due to fewer setups (since there are fewer lots to process).

Integrating ALI into the VFTS

The VFTS, which was developed at the University of Southern California (USC), is used in a senior-level production course at USC and several other universities to illustrate forecasting, planning, and scheduling concepts. It uses the Visual SLAM simulation package (Pritsker and O'Reilly, 1999) as its core. That is, the VFTS provides an interface between a student and a

Visual SLAM factory simulation model running in the background.

The VFTS allows students to create and run their own factories. To create a factory, students set a variety of parameters such as routes, processing times, setup times, demand, due dates, number of machines at each work center, dispatching rules, factory release rule (push versus pull), and lot size, as shown in Figure 3. In this figure, the arrival rate, λ_i , specifies the inverse of the time between arrivals of the lots. As previously mentioned, a new lot of type *i* is released to the factory every $1/\lambda_i$ time units until n_i lots are released. The total number of parts in the figure is equivalent to the variable D_i. The due date factor is the desired ratio of completion time to the sum of the processing times. This factor is used to set the due dates for each lot.

At run time, students simulate the factory and change decision parameters to see their effect on factory performance. Students watch the simulation via an animation window that shows the factory dynamics, including lots moving from machine to machine, queues rising and falling, machines becoming busy or idle, and machines going down for repair, as shown in Figure 4. At the end of a simulation run, students view a variety of summary statistics such as average cycle time and work in process, and they can compare these statistics to those from the previous simulation run to evaluate the effects of their decision changes, as shown in Figure 5.

Readers interested in learning more about the VFTS can refer to the papers by Dessouky et al. (1998, 2001) or access it through our web page at http://vfts.isi.edu/ (specify the group id as "guest-agent", the user id as "guest", and the password as "guest"). An example that illustrates setting up a factory within the VFTS can be found in the tutorial at http://vfts.isi.edu/docs/vfts.htm. The remainder of the paper focuses only on the agent

interaction within the VFTS.

The Agent Interaction window, shown in Figure 6, supports ALI's dialogue with a student. The *Speech Area* displays ALI's comments and explanations. As new text appears in the speech area, old text is moved into the *Speech History* area so that the student always has access to ALI's prior comments for reference. To the right of the speech area is a *Quiz Panel* in which ALI asks multiple-choice questions to test the student's understanding.

Because ALI may introduce terms with which the student is unfamiliar, the student should be able to ask ALI to explain them at any point during the interaction. To the right of the speech history area is a list of terms that the student can select and ask ALI to describe. ALI updates this list automatically whenever it introduces a new term.

Figure 7 shows the software architecture of our learning environment. The student interacts with the VFTS Interface Manager via a graphical user interface (GUI). When a student initiates a new simulation, the interface manager sends the factory model to the simulator (running separately on a server), which broadcasts the factory state at each simulation tick back to the interface manager. Each update from the simulator is displayed to the student via the animation window. Since the simulation typically runs very fast, the animation window allows students to control the speed of these animation updates and even pause the animation in order to give them time to comprehend the dynamics. At the end of the simulation run, the summary statistics are displayed to the student, and they are also passed to ALI. ALI uses these statistics to decide whether to initiate a dialogue with the student.

To provide guidance to students, ALI must know the target knowledge that students are expected to learn. To allow ALI to be integrated easily into new virtual laboratories such as the

VFTS, course authors provide this target knowledge to ALI via a text file using a simple yet general knowledge specification language designed for virtual laboratories (D'Souza et al., 2001). Thus, to get ALI to help students understand the relationship between lot size and cycle time, we had to represent the knowledge from the previous section using this language. We illustrate each type of item in the language with representative examples from our encoding of the VFTS knowledge.

Entities are objects in the simulation that have properties (described below). Names of entities are placed in the "Describe" area of the agent interaction window, and the description is given to students when they select the entity name in that window.

```
Entity {name: product;
   description: An item manufactured by the factory.}
Entity {name: machine;
   description: An entity that processes input materials to make a product.}
Entity {name: workcenter;
   description: A collection of one or more machines with similar capabilities.}
```

Variables in the simulation model are represented as properties of entities. When ALI introduces a new property in its dialogue with a student, it adds the property name to the "Describe" area of the agent interaction window, and the description is given if the student selects the entity name in that window.

```
Property {name: lot size; ofEntity: product;
   description: The size of a batch for a particular product.}
Property {name: number of setups; ofEntity: product;
   description: Number of times a machine was prepared to process a new
```

product type.}

Property {name: utilization; ofEntity: workcenter;
description: Ratio of busy time to total time available for a particular workcenter.}

Property {name: queue time; ofEntity: product;
description: The total time a product spends waiting before being processed.}

Property {name: cycle time; ofEntity: product;
description: Cycle time is counted from the time a lot is released to the factory until the time it finishes processing at the last workcenter on its route.}

Relations represent the primary relationships among simulation variables (properties) that we want students to learn. Thus, for teaching the relationships between lot size and cycle time, we would define one relation for each path in Table 2.

Relation { name: lotSize cycleTime;
description: When the cycle time decreases as lot size increases and vice versa, it often means the savings in queue time due to fewer setups dominates the direct effect of longer lot processing times.;
properties: lot size, cycle time;
explanationList: lotSize numLots, numLots numSetups, numSetups utilization, utilization queueTime, queueTime cycleTime;
type: negative;
isHeuristic: true}

The relation above represents path 2. Each of the other paths in Table 2 is defined as a separate relation similar to the example above. The relation type, negative, specifies that there is

an inverse relation between the two variables (e.g., cycle time decreases when lot size is increased). The relation is marked as a heuristic (in contrast to a mathematical law) because the relation is not guaranteed to hold. ALI will alert students to violations of laws (which could be due to simulation errors or approximations) to prevent them from forming misconceptions, but it assumes that violations of heuristics are to be expected. The chain of relations listed in the explanation list represents the path of lower-level relations that collectively explain this high-level relation, as described in the previous section. Each of the relations in the explanation list is defined similar to the above example, except that explanation relations (called "atomic" relations in ALI) do not have explanation lists. For example, the following atomic relation defines the last link in the explanation chain above.

```
AtomicRelation {name: queueTime cycleTime;
   increaseDescription: Since the time a lot spends waiting in the queue is a
                        component of the cycle time, large queue times will result
                        in higher cycle times.;
   decreaseDescription: Since the time a lot spends waiting in the queue is a
                        component of the cycle time, small queue times will result
                        in lower cycle times.;
   properties: queue time, cycle time;
   type: positive}
```

Note that this relation is not heuristic. That is, cycle time is a function of queue time and other variables defined by other atomic relations; if queue time increases and these other variables are held constant, cycle time is guaranteed to increase. Also, as shown by this example, relations and atomic relations can have separate descriptions for the cases where the

influencing variable (in this case, queue time) increases and decreases. When descriptions for both cases are provided, ALI uses the one that matches the student's actual simulation results; this can result in clearer explanations. If only a single description of the relation is provided, as with the previous example, it is used in both cases.

Once the target knowledge from the previous section is encoded in ALI's knowledge specification language, ALI has all the information it needs to monitor a student's experiments, identify learning opportunities, generate quizzes to test the student's understanding of the factory relationships, and provide explanations. The next section describes how ALI does this.

Interacting with ALI

As ALI monitors a student's experiments, it compares the results of the current simulation against the results of the previous simulation. This allows ALI (and the student) to see the effect of a change (e.g., an increase in lot size). A student that changes a single independent variable at a time (e.g., lot size) can isolate the effect of that variable on the dependent variables (e.g., cycle time). If a student changes multiple independent variables at once, ALI will use the opportunity to provide instruction in the scientific method, and caution the student against such changes. (A parameter controls how often ALI provides such reminders.)

To simplify the discussion, we will assume that ALI's model of the student is in its initial state (i.e., the student has not yet begun experimenting with the simulation). A discussion of the effect of the student model on ALI's behavior is deferred to the end of this section.

Each relation in the knowledge base, together with the atomic relations that explain it,

represents a learning objective. The task of ALI's low-level reasoner is to identify the instances in which a relation is illustrated during the student's experiments. These instances are learning opportunities that ALI exploits to engage the student in a dialogue about the relation. When a simulation ends, the low-level reasoner scans all the relations in its knowledge base. For each relation, it determines whether the relation was satisfied. A relation is satisfied if its two variables changed in a manner consistent with its relation type (positive or negative) and all its explanation relations were also satisfied. For example, the relation between lot size and cycle time in the previous section would be satisfied if the student increased lot size, cycle time decreased, and each of the five atomic relations in its explanation list was similarly satisfied. If so, this simulation run provides a good example for illustrating the relation, so ALI's low-level reasoner sends the high-level reasoner a message notifying it of the specific relation that was illustrated. This causes ALI's high-level reasoner to initiate a dialogue with the student.

The high-level reasoner determines ALI's behavior as a function of the student model (described later) and the knowledge events generated by the low-level reasoner. To ensure that the student noticed this learning opportunity, ALI first generates a quiz. The quiz asks the student if he/she noticed the effect on cycle time that results from the change in lot size. The quiz does not have to be created by the course author; ALI has quiz templates into which the names of appropriate properties are added from the knowledge base to construct the complete quiz. Quizzes are multiple-choice questions of the following form:

> Did you notice what happened to the cycle time of product 1 when the lot size of product 1 was increased?
>
> - Cycle time of product 1 increased
> - Cycle time of product 1 decreased

- Did not notice

If the student answers the question incorrectly or did not notice the change, ALI will suggest that the student try again:

> Why don't you try increasing lot size again, and this time keep a close eye on cycle time?

Otherwise, since this is the first time that the student has encountered this relation, ALI will describe the reason for this observed effect on cycle time. The description is taken from the definition of the relation.

> Very good! When the cycle time decreases as lot size increases and vice versa, it often means the savings in queue time due to fewer setups dominates the direct effect of longer lot processing times. Now let's see why this happens...

After describing the relation, ALI will engage the student in a dialogue about its underlying causes. It traverses each link in the explanation chain for this relation, and asks the student a question about each atomic relation in turn. For our example relation, there are five atomic relations in the explanation chain. Starting with the first one, ALI would pose the following question:

> What do you think happened to the number of lots of product 1 when the lot size of product 1 increased?

- Number of lots of product 1 increased
- Number of lots of product 1 decreased
- No change

At the same time, ALI knows that the term "number of lots" is new and places it in the list of describable elements in case the student needs to have it elaborated. Next, ALI gives the student feedback on his/her answer ("yes" or "no") followed by a textual description of the relation. For example, if the student answered the above quiz incorrectly, ALI would respond as follows:

> No. The number of lots has an inverse relationship with the lot size to be produced. Hence, a bigger lot size will generate fewer lots to produce.

Regardless of the student's answer, ALI will proceed through the remaining relations in the explanation list:

> Try answering this question. What do you think happened to the number of setups of product 1 when the number of lots of product 1 decreased?
>
> - Number of setups of product 1 increased
> - Number of setups of product 1 decreased
> - No change

ALI continues the dialogue in the same manner as it did with the first explanation relation, adding new terms to the list of describable elements. For example, if the student answers the above quiz correctly, ALI would respond as follows:

> Very Good! If the sequence of lot arrivals to a workcenter alternates between several product types, it is more likely a setup is required at a workcenter for each new lot arrival. In this case, fewer lots will decrease the number of setups required at a workcenter.

ALI continues in this manner until the entire explanation chain has been discussed. At

that point, ALI summarizes the relation and its explanation:

> In summary...
>
> 1. An increase in lot size of product 1 decreases the number of lots of product 1.
> 2. A decrease in the number of lots of product 1 decreases the number of setups of product 1.
> 3. A decrease in the number of setups of product 1 decreases the utilization of workcenter 1.
> 4. A decrease in the utilization of workcenter 1 decreases the queue time of product 1.
> 5. A decrease in the queue time of product 1 decreases the cycle time of product 1.
>
> Hence, an increase in lot size of product 1 decreases the cycle time of product 1

ALI keeps a record of the student's presumed knowledge in order to avoid discussing relations that the student has already mastered. This student model also allows ALI to distinguish between new information presented to the student and references to material that has already been discussed; without making this distinction clear to students, they may tune out repeated material and miss new information (Moore, 1996).

The student model is stored as an overlay (Goldstein, 1977) that adds attributes to each relation and atomic relation in the knowledge base. The attributes are represented as simple flags: whether or not the relation has been encountered, whether it has been explained, and whether its quiz was answered correctly. These attributes are serialized out to permanent storage so that they can be maintained across student sessions with the VFTS.

As an example of the effect of the student model on ALI's interaction, we return to our example. When our student encounters the particular relation between lot size and cycle time

again, ALI will check to see if the relation has been understood previously. If it hasn't, then ALI will proceed to go through the explanation chain again. This time, however, ALI will only describe the reason for an atomic relation if the student gets an answer wrong. If ALI described an atomic relation previously, and the student's answer demonstrates a correct understanding of it, there is no reason to repeat the description.

When walking a student through the explanation chain of a relation, ALI may encounter an atomic relation that was already discussed in the context of this or another high-level relation. In such a situation, ALI modifies its text to clearly identify the new discussion as referring back to a prior discussion:

> Do you remember what happens to the number of lots of product 1 when the lot size of product 1 increased?

Experience with the VFTS and ALI

During the fall semester of 2001, students in an undergraduate production course at the University of Southern California used the VFTS and ALI in a large-scale project. Using the VFTS, the project required the students to forecast demand, plan weekly production volume, and develop effective scheduling strategies by determining release and dispatching rules and the appropriate lot size for each product type. In addition to ALI's ability to teach students about the relationship between lot size and cycle time, ALI also included knowledge for teaching them how changes in release rules and dispatching rules affect cycle time.

The project was completed in teams of 2-3 students. Enrollment in the class supported the creation of 14 teams. The student teams were given four weeks to complete the project. In

order to assess the benefit of incorporating ALI into the VFTS, half the teams used a version of the VFTS with ALI and the other half without.

The project was assigned on the latter part of the semester. Thus, most of the forecasting, planning, and scheduling methodologies incorporated within the VFTS were discussed in the classroom. Before the assignment of the project, one class session was used for a demonstration of the VFTS system.

An empirical comparison between the teams that used ALI and those that did not showed no difference in the average project grades of the two groups. This does not necessarily imply the agent did not benefit the students in completing their project. First, since the agent, ALI, is only incorporated in the scheduling module of the VFTS, no agent was used in a large portion of the project (forecasting and planning modules) for either group. Second, the project could be successfully completed without the agent since the material was covered in class. The agent in this exercise was used as a reinforcement of the concepts taught in class.

Although the empirical comparison showed no net gain in project scores, a survey of the teams using ALI showed a positive self-evaluation of the agent. The results of the survey are listed below.

	N	Minimum	Maximum	Mean	Std. Deviation
Part I, AQ1. "ALI's questions and comments caused me to think about the simulation results more than I would have otherwise."	13	2	5	4.08	1.12
Part I, AQ2. "ALI's explanations helped me understand the material."	13	2	5	4.08	1.26
Part I, AQ3. "Having ALI require me to answer his questions forced me to think about the question more than I would have otherwise."	13	2	6	4.46	1.05

(Scale: 1 = I strongly disagree, 2 = I disagree; 3 = I somewhat disagree; 4 = I somewhat agree; 5 = I agree; 6 = I strongly agree)

We also have used the VFTS with ALI in a workshop at the 2002 Regional IIE Student

Conference. In addition to the VFTS workshop, the conference included industry speakers, student project presentations, and factory site tours. Students from USC, Cal Poly - Pomona, and Arizona State University attended the workshop to compete in a contest using the VFTS to determine the best production strategies that minimize manufacturing cycle time in a given example (see the appendix for a copy of the exercise). As opposed to the class project, this exercise only consisted of the scheduling module due to time constraints. In the workshop, the students were given a ten minute demonstration of the VFTS and only the agent provided the necessary tutoring.

Out of the fourteen students completing an overall evaluation of the student conference, eleven reported the VFTS Workshop as the most enjoyable activity. Students especially liked the opportunity to apply their knowledge to a real-world problem, work in teams, and see how different strategies and methods would affect the outcome of the simulated factory, and they reported that the software was easy to use, fun, and stimulating.

Conclusion and Future Efforts

One of the fundamental concepts taught in most industrial engineering curriculums is just-in-time manufacturing. Basic tenets of JIT are setup time and lot size reduction, which have a direct impact on the cycle time. Therefore, it is important for students to understand the complex relationship between the lot size and cycle time. In this paper, we have presented an agent-based learning approach for teaching this relationship. This learning paradigm is embedded in the Virtual Factory Teaching System, which is a factory simulator used to teach students forecasting, planning, and scheduling concepts. The current agent framework is being

expanded to include social intelligence in order to make the agent more sensitive to student attitudes (e.g., confidence vs. frustration) as they interact with the system.

Acknowledgements

The research reported in this paper was partially supported by the National Science Foundation under grants CDA–9616373 and EEC-9872488 and the Powell Foundation. We thank Diane Bailey, Sushil Verma, Edward Kazlauskas, Lewis Johnson, George Bekey, and Sadashiv Adiga for their contribution and input to the project. We thank Ajay Singh, Mohammad Reza Kolahdouzan, Chat Srivaree-ratana, Parthiv Patel, H. O. Reed, Rajaram Ganeshan, Erin Shaw, and Zachary Baker for their help in developing the VFTS. We thank Kristin Sahyouni for helping to create the exercise included in the appendix. We also would like to thank the members of the advisory board consisting of Mr. Jack Ferrell, Dr. Patricia Heffernan-Cabrera, Dr. Elizabeth Morris, and the late Dr. A. Alan B. Pritsker for their input.

References

Anderson, J.R., A.T. Corbett, K.R. Koedinger, and R. Pelletier (1995), "Cognitive Tutors: Lessons Learned," *Journal of the Learning Sciences,* 4(2):167-207.

De Jong, T., and W.R. van Joolingen (1998), "Scientific Discovery Learning with Computer Simulations of Conceptual Domains," *Review of Educational Research,* 68(2): 179-201.

Dessouky, M. M., Bailey, D. E., Verma, S., Adiga, S., Bekey, G. E. and Kazlauskas, E. J. (1998), "A Virtual Factory Teaching System in Support of Manufacturing Education," *Journal of Engineering Education*, Vol. 87, pp. 459-467.

Dessouky, M. M., (1998), "Using Queueing Network Models to Set Lot-sizing Policies for Printed Circuit Board Assembly Operations," *Production and Inventory Management,* Vol. 39, pp. 38-43.

Dessouky, M. M., S. Verma, D. Bailey, and J. Rickel (2001), "A Methodology for Developing a Web-based Factory Simulator for Manufacturing Education," *IIE Transactions*, Vol. 33, pp. 167-180, 2001.

D'Souza, A., J. Rickel, B. Herreros, and W.L. Johnson (2001), "An Automated Lab Instructor for Simulated Science Experiments," *Proceedings of the Tenth International Conference on Artificial Intelligence in Education*, pp. 65-76, IOS Press.

Finch, F. (1993), "Japanese Management Techniques in Small Manufacturing Companies: A Strategy for Implementation," *Production and Inventory Management*, Vol. 27, pp. 30-38.

Gertner, A.S., and K. VanLehn, "ANDES: A Coached Problem-Solving Environment for Physics," *Proceedings of the Fifth International Conference on Intelligent Tutoring Systems,* pp. 133-142, Springer.

Goldstein, I.P (1977), "Overlays: A Theory of Modelling for Computer-Aided Instruction,", Artificial Intelligence Laboratory Memo 495, Massachusetts Institute of Technology, Cambridge, MA.

Handfield, R. (1993), "Distinguishing Features of Just-in-Time Systems in the Make-to-Order Assemble-to-Order Environment," *Decision Sciences*, Vol. 24, pp. 581-602.

Hill, A. V., and A. S. Raturi (1992), "A Model for Determining Tactical Parameters for Materials Requirements Planning Systems, *Journal of the Operational Research Society,* Vol. 43, pp. 605-620.

Im, J. H., and S. M. Lee (1989), "Implementation of Just-in-Time Systems in U.S. Manufacturing Firms," *International Journal of Operations and Production Management***,** Vol. 9, pp. 5-14.

Karmarkar, U. S., S. Kekre, and S. Kekre (1992), "Multi-item Batch Heuristics for Minimization of Queueing Delays," *European Journal of Operational Research*, Vol. 58, pp. 99-111.

Lester, J.C., S.A. Converse, B.A. Stone, S.E. Kahler, and S.T. Barlow (1997), "Animated Pedagogical Agents and Problem-Solving Effectiveness: A Large-Scale Empirical Evaluation," *Proceedings of the Eighth World Conference on Artificial Intelligence in Education,* pp. 23-30, IOS Press.

Mehra, S. and R. A. Inman (1992), "Determining the Critical Elements of Just-in-Time Implementation," *Decision Sciences*, Vol. 23, pp. 160-174.

Meyer, T.N., T.M. Miller, Kurt Steuck, and M. Kretschmer (1999), "A Multi-Year Large-Scale Field Study of a Learner Controlled Intelligent Tutoring System," *Proceedings of the Ninth International Conference on Artificial Intelligence in Education,* pp. 191-198, IOS Press.

Moore, J.D. (1996), “Making Computer Tutors More Like Humans,” *Journal of Artificial Intelligence in Education,* 7(2): 181-214.

Pritsker, A. A. B. and O’Reilly, J. J. (1999), *Simulation with Visual SLAM and AweSim,* 2nd Edition*,* John Wiley & Sons, New York, New York.

Schonberger, R. J. (1986), *World Class manufacturing: The Lessons of Simplicity Applied*, Free Press, New York.

Voss, C. A. and S. J. Robinson (1987), "Application of Just-in-Time Manufacturing Techniques in the United Kingdom," *International Journal of Operations and Production Management*, Vol. 7, pp. 46-52.

APPENDIX

IIE Conference

Production and Scheduling Competition
March 23, 2002

Congratulations! After earning millions of dollars on the dot.com craze, you have decided to put your IE education to good use and have used your savings to purchase a manufacturing firm, I.I.E. Manufacturing Co. (IIEM). But, things at IIEM aren't going *exactly* as planned.

IIEM produces four different electrical component products that are used exclusively in a variety of appliances. IIEM holds patents on all four products and as a result currently faces no competition. However, their patents will be expiring in less than one year and if production in your plant continues as it is currently, you will soon be out of business.

IIEM's former owners had a poor history of meeting their demand on time – they often had high demand backordered as well as excessive inventory due to poor forecasting and scheduling. They also did very little to contain their costs and consistently incurred unnecessary additional production expenses. They maintained high work-in-process (WIP) inventory, leading to long production cycle times. The high cycle times and WIP were the result of many factors including manufacturing inappropriate quantities of the four products (i.e. little to no forecasting was done), poor utilization of the machines, running inappropriate lots sizes, and a lack of formal scheduling methodology. None of this was of any concern to the former owners since, as the sole providers of these products, they could simply pass the additional costs on to their customers.

Unfortunately, these are the conditions under which IIEM has operated for many years. In order to save your company and win back a lot of the business that will probably be lost next year when your patents expire and other manufacturing firms enter the market, there are many changes that must take place. In order to accomplish everything on your agenda, you have hired an independent consulting team to reorganize your production.

After a few months of observation and analysis, the consulting team has redesigned the floor layout and setup your production line as a simple flowshop, meaning each product goes through workcenters 1, 2, & 3 in the same order. Two members of the team have dedicated a significant portion of their work to conducting time studies and have determined accurate transportation times between workcenters and processing times for each product. In addition, they have given you a setup time matrix for the various products and workcenters in the factory. Lastly, the team has analyzed historical demand data and determined that in order to satisfy your current demand, you must produce 50 units of product 1, 75 units of product 2, 30 units of product 3, and 40 units of product 4 each week. Thanks to these consultants, you are almost back on track. The only thing left for you to do is determine the proper scheduling methodology to use in order to satisfy your demand under the given cost and time constraints.

The 50, 75, 30, and 40 units of products 1, 2, 3, and 4, respectively, must be produced each week in order to satisfy your demand and rebuild IIEM as a successful manufacturing company. IIEM is currently unable to produce these quantities within the 2400-minute workweek (5 days x 8 hours per day x 60 minutes per hour = 2400 minutes per workweek) and currently has an average cycle time of 2000 minutes. Cycle time can be defined as a product's lead-time, or the time from when a product is introduced to the factory to the time it has completed production. Your goal is to reduce the cycle time. In addition, you must meet the demand within the week. The standard workweek is 2400 minutes. There

is an option of running the factory for one hour of overtime each day, thus creating a 2700-minute workweek – but this option will incur additional expenses. In the VFTS simulation, the process finish time represents the total time in a workweek required to complete production. For each hour of overtime in your schedule, the factory operation costs an additional $4,000. For each half-hour reduction in cycle time, you will save $180 in inventory costs.

All of the processing, transportation, and setup times have been accurately determined by the consulting team and entered into the VFTS simulation. You must now determine the proper release rule for the factory, dispatching rules for each of the work centers, and appropriate lot sizes for each product. **Do not change any other variable in the simulation!** In addition to determining the proper release rule, dispatching rule, and lot sizes, also calculate the savings (cost) to your company as a result of (1) cycle time reduction savings, (2) additional overtime costs (if any), and (3) the net savings (cost) of meeting this demand using your scheduling methodology.

Number of Machines

Work Center	Number of Machines
1	2
2	2
3	3

Demand (in units)

Product 1	Product 2	Product 3	Product 4
50	75	30	40

Setup Times (in minutes per lot)

	Work center 1	Work center 2	Work center 3
Product 1	65	72	61
Product 2	52	53	50
Product 3	61	57	59
Product 4	54	51	52

Processing Times (in minutes per unit)
(Standard deviation of processing times is negligible.)

	Work center 1	Work center 2	Work center 3
Product 1	5	10	15
Product 2	9	7	30
Product 3	29	25	22
Product 4	8	12	25

Transportation Times (in minutes)
(Rows represent source work centers, columns represent destination work centers.)

	Work center 1	**Work center 2**	**Work center 3**
Work center 1	1	2	3
Work center 2	2	1	2
Work center 3	3	2	1

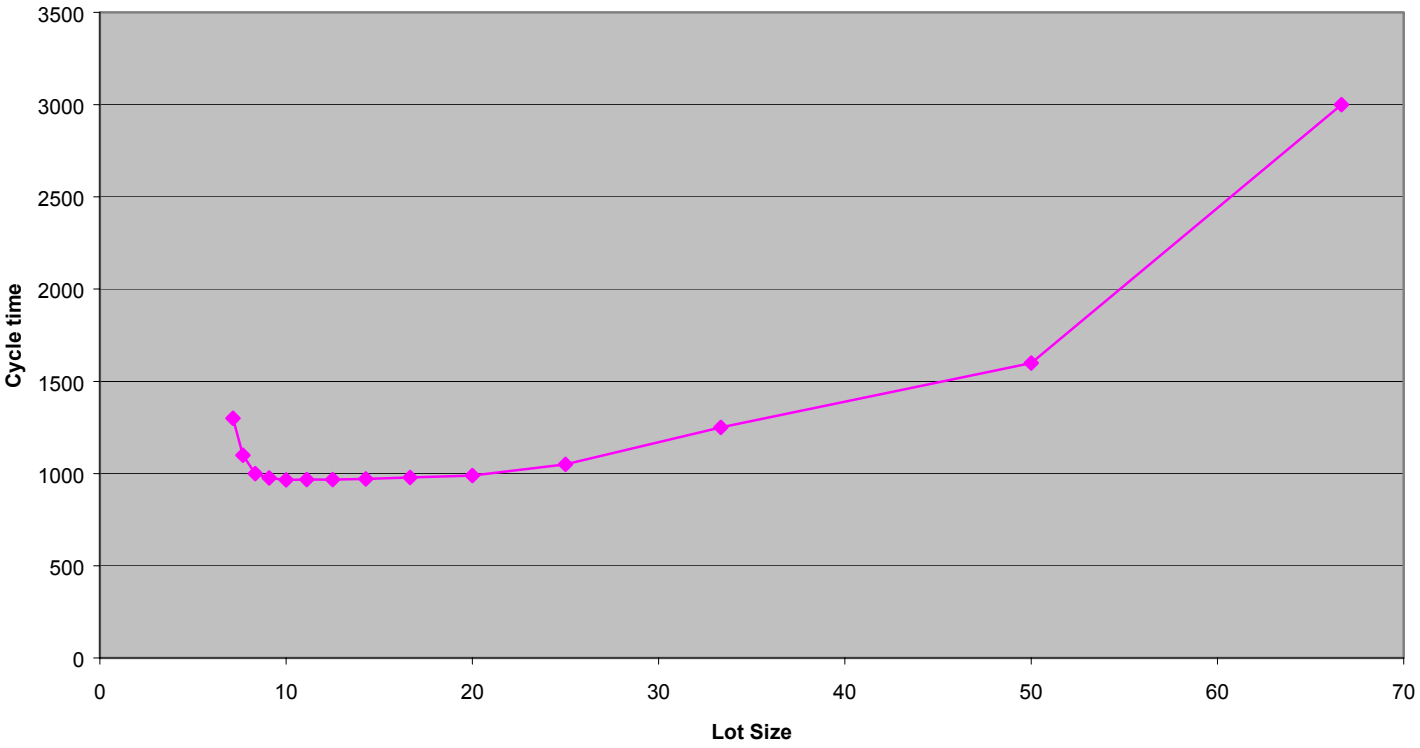

Figure 1. A sample plot between cycle time and lot size

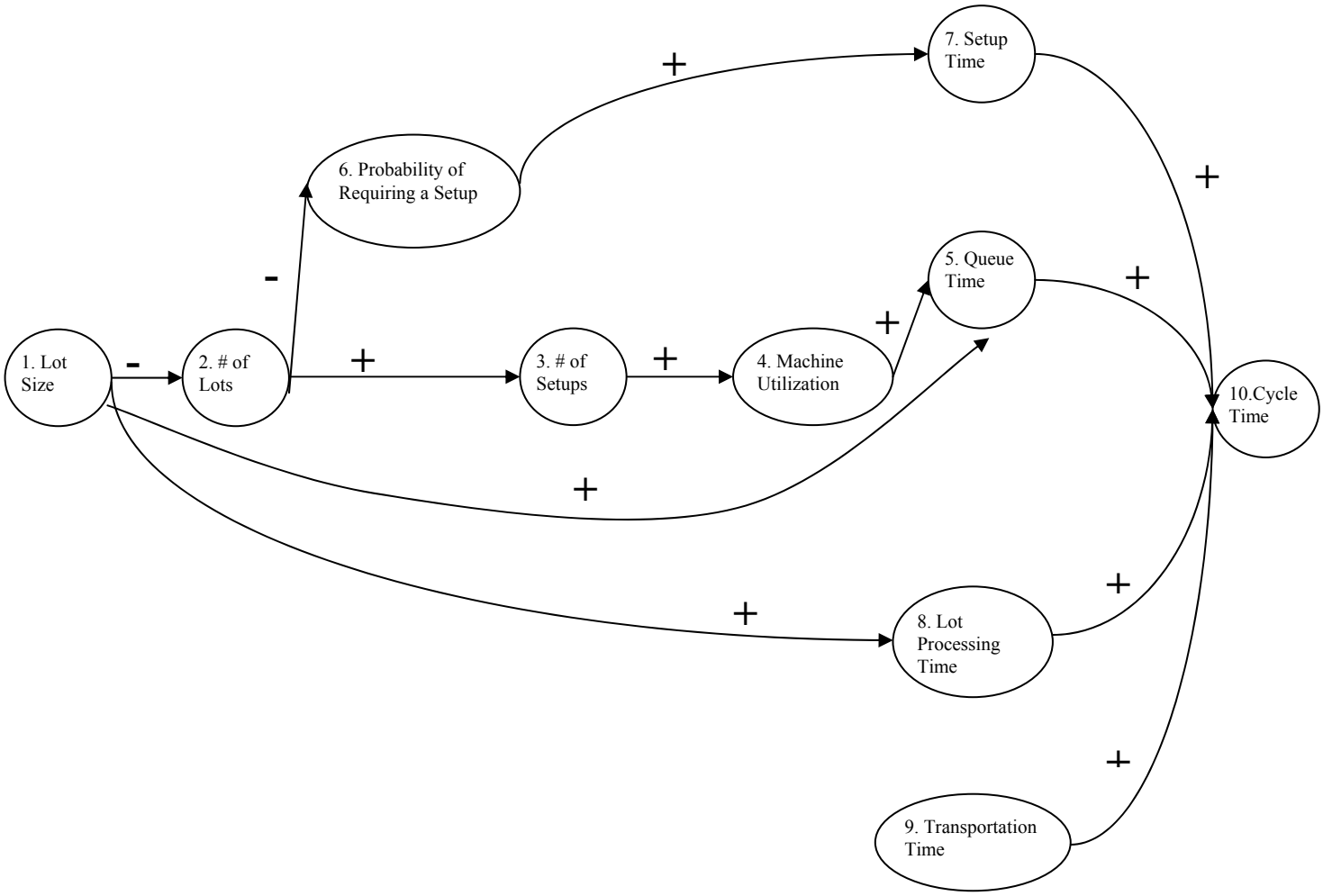

Figure 2. The relationship between cycle time and lot size

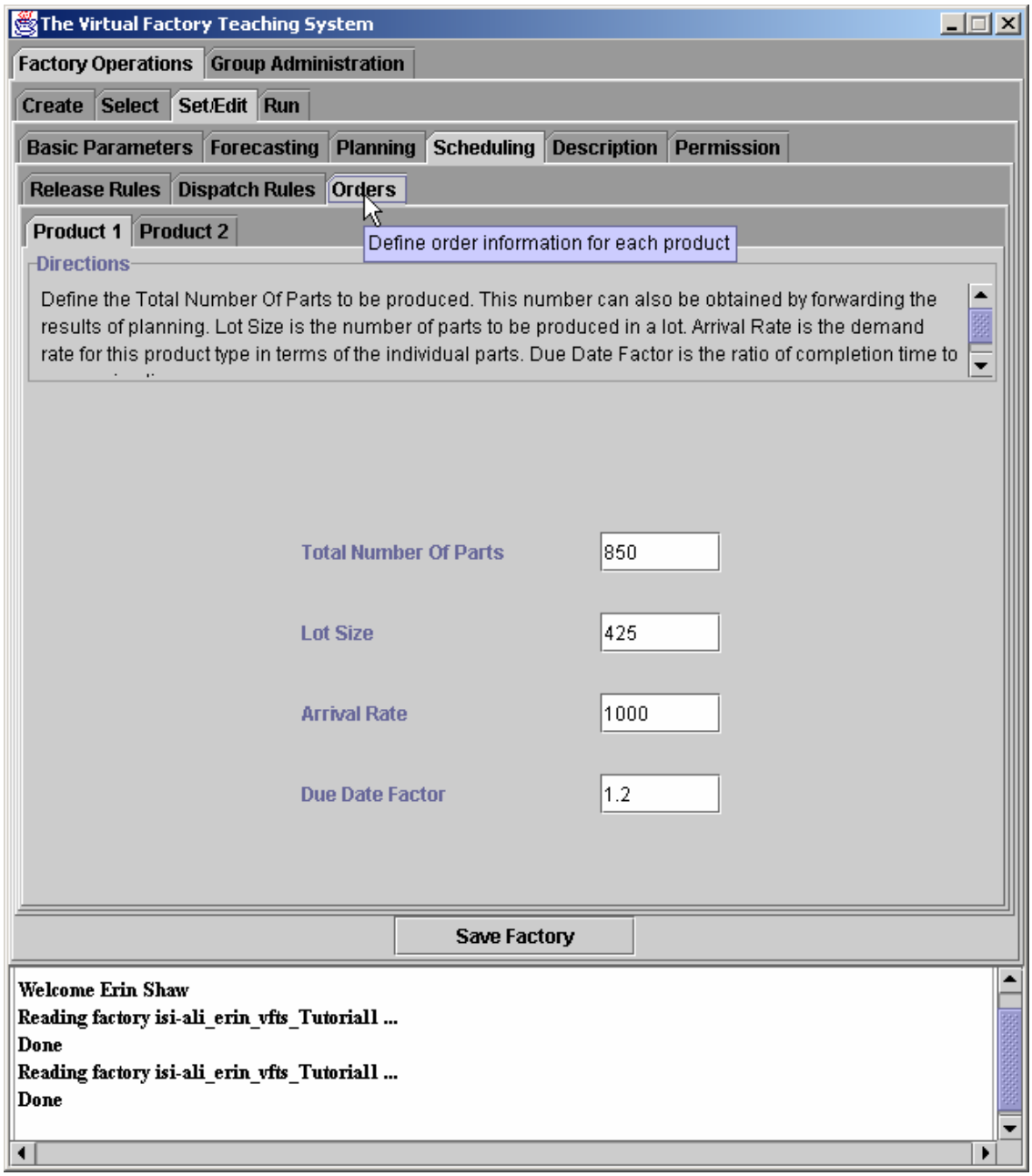

Figure 3. Students make factory decisions via a graphical user interface

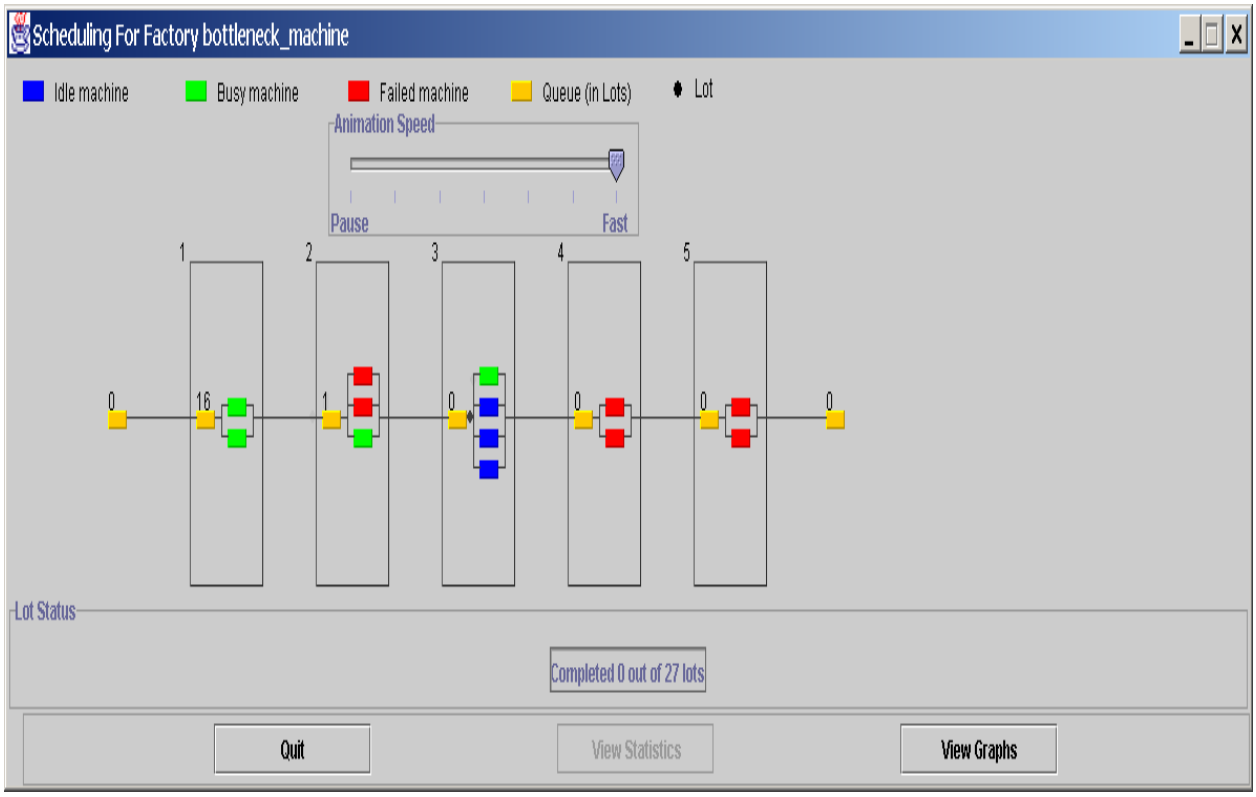

Figure 4. Simulation animation

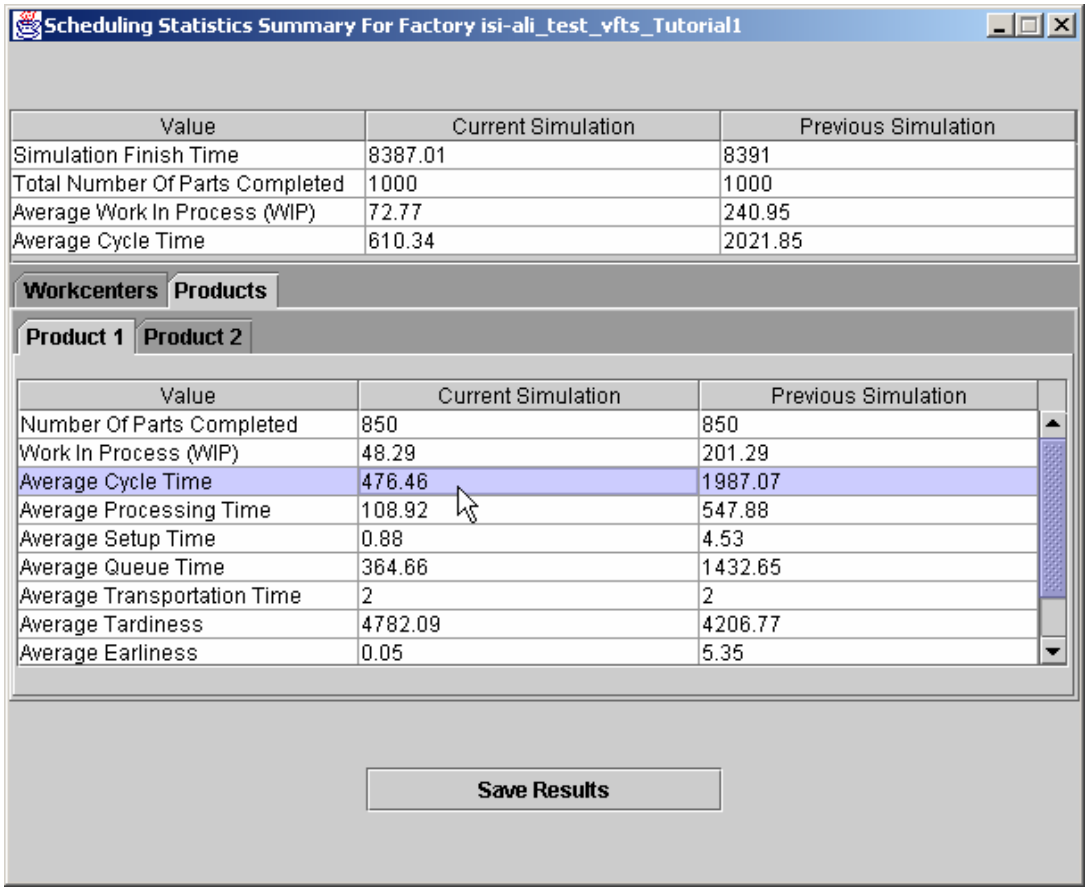

Figure 5. Simulation results

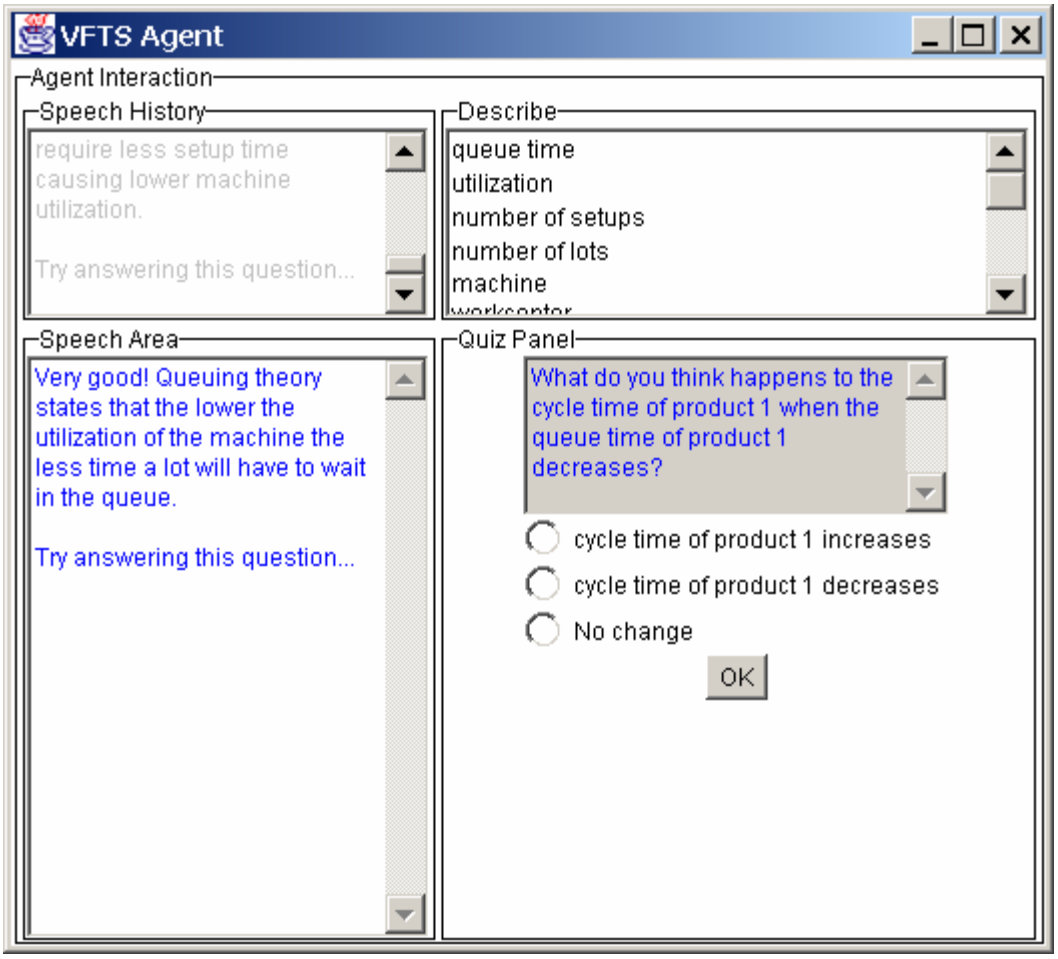

Figure 6. Agent interaction window

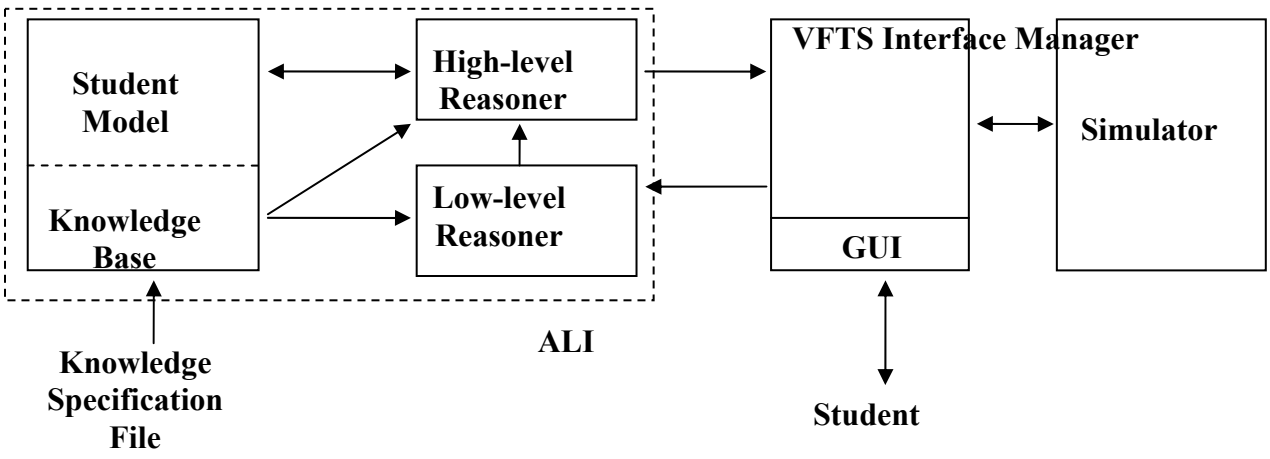

Figure 7. VFTS/ALI architecture