International Journal of Production Research
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/tprs20

Intelligent scheduling controller for shop floor control systems: A hybrid genetic algorithm/decision tree learning approach

C.-T. Su [a] & Y.-R. Shiue [b]

[a] Department of Industrial Engineering and Management , National Chiao Tung University , Hsinchu, Taiwan, ROC

[b] Department of Industrial Management , Van Nung Institute of Technology , Chungli, Taiwan, ROC

Published online: 14 Nov 2010.

To cite this article: C.-T. Su & Y.-R. Shiue (2003) Intelligent scheduling controller for shop floor control systems: A hybrid genetic algorithm/decision tree learning approach, International Journal of Production Research, 41:12, 2619-2641, DOI: 10.1080/0020754031000090612

To link to this article: http://dx.doi.org/10.1080/0020754031000090612

INT. J. PROD. RES., 2003, VOL. 41, NO. 12, 2619–2641

Intelligent scheduling controller for shop floor control systems: a hybrid genetic algorithm/decision tree learning approach

C.-T. SU†* and Y.-R. SHIUE‡

This work develops an intelligent scheduling controller (ISC) to support a shop floor control system (SFCS) to make real-time decisions, robust to various production requirements. Selecting near-optimal subset system attributes (or features) based on various production requirements to construct ISC knowledge bases is a critical issue because of the existence of much shop floor information in an SFCS. Accordingly, this work developed a learning-based ISC methodology to acquire knowledge of a dynamic dispatching rule control mechanism. The proposed approach integrates genetic algorithms (GAs) and decision trees (DTs) learning to evolve a combinatorial optimal subset of features from possible shop floor information concerning a DT-based ISC knowledge classifier. A GA is employed to search the space of all possible subsets of a large set of candidate features. For a given feature subset, a DT algorithm is invoked to generate a DT. Applying the GA/DT-based knowledge learning mechanism to the experimental results demonstrates that the use of an optimal subset of system attributes to build scheduling knowledge bases enhanced generalization ability of the learning bias above that in the absence of an attribute selection procedure, in terms of prediction accuracy of unseen data under various performance criteria. Furthermore, simulation results indicate that the GA/DT-based ISC improves system performance in the long run over that obtained with classical DT-based ISC and the heuristic individual dispatching rule, according to various performance criteria.

1. Introduction

In the 21st century, many manufacturing enterprises implement computer-integrated manufacturing (CIM) and flexible manufacturing technology to increase manufacturing productivity, improve customer service and enhance product quality. However, the shop floor control system (SFCS) will play a key role in the success of CIM and flexible manufacturing technology (Jones and Saleh 1990, Smith *et al.* 1996).

Previous works (Jones and Saleh 1990, Cho and Wysk 1995, Smith *et al.* 1996, Chang *et al.* 1998) have suggested that the SFCS function can be hierarchically decomposed into planning, scheduling and execution tasks. Planning has been described as selecting the tasks to be performed by the manufacturing system. Scheduling then identifies a good dispatching rule for these planned tasks based on the system's status and performance criteria. The execution function performs

Revision received January 2003.

† Department of Industrial Engineering and Management, National Chiao Tung University, Hsinchu, Taiwan, ROC.

‡ Department of Industrial Management, Van Nung Institute of Technology, Chungli, Taiwan, ROC.

* To whom correspondence should be addressed. e-mail: ctsu@cc.nctu.edu.tw

International Journal of Production Research ISSN 0020–7543 print/ISSN 1366–588X online © 2003 Taylor & Francis Ltd
http://www.tandf.co.uk/journals
DOI: 10.1080/0020754031000090612

scheduling tasks through directly interfacing with physical equipment. Information flow within a controller during normal operation occurs in a top-down fashion from planning, through scheduling, to execution. When an exception or error occurs, the recovery information flow is reversed. Therefore, some researchers (Cho and Wysk 1993, Chiu and Yih 1995, Park *et al.* 1997) have claimed that developing an efficient scheduling mechanism is the core of operating an SFCS to satisfy various performance criteria.

A sophisticated scheduling task in SFCSs, involves possess the following characteristics (Cho and Wysk 1993, 1995, Park *et al.* 1997, Chen and Yih 1999):

- It can run in real-time mode to conform to the response requirements of the execution module in the SFCS.
- It provides a learning mechanism that is robust against various part mix ratios and changes in the production requirements in the target environment.
- It possesses a system attribute-selection mechanism to identify near-optimal subsets of system attributes (information) from many types of shop floor information and then enhance the generalization ability for of ISC knowledge bases.

Most currently researched works do not possess all of the above. However, these essential characteristics are especially evident in the development of identifying near-optimal subsets of system attributes to construct a scheduling knowledge base in SFCS. This study designs an intelligent scheduling controller (ISC) in the scheduling function of SFCS to satisfy the above characteristics.

The use of a decision tree (DT) learning approach in adaptive dispatching mechanisms to improve the production of a manufacturing system has produced outstanding outcomes in recent research (Shaw *et al.* 1992, Park *et al.* 1997, Kim *et al.* 1998, Arzi and Iaroslavitz 2000, Shiue and Su 2003). The method approximates the numerically or symbolically valued target function that is robust to noisy data and capable of learning disjunctive expressions. Knowledge is finally presented in the form of a decision tree.

Selecting suitable system information based on various production requirements to construct an ISC knowledge base in DT-based learning is a crucial research issue due to the existence of a large amount of shop floor information in SFCS. Using too many attributes to learn a concept causes overfitting of training data in DT-based learning and degrades the generalization ability of an ISC knowledge base. Besides, omitting one important system attribute will greatly influence the learning and may harm the ability of the ISC to classify knowledge.

Genetic algorithms (GAs) represent a parallel, iterative optimization technique that facilitates efficient probabilistic searching in high dimensional space (Goldberg 1989). The problem of selecting attributes in SFCS is well suited to formulation as an optimization problem. Given a set of f-dimensional input SFCS attributes, the task of GA is to find a transformed set of SFCS attributes in a d-dimensional space $(d < f)$ that maximizes given performance criteria.

This paper seeks to develop an ISC to support an SFCS. It introduces a hybrid learning methodology that integrates GAs and DT learning to select a combinatorial optimal subset of features from possible shop floor information for a DT-based ISC knowledge classifier to support real time decision-making, that is robust against various production requirements, and to select an optimal subset of system information from large sets of shop floor information.

2. Review of related research

2.1. *Overview of machine learning approaches to ISC and their limitations*

Scheduling decisions are typically implemented through intelligently dynamic dispatching rules on the basis of current system status at each time decision point over a series of scheduling period horizons to meet the prerequisites for ISC in the SFCS (Cho and Wysk 1993, 1995). The advantage of using intelligent dispatching rules is that they can quickly give fast, acceptable solution and conform to operation characteristics in the SFCS. Basically, two machine-learning approaches to solving problems with the adaptive dispatching mechanism exist: artificial neural networks (ANN) approach and DT learning approach (Priore *et al.* 2001).

ANNs as learning tools have demonstrated an ability to capture the general relationship between variables that are difficult or impossible to relate to each other analytically by learning, recalling and generalizing from training patterns or data. In applying the ANNs' approach to adaptive dispatching mechanisms (Cho and Wysk 1993, Sun and Yih 1996, Arzi and Iaroslavitz 1999), the ANN established for a set of training samples each of which consists of a vector of system attribute values and corresponding dispatching rules that are generated by simulation can suggest a preference indicator of all dispatching rules for a given system status. The reasons for not using ANN as a learning tool in this study are as follows:

- ANN does not respond more quickly than DT learning for various production requirements in the SFCS. Even if the recall process is fast, the training of ANN tends to be time-consuming and relatively slow.
- Proposed GA-based attribute selection procedure cannot be applied in a neural network-based classifier because the neural network architecture and parameters are empirically determined and cannot be obtained beforehand.

The major advantages of the DT learning approach in constructing ISC can be stated as follows:

- Learned function can be represented by either a DT or sets of if–then rules to improve readability by humans. It can also be easily coded into an ISC mechanism.
- Concept (knowledge) learned from training examples not only accurately classifies the given examples, but also accurately predicts the unseen (test) examples.
- DT learning methods are robust against errors in classifying the training examples and errors in the attribute values that describe these examples.

These capabilities of DT learning methods satisfy the needs generated by the dynamic nature of a manufacturing environment. Therefore, the DT-based learning approach used for ISC is applicable to decision problems in SFCS. Such problems, in which training examples are classified into one of a discrete set of possible categories (scheduling rules), are often called 'supervised classification problems'.

The Interactive Dichotomimizer 3 (ID3) algorithm (Quinlan 1986, 1987) and its successor, C4.5 (Quinlan 1993), are the primary foci of research in the field of DT learning. Generalization is an important ability specific to DT learning that predicts unseen data with high accuracy, according to concepts learned from training examples (Mitchell 1997). Figure 1 shows the generalization ability in DT learning. Real shop floor information data in the examples are sometimes misclassified or have

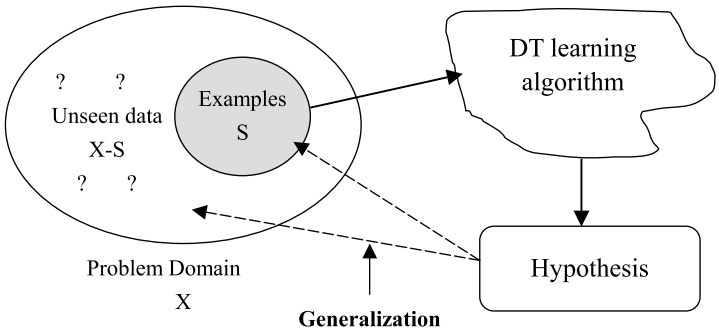

Figure 1. DT learning from examples and its generalization ability.

incorrect attribute values. Hence, further refinement is required to handle noise data and thus enhance the generalization ability of DTs.

So far most research work (Quinlan 1986, 1987, Mingers 1989a, b) has focussed on the impact of generalization ability in DT learning on the choice of attribute selection measure and post-pruning method where the hypothetical attributes are limited and given in advance. However, a problem remains about how to select appropriate attributes to describe the training examples and represent knowledge bases in the early stage due to the existence of many various types of attributes in SFC systems. If too many irrelevant or redundant attributes are used to describe the training examples, longer DTs will be created to fit the training examples and thus the generalization ability of knowledge bases will be detrimentally affected.

2.2. *Attribute selection*

Concerned with selecting attributes in the ISC problem domain, Chen and Yih (1999) presented an FSSNCA (feature subset selection based on nonlinear correlation analysis) algorithm that consists of two stages: filter and search. The filter stage can eliminate irrelevant and redundant features to define a good starting place for a search of a feature subset space. The search stage searches using feedback from the selected learning classifier to optimize the subset of the features to the chosen learning algorithm. However, Cover (1974) pointed out that even when such an independence of features was assumed and the features were selected based on individual merit, a simplified model cannot guarantee the optimality of a selected feature subset.

Shiue and Su (2003) presented an approach based on weights of ANNs to identify the important attributes for DT-based learning adaptive dispatching mechanism. Their primary conclusion was that the scheduling knowledge base that uses a set of selected attributes was superior in choosing desired dispatching rules under unknown production conditions than a knowledge base built that includes other sets of attributes. A weakness of the approach is that the attribute reduction process attributes still has redundant attributes in SFCS and cannot guarantee selection of a near-optimal subset of attributes to build a knowledge base.

Siedlecki and Sklansky (1988) gave an overview of combinatorial feature selection methods and described the limitations of methods such as probabilistic measures of class separable ability, the branch and bound search algorithm, and artificial intelligence (AI) for selecting features in the design of a classifier. They

demonstrated the unfeasibility of tackling a large-scale problem (by considering a 20-element selection problem in the large-scale domain). They described the potential benefits of Monte Carlo approaches such as simulated annealing and the GA to handle large-scale problems. A direct approach to use GAs for attribute selection was introduced by some researchers (Siedlecki and Sklansky 1989, Raymer *et al.* 2000). In their work, a GA was used to find an optimal binary vector. Each resulting subset of features was evaluated according to its classification accuracy on a set of unseen data using a K nearest neighbour classifier. Accordingly, embedding GAs used to select optimal features in DT learning are worthy of study in designing ISC methodology.

3. ISC framework and problem description

3.1. *Overview of ISC operation*

In this study, an ISC links the planning and execution functions in SFCS. It receives information such as part type, part routing, part mix ratio, planned scheduling time horizon, and performance criteria from the production requirement and the planning function in SFCS. It then sends an output control command to the execution function that interfaces with physical equipment. Under this operating architecture (figure 2), the ISC plays an important role in operating the SFCS to meet various production requirements and the quick response requirement of an execution module in the SFCS. The major task in ISC is identifying a good dispatching rule for these planned tasks, according to the current system status and performance measures in a prespecified control interval. The following section uses an FMS to illustrate the concept of an ISC operating scheme.

3.2. *Description of problem*

The case study involves a modification of the model used by Montazeri and Van Wassenhove (1990). The case involves three machine families; three load/unload stations, three AGVs, an input buffer and a central WIP buffer with sufficient capacity to prevent deadlock, and a computer-controlled local area network. The first two machine families have two machines and the third has a single machine. Eleven different types of parts are produced in this model and their processing times are identical to those used by Montazeri and Van Wassenhove. Several operating assumptions are as listed below.

- Raw materials for each type part are readily available.
- Each job order arrives randomly at FMS and consists of only one part with an individual due date.
- Part with a pallet travels to each machine or load/unload station to achieve operational flexibility, and the part-type match for one specific pallet problem is ignored.
- Each machine can execute only one job order at a time.
- Each machine is subject to random seed failures.
- Processing times are assumed to be predetermined.
- Idle machine in a family has a higher priority than other machines to process a part. If no machine in the family is idle, then the part goes to the least utilized machine.
- When the part finishes each step of the process, it must return to one of available load/unload stations for reorientation. Otherwise, it will go to central

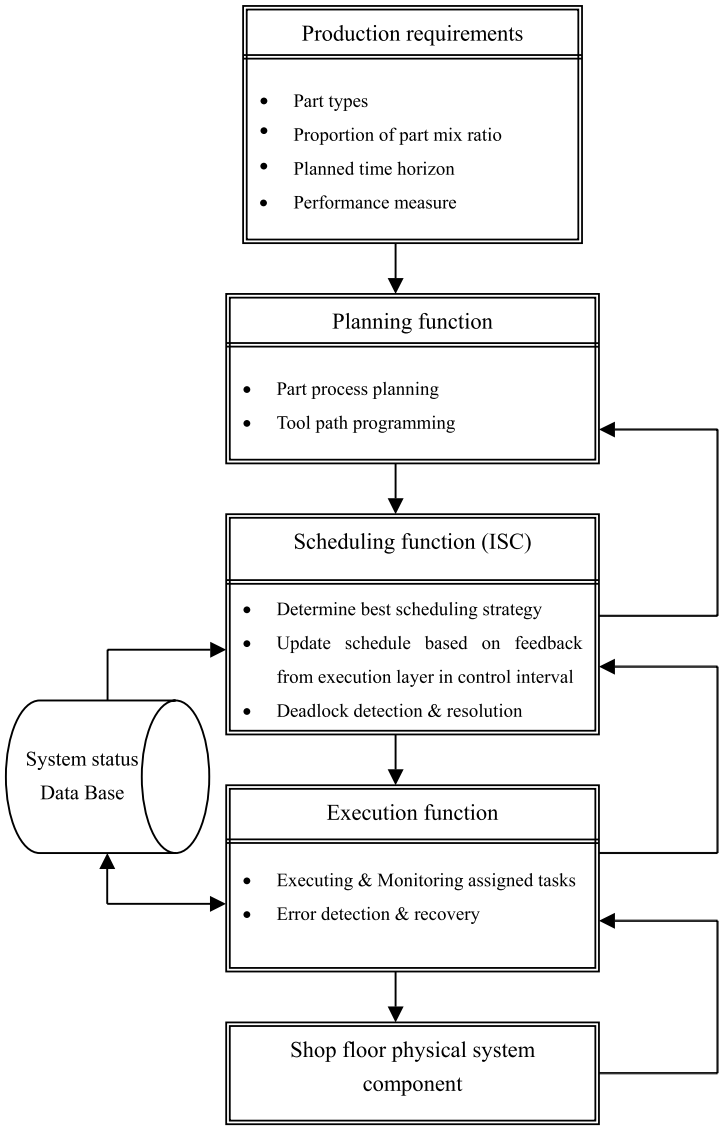

Figure 2. Shop floor control system generic function structure.

WIP buffer to wait for next operation (part reorientation in load/unload stations).

- AGV can carry only one part at a time and move only in a counterclockwise direction.
- All material movements that do not involve the AGV system are assumed to be negligible.

Based on this case study, the ISC is responsible for generating a series of dispatching strategy commands to the execution function that interfaces with physical components of FMS in this study. The part with the highest priority is

chosen for immediate processing within this shop, when the machines (or load/unload stations) are available. Intelligently and dynamically evaluating the current status of the shop floor system, using the ISC execution, and then selecting the most appropriate dispatching rule for the next scheduling period will lead to improve the performance of the system in the long run.

A robust learning-based ISC mechanism must be developed to enhance knowledge representation in a scheduling function to reflect various production requirements for SFCS and thus attain the goals stated above. Moreover, selecting near-optimal subsets of system attributes to improve the generalization ability of the knowledge base is another crucial research issue in the ISC problem domain since essential attributes are uncertain in SFCS.

4. Overview of the proposed learning-based ISC mechanism methodology

A learning-based ISC mechanism methodology is developed to obtain knowledge for a dynamic dispatching rule control mechanism by means of a hybrid GA/DT approach with the ability to select attributes and thus efficiently solve ISC problems. Figure 3 shows the architecture of the proposed learning-based ISC mechanism methodology in SFCS. The proposed ISC knowledge learning mechanism learns from training examples to evolve an optimal subset of discriminatory attributes for ISC, based on various performance criteria. Eventually, scheduling knowledge, represented as a decision tree in a dynamic dispatching rule control mechanism, sends the best dispatching strategy control command to the execution function layer.

In this architecture, the simulation-based training example generation mechanism collects a set of training examples provided as inputs for learning the concept. The input data of this phase include physical system and product specifications, product data; process plans database and training example specifications (to be described in Section 5). These are all used to build a simulation model for off-line learning training examples. In learning the knowledge phase, a GA is used to search the space of all possible subsets of a large set of system features. For a given feature subset, the DT learning algorithm is invoked to generate a DT. The performance of the DT in classifying unseen data is used to measure the fitness of the given feature set, which, in turn, is used by the GA to evolve superior feature sets. This GA/DT process iterates until a feature subset is found with satisfactory classification performance. Then, the DT learning algorithm learns the whole set of training examples with a near-optimal subset of system attributes to generate the ISC knowledge base and build a dynamic dispatching rule control mechanism.

In the dynamic dispatching rule control phase, the control mechanism in the form of a decision tree triggers a signal in a fixed control interval (determined by a preliminary simulation run based on various performance measures). The control mechanism then inputs the current system status from the execution layer, linked to the shop floor by physical system components, into the dynamic dispatching rule control mechanism for on-line scheduling control.

Under this architecture, the ISC can easily identify an optimal subset of system attributes to build sound ISC knowledge and enhance the generalization ability of a learning bias. Furthermore, the ISC can significantly improve the performance of the SFCS over that of the classical DT-based learning ISC in the absence of an attribute selection procedure under various performance criteria in the long run.

The following sections focus on two areas of this system—a simulation-based training example generation mechanism that will influence the quality of knowledge

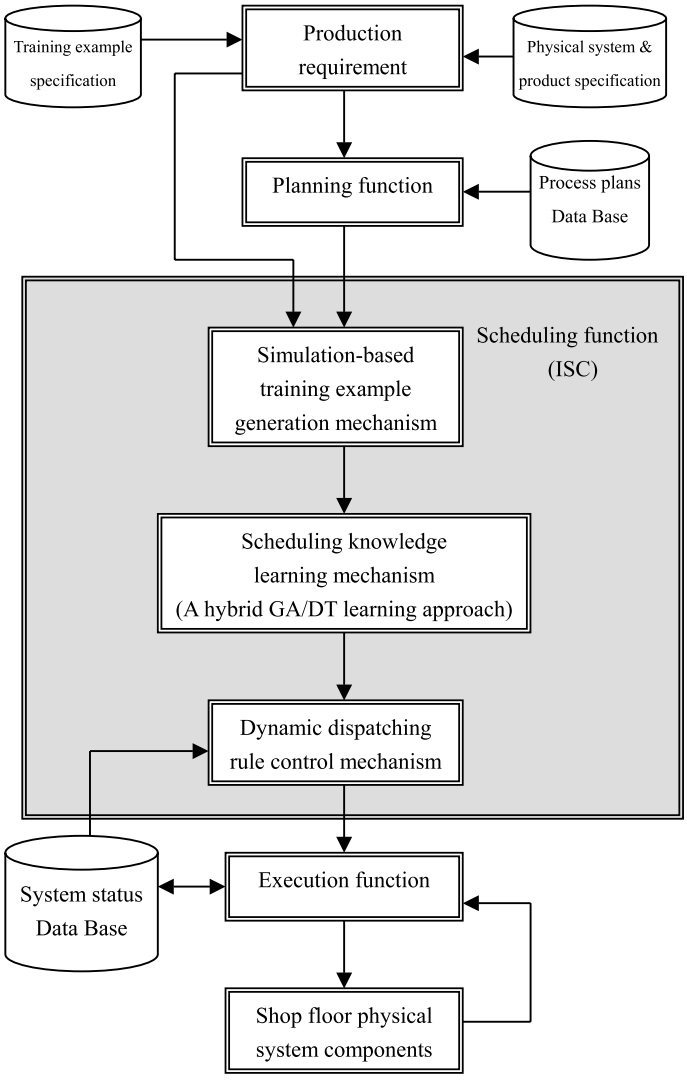

Figure 3. Architecture of the proposed learning-based ISC mechanism methodology.

representation for ISC, and a scheduling knowledge learning mechanism as the core mechanism of ISC.

5. Simulation-based training example generation mechanism

A set of training examples is provided as system information to learn the concept representing each class. A given training example consists of a vector of attribute values and the corresponding class. It is known that the relative effectiveness of a dispatching rule will heavily depend on the status of the system, given by performance criteria. Hence, in order to build the scheduling knowledge base, training examples in ISC must have enough information to reveal this property.

In the ISC knowledge base, a set of training examples can be represented by triplet $\{PM, F, d_o\}$, where PM is the user-defined management performance measures; F is the subset of system features, and d_o is the optimal control strategy (dispatching rule) under such performance criteria and system status.

Three kinds of performance criteria are typically studied in scheduling problem domain: throughput based, flow time based, and due date based. Table 1 presents the three performance criteria used in this study.

Performance criteria	Description
TP	throughput
MF	mean flow time
NT	number of the tardy parts

Table 1. Performance criteria.

System attribute	Description
Nj	number of the jobs in the system
MeUM	mean utilization of machines
SdUM	standard deviation of machine utilization
MeUL	mean utilization of load/unload stations
MeUB	mean utilization of pallet buffers
MeUA	mean utilization of AGVs
MiOT	minimum imminent operation time of candidate jobs within the system
MaOT	maximum imminent operation time of candidate jobs within the system
MeOT	mean imminent operation time of candidate jobs within the system
SdOT	standard deviation of the imminent operation time of candidate jobs within the system
MiPT	minimum total processing time of candidate jobs within the system
MaPT	maximum total processing time of candidate jobs within the system
MePT	mean total processing time of candidate jobs within the system
SdPT	standard deviation of the total processing time of candidate jobs within the system
MiRT	minimum remaining processing time of candidate jobs within the system
MaRT	maximum remaining processing time of candidate jobs within the system
MeRT	mean remaining processing time of candidate jobs within the system.
SdRT	standard deviation of the remaining processing time of candidate jobs within the system
MiST	minimum slack time of candidate jobs within the system
MeST	mean slack time of candidate jobs within the system
SdST	standard deviation of the slack time of candidate jobs within the system
MaTA	maximum tardiness of candidate jobs within the system
MeTA	mean tardiness of candidate jobs within the system
SdTA	standard deviation of the tardiness of candidate jobs within the system
MaWL	maximum workload in front of any machine/station within the system
ToWL	total workload in front of any machine/station within the system
MeSO	mean sojourn time of candidate jobs within the system
SdSO	standard deviation of the sojourn time of candidate jobs within the system
MeTD	mean time now until due data of candidate jobs within the system
SdTD	standard deviation of the time now until due data of candidate jobs within the system

Table 2. SFCS attributes.

Dispatching rule	Description
FIFO	select the job according to the rule of first in first out
SPT	select the job with the shortest processing time
SIO	select the job with the shortest imminent operation time
SRPT	select the job with the shortest remaining processing time
CR	select the job with the minimum ratio between time now until due-data and its remaining processing time
DS	select the job with minimum slack time
EDD	select the job with the earliest due-data
MDD	select the job with the minimum modified due-data
MOD	select the job with the minimum modified operation due-date

Table 3. Dispatching rules.

This work seeks to identify essential system attributes under various performance criteria. Therefore, all possible system attributes are exhaustively examined. Table 2 lists the 30 candidate attributes examined in this study. The criteria for selecting system attributes are based on both earlier research (Cho and Wysk 1993, Park *et al.* 1997, Chen and Yih 1999, Arzi and Iaroslavitz 2000), which used machine learning to develop scheduling knowledge bases, and the case environment considered in this research.

The need for dynamic dispatching rules arises from the fact that no single dispatching rule has been proven to be optimal than all other rules under a variety of shop configurations and operating conditions (Blackstone *et al.* 1982, Baker 1984, Montazeri and Van Wassenhove 1990, Sabuncuoglu 1998). Consequently, excessive efforts in studying the best dispatching heuristics in various environments are unnecessary. Table 3 specifies nine dispatching rules that were found effective in terms of the three performance criteria in this study. The mathematical definition in performance criteria, system attributes, and dispatching rules are equivalent to those used by Shiue and Su (2003).

Training examples must constitute a comprehensive initial knowledge base that represents a broad range of possible system states (Arzi and Iaroslavitz 2000). A multipass simulation (Wu and Wysk 1989) approach is used as a tool in the training example collection mechanism, which is utilized to collect training examples including the state variable of the system attribute recorded at the beginning of scheduling decision point and the performance measure of each dispatching rule recorded at the end of a scheduling decision point. This training examples collection mechanism offers the advantage of being able to provide various patterns of job arrivals that represent a breadth of possible system states to learn the concept for the ISC knowledge learning mechanism. Moreover, some training examples generated by the training examples collection mechanism can be used as unseen data to determine the optimal subset of shop floor information.

6. Architecture of a hybrid GA/DT-based knowledge learning mechanism

Figure 4 shows the proposed approach that consists of two modules: a GA module and a DT learning module. A GA module is used to search the spaces of possible subsets of a large set of candidate shop floor system attributes, to determine an optimal subset of system attributes according to various performance measures, whereas the DT learning module generates training examples that were applied to

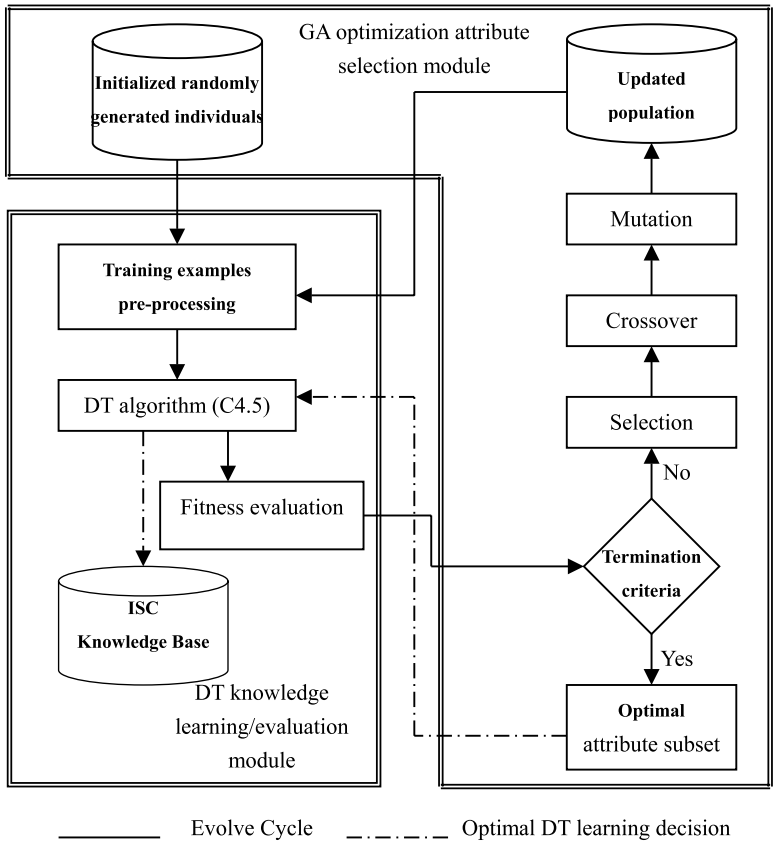

Figure 4. Architecture of the GA/DT-based knowledge learning mechanism.

evaluate the fitness of a given subset of system attributes determined by the GA module. This GA/DT process iterates until a feature subset is obtained with the best performance in classifying unseen data, and the corresponding DT module is obtained with an optimal subset of system attributes to generate the ISC knowledge base.

A more concise algorithm for the ISC knowledge learning mechanism follows.

Step 1. Initialize GA and DT module parameters.
Set $g = 0$, where g is the current generation.

Step 2. Initialize a random population of individuals, $P(g)$, which represents a population of candidate subsets of the attributes in SFCS at generation g.

Step 3. Assign $\mathrm{P}(g)$ to a collection of training examples, S, denoted, $S_{\mathrm{p}(g)}$.

Step 4. Evaluate initial $\mathrm{S}_{\mathrm{p}(g)}$.

Step 4.1. Use the DT algorithm to build DTs for all $\mathrm{S_P}(g)$ in DT module.

Step 4.2. Obtain the performance of the DT algorithm in classifying unseen data as a fitness function for all $S_{\mathrm{P}(g)}$ in DT module, and then feed this performance into GA module.

Step 5. While the termination criteria are false, performs *Steps* 6–10 (Evolution Cycle).

Step 6. Set $g = g + 1$.

Step 7. Select $S_{P(g)}$ from $S_{P(g-1)}$ in GA module.
Step 8. Crossover $S_{P(g)}$ in GA module.
Step 9. Mutate $S_{P(g)}$ in GA module.
Step 10. Evaluate Sp(g).
Step 10.1. Use the DT algorithm to build DTs for all SP(g) in DT module.
Step 10.2. Obtain the performance of the DT algorithm in classifying unseen data as a fitness function for all SP(g) in DT module, and then feed this performance into GA module.
Step 11. While termination criterion is true. For optimal $S_{P(g)}$, using the DT algorithm generates an ISC knowledge base in the DT module.

6.1. *GA optimization attribute selection module*

A GA is used here to evolve the optimal subset of system information used as the input attributes for the DT-based ISC knowledge learning mechanism. GAs involve five basic procedures: encoding the problem parameters, generating the initial population, setting the fitness function, applying genetic operators based on an objective function to create a new population, and determining termination criteria. Each of these procedures is discussed below in relation to the algorithm considered here.

6.1.1. *Chromosomal representation*

A chromosome in a GA is a sequence of symbols that represents an individual or candidate solution to a problem. These symbols are often encoded as numbers.

In the application of a GA to determine optimal subset attributes for SFCS, each chromosome represents a subset of attributes. A GA is employed to find the optimal binary chromosome, where each bit is associated with an attribute of the SFCS (figure 5). The dimension of the string corresponds to the number of studied attributes in the training examples. If the *f*-th bit of chromosome equals '1', then the *f*-th attribute is allowed to participate in learning scheduling knowledge; if the bit equals '0', then the corresponding attribute does not participate in learning scheduling knowledge.

6.1.2. *Initial population and choice of fitness function*

The collection of individuals in each iteration is the population. For the initial population, the value of each bit of the string is randomly generated as either '1' or '0'.

The percentage of correct classifications of unseen data is the fitness function that represents a measure used to judge the performance of the DT-based scheduling knowledge learning mechanism. The fitness function is given by:

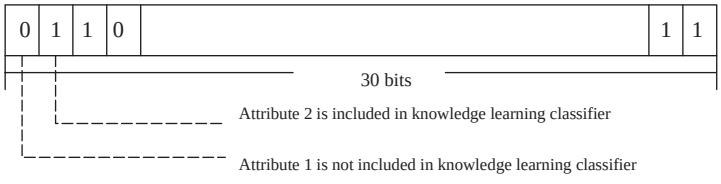

Figure 5. Chromosome structure of system attributes.

$$fv = \frac{nc}{nu} \times 100, \tag{1}$$

where *nc* is the number of unseen data correctly classified by the DT, and *n*u is the total number of unseen data.

6.1.3. *Genetic operators*

Genetic operators modify individuals within a population to produce a new individual for testing and evaluation. Historically, crossovers and mutation have been the most important and best understood genetic operators.

The crossover operator takes two chromosomes to produce two new chromosomes by swapping some bits, thereby enlarging the search space and accelerating the process of reaching the optimal solution. In this study, a chromosome is represented by a binary string as in feature selection. Hence, crossover can be performed by arbitrarily choosing a point called the crossover point, at which two chromosomes exchange their parts to create two new chromosomes. The mutation operator increases the variability of the population, and then provides a solution with an opportunity to escape the local optimal area and search other terrains for a global optimum. During the mutation process, the GA module scans each bit and generates a random number for each bit in the interval of 0 to 1 for each chromosome.

Choosing crossover and mutation rates as control parameters can involve solving a complex nonlinear optimization problem. Furthermore, the settings of these rates depend critically on the nature of the problem domain. In this work, the population size, crossover rate and mutation rate of the GA optimization attribute selection module are set as 100, 0.6, and 0.001, respectively, which values are based on the suggestions from DeJong (1975). All the strings are randomly generated.

6.1.4. *Creation of a new population*

Selection is an operation that determines the combination that performs GA operators. In this hybrid system, selection is performed by the roulette wheel method. The probability that a chromosome selected for inclusion in the mating pool is proportional to its fitness. After the crossover and mutation operations, a new population is generated and then evaluated using the fitness function.

Only a desired, predefined number of the best chromosomes survive to the next generation since the population size is finite. In this study, an elitist strategy (De Jong 1975) is incorporated into the selection process, forcing the GA to retain the best 10% of individuals. The fittest 10 individuals in each generation survive to the succeeding generation. This strategy prevents oscillation of the fitness function with each generation and significantly improves the performance of the GA.

6.1.5. *Termination criteria*

Two termination criteria are used: the best fitness remains the same for defaulted generations and the maximum number of generations is reached. The best fitness remains unchanged for the defaulted generations to determine whether the population has converged, and the maximum number generations of is considered to prevent use of excessive computer time. If either the best fitness remains the same for defaulted generations or a predefined number of generations is reached. Then, the GA process stops and the chromosome with the best fitness value, which contains an

optimal subset of system attributes. Otherwise, the GA module performs selection, crossover and mutation operations to generate a new population.

6.2. *DT knowledge learning/evaluation module*

The DT module has two primary functions. These are evaluating the fitness of the GA module and generating the DT-based ISC knowledge base. After the initial population is generated, each chromosome in the population representing a subset of SFCS information attributes is fed in the form of training examples into the DT module to evaluate fitness to determine the optimal subsets of system attributes in terms of various measures of performance. When one termination criterion is reached in the GA optimization module, and the corresponding DT module generate the ISC knowledge base. Knowledge is finally presented in the form of a decision tree and uses for dynamic dispatching control mechanism.

The DT learning algorithm, C4.5, applied here has become a standard learning tool for building the DTs used in supervised classification problem domains. C4.5 is an extended form of ID3 with some additional specific characteristics such as the ability to handle continuous attribute values, noise data, alternative measures for selecting attributes, and pruning DTs.

Rule induction in C4.5 has three phases: first, an initial, large rule tree is created from the sets of examples according to attribute selection measures; second, this tree is pruned by removing the branches with little statistical validity, and third, the pruned tree is processed to improve its understandability. The generalization ability of C4.5 can be enhanced through the first two phases even if the classifications of the training examples or the attribute values that describe these examples contain errors. Besides, according to various problem domain characteristics, C4.5 can be tuned by some parameters. Each of these aspects involved in the first two phases is discussed below in relation to this work.

6.2.1. *Constructing decision trees*

In the tree creation phase, C4.5 is a greedy algorithm that grows the tree from the top-down, selecting at each node the attribute that best classifies the local training examples. This process continues until the tree perfectly classifies the training examples or until all attributes have been tested. The system attributes in SFCS include some characteristics such as continuous value attributes and a bias in favour of features with more values. C4.5 introduces a gain ratio to avoid a bias toward the attributes with many values over those few values. Besides, C4.5 can dynamically define new discretely valued attributes that partition the continuous attribute value into discrete set intervals (called threshold values). Appendix A gives a more detailed description.

6.2.2. *Pruning decision trees*

The tree creation phase can lead to difficulties when the data includes noise. When DT classifies such data, the resulting tree tends to be very large. However, the number of branches reflects the probability of occurrence of particular data rather than underlying relationships. These relationships are very unlikely to occur in further examples. In this situation, the DT creation algorithm will produce trees that overfitting the training examples. A few training examples collected as noisy data may not generate suitable ISC knowledge. The Post-pruning DTs approach can identify the least reliable branches and remove them.

C4.5 can be tuned by two parameters during DT pruning: the minimal number of examples represented at any branch of any feature-value test (*m* parameter) and the confidence level of pruning (c parameter). Appendix B gives a more detailed description.

7. Experiment

7.1. *Constructing a simulation model and generating training examples*

A discrete event simulation model is used to generate training examples and verify the presented methodology. The simulation model is built and executed using SIMPLE++ (2000) object-oriented simulation language, and is run on a Pentium IV 1.5G MHz PC in the Windows 2000 system.

Several parameters are determined by a preliminary simulation run. The time between the arrivals of various jobs is exponentially distributed with a mean of 31 min. The period after which each job is due to be completed, is randomly assigned from six to ten times the total processing time and is uniformly distributed. The maximum number of pallets (jobs) that are allowed to be handled by the FMS system is 100. Table 4 shows five product-mix ratios considered to achieve various conditions with respect to machine loading and shifting of the bottleneck. The proportions of part types varied continually every 20 000 min.

Forty random seeds were used and 1000–5000 min (1000 min for one unit) were chosen from the beginning in simulation clock (after a warm-up period) to generate 200 different patterns of job arrivals and thus generate training examples. The warm-up period for each run was 10 000 min, followed by 10 multipass scheduling periods, each of which was from 2000 to 10 000 min (1000 min for one unit) depending on a trial and error process in each performance criterion. A total of 2000 training examples were collected.

7.2. *Optimal subset of system attributes determined by the GA/DT-based approach*

The proposed GA/DT-based approach in the GA module was implemented using a program written by the authors, coded in Visual C++ and linked to See5 (C4.5 commercial windows version) by a batch file mode. Tables 5 and 6 show the primary parameter settings of the GA/DT-based approach for the ISC knowledge

	Part-mix ratios (%)				
Part ID	Mix 1	Mix 2	Mix 3	Mix 4	Mix 5
1	11.00	14.00	6.00	9.00	14.00
2	11.00	14.00	6.00	9.00	14.00
3	11.00	15.00	6.00	9.00	14.00
4	12.00	10.00	15.00	8.00	15.00
5	6.00	12.00	15.00	13.00	7.00
6	8.00	8.00	9.00	12.00	5.00
7	8.00	5.00	8.00	3.00	5.00
8	7.00	3.00	8.00	9.00	4.00
9	7.00	3.00	7.00	8.00	4.00
10	2.50	1.00	4.00	1.00	6.00
11	16.50	15.00	16.00	19.00	12.00

Table 4. Part-mix ratios.

GA parameter	Value
Population size	100
Crossover rate	0.6
Mutation rate	0.001
Elitism strategy	0.1
maximum generation	500
Best fitness remains the same for generation	10

Table 5. GA parameters.

See5 patameter	Definition	Value
m	minimum number of instances represented by a node	2
c	confidence level for pruning	25
S	training data percentage	50

Table 6. See5 parameters.

Performance criteria	Optimal subset of system attributes	No. of attributes selected
TP	{MeUB, MaPT, SdPT, MeTA}	4
MF	{SdOT, MaPT, MeRT, SdRT}	4
NT	{Sd PT, SdST, MaTA, MeTA, ToWL}	5

Table 7. Results of selected attributed for each performance criterion.

building used in this study. Table 7 shows the results of optimal subset of system attributes according to each performance criterion.

This study has two goals. The first to investigate whether the selected optimal subset of attributes can generate more effectively generalization ability in ISC knowledge bases; the other is to determine whether using the optimal subset of attributes can yield superior production performance with respect to each criterion than in the absence of an attribute selection procedure. Hence, two experiments were conducted, and are described below.

7.3. *Verifying experiment 1 design and results*

The experiment described below is performed to examine whether the proposed GA/DT-based ISC can be more robustly generalization ability than the classical DT-based ISC (without feature selection procedure) in various scenarios.

First, two groups of attribute subsets are designed for the training examples, one of which includes all 30 attributes and the other of which includes an optimal subset of system attributes selected by GA/DT-based knowledge learning mechanism. Second, the 2000 training examples are divided arbitrary into a learning set and an unseen set by a series of 15 runs. Each run uses a different random seed. Each set contains 1000 training examples. Finally, the learning set is used to build ISC knowledge bases. The unseen set is used only to estimate the generalization ability

after the attribute has been selected and the ISC knowledge base has been constructed. Figure 6 shows the results of the experiment.

These results were then analysed by statistical independent samples t-test for each criterion to detect a statistically significant difference between the GA/DT-based and the Classical DT-based approach. Table 8 summarizes the results which show significant differences ($p = 0.0000$) in generalization ability between using and not using the feature selection procedure to build ISC knowledge bases, as measured by various performance criteria.

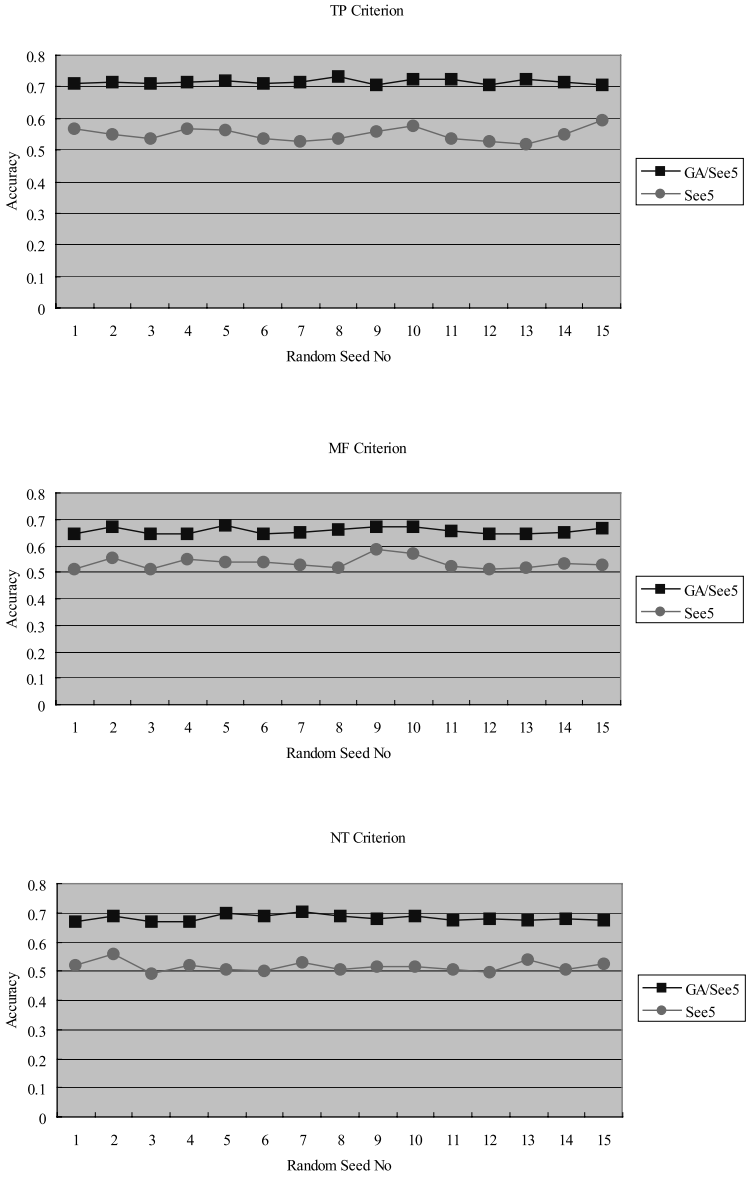

Figure 6. Accuracy of GA/See5 versus See5 using different random seeds of sampling under various performance criteria.

	TP		MF		NT	
	Mean	SD	Mean	SD	Mean	SD
GA/DT-based	0.7157	0.0076	0.6567	0.0123	0.6825	0.0104
DT-based	0.5496	0.0214	0.5347	0.0222	0.5154	0.0181
P	0.0000		0.0000		0.0000	

SD, standard deviation.

Table 8. Summary of generalization ability of the GA/See5 versus See5 approach.

7.4. *Verifying experiment 2 design and result*

A stream of arriving jobs, was generated in a simulation of 160 000 min using a different set of random seeds to investigate whether the proposed GA/DT-based ISC is more effective based on various performance measures in the long run than the classical DT-based ISC and the single heuristic rule in various scenarios. The performance of GA/DT-based ISC was compared with that of classical DT-based ISC and individual dispatching rules were compared using 30 random seeds, based on three performance criteria. Table 9 shows the mean and standard deviation of 30 simulation runs under different scheduling strategies. The proposed approach has been shown to be able to achieve better results, as measured by all performance criteria, because of superior efficiency.

A paired-sample *t*-test was performed to examine whether the proposed GA/DT-based ISC provides evidence significantly superior to that provided by the classical DT-based ISC and individual dispatching strategies (due to the use of common random number seeds which are not independent in this study). The null hypothesis is that the mean values of all the scheduling strategies are equal. An overall significance level of 95% was selected for this analysis. Table 10 summarizes the results of the paired-sample *t*-test.

The paired sample *t*-test shows that the hypothesis is rejected at a significance level of 95% for all control strategies. Therefore, the proposed approach outperforms classic DT and the other dispatching strategy.

Scheduling strategy	TP		MF		NT	
	Mean	SD	Mean	SD	Mean	SD
GA/DT-based	5248.7333	19.8771	1066.4833	206.4070	383.0667	83.4489
DT-based	5246.7667	19.8124	1151.1685	270.7587	468.0333	106.8649
FIFO	5226.9000	56.9648	1769.9067	541.3510	3096.5667	771.5670
SPT	5243.5337	22.7000	1086.0193	221.6849	419.7667	94.3371
SIO	4236.9667	37.8709	1237.8833	286.8156	768.9000	650.3615
SRPT	5245.5000	22.5048	1165.3360	358.7010	431.5333	105.9941
CR	5200.7333	48.9412	1669.0660	304.9825	2057.6667	728.3847
DS	5219.9667	72.8117	1163.1480	346.7107	958.6000	886.2119
EDD	5231.4000	56.2915	1536.2867	585.8137	2306.0333	1115.0344
MDD	5227.8000	63.8692	1532.5973	590.4579	2315.8667	1124.5156
MOD	5226.3000	66.3762	1550.0260	577.1296	2318.4000	1105.3951

Table 9. Comparison of the mean and SD between GA/DT and the other scheduling strategy (min).

Performance measure	*P*									
	DT	FIFO	SPT	SIO	SRPT	CR	DS	EDD	MDD	MOD
TP	0.0007	0.0072	0.0000	0.0378	0.0001	0.0000	0.0115	0.0304	0.0299	0.0268
MF	0.0045	0.0000	0.0096	0.0000	0.0264	0.0000	0.02020	0.0000	0.0000	0.0000
NT	0.0000	0.0000	0.0000	0.0016	0.0006	0.0000	0.0000	0.0000	0.0000	0.0000

Table 10. Comparison of the paired-sample *t*-test between GA/DT and the other dispatching strategy.

8. Conclusions

A successful ISC for supporting an SFCS would facilitated real time decision-making, be robust to various production requirements and select a near-optimal subset of system information from a large amount of shop floor information to establish ISC knowledge. This work introduces a hybrid learning methodology that integrates GAs and DT learning to select a combinatorial optimal subset of features from potential shop floor information for use by a DT-based ISC knowledge base classifier.

The following inferences can be drawn.

- Many types of shop floor information exist in SFCS. However, considering a few essential system attributes is adequate for building a knowledge base in ISC.
- Some classical DT learning approaches such as C4.5 (or See5) have provided several measures for selecting attributes in the tree creation phase and for rule post-pruning to enhance the generalization ability in learning bias. However, these DT learning approaches to constructing DT knowledge bases are usually not considered when some irrelevant and redundant attributes exist in the problem domain. The generalization ability of DT learning still deteriorates. Besides, the performance under various production criteria may be worse than that of the heuristic individual dispatching rules in the long run.
- By the GA/DT-based knowledge learning mechanism, the experimental results show that the use of the optimal subset of system attributes to build scheduling knowledge bases delivers better generalization ability than without an attribute selection procedure in terms of the accuracy of prediction on unseen data under various performance criteria.
- Proposed GA/DT-based ISC offers an intelligently dynamic dispatching rule control mechanism that evaluates the current status of SFCS, and then sends the most appropriate dispatching rule to the execution function of SFCS for the next scheduling period. The results of the simulation reveal that will lead to better system performance than that of the classical DT-based learning ISC under various performance criteria in the long run.

Some potential directions for improvement and future work are clear. First, the neural network architecture and parameters are empirically determined and unable resolve beforehand. Integrating the GA-based attribute selection approach with the neural network based ISC knowledge classifier is an area for future research. Second, the various presentation data concerning system attributes in SFCS may influence the selection of the optimal subset of features. This problem can be effectively

addressed by a sensitivity analysis and more experimental study. Moreover, how to improve the generalization ability of ISC knowledge bases in continuously changing product mix ratios is another important area for further research. A possible research direction will incorporate on-line learning mechanism that integrates reinforcement learning methodology such as Q Learning, temporal difference learning (Mitchell 1997) and transient state detection algorithm (Ishii and Talavage 1991, Moses 1999) in ISC to solve this problem.

Acknowledgements

This paper was supported in part by the National Science Council, ROC, under Contract NSC-91-2213-E-009-072.

Appendix A: C4.5 creates DTs methods used in this study

Quinlan (1986) used an evaluation function called information gain to measure how well a given attribute separates the training examples according to the target classification. In order to define information gain precisely, he defined a measure used in information theory called entropy that characterizes the impurity of a collections of examples:

$$\text{Entropy}(S) = -\sum_{d \in D} P(d) \log_2 P(d), \tag{2}$$

where S is a collection of examples, D is the set of class labels and is the dispatching rules used in this study, and $P(d)$ is the proportion of S belonging to class d.

The information gain is simply the expected reduction in entropy caused by partitioning the examples according to their features. Hence, the information gain is:

$$\text{Gain}(S,f) = \text{entropy}(S) - \sum_{v \in V_f} P_s(v)\text{entropy}(S_v), \tag{3}$$

where f is a feature relative to a collection of examples S, V_f is the set of all possible values for feature f, and S_v is the subset of S for which feature f has value v (i.e. $S_v = \{s \in S | f(s) = v\}$). $P_S(v)$ is the probability that S belong to feature value v. The probability is estimated from relative frequency of the training set.

There is a natural bias in the information gain that favours tests with many outcomes. One way to avoid difficulty is to select decision attributes based on some measure that has been used successfully in the gain ratio. The gain ratio measure penalizes attributes by incorporating the normalizing factor called split information, which is sensitive to how broadly and uniformly the attribute splits the data. The following equations define split information:

$$SI(S,f) = -\sum_{v \in Vf} P(v) \log_2 P(v). \tag{4}$$

The gain ratio measure is defined in terms of the earlier information gain, as well as split information:

$$\text{Gain ratio}(S,f) = \frac{\text{gain}(S,f)}{\text{SI}(S,f)}. \tag{5}$$

C4.5 can easily handle continuous value attributes incorporated into the learned tree. This can be accomplished by dynamically defining new discrete valued attributes that partition the continuous attribute value into a discrete of intervals (called threshold

values). For a collection of examples S, the set of all possible values for feature f are sorted on the values to give the sequence $v_1, v_2, \ldots, v_{t-1}, v_t$. Each pair of values v_s, v_{s+1} suggest a possible threshold:

$$v_{th} = \frac{v_s + v_{s+1}}{2}. \tag{6}$$

Equation (6) divides a collection of examples S for feature f into two subsets, those with a value of f above and below the threshold, respectively. Hence, there are $t - 1$ possible splits on f all of which are examined. The gain ratio of this section can then be investigated as above.

Based on above-mentioned, C4.5 creates DTs in this study involves the following steps:

Step 1. Initialization

Initialize GA and See5 parameters.
A set of training examples, S, is given.
Set F is a list of features that may be tested by the learned DT.

Step 2. For each feature $f \in F$ uses equations (2–6) calculates an initial value of gain ratio. Select feature f that results in the maximum gain ratio in information to serve Tf as the target feature whose value is to be predicted by DT

Step 3. Let $EX_{v_{th}}^{Tf}$ is the subset of examples that have threshold value v_{th} for Tf. For each $EX_{v_{th}}^{Tf}$ creates a new branch tree.

Step 4. For each $EX_{v_{th}}^{Tf}$ use equations (2–6) to calculate a gain ratio, perform Steps 5–7.

Step 5. If $EX_{v_{th}}^{Tf}$ gain ratio = 1 is true, perform Step 6. Otherwise, perform Step 7.

Step 6. This value v_{th} for Tf branch becomes a leaf node.

Step 7. $F = F - Tf$.
Below this value v_{th} for Tf branch creates new branches for F.

Step 8. Repeat Steps 2–7. Continue the procedure until all subtrees are of a single class and the system entropy is zero.

Appendix B: C4.5 prune DTs method used in this study

The m parameter determines the minimum number of instances represented by a node. A higher value of m leads to an increasing level of abstraction and thus less recoverable information about individual instances. Setting $m > 1$ (the defaulted value is $m = 2$), allows C4.5 to avoid creation of long paths that involve a minority of obscure individual instances that are most likely represent noise.

The c parameter denotes the confidence level of pruning, which ranges from 0 to 100%. No pruning occurs if setting value $c = 100\%$ is set (the defaulted value is 25%). As more pruning is performed, less information about the individual example is remembered in the abstracted decision tree. The origin of this parameter comes from Quinlan's (1987) pessimistic error pruning method for pruning DTs. A pessimistic error pruning method aims to avoid the requirement for a separate test data set. As has been seen, the misclassification rates produced by a tree on its training data are overly optimistic. If used for pruning, they produce overly large trees. Quinlan suggests using the continuity correction for the binomial distribution to obtain a more realistic estimate of the misclassification rate. For a more detailed description about estimated pessimistic misclassification rate, see Quinlan (1987).

For easy of computation, C4.5 uses the following equation to calculate the number of predicted misclassifications of N training examples at leaf:

$$N \times U_{CF}(E, N), \tag{7}$$

where E is the number of training examples misclassification at leaf, CF is confidence level (i.e. c parameter), and $U_{CF}(E, N)$ is the upper limit error probability of a leaf covering N training examples with E training examples misclassification.

If the presence of a leaf node leads to a higher predicted number of errors than its absence through a tuned c parameter, then it is pruned from the tree. C4.5 therefore prunes DTs involved following six steps.

Step 1. For each non-leaf node in the bottom layer replaced by a leaf, count the number of misclassifications in the training examples and calculate the predicted number of errors at the leaf.
Step 2. For each non-leaf node in the bottom layer if a subtree is kept, count the number of misclassifications in the training examples and calculate the predicted number of error for each leaf. Then, sum over the predicted number of errors.
Step 3. IF a leaf predicted a given number of errors in Step 1 is less than the subtree kept in Step 2 is true, perform Step 4. Otherwise, perform Step 5.
Step 4. Replace subtree with a leaf.
Step 5. Keep subtree.
Step 6. Repeat Steps 1–5. Until further pruning would increase predicted misclassification rate.

References

Arzi, Y. and Iaroslavitz, L., 1999, Neural network-based adaptive production control system for flexible manufacturing cell under a random environment, *IIE Transactions,* **31**, 217–230.

Arzi, Y. and Iaroslavitz, L., 2000, Operating an FMC by a decision-tree-based adaptive production control system. *International Journal of Production Research,* **38**, 675–697.

Baker, K. R., 1984, Sequencing rules and due-date assignments in a job shop. *Management Science,* **30**, 1093–1104.

Blackstone, J. H., Philips, D. T., Jr. and Hogg, G. L., 1982, A state-of-the-art survey of dispatching rules for manufacturing job shop operations. *International Journal of Production Research,* **20**, 27–45.

Chang, T. C., Wysk, R. A. and Wang, H. P., 1998, *Computer-Aided Manufacturing,* 2nd edn (Englewood Cliffs: Prentice-Hall).

Chen, C. C. and Yih, Y., 1999, Auto-bias selection for learning-based scheduling systems. *International Journal of Production Research,* **37**, 1987–2002.

Chiu, C. and Yih, Y., 1995, A learning-based methodology for dynamic scheduling in distributed manufacturing systems. *International Journal of Production Research,* **33**, 3217–3232.

Cho, H. and Wysk, R. A., 1993, A robust adaptive scheduler for an intelligent workstation controller. *International Journal of Production Research,* **31**, 771–789.

Cho, H. and Wysk, R. A., 1995, Intelligent workstation controller for computer-integrated manufacturing: problems and models. *Journal of Manufacturing Systems,* **14**, 252–263.

Cover, T. M., 1974, The best two independent measurements are not the two best. *IEEE Transactions on Systems, Man, and Cybernetics,* **4**, 116–117.

DeJong, K., 1975, The analysis and behavior of a class of genetic adaptive systems. PhD thesis, Department of Computer and Communication Sciences, University of Michigan.

Goldberg, D. E., 1989, *Genetic Algorithms in Search, Optimization and Machine Learning* (Reading: Addison-Wesley).

http://www.rulequest.com/see5-win.html.

Ishii, N. and Talavage, J., 1991, A transient-based real time scheduling algorithm in FMS. *International Journal of Production Research,* **29**, 2501–2520.

Jones, A. T. and Saleh, A., 1990, A multi-level/multi-layer architecture for intelligent shop floor control. *International Journal of Computer Integrated Manufacturing,* **3**, 60–70.

Kim, C. O., Min, H. S. and Yih, Y., 1998, Integration of inductive learning and neural networks for multi-objective FMS scheduling. *International Journal of Production Research,* **36**, 2497–2509.

Mingers, J., 1989a, An empirical comparison of selection measures for decision-tree induction. *Machine Learning,* **3**, 319–342.

Mingers, J., 1989b, An empirical comparison of pruning methods for decision-tree induction. *Machine Learning,* **4**, 227–243.

Mitchell, T. M., 1997, *Machine Learning* (New York: McGraw-Hill).

Montazeri, M. and Van Wassenhove, L. N., 1990, Analysis of scheduling rules for an FMS. *International Journal Production Research,* **28**, 785–802.

Moses, S. A., 1999, Transient state estimation for a discrete manufacturing system. *Journal of Manufacturing System,* **18**, 256–267.

Park, S. C., Raman, N. and Shaw, M. J., 1997, Adaptive scheduling in dynamic flexible manufacturing systems: a dynamic rule selection approach. *IEEE Transactions on Robotics and Automation,* **13**, 486–502.

Priore, P., De La Fuente, D., Gomez, A. and Puente, J., 2001, A review of machine learning in dynamic scheduling of flexible manufacturing systems. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing,* **15**, 251–263.

Quinlan, J. R., 1986, Induction of decision trees. *Machine Learning,* **1**, 81–106.

Quinlan, J. R., 1987, Simplifying decision trees, *International Journal of Man-Machine Studies,* **27**, 221–234.

Quinlan, J. R., 1993, *C4.5: Programs for Machine Learning* (San Manteo: Morgan Kaufmann).

Raymer, M. L., Punch, W. F., Goodman, E. D., Kuhn, L. A. and Jain, A. K., 2000, Dimensionality reduction using Genetic Algorithms. *IEEE Transactions on Evolutionary Computation,* **4**, 164–171.

Sabuncuoglu, I., 1998, A study of scheduling rules of flexible manufacturing systems: a simulation approach. *International Journal Production Research,* **36**, 527–546.

Shaw, M. J., Park, S. and Raman, N., 1992, Intelligent scheduling with machine learning capabilities: the induction of scheduling knowledge. *IIE Transactions,* **24**, 156–168.

Shiue, Y. R. and Su, C. T., 2003, An enhanced knowledge representation for decision-tree based learning adaptive scheduling. *International Journal of Computer Integrated Manufacturing,* **16**, 48–60.

Siedlecki, W. and Sklansky, J., 1988, On Automatic feature selection. *International Journal of Pattern Recognition and Artificial Intelligence,* **2**, 197–220.

Siedlecki, W. and Sklansky, J., 1989, A note on genetic algorithms for large-scale feature selection. *Pattern Recognition Letters,* **10**, 335–347.

SIMPLE ++, 2000, *Reference Manual Version* 7.0 (Stuttgart: AESOP).

Smith, J. S., Hoberecht, W. C. and Joshi, S. B., 1996, A shop-floor control architecture for computer-integrated manufacturing. *IIE Transactions,* **28**, 783–794.

Sun, Y. L. and Yih, Y., 1996, An intelligent controller for manufacturing cells. *International Journal of Production Research,* **34**, 2353–2373.

Wu, S. D. and Wysk, R. A., 1989, An application of discrete-event simulation to on-line control and scheduling in flexible manufacturing. *International Journal of Production Research,* **27**, 1603–1623.