turing Decision Systems. He holds a B.S. degree in Mathematics from Westminster College, and M.S. and Ph.D. degrees in Computer Science from the University of Pittsburgh. His research interests include constraint-based planning and scheduling, integration of predictive and reactive decision-making, distributed problem solving, temporal reasoning, machine learning, and knowledge-based production management. He has been a principal architect of several knowledge-based scheduling systems for complex manufacturing and space applications.

Claude Le Pape is a visiting researcher in the Robotics Laboratory at Stanford University. He received a Ph.D. in Computer Science from University Paris XI in 1988. He also attended "Ecole Normale Superieure de Paris" from 1982 to 1987 and received a management degree from "College des Ingenieurs" in 1988. His main interests are constraint satisfaction, knowledge base consistency checking and the application of symbolic reasoning techniques to manufacturing and engineering problems. His current worked is aimed at determining to what extent various constraint propagation techniques allow autonomous robots to construct and execute efficient schedules with minimal centralized control.

Karl Kempf is Principal Scientist in the Knowledge Application Laboratory at Intel Corporation located in Santa Clara, California, and is a Mentor in the Center for Integrated Systems at Stanford University. He was the founding co-chairman (with Mark Fox of Carnegie Mellon University) of the AAAI Special Interest Group in Manufacturing (SIGMAN), and currently serves as its benchmark secretary. He is also on the editorial board of IEEE Expert. Karl received a B.S. in Chemistry and a B.A. in Physics from Otterbein College (as a mathematics major), and did graduate work at Stanford University and The University of Akron, receiving his PhD in Computer Science. His research interests center on spatial and temporal reasoning systems applied to robots specifically and to manufacturing systems in general. Karl has been involved in the design, development, and delivery of six successful AI-based manufacturing applications, and has produced over 40 publications in the area of AI and Manufacturing.

AI Magazine Volume 11 Number 5 (1990) (© AAAI)

AI-Based Schedulers in Manufacturing Practice: Report of a Panel Discussion

Karl Kempf, Bruce Russell, Sanjiv Sidhu, and Stu Barrett

Abstract

There is a great disparity between the number of papers which have been published about AI-based manufacturing scheduling tools and the number of systems which are in daily use by manufacturing engineers. It is argued that this is not a reflection of inadequate AI technology, but is rather indicative of lack of a systems perspective by AI practitioners and their manufacturing customers. Case studies to support this perspective are presented by Carnegie Group as a builder of scheduling systems for its customers, by Texas Instruments and Intel Corporation as builders of schedulers for their own use, and by Intellection as a consulting house specializing in scheduling problems.

Introduction

Given the impact of manufacturing on the gross national product of any developed nation, it is not surprising that an increasing number of AI practitioners are becoming involved in the manufacturing domain. (Although the AAAI SIGMAN (Special Interest Group in MANufacturing) held its first formal business meeting at AAAI-88, its membership already includes roughly one-third that of the AAAI parent organization.) From an optimistic viewpoint, this blossoming of interest could bring about two important results. One is the revitalizing effect that strong solutions for outstanding problems would have in manufacturing. The other is the validating effect that successful solution of large scale problems would have on AI theories.

The pessimistic reality is that these results have not yet been realized in spite of long-term efforts by talented people. In this paper, we try to examine some of the reasons for this limited progress. While there are clearly technical problems encountered in applying AI techniques to manufacturing problems, our experience is that it is more likely that "people problems" block the march of progress. Although specifying and implementing the things which go on inside the computer are difficult, handling the things which go on outside of the computer are even more troublesome. While a few comments on technological matters will inevitably slip into the discussion, our intended focus is the manufacturing personnel and AI technologists involved in the projects with which we are familiar.

Artificially intelligent schedulers of manufacturing production serve as our example application. A variety of scheduling problems, including flowshops and job-shops, have been shown to be NP-complete [Garey and Johnson 1979]. A wide selection of AI-based solutions have been proposed including at least the 100 referenced in a recent overview article [Kempf 1989a]. But very few AI-based schedulers have found their way into daily manufacturing practice. Even the most publicized of AI-based manufacturing schedulers, the ISIS/OPUS systems produced at Carnegie Mellon University starting in 1980, have yet to enter productive service and are not considered to be on the verge of doing so [Smith 1989].

A panel was convened at the AAAI co-sponsored Third International Conference on Expert Systems and the Leading Edge in Production and Operations Management with the charter of addressing the disparity in manufacturing production scheduling between the number of papers in print and the number of systems in service. Under the auspices of the Management Sciences Department in the College of Business Administration of the University of South Carolina, a number of speakers were invited to participate. With direction from Timothy Fry, one of the Program Chairmen, the aerospace, automotive, and electronics industries were targeted. Systems were sought which had been built from scratch and/or using commercial tools, and

which included the efforts of internal company personnel and/or external consultants.

An invitation was issued to Carnegie Group Incorporated on the basis of their commercial tool offerings, consulting experience in manufacturing, and scheduling applications in aerospace and electronics. INTEL Corporation and Texas Instruments were invited based on their ongoing interest in applying AI to semiconductor manufacturing including production scheduling systems. Intellection's invitation was based on consulting access to a wide variety of manufacturing companies as well as broad experience in the scheduling arena. Each of these invitations was accepted and the nominees were the speakers during the panel and the authors of this article.

Two other invitations were issued, but declined. They are mentioned here simply because the reasons given for not participating speak to the central theme of this paper. One went to General Motors Technical Center where it was known that KEE (from Intellicorp) was being used to build a scheduler for a GM flexible manufacturing system. The invitation was declined citing secrecy during the pursuit of patents. We took this as an indication of a successful application. The other unsuccessful invitation went to Westinghouse Electric which has been working for some years using ART (from Inference Corporation) to build a scheduler for a flexible manufacturing system located at a General Dynamics facility in Fort Worth, Texas [Reynolds 1986]. An obvious question for a panel member representing this project would have contrasted Westinghouse sponsorship of ISIS with their utilization of ART. Unfortunately, the question was never answered since the invitation was declined based on the transfer of the initiator and principal architect of the project to another department within Westinghouse.

Panel members were each furnished with a list of questions prior to the event. This list was intended to serve at least as a motivator of discussion or, at most, as a partial outline for their presentation. The questions included the following .....

1) At a sufficiently detailed level to give some context to your remarks, what was the scheduling problem addressed and what was the technical approach you took in your solution? Had there been previous attempts in the manufacturing organization to provide similar functionality?

2) What were the steps that you went through to get the system into actual use? Were there laboratory trials? Were there factory floor trials? Was the system delivered incrementally in terms of functionality or all at once?

3) Were there knowledge collection issues? Were there human schedulers? Were they experts producing high quality solutions quickly? Was their knowledge collected and used? Was their knowledge augmented?

4) Were there data collection or interfacing issues? Was there a manufacturing data collection system? How difficult was the interfacing task? Was the data available complete, correct, and consistent?

5) Were there user interfacing issues? Who are the users - the experts or the shop floor personnel? How does the user interact with the system? How often do they interact? Were the users computer literate at the start of the project?

6) Does the system work? What are the metrics used to assess system performance? How much better is the automated system than the system it replaced? Do the users and the management think it is better?

7) Is the system accepted in the company? Do the intended users actually use the system? Does the management actually rely on the system? How do personnel not directly in contact with the system view it?

8) Where is the system going? Is it done? Will it ever be done? Is it expanding or contracting? Has it turned users on or off to AI? Has it turned management on or off to AI? Has it set up other AI projects?

Each speaker addressed these and other issues for about 15 minutes. Then the session was opened for comments from the audience. Surprisingly, discussion continued for over an hour with roughly one half of the one hundred people in attendance participating either by asking a question or volunteering an answer, example, or comment. The following are the texts of the speakers' remarks, expanded as a result of the stimulating audience interaction.

A Viewpoint from Carnegie Group Incorporated

Two different applications will be examined in this section. The first is called DISPATCHER [Acock and Zemel 1986, Zemel and Acock 1986], a currently deployed and operational system at Digital Equipment Corporation (DEC). It was developed collaboratively by Carnegie Group Incorporated (CGI) and DEC, and is currently being maintained and enhanced by DEC. The second application is currently under development by CGI for Ellwood City Forge.

The DISPATCHER environment consists of different printed circuit board assembly facilities containing on the order of 75-100 workcenters performing various process steps. An automated material handling system moves work between the processing stations. After each process step, work in process is routed back to a central location equipped with storage carousels where dispatching decisions are made. Once the decision making software has made the assignment of a tote to a workcenter, the material handling system carries out the movement to the nest workcenter. These assignments are based on tote due dates and priorities, and station availability and capability. Of course, the environment is dynamic and a given workcenter may be unavailable at a point in time or may change over from one process step to another. The decision making software is coded in OPS-5, BLISS, and C. It is integrated with conventional software; namely, the material handling systems and the factory information system. In some sense, the decision-making logic is relatively simple in this particular application. The strength of the system lies in its ability to maintain the quality and speed of its decision making under load. Load is driven both by the frequency with which the decisions have to be made and the rate of change in the status on the factory floor. Under conditions of heavy load, human decision makers tend to fall behind and have to place totes into the carousels; whereas, the automated decision making keeps pace and continues to turn totes around and assign them to available workstations.

A finite capacity planning/scheduling system for Ellwood City Forge, the second application, is being built

for a $70 million a year specialty steel forgings manufacturer located near Pittsburgh. Their business depends on producing high quality product efficiently while being very responsive to their customers. The process employed includes four sequential sets of operations. Hot steel is created in batches called melts which are then poured into ingots. Ingots in turn move to the forging process where they are forged into their required shape. Subsequently, they are heat treated according to a certain temperature profile. Finally, the parts are machined to meet their final specifications. The approach taken is rather different from the previous example. Instead of focusing on transaction-by-transaction decisions, the overall flow of orders across all four steps must be smoothed and compromises made to maximize overall flow. Significant cost reductions can be achieved through improving yield, reducing fuel consumption, and reducing the stock and work-in-progress inventories. The application is being built on top of Knowledge Craft to facilitate the explicit modeling of the manufacturing environment and various process details. The system will be deployed in the production planning department. Current pilot versions are in operation with partial functionality.

A number of issues that arise in actually deploying AI-based scheduling systems will be addressed in this section. In each paragraph an issue will be raised and how the issue was addressed in the two cases described above will be contrasted.

What were the steps or stages of the project? The DISPATCHER project went through a fairly traditional software engineering life cycle with significant emphasis on testing prior to deployment. In the Ellwood City Forge case, due to the complexity of the environment, an incremental build strategy is being followed where deployment of increasingly complex functionality will be staged. Again, this places a significant emphasis on testing in the environment of intended use.

Where did the scheduling knowledge come from? In DISPATCHER there were several human dispatchers who had previously performed this function. Most of their knowledge was sufficient for the task; and as noted above, the benefits of the system were due to its maintaining the quality and speed of the decision making under load. In the case of Ellwood City Forge, there are a number of experts who contribute knowledge. For example, metallurgists have provided knowledge of steel types and properties to meet various users' requirements. Forging experts have supplied forging process knowledge. The president of the company has supplied knowledge in terms of overall strategies that he would like to employ in running his manufacturing environment. However, any one of these sets of expertise is not sufficient to solve the overall scheduling problem. It is the integration and the use of all these sources in a coordinated fashion that leads to the solution.

What were the data interface issues? With DISPATCHER, interfaces were needed to obtain both the shop floor status and the released orders. In addition, an interface to a material handling system was required to carry out the decisions of the expert software. In the Ellwood City Forge example, the required shop floor status and order information will come out of their MRP system. The integration between our system and the MRP system provides the focal point for the teamwork between the customers' MIS organization and CGI's knowledge-based systems development group. A significant challenge to the project is the fact that the data from the MRP system may be unavailable, may be incomplete, and in some cases may be inaccurate. The decision making software must be prepared to make the best decisions it can in an environment of imperfect information.

What were the user interface issues? For DISPATCHER, a relatively simple text-oriented display revealing the decisions made to the operators was the basic online information. Summary status and performances were also developed and have been extended as operational experience has been gained. Since the Ellwood City Forge application is used by production planners, a more sophisticated information-rich interface is required. Various graphic forms of presentation tables and information need to be displayed and the user needs to be able to interact with the system to change certain key assumptions or override certain decisions. Clearly, different users required radically different interfaces.

How is system success measured? In the case of DISPATCHER, very clear measurements of throughput were made. After system deployment, throughput was double what it was prior to system deployment. Of course, there were other changes taking place in the manufacturing environment concurrent with the system deployment that somewhat dilute the correlation. However, in this particular case, the payoff for the system is very clear. In the Ellwood City Forge case, due to the complexity of the system and its phased delivery, it is unlikely that as clear a measure of impact will be available. A great deal of subjective human evaluation of the quality of its decisions throughout the phased deployment will be the main measuring stick. If the system passes those measures, it will then be fully deployed.

How is the system accepted? The acceptance of DISPATCHER was very high. The previous approach to the problem was manual and that has now been replaced by the automated decision-making embedded in the DISPATCHER software. In the case of Ellwood City Forge, the current approaches are manual and the various individuals involved in making those decisions are yet to be fully convinced that the system will improve those decisions. An important challenge of the deployment process is obtaining the buy-in and commitment of all concerned to the success of the system. In the Ellwood City Forge case, the jury is still out.

Is the system developing further and has it spawned other AI-based systems? In the DISPATCHER case, the answer is clearly yes. It is an operational system undergoing the usual enhancement and maintenance. Within DEC the commitment to AI-based decision software in the manufacturing environment is very high. Many factors contribute to that but some of the early successes such as DISPATCHER and, of course, XCON (a system that configures computer equipment) have contributed to this. In the Ellwood City Forge case, it is to be determined whether the success and follow-on commitment will be high. However, given the scope of the system and the fact that it embodies a very broad range of expertise, we anticipate significant on-going enhancement and development of the system.

The two case studies illustrate a fairly general point. It is fairly easy to automate some relatively well constrained, localized decision making. If there are significant gains to be made by maintaining the quality and speed of that decision making under load, a very high return opportunity exists. The implementation cycle is likely to be short, measurement of success localized and relatively easy, and commitment should be high. However, these opportunities are fairly rare. Ellwood City Forge represents the more typical manufacturing scheduling challenge, where the gains are only achieved by taking a broader view of the entire operation, bringing many diverse sets of expertise to bear including basic strategies for how you want to run your manufacturing operation. Deployed systems in these categories are few and represent significant challenges to the system implementor. However, the payoff for successful deployment of these systems should be substantial and will represent a major contribution of our field to manufacturing productivity.

A Viewpoint from Texas Instruments

The Texas Instruments (TI) case study dealt with the "real world" problems involved in getting a knowledge-based manufacturing scheduling system into production. The factory is located in Lubbock, Texas, and mainly produces the educational toy products of TI's consumer products group.

The factory consists of three semi-autonomous work centers: plastics molding, printed wire board assembly, and final assembly. The factory produces almost 100 different electronic assembly products. The goal of the system is to generate the daily work schedules for each work center, coordinate the operation of the three work centers in order to meet the factory demand goals, and comprehend the raw material constraints imposed by the higher level MRP system. In order for the system to utilize material constraints in the generation of the schedule, the system has to keep track of all of the current inventory of both raw materials and work in process, and needs to know about the future expectation of delivery of vended materials.

The most difficult scheduling problems concern the plastics shop. The scheduling of the production of the plastics parts has to comprehend molds, presses, operators, technicians, raw materials, and sequence dependent change-overs. The shop has 117 molds and 25 presses. A given mold can fit into multiple presses. The system has to optimize according to preference.

Another major aspect of this system is its ability to understand that the factory has to deal with seasonal variations of products, demand, and resources. All of the resource representations have to be built on a temporal foundation so that the system can understand, for example, that the number of resources (machines and operators) varies over time.

Another interesting problem concerns providing the users of the system insight into WHY a particular scheduling decision was made. Because the implementation of the system is not built on a conventional rule engine, the analysis portion did not come for free. The why or why-not analysis sometimes proved to provide as much value as the generated schedule, especially when raw material constraints were encountered.

However, the point of the panel was not to dwell on the particulars of the scheduling problem or the algorithms used for solution. The panel presented case studies of problems dealing with the difficulties involved in integration and bringing the system into production.

During the development phase of the project we encountered numerous problems. The system was envisioned to be used by multiple departments, each with their own needs and priorities. It was sometimes difficult to know which department had authority for certain aspects of system definition. The factory data was distributed over different databases, requiring the system implementors to coordinate different customer teams to work together. Some of the domain knowledge was in the form of PC spreadsheets. Some "experts" gave us conflicting information about how the operations were run. We had a recurring problem of understanding the difference between how the operations were currently organized versus how some expert thought they should be organized.

We had numerous prototype milestones during the development of the system. One problem that we had difficulty with was that some of the milestones were not written in enough detail to unambiguously define the deliverables for a particular milestone. For example, a milestone that was defined as a "functional prototype" had a wide gap of expectations between the users and the implementors. Some of this was expected since this was an intra-company effort. The contract would have been much more rigorously constructed if this had been an inter-company project.

There were numerous factors working for us however. For example, the user population was very computer literate. We did not have problems with the users understanding high level user interfaces. They really were excited about using a "workstation" class tool. We had an even more important advantage in that the local Management Information Systems (MIS) department was behind the effort. They had identified a person that received formal training on the Explorer symbolic processing system. This person also worked with the development team so that he had a good understanding of the system architecture. In retrospect, this commitment by the MIS organization was the key to the success of the project.

From the beginning, the goal was to have the system be driven by the information that was currently in the factory databases. This information included: bill of materials structures, demand profiles, outstanding purchase orders to raw material vendors, and current inventory information. In addition, the knowledge engineers had to gather information about the specific characteristics of the domain that was not available in a formal database. The engineering and preferential constraints of which plastic molds can operate in which injection machines serve as one example.

It was thought that by having the system integrated with the current factory data information structure, the system would be easier to maintain. Someone could change the bill of materials for a product in one place and the system would automatically adjust. During the implementation of the system, the customer was repeatedly told that the integrity of the factory data system was KEY to the successful operation of the sys-

tem. The system implementors stressed the importance of putting the appropriate administrative mechanisms in place to assure the factory data was accurate and up to date. The various factory operation groups realized that this was important. They believed that the current mechanisms were adequate. They were wrong.

Although the final acceptance test of the system was completed in the summer of 1988, the system is currently in production only in a limited capacity. The primary reason for the latency between acceptance and full scale production is the inability of the various organizations to maintain the factory databases to a level that allows the scheduling system to provide quality operations management support. The system is expected to provide very detailed schedules, taking into account detailed information about the current state of the factory floor. The more detailed the analysis expected, the more detailed the data required.

The nature of the database problems were the usual ones encountered with CIM environments: data gathering procedures that were not enforced, unsupportable MIS software (the original author was no longer with the organization), and cross-organization responsibilities resulting in no one being responsible.

It is interesting to note that while the database problems encountered were the usual CIM problems, the system does provide some interesting side benefits. For example, the analysis feature of the scheduling system provided a benefit during the integration phase. Because the system could give reasons for its decisions, the users of the system could debug the factory databases. For example, if there was an error in the bill of materials description for a particular product, the system usually could not schedule the production of that part. The system could provide "why not" analysis that would give the users insight into where the factory data was invalid. In fact, during the time that the system was not in production, it still provided a valuable tool that allowed the factory administrators to better understand what parts of the factory databases and administrative procedures needed to be enhanced.

Even though the system is currently only in limited operation, it is an important tool to allow the various departments to quantify the quality of their operation methodologies. The system is the central focal point for all factory data and production methodologies and provides a sanity check that is not encumbered by departmental boundaries.

A Viewpoint from INTEL Corporation

At INTEL Corporation, work is underway on the problem of scheduling semiconductor wafer fabrication facilities, better known as "fabs". These fabs typically run 4 to 6 distinct processes, each containing roughly 250 processing steps. It is not uncommon for 6 to 8 different products to be produced with each process with as many as 1000 jobs active in the factory at any point in time. A typical fab contains about 300 pieces of schedulable production equipment as well as roughly 1000 tools (lithography reticles, diffusion boats, implantation sources) and 50 operators per shift. Fabs often operate 24 hours per day, seven days per week. They are scheduled strategically on a monthly and weekly basis, tactically at the beginning of each 12 hour shift, and rescheduled in real-time as often as necessary.

Given this level of complexity, it is not surprising that the main goal which the manufacturing engineers who use scheduling systems are seeking is predictability and manageability. Given the complexity of the software involved in the solution to this set of scheduling problems, neither is it surprising that the main goal of the development team has been to insure that the system exhibits a high degree of usability and maintainability. Of course, both the users and the developers are interested in employing the scheduling systems to increase fab performance in terms of minimizing work-in-progress and throughput-time and maximizing on-time-deliveries and machine-utilization.

The approach we have taken contains three components, one each for the strategic, tactical, and real-time problems [Kempf 1989b]. The strategic component is very useful for answering longer term "what-if" questions about increasing, decreasing, and rearranging production capabilities and marketing requests. The tactical and real-time components are tightly integrated and powerfully address the shorter term operations of each shift. While the strategic scheduler is built on conventional scheduling technology, the tactical scheduler is based on a novel approach [Kempf 1989c]. The tactical component contains a number of scheduling techniques, each applicable under a different set of conditions. These techniques range from machine-centered scheduling for use when bottleneck machines are the main problem, to job-centered scheduling when late jobs are the focus of attention, to hybrid approaches for a range of intermediate problems. Situation assessment knowledge is included for use in determining which technique to select as well as determining when a change of technique is indicated.

Our approach to this problem began with an extended period of study. We felt that it was appropriate to spend time with the manufacturing personnel who would be the ultimate users of the system until two events occurred - we felt we understood the problem, and they felt we understood the problem. With both the strategic and tactical components, this study period was followed by rapid delivery of a prototype user interface. This was important for the users because it crystallized in their minds the fact that we understood their problem and helped set their expectations of what the system would do once delivered. This was important for the developers because it got the users involved in the project at a very early stage and provided a harness in which to test the evolving modules of the scheduler under realistic conditions. After a series of such tests, the strategic scheduler is entering daily service in late 1989. The tactical scheduler is well into its test cycle with installation expected in early 1990. The real-time scheduler is just entering its initial implementation stage, but may be in service by the end of 1990.

The knowledge collection scenario was set early in this project when it was realized that the scheduler should not, in fact could not, be an expert system. The users made it clear early in our interactions that, although they had some good ideas concerning how to schedule production, they certainly did not want a computational model of current practice. They felt that their

knowledge was necessary but not sufficient. These circumstances have necessitated both knowledge collection and knowledge augmentation. In terms of mechanism, this means a base scheduling mechanism which operates strictly on the firm constraints in the system and aims at overcoming the fundamental combinatorics, augmented by a secondary mechanism which applies user-generated heuristics to mediate between conflicting constraints and makes decisions left open after all the relevant constraints have been applied. Given the availability of the user interface to run test cases, and given a user friendly knowledge editor (which has not yet been implemented), we have not yet experienced major problems with knowledge collection.

Data collection issues have been a quite different story. Although a data collection and management system was in place in the initial fab targeted to assist in the development of our scheduling tools, many problems arose in trying to use it. In the first place, it was not architected to provide interfaces to the kinds of systems that the schedulers represented, and so special interfaces had to be written. Furthermore, as instantiated by the manufacturing organization, the system did not track each manufacturing step about which any effective scheduler should have data. And finally, the data entered by the operators on the fab floor proved to be incomplete, incorrect, and inconsistent when checked in detail. These last two problems have been especially difficult to handle and are not completely resolved at this time. There has always been a circular argument revolving around a) promises from the manufacturing organization to address the problems as soon as the scheduling system was working and b) statements from the development team that the system could not be expected to work until the problems were eliminated. This conflict was clouded by lack of anyone being clearly identified as owning the problems.

There could have been equally serious user interface problems in this project, but they were avoided by hard work on the part of the development team. It became obvious that the intended community contained a wide mix of users. On one hand, there were users who were very computer literate, a few with personnel computers or terminals on their desks. On the other hand, there were users who were skeptical (if not antagonistic) based either on lack of exposure or bad previous experiences. We managed this mix by employing three fundamental ideas. The first was to make the interface as simple as possible. The interface was made to be understandable by the users, and did not exhibit all the tricks which the developers could think of including. The second idea was to show the users a prototype as soon as possible, and to work through many iterations until the users were comfortable. The third was to include in our user interface many of the standard reports that the users were accustomed to seeing from the existing data system, in addition to the same information displayed using the power of our high-resolution, bit-mapped, color graphic terminal. This allowed each user to adapt to the system, moving from the old system to the new at their own pace.

Assessing whether this system works or not is very difficult. This is because we did not simply automate the current approach, so it is not possible to compare performance before and after, or with the system switched off then switched on. The scheduling tools will (and have begun to) actually change peoples jobs and the way in which production is accomplished. This is a much more difficult circumstance in which to measure. But it is safe to say that the users like the system and believe it will (and has begun to) help them. Once the tactical version is in production use on a daily basis, it may be possible to more accurately quantify system performance in terms of the standard manufacturing metrics of minimizing work-in-progress and throughput-time and maximizing on-time-deliveries and machine-utilization.

It is even more difficult to access whether the system is accepted in the company [Meieran and Kempf 1989]. We have noticed cases, fortunately isolated ones, in which the further an individual is up the management chain, the less important the hour by hour scheduling problem seems, and so the less important our system seems. Others in the management chain appreciate the set of scheduling tools as a competitive weapon. There have been further cases, also fortunately isolated, of non-acceptance surfacing among those who supply data to the system. This has mainly been due to the difficulties with the quality of the data required by the tools. Others in the data chain are happy to see something being done with the data system which they have worked so hard to install, even if they do need to do further work to improve it.

The fab scheduling project at INTEL is a long way from being completed. Given that fabs are dynamic entities, always changing processes, products, and equipment, not to mention scheduling strategies in response to conditions in the marketplace, it is possible that the scheduling tools will command a certain amount of attention for a very long time. From a larger perspective, this AI-based scheduling project has stimulated a lot of interest in AI techniques to help build other competitive weapons in the manufacturing arena. We believe that our real leverage will come when we have a set of AI programs in each of our manufacturing facilities, all interfaced together bringing a new meaning to the CIM concept [Kempf 1987]. In this case, we will certainly be involved with the scheduling tools for many years to come.

A Viewpoint from Intellection

Intellection is in the business of providing effective solutions for complex manufacturing scheduling problems. We have had the chance to review many existing traditional and AI-based scheduling systems that failed to meet the users' expectations. We are rarely called upon to address a problem for which no previous solution attempts have been made. In fact, it is fairly common for us to encounter situations where several techniques have been tried unsuccessfully.

Analysis of many of these situations shows that there are a few typical mistakes that are commonly made in both AI-based and traditional approaches. In this section, a classification of these shortcomings will be provided based on conceptual and technical issues related to scheduling system designers and implementors. Other important difficulties such as organizational politics and integra-

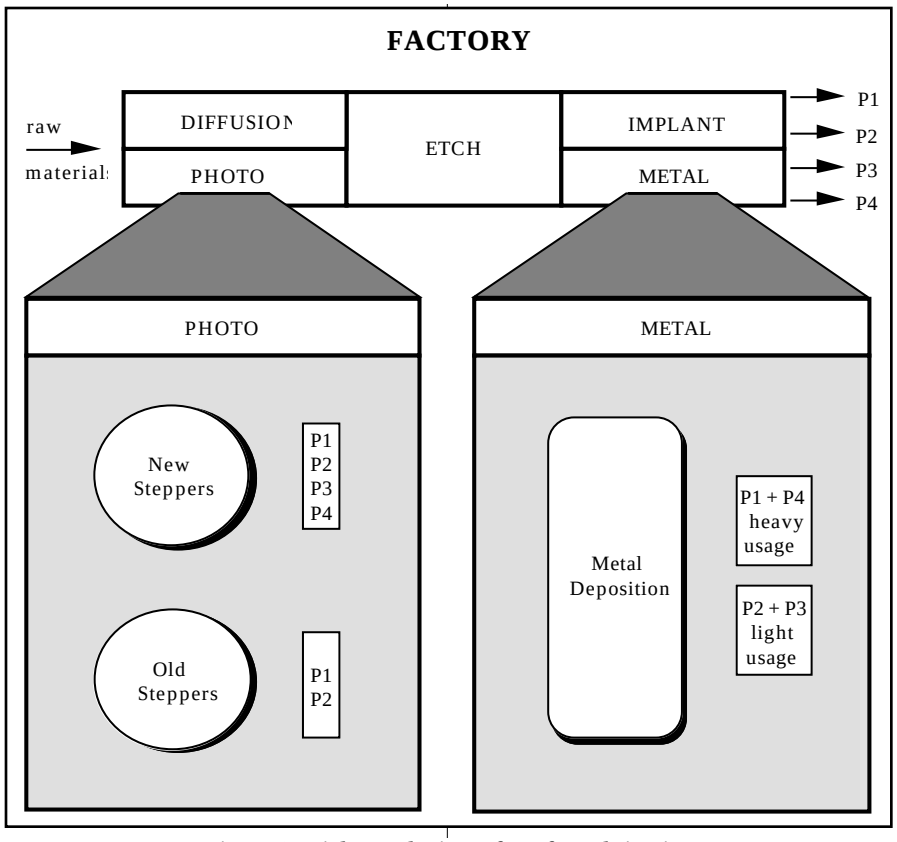

Figure 1: High-Level View of Wafer Fabrication

tion with other factory systems have been discussed previously in this article.

A semiconductor wafer fabrication plant (fab) will be used as a source of examples to clarify some of the points that will be raised. Assume that the major goals for the wafer fab being discussed are meeting due dates and minimizing cycle times. Example-1 provides a high-level view of fab operations while example-2 focuses on one particular area, providing greater detail into the lot sequencing requirements there.

Example-1 is shown diagrammatically in Figure 1. Assume that the wafer fab produces four types of products including P1, P2, P3, and P4. Consider two key areas of the factory including photo lithography and metal deposition. The photo area consists of two types of machines called steppers which index the lithograpy pattern across the surface of the wafers. The older steppers cannot handle newer products with narrow geometries like P3 and P4. The newer steppers can run all products that the fab produces. It is preferred that the newer steppers concentrate on products P3 and P4 since products P1 and P2 can run on the older steppers. In the metal deposition area, products P1 and P4 require much more service than products P2 and P3. Thus, if too many P1 and P4 type lots arrive together at this area, it can get clogged.

The fashion in which lots arrive at any area depends on the lot release and sequencing policies being used at upstream areas. If the lot release strategy is inappropriate, the mix of lots in the factory will be such that only limited improvements can be achieved by intelligent lot sequencing. The planning problem is thus one of releasing lots in such a fashion that they arrive at the above type of areas in an appropriate fashion. The appropriate fashion is one where the probability of minimizing the average cycle time for all lots and the probability of meeting due dates are maximized.

Example-2 is depicted in Figure 2. This example focuses on the lot sequencing decisions that need to be made once the lots are released to the fab. M1 is a machine which can process more than one lot of wafers at a time, such as the diffusion furnace described above, with a capacity of 2 lots per batch. Lot1 is queued for step1 to be performed on machine M1. The next step, step2, is to be performed on machine M3. Similarly, Lot2 and Lot3 are queued for step11. Step11 is followed by step12, which is to be performed on machine M2. M2 is broken and has a large queue. Machine M1 is to be loaded with either step11 (with Lot2 and Lot3) or step1 (with Lot1). The time required is the same for step1 and step11 and is independent of the number of lots loaded.

If Lot2 and Lot3 are loaded first it is certain that they will wait a long time in the queue of machine M2. Since this decision does nothing to reduce the stay of these in the plant, it does not help meet their due dates, nor does it help in reducing their contribution to plant WIP and throughput-time. On the other hand if Lot1 is loaded, only 50% of M1s capacity is utilized. If M1 needs to be utilized at a high level this decision may cause higher queuing at a later stage.

The following is a classification of the mistakes that we commonly come across in systems designed to solve the scheduling problems contained in these examples. The reason for the failure of a given effort is usually a combination of some of the points that are mentioned below. No order of importance is implied.

1. Inadequate understanding of dominant domain characteristics: In any domain, there usually are dominant characteristics that can be exploited in order to simplify solution strategies. The existence of dominant bottlenecks is a good example of such a characteristic. Decisions related to these dominant characteristics have significant impact on the quality of the overall solution. Many scheduling systems are incapable of recognizing and making use of this information. The resulting schedules are usually unsatisfactory.

There are other cases where systems do attempt to exploit a domain characteristic, but cause other damage in this attempt. This is very common in bottleneck based solution strategies. The designers of such strategies believe that the only thing that the scheduling system should worry about is the need of the capacity bottleneck resource. The reliance on any one characteristic at the expense of other important ones leads to poor schedules. In Example-1, assume that the photo area is the capacity bottleneck. In such a situation, it is quite possible for a bottleneck based strategy to feed the plant with a mix that is P2 and P3 intensive. This mix adequately utilizes the photo area, but there is a possibility of starving the metal area because P2 and P3 do not

need much processing in metal.

From another perspective, it is fairly common to come across examples of inadequate analysis of domain characteristics when pre-packaged scheduling tools are used. In many such cases, system designers concentrate on understanding tools instead of concentration on understanding the problem. Attempts are then made to fit the problem into the framework provided by the chosen tool.

2. Inappropriate reliance on locally greedy strategies: Because most scheduling problems are fairly complex, they are often simplified by using simple local dispatching rules such as First-In-First-Out (FIFO), Shortest-Remaining-Processing-Time (SRPT), and so on. As is evident from Example-2, in many situations it makes no sense to give priority to a job just because it arrived at a workstation first. Though SRPT has some attractive properties, it does not make sense to give priority to a lot with a short remaining processing time, when it is known that it will get stuck in a downstream queue. When critical decisions are made purely on local criteria, without considering global impact, systems do not perform well. This is one of the most common mistakes that we encounter.

3. Misuse of shallow expert knowledge: Experience indicates that in the domain of scheduling, experts often tend to present an inadequate picture to knowledge engineers. We come across situations where the number and nature of the constraints are such that humans are forced to oversimplify, or where situation-dependent knowledge is presented as general purpose knowledge. Once again consider Example-2 and assume that for the past year customer demand (and thus the mix running through the plant) was such that machine M1 was idle 50% of the time. When queried, the expert might say that the proper way to schedule M1 is to give highest priority to the job with the SRPT, unless the downstream process for that job has a queue of more than three jobs. Now assume that the mix changes and M1 needs to be utilized at a 96% load. In this situation the previous rule is not good because the penalties associated with underutilizing M1 become very high. In Example-2, even though Lot2 and Lot3 join a existing queue of four jobs, loading them is better than wasting 50% of M1's capacity, by loading Lot1 alone. Experts are either aware of such possibilities a priori, or adapt very fast. Expert systems that use such shallow knowledge without adaptation do not perform well.

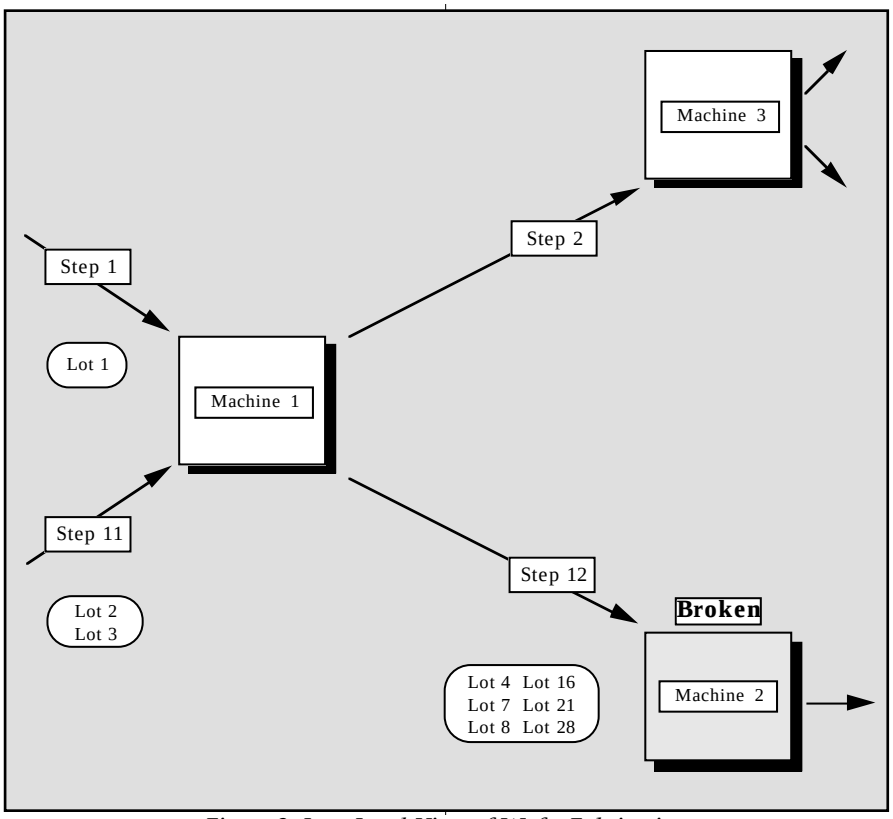

Figure 2: Low-Level View of Wafer Fabrication

As another facet of this same problem, we often notice that expert knowledge is used as a crutch and not enough effort is made to understand the basic properties of the problem being solved.

4. Excess concern about trivialities: We often come across so-called intelligent systems that are extremely dumb. One sign of this dumbness is the amount of effort these systems spend on meaningless activity. Consider a situation where a system is building a schedule for a factory for a shift of eight hours. During the course of developing the schedule, the system comes across two jobs that overlap at a machine for one minute, seven hours from the current time. Assume that there are bound to be fluctuations. It is as meaningless to label a schedule with a one minute overlap as infeasible as it is to label a schedule with a one minute gap as feasible. Systems that do not recognize such facts waste a lot of computation.

5. Improper problem segmentation: When people attempt to solve large scheduling problems, they usually start with a prototype. The prototype usually attempts to solve a simplified problem as a first step. A subsection of the plant is usually picked. Often, the proper way to design an appropriate solution for the large problem is different from the methodology needed for this small subsection. We have come across situations where this fact was not recognized. A solution that was a great success as a prototype failed at solving the scaled up version of the problem.

A related problem occurs with incremental approaches to system design. When the initial system is build, all the important constraints are not considered up front. A prototype system is built. Later attempts are made for the system to accommodate more and more criteria. This attempt to incrementally slap on additional constraints works only in extremely well designed systems. A simple example of this is the presence of holidays. Assume that an initial prototype ignored holidays while building schedules. Now the prototype is being enhanced to accommodate the constraint that no jobs can be scheduled during holidays. Assume that there is a job that requires a processing time of 8 hours. Also assume that this job is being

scheduled in such a way that it needs seven hours of processing during regular working hours and one hour of processing during a holiday period. One option is to follow a rule that does not allow you to schedule any job that runs into a holiday. This is the simple way of solving the problem. What really needs to happen is that the system must determine if it is important enough to schedule the job before the holiday starts. If this is the case, then it should see if there is any way to start the job one hour earlier. Such enhancements are not straightforward if the initial prototype was not designed properly.

In summary, we feel that there is a set of useful abstractions over the detailed problems related in points one though five. The first abstraction has to do with problem definition and spans points one and two. The lesson from our experience is that in AI, as in any other discipline, defining the problem - being sure that all of the issues have been made explicit and correctly evaluated - is absolutely necessary (but certainly not sufficient) for success. The second abstraction deals with knowledge including the Operations Research type of knowledge found in point two, the expert knowledge mentioned in point three, and the common sense knowledge referred to in point four. The lesson, possibly unique to AI, is that defining the knowledge - being sure that all of the knowledge from all of the sources has been made explicit and correctly evaluated - is absolutely necessary (but certainly not sufficient) for success. The third abstraction deals with inference as reflected in the problems described in point five. The lesson here is that the approach, or combination of approaches, selected must be able to solve the whole problem using all of the knowledge even if the system is going to go through prototype stages and be delivered in stages. Of course, we all know these things, but apparently, based on the results so far, they are frequently forgotten in the face of pressure to demonstrate a working system as soon as possible.

Conclusions

We all agree that AI techniques that are accessible today can and should be applied to the solution of manufacturing scheduling problems. Successful application benefits both the AI community and the manufacturing community. We present the status of a number of such attempts. Unfortunately, only one of the examples described here is currently fully functional and operating on a daily basis, and it addresses a relatively simple scheduling problem. Of the projects addressing more complex scheduling problems, none of the ones reported here have been fully successful yet.

We believe that it is interesting to note that there are a very large number of non-AI issues which contribute strongly to the prolonged development periods of the AI projects described. Looking critically at the state of the examples above, it becomes clear that to deliver a working AI-based scheduler into a manufacturing environment requires attention to system issues which can and do occupy at least as much person-time as the AI core. It is often the case that this situation surprises many AI practitioners who want to solve the AI part of the problem and then move on. Project personnel with this perspective are almost sure to fail to provide either relevant tests of AI theory or useful tools to manufacturing personnel. A number of problems should be anticipated from project inception, and should be identified and addressed as early as possible if the AI project is to succeed. These potential problems are reflected in the question set presented in the first section of this article.

If the application is difficult enough to provide a good test of AI theory, there may be broad disagreement in the user community whether the problem is real or is simply an artifact of inappropriate operating procedures elsewhere. If it is agreed that there is a problem, there may be broad disagreement whether to automate existing methods or to try for something better. In either case, there may be broad disagreement about who should be considered the manufacturing "experts" participating in the project. In brief, defining the problem may occupy significant time.

There will need to be a data interface to a non-AI system which may not be architected to interface to AI systems. If a data interface is successfully implemented, the data which it supplies may be incomplete, incorrect, inconsistent, and untimely in the context of the AI system. The personnel in charge of the conventional data system may not agree that there are any shortcomings in their system, and even if they do agree, may be slow to respond. In brief, designing and implementing the data interface may occupy a lot of time.

There will need to be a user interface to potentially computer illiterate personnel who may have a negative disposition toward computers based on previous bad experiences. They will be able to describe what they do not like about existing applications in their factory, but may not be able to provide adequate descriptions of user interface features which they would find helpful. If a user interface is successfully implemented, the users may continuously request modifications and enhancements. In brief, designing and implementing the user interface may occupy a lot of time.

Since the AI-based system, as delivered, will require a new and different operating procedure (even if it was meant to simply automate existing methodology), it may be impossible to prove analytically that the system is a success. Since many procedural changes will have been made in manufacturing during the development and installation of the AI-based system, performance improvements may be credited to these changes rather than to the AI system.

Designers and implementors of conventional software have encountered and learned to recognize these problems over the course of their experience. But AI practitioners have not yet delivered their technology often enough to be so aware. Until we learn that the AI core of our applications is no more than the tip of the iceberg, the AI community will continue to have a set of untested theories and the manufacturing community will continue to have a set of poorly solved problems.

References

Acock, M. and R. Zemel 1986. DISPATCHER: AI software for automated material handling systems. Proc. SME ULTRATECH - AI in Manufacturing (Long Beach), p. 2/139-2/146.

Garey, M. R., and Johnson, D. S. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York, N.Y.: W. H. Freeman.

Kempf, K. G. 1987. Integrating Artificial Intelligence into the CAD/CAM Environment. Proc. IEEE Systems Design and Integration Conf. (Santa Clara), p. 9/1-9/7.

Kempf, K. G. 1989a. Manufacturing Process Planning and Production Scheduling: Where We Are and Where We Need To Be. Proc. IEEE 5th Conf. on Artificial Intelligence Applications (Miami Beach), p. 13-19.

Kempf, K. G. 1989b. Scheduling Wafer Fabrication - The Intelligent Way. SME Electronics in Manufacturing, Vol. 4, No. 3, p. 1-3.

Kempf, K. G. 1989c. Manufacturing Scheduling - Intelligently Combining Existing Methods. Proc. AAAI Stanford Spring Symposium, p. 51-55.

Meieran, E., and Kempf, K. G. 1989. Applications of Artificial Intelligence in Factory Management. Proc. IEEE 7th Inter. Electronic Manufacturing Technology Symposium (San Francisco), p.18-22.

Reynolds, R. F. 1986. FMS Scheduling with an Expert System. Proc. Inter. Cong. Technology and Technology Exchange (Pittsburgh), p. 204-206.

Smith, S. F. 1989. Private communication, July.

Zemel, R. and M. Acock 1986. DISPATCHER: an intelligent approach to factory control. Proc. Amer. Control Conf. (Seattle), p. 152-155.

Credits

The system at Texas Instruments was built on the contributions of many people including: Knowledge Engineers Mike Delucia, Hank Ivy, and Joe Marshal, primary experts Kermic Powell and Benie Ellis, Rusty Jackson and Dick Woodcock from the plastics shop, Steve Lyle's MIS Department, Cub Curtiss for his initial start-up efforts, and Bobby Fain for his vision of how knowledge-based systems could improve the quality of his operations. The work at INTEL has been a collaboration between a number of the members of the Knowledge Applications Laboratory, including Khanh Ho, Joe Holman, Gary Scott, Chee Yu, Mary Murphy-Hoye, Marcos Paz, and Naiping Keng, and a number of members of the Manufacturing Operations Staff, including Mark Goranson, Mike Hagen, Katie Craven, Jeff Evans, Brian Verwer, Dennis Danielson, and Craig Hale, as well as Gene Meieran, INTEL Fellow, who has provided the environment in which this work could successfully take place.

About the Authors

Stu Barrett received his BSEE from Michigan Technological University in 1972. He started working for Texas Instruments in the Computer Systems Division in 1973. Early on at TI Stu spent much time working with Operating Systems, Communications, Networking and Terminal Systems. In 1985 Stu was nominated to the Senior Member of the Technical Staff position at TI. In 1986 he became involved with AI and joined the Austin Knowlege Engineering group in TI. He designed and implemented the Semantic Network Representation Language (SNRL), a frame system that is being used for most of the Lisp based KE projects at TI. Stu was the technical project manager for the TIMASS system that is used to provide decision support for factory planning and scheduling.

Sanjiv Sidhu is the president of Intellection Inc. where he provides scheduling expertise to companies such as General Dynamics, Ford, Catepillar, and Timken Steel. He has taught courses on building intelligent scheduling systems at companies such as Monsanto, McDonnell Douglas, Boeing, Alcoa, and Armco Steel. Previously, Sanjiv worked at Texas Instruments in the Artificial Intelligence Laboratory where he was the technical leader of major scheduling software projects for TI and TI's customers.

Bruce Russell, Vice President of the Government, Transportation, Steel, and Aerospace Division, joined Carnegie Group with 22 years of experience in software engineering. His most recent position was Vice President of Product Development at Non-Procedural Systems. In addition, he has held software management positions with such companies as Pitney Bowes and Dun and Bradstreet, acquiring a broad range of software product development and support experience. Dr. Russell holds degrees from Harvard, the University of Wisconsin, and the National University of Ireland.

Karl Kempf is Principal Scientist in the Knowledge Application Laboratory at Intel Corporation located in Santa Clara, California, and is a Mentor in the Center for Integrated Systems at Stanford University. He was the founding co-chairman (with Mark Fox of Carnegie Mellon University) of the AAAI Special Interest Group in Manufacturing (SIGMAN), and currently serves as its benchmark secretary. He is also on the editorial board of IEEE Expert. Karl received a B.S. in Chemistry and a B.A. in Physics from Otterbein College (as a mathematics major), and did graduate work at Stanford University and The University of Akron, receiving his PhD in Computer Science. His research interests center on spatial and temporal reasoning systems applied to robots specifically and to manufacturing systems in general. Karl has been involved in the design, development, and delivery of six successful AI-based manufacturing applications, and has produced over 40 publications in the area of AI and Manufacturing.