South African
Computer
Journal
Number 11
May 1994

Suid-Afrikaanse
Rekenaar-
tydskrif
Nommer 11
Mei 1994

Computer Science and Information Systems

Rekenaarwetenskap en Inligtingstelsels

The South African Computer Journal
An official publication of the Computer Society of South Africa and the South African Institute of Computer Scientists

Die Suid-Afrikaanse Rekenaartydskrif
'n Amptelike publikasie van die Rekenaarverenging van Suid-Afrika en die Suid-Afrikaanse Instituut vir Rekenaarwetenskaplikes

Editor
Professor Derrick G Kourie
Department of Computer Science
University of Pretoria
Hatfield 0083
Email: dkourie@dos-lan.cs.up.ac.za

Subeditor: Information Systems
Prof John Shochot
University of the Witwatersrand
Private Bag 3
WITS 2050
Email: 035ebrs@witsvma.wits.ac.za

Production Editor
Dr Riël Smit
Mosaic Software (Pty) Ltd
P.O.Box 16650
Vlaeberg 8018
Email: gds@cs.uct.ac.za

Subscriptions

	Annual	Single copy
Southern Africa:	R45,00	R15,00
Elsewhere:	$45,00	$15,00

to be sent to:
Computer Society of South Africa
Box 1714 Halfway House 1685

Phone: +27 (11) 315-1319 Fax: +27 (11) 315-2276

Intelligent Production Scheduling: A Survey of Current Techniques and An Application in The Footwear Industry

Vevek Ram
Department of Computer Science, University of Natal, PO Box 375, Pietermaritzburg 3200, South Africa

Abstract

Computer Based Scheduling is an important activity especially in automated environments such as CIM (Computer Integrated Manufacturing) where it is an integral part of the total manufacturing function. Developments in Artificial Intelligence have allowed the scheduling problem to be solved with the use of the Knowledge-Based approach. This thrust, known as intelligent scheduling makes use of many varied techniques in both knowledge representation and inference design. Also, some approaches view the scheduling problem as a subset of the larger problem of Planning. This paper examines the various Knowledge-Based approaches to the solution of the scheduling problem in order to establish the state of the art. A prototype system which is being developed for application in the footwear manufacturing industry is also presented. This system makes use of a hierarchical planning system to aggregate ordered items into production units which are then scheduled by the scheduling subsystem using multiple context reasoning with truth maintenance. The system is discussed in terms of its design and implementation and also the unique advantages it offers over other approaches.
Keywords: *Intelligent Scheduling, Multiple context reasoning*
Computing Review Categories: *I.2.1, J.1.*

Received: June 1993, Accepted: October 1993, Final version: February 1994.

1 Introduction

Scheduling has proved to be a rich research area and many useful and interesting techniques have been developed. Unfortunately, not all researchers working in the area share the same view or definition of scheduling. Classical operations research defines scheduling as a process of defining a sequence of operations whose execution realizes some objective and where each operation is defined in terms of the time and resources required for its successful execution [17]. Many Artificial Intelligence researchers would argue that this is also an accurate description of Planning. Barr and Feigenbaum [2], and Chang and Wee [5] for example, describe the planning process as "the process of developing a sequence of actions to achieve a goal". This is because planning has been a focus of Artificial Intelligence for many years and many techniques used in planning can also be used in scheduling. Some researchers in Artificial Intelligence however, do make a clear distinction between planning and scheduling. Findler and Lo [14], argue that planning tries out alternative methods of attack in a coarse grained way, with the implication that scheduling is finer or a subsequent activity. Noronha and Sharma [27], define planning as the process of devising, designing or formulating something to be done and scheduling as the process of devising or designing a procedure for a particular objective, specifying the time and sequence for each item. Solberg [33] suggests that apart from distinctions in the definition of scheduling, there are three different perspectives or paradigms in the scheduling research area. Firstly, there is the optimization view which examines planning and scheduling problems as optimization problems. An example of this view is discrete-event simulation. Secondly there is the data centred view which emphasises the data management issues of manufacturing and scheduling. MRP is an example of this view. Finally, there is the control view which focuses on aspects necessary to keep a schedule under control in the presence of disturbances. Real time scheduling and reactive planning are examples of the control view. Naronha and Sharma [27] have developed an excellent taxonomy which resolves the ambiguity in definitions and serves as a practical research framework. In this taxonomy, shown in Figure 1, scheduling is viewed as a subset of planning. This view is espoused in this paper.

Traditional or conventional approaches to scheduling have experienced difficulties due the large number of variables considered and the formal algorithmic methods used which often lead to combinatorial explosion. In fact, large scheduling problems are NP complete and a simple situation with j jobs and m machines can lead to $(j!)^m$ solutions. Recent developments in Artificial Intelligence have resulted in a range of tools and techniques that can be readily applied to scheduling problems. The following section describes some of the more important work in this area.

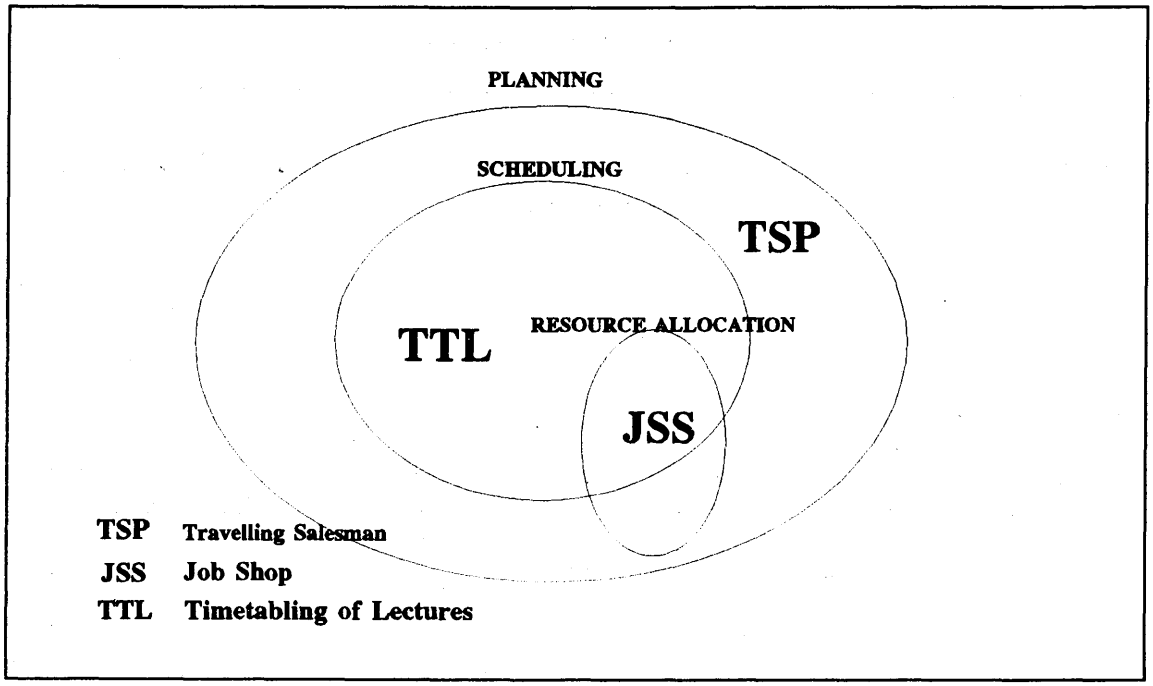

Figure 1. The Naronha-Sharma taxonomy of scheduling and planning

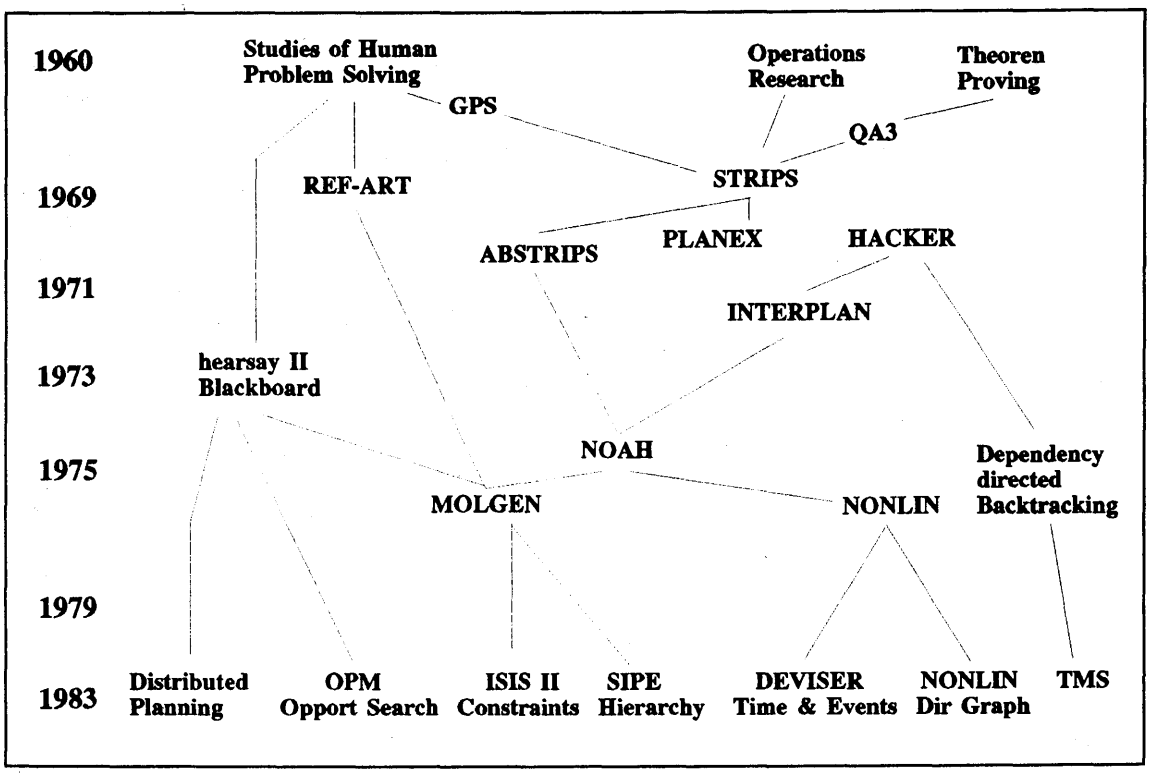

Figure 2. Historical development of AI planning and scheduling systems (adapted from [23])

Table 1. AI-based planning systems

System	Year	Reference
Graph Traverser	1966	Michie [26]
GPS	1969	Ernst and Newell [9]
STRIPS	1971	Fikes and Nilsson [11]
ABSTRIPS	1974	Sacerdoti [29]
HACKER	1975	Sussman [35]
INTERPLAN	1975	Tate [38]
NOAH	1977	Sacerdoti [30]
NONLIN	1977	Tate [38]
MOLGEN	1981	Stefik [34]
SIPE	1983	Wilkins [40]
ISIS	1983	Fox and Smith [15]
DEVISER	1983	Vere [39]
BATTLE	1985	Slagle [31]
BB1	1985	Hayes-Roth [18]
OPLAN	1985	Bell and Tate [3]
OPIS	1986	Smith *et al* [32]
DISPATCHER	1986	Acock and Zemel [1]
FACTOR	1986	Factrol [10]
KIP	1987	Luria [24]
ISA	1987	Kanet [20]
SONIA	1988	Collinot [6]
PLATO-Z	1988	O'Grady [28]
PLANEX	1989	Zozaya-Gorostiza [41]
LIBRA	1989	Fiksel and Hayes-Roth [12]
CORTES	1991	Sycara *et al* [36]
LOGOS	1991	Meng and Sullivan [25]

2 AI Approaches to Scheduling

Figure 2 illustrates the historical development of Artificial Intelligence planning and scheduling systems. As a discussion of each of the systems indicated is beyond the scope of this paper, a comprehensive list of systems and their accompanying references is shown in Table 1.

This section examines the broad categories of Artificial Intelligence based scheduling.

Hierarchical scheduling concerns the development of schedules by using different levels of abstraction. These levels form a hierarchy of representations of a schedule in which the highest level is an abstraction of the schedule and the lowest level is a detailed schedule, sufficient to solve the problem. An abstraction level is distinguished by the granularity, or the fineness of detail of the discriminations it makes in the world [40]. Some known hierarchical scheduling systems include NOAH [30], ABSTRIPS [29] and SIPE [40].

Non-hierarchical scheduling concerns the development of schedules by ordering operations at a single level of abstraction. A non-hierarchical scheduler develops a sequence of problem solving actions that achieves a goal. Any goal can be reduced to set of simpler subgoals, or, means-end analysis can be used to reduce the differences between the current state and the desired state of the world. A disadvantage with non- hierarchical scheduling systems is their failure to distinguish between actions which are critical to the success of the problem solving process and those that are just details. STRIPS [11], and HACKER [35] are examples of non-hierarchical scheduling systems.

Script-based scheduling makes use of skeleton schedules that are recalled from a store of skeleton schedules instead of generating them as in the case of hierarchical schedulers. These stored schedules contain the outlines for solving many different kinds of problems. The scheduling process proceeds in two steps: first, a skeleton schedule is found that is applicable to the given problem and then the abstract steps in the schedule are filled in with problem solving operators from the particular problem context. Script-based planning or scheduling is a subcategory of hierarchical planning. An example of this type of scheduling is found in the MOLGEN system [34].

Opportunistic scheduling methods are characterized by their flexible approach. Schedules are developed in a fragmentary manner and synthesised as opportunities arise. Opportunistic scheduling has been developed by Hayes-Roth and Hayes-Roth [19] who argue that it is more efficient than hierarchical scheduling when the scheduling problem is complex and also that it is closer in nature to the way in which human schedulers schedule. An important concept in the framework of opportunistic scheduling is that of island driving, in which a problem solver finds part of a solution that he thinks is correct, an island, and extends the solution from there, possibly toward another island. These "islands" can drastically reduce the range or solution space of the strategic problem. Opportunistic scheduling makes use of the blackboard framework to represent the complex control structure of human scheduling.

Constraint-Based scheduling uses constraints to represent the interactions between subproblems. Constraints are dynamically formulated during planning and used to coordinate the solutions of nearly independent subproblems. MOLGEN [34] makes use of constraint formulation, which is the adding of new constraints as commitments in the overall design or planning process, constraint propagation, which is the creation of new constraints from old ones and constraint satisfaction, which is the operation of finding values for variables so that a set of constraints on the schedule variables is satisfied. Other constraint-based systems are ISIS [15], and CORTES [36].

Reactive Planning and Scheduling is the process of replanning when unexpected changes in the environment occur during plan or schedule execution that invalidate the original plan or schedule. This approach is useful when the application domain is characterised by uncertainty or illspecification of the initial configuration. Noronha and Sarma [27] argue that in non real-time situations, reactive planning and non-reactive planning are no different since the techniques used to generate the original plan or schedule can be used to generate a new one in the restated problem. OPIS [32] and LOGOS [25] are examples of reactive planning.

Other Approaches to scheduling concern techniques from diverse areas such as genetic engineering and statistical mechanics that have been used successfully in the scheduling area. Simulated annealing [22] is a local optimization procedure that "melts" a system or schedule being

optimized at a high temperature and slowly lowering the temperature until the system "freezes" in an optimum state. Since it is an optimization technique, it does not generate schedules but is capable of optimizing any random configuration. Scheduling using this technique has been used by Brandimarte *et al* [4]. Genetic algorithms [16] can also be used to locally optimize schedules. A simplified form of this technique uses three operators viz; reproduction, crossover and mutation to generate successive populations (schedules) of increasing optimality. Syswerda [37] describes an application using this method in the area of laboratory scheduling.

Nonmonotonic and truth maintenance based reasoning have the potential to be used for generating schedules although this has not been explored to any great depth [13, 21]. The scheduling system described later exploits this alternative.

3 ORDPLAN: An Intelligent Planning and Scheduling System

ORDPLAN is a combination of a hierarchical planner and a scheduler that is being developed for the footwear manufacturing industry. Although the system is designed specifically for this application, it is readily applicable to other manufacturing situations. The footwear manufacturing industry in most countries is extremely competitive and competitive advantage can be gained through timely delivery and low cost. Both these may be achieved through effective planning and scheduling. The system is referred to as a hierarchical planner in the sense that it makes use of three planning levels or spaces in the mold of Molgen [34]. This is different from strict hierarchical planning systems which form abstractions of the problem at different levels [40]. The topmost level of the system is concerned with allocating orders to different production facilities. These facilities are physically distributed manufacturing plants with differing capabilities. At the second level, the system uses due-date and machine loading constraints to convert ordered items into homogeneous production batches. The lowest level converts these production batches into schedules. The two upper levels are essentially expert systems which use heuristic rules about construction intricacies, plant capabilities, due dates and other constraints in order to decompose the aggregate production requirements into batches. As the area of expert systems is generally well known, the following discussion will be confined to the description of the scheduling subsystem.

4 The Scheduling Subsystem

The traditional method of inference in many Artificial Intelligence problem solvers is monotonic reasoning. This assumes that axioms do not change and that conlusions drawn from them are always true even if assumed facts are later found to be false. Problem solving in the real world is characterised by making assumptions about uncertain information all the time and retracting these as they are subsequently unfounded. Nonmonotonic reasoning handles uncertainty by making the most reasonable assumptions and reasoning as if these assumptions were true. If these assumptions are proved untrue at a later stage, both the assumptions and all conclusions dependent on them must be retracted. The monitoring of assumptions and their validity or justification is done by what is generally known as a truth maintenance system [8]. The reasoning system used in ORDPLAN is made up of a rule based problem-solver which makes inferences and a truth maintenance system which maintains a record of the justification of these inferences. This is illustrated in Figure 3.

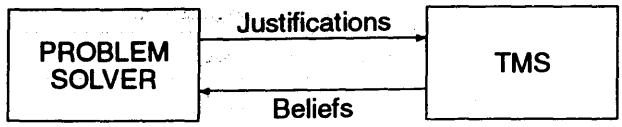

Figure 3. ORDPLAN's reasoning system

The particular form of truth maintenance that is used is based on the work of De Kleer [7] and known as Assumption Based Truth Maintenance (ATMS). In this form of reasoning or problem solving, a set of related facts and assumptions constitutes a context or world. If an additional piece of information needs to be added to the context, all possible assumptions will spawn as many new contexts. Each of these new contexts will differ by the different assumption of the last piece of information. The ATMS will destroy contexts which are inconsistent when exact information is received and assumptions are justified. Only contexts with justified assumptions (beliefs) are preserved. In the scheduling application, contexts are created for a given unit of production time which reflect all the production assumptions made in that time unit. When a context is consistent, that is, there are no constraints violated, the next set of contexts are generated. Constraints are propagated across worlds and invalid contexts in which some production constraint is violated are immediately discarded. The process of generating contexts ends when there are no more remaining time units, in this case, the end of a production day. The partial solutions represented by each world can be merged to form a complete solution or schedule. The following simple example illustrates the scheduling procedure.

Assume that there are 5 production job units. There are generally 10 or 12 pairs of footwear items in each unit, but each unit is treated as a single job since the operations performed on each item are identical. Each unit has to pass through 4 manufacturing processes. These are Clicking (leather cutting), Closing (stitching), BottomPrep (cutting of soles and heels), Lasting and Finishing (joining top and bottom halves and polishing etc). Units require different amounts of time in each process depending on the nature of the design of the items in the units. Table 2 is an example of such a production unit. Each process, consisting of many varied machine operations, is regarded as a single operation for the sake of preserving the simplicity of this example. The number in each cell is the time required for the process in minutes.

Table 2. Job processes

JOB	PROCESS			
UNIT	1	2	3	4
Job1	12	24	15	15
Job2	12	32	10	12
Job3	18	36	18	18
Job4	10	18	10	10
Job5	15	22	12	15

Table 3. A job object

Job	11234
Type	Moccassins
Customer	XYZ Bazaars
Order	1549
Operations	$[(o_1, p_1, d_1, s_1), \ldots]$
Costing	
...	
...	

' The knowledge base of the scheduling system consists of objects for each entity in the environment and rules that prescribe the procedure used for scheduling. In this case, the objects are machines, jobs, and a timerecorder. Every object contains slots which are used to store attributes of the object. The job object is illustrated below and contains slots for the type of job, the customer for whom the job is being manufactured, the order number, the operations slot and other financial and costing slots. The operations slot contains a list of tuples, each of which contains the operation description, o, the process which performs it, p, the duration, d, and the sequence position, s, of the operation.

The operations slot is effectively a guide to the fabrication of the product. All Job and Process objects inherit attribute types, default values and other values from parent objects in the same class. Rules are used to guide the scheduling process. The following rule is an example

if (process is idle) and
(job is waiting) and
(job is in_need_of process) and
(job is not complete_process)
then (assign job to process) and
(update process-queue) and
(update job-sequence)

The system uses such rules to assign jobs to machines in a given time unit that is recorded by the timerecorder object. If no contradictions (constraint violations) arise, such as an assigned process which is out of sequence, not enough time left in the production day to complete a process, or sometimes even two jobs assigned to one machine or one job assigned to two machines, then the contexts at this level are preserved and the timerecorder moves the process into the next time unit for a new level of contexts to be generated. In the first time unit (minute) for example, job1 is assigned to process 1, job2 to process 2 , job3 to process 3 and job4 to process 4. This is the first context. There are two inconsistencies in this first context. Process 2 (stitching) cannot be done on job 2 before process 1 (cutting). Also, process 4 cannot be done to job4 until it has gone through all the other processes. The TMS detects that the inconsistencies arise not out of the assignment of job2 to process2 and job4 to process4 but in the violation of the process sequence of these two jobs. These assignments can become valid later as job2 and job4 have the prerequisite process performed on them and the ATMS will then accept any context with these assignments in them at that stage as being consistent. The first valid and consistent context then is one which contains the assignments: Job1 in process1, job2 in process3, process2 and process4 are both idle. The next set of contexts are explored 10 timerecorder units (minutes) later when process3 on job2 is complete. Contexts are created in this manner until there are no more time units remaining. The chains of contexts created at this terminating stage are all consistent and thus represent a valid schedule. Contexts that are marked as inconsistent can be examined to investigate the cause of the inconsistency and to identify the constraint violation. A graphical illustration of the created contexts is shown in Figure 4.

In the implementation of the system, in smalltalk, contexts were themselves objects. Broadcast messages (messages to all objects in a certain class) would then trigger methods in each context object to re-evaluate its internal consistency with the new information contained in the broadcast. A Prolog interpreter is then accessed by the context object to perform the consistency maintenence. The Prolog inference engine controls the creation and deletion of context objects. More detail regarding the theoretical foundations of context and consistency maintenance can be found in De Kleer [7].

5 Conclusion

The use of multiple context reasoning for scheduling offers an interesting alternative to other methods that are more difficult to implement. Because of the removal of inconsistent worlds, the search space is reduced considerably and this makes the approach more efficient than other methods. Although the use of fixed dispatching rules in other methods also restrict search, in most cases only one measure is optimized and solutions which optimize more than one measure are often not considered. The examination of the worlds generated at any stage provides useful insight into the implications of choices made and the reasons for inconsistency. Systems can also be designed to offer automatic explanations of why contexts fail before they are pruned. The prototype system described has been implemented using a combination of Prolog and Smalltalk. This hybrid environment, while severely restrictive for real applications, is easily available at minimal cost for prototyping purposes. Fortunately, several large development shells implement context based reasoning mechanisms with truth maintenance and these can be used for more serious applications.

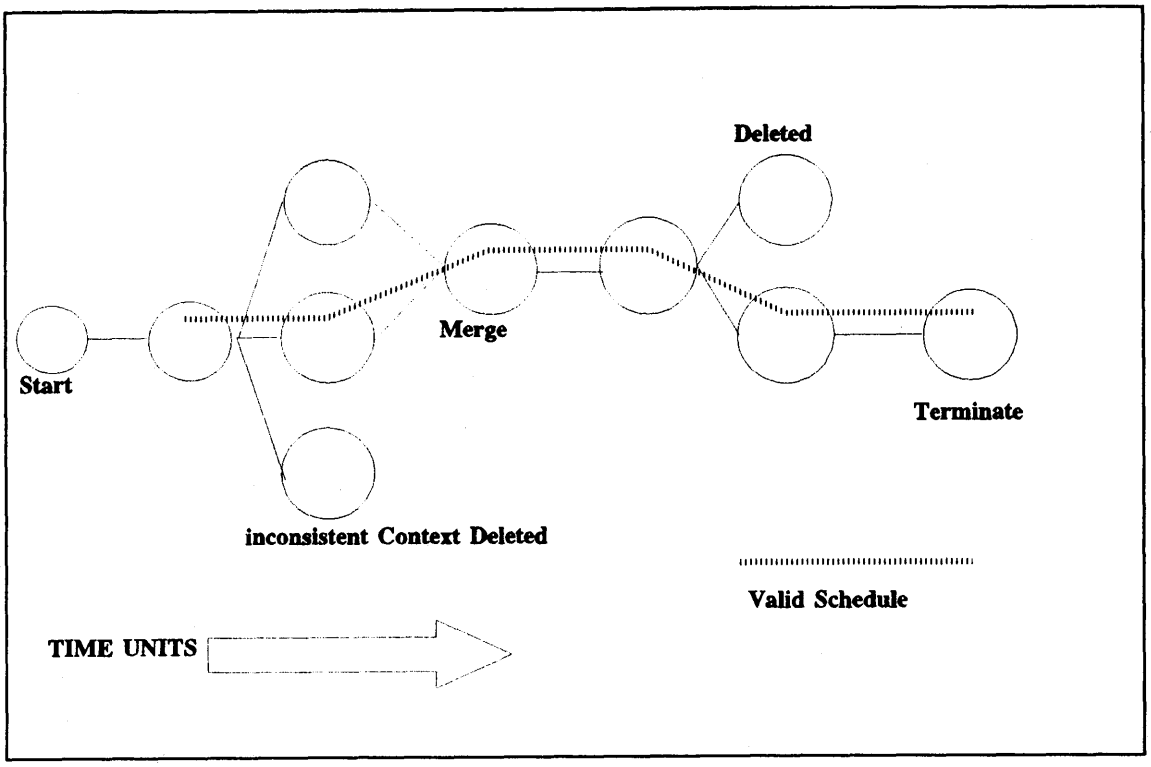

Figure 4. The generated contexts and the final schedule

References

1. M Acock and R Zemel. 'DISPATCHER: AI software for automated material handling'. In *Proc of the SME AI in Manufacturing Conference*, (1986).
2. A Barr and E Fiegenbaum. *Handbook of AI, Vol 2.* William Kaufman, Los Altos Calif, 1981.
3. C Bell and A Tate. 'Using temporal constraints to restrict search in a planner'. In *Proc of the 3rd workshop of the ALVEY planning SIG*, Sunningdale, UK, (1985).
4. P Brandimarte *et al.* 'FMS production scheduling by simulated annealing'. In *Proc of the 3rd Int Conf on Simulation in Manufacturing*, (1987).
5. K Chang and W G Wee. 'A knowledge-based planning system for mechanical assembly using robots'. *IEEE Expert*, pp. 18–30, (Spring 1987).
6. A Collinot *et al.* 'SONIA: A knowledge-based scheduling system'. *International Journal of AI in Engineering*, **3**:86–94, (1988).
7. J De Kleer. 'An assumption-based TMS'. *Artificial Intelligence*, **28**(1):127–162, (1986).
8. J Doyle. 'A truth maintenance system'. *Artificial Intelligence*, **12**(3):231–272, (1979).
9. G Ernst and A Newell. *GPS: A Case Study in Generality and Problem Solving*. Academic Press, New York, 1969.
10. Factrol Incorporated, W Lafayette, Indiana, USA. *Factrol Reference Manual.*
11. R Fikes and N Nilsson. 'Strips: A new approach to the application of theorem proving to artificial intelligence'. *Artificial Intelligence*, **1**(2):251–288, (1971).
12. J Fiksel and F Hayes-Roth. 'Knowledge systems for planning support'. *IEEE Expert*, pp. 17–23, (Fall 1989).
13. R Filman. 'Reasoning with worlds and truth maintenance in a knowledge-based programming environment'. *CACM*, **33**(4):382–401, (1988).
14. N Findler and R Lo. 'An examination of distributed planning in the world of air traffic control'. In A Bond and L Gasser, eds., *Readings in Distributed Artificial Intelligence*. Morgan Kaufman, California, (1988).
15. M Fox and S Smith. 'ISIS: A knowledge-based system for factory scheduling'. *Expert Systems*, **1**(1):25–49, (1984).
16. D Goldberg. *Genetic Algorithms: In Search of Optimization and Machine Learning*. Addison-Wesley, Mass, USA, 1989.
17. S Graves. 'A review of production scheduling'. *Operations Research*, **29**(4), (1981).
18. B Hayes-Roth. 'A blackboard architecture for control'. *Artificial Intelligence*, **26**:251–231, (1985).
19. B Hayes-Roth and F Hayes-Roth. 'Cognitive processes in planning'. Technical Report R2366ONR, Rand Corporation, California, (1978).
20. J Kanet *et al.* 'Expert systems in production scheduling'. *European Journal of Operations Research*, **20**:51–59, (1987).
21. S Kimborough and F Adams. 'Why nonmonotonic logic?'. *Decision Support Systems*, **4**:111–127, (1988).
22. S Kirkpatrick *et al.* 'Optimization by simulated an-

nealing'. *Science*, **220**(4598):671–680, (1983).
23. P Kuczora. 'Planning and project management'. In G Winstanley, ed., *Artificial Intelligence in Engineering*. Wiley, West Sussex, England, (1991).
24. M Luria. 'Concerns: A means of identifying potential plan failures'. In *Proc of the 3rd IEEE conf on AI Applications*, (1987).
25. A Meng and M Sullivan. 'LOGOS: A constraint-directed reasoning shell for operations management'. *IEEE Expert*, **6**(1):20–28, (1991).
26. D Michie and J Doran. 'Experiments with the graph traverser program'. In *Proc of the Royal Society*, volume 294, pp. 235–259, (1966).
27. S Noronha and V Sarma. 'Knowledge-based approaches for scheduling problems: A survey'. *IEEE Trans on Knowledge and Data Engineering*, **3**(2):160–171, (1991).
28. P O'Grady and P Lee. 'An intelligent cell control system for automated manufacturing'. *Int Jnl of Production Research*, **26**(5):845–861, (1988).
29. E Sacerdoti. 'Planning in a hierarchy of abstraction spaces'. *Artificial Intelligence*, **5**:115–135, (1974).
30. E Sacerdoti. *A Structure of Plans and Behaviour*. Elsevier, New York, 1977.
31. J Slagle and H Hamburger. 'An expert system for a resource allocation problem'. *CACM*, **28**(9):994–1004, (1985).
32. S Smith *et al.* 'Constructing and maintaining detailed plans: Investigations into the development of knowledge-based factory scheduling systems'. *AI Magazine*, pp. 45–61, (Fall 1986).
33. J Solberg. 'Production planning and scheduling in CIM'. In Ritter, ed., *Information Processing '89*. Elsevier, Holland, (1989).
34. M Stefik. 'Planning with constraints (Molgen Part1)'. *Artificial Intelligence*, **16**:111–139, (1981).
35. G Sussman. *A Computer Model of Skill Acquisition*. Elsevier, New York, 1975.
36. K Sycara *et al.* 'Resource allocation in distributed factory scheduling'. *IEEE Expert*, **6**(1):29–40, (1991).
37. G Syswerda. 'Schedule optimization using genetic algorithms'. In Davis, ed., *Handbook of Genetic Algorithms*. Van Nostrand, New York, (1991).
38. A Tate. 'Generating project networks'. In *Proc of the IJCAI-77*, Los Altos, (1977). Morgan Kaufman.
39. S Vere. 'Planning in time: Windows and durations for activities and goals'. *IEEE Trans on Pattern Anal and Machine Recog*, **5**(3):246–267, (1983).
40. D Wilkins. 'Hierarchical planning: Definition and implementation'. In B DuBoulay, D Hogg, and L Steels, eds., *Advances in Artificial Intelligence*. Elsevier, Holland, (1987).
41. C Zozaya-Gorostiza *et al. Knowledge-Based Process Planning for Construction and Manufacturing*. Academic Press, London, 1989.

Notes for Contributors

The prime purpose of the journal is to publish original research papers in the fields of Computer Science and Information Systems, as well as shorter technical research papers. However, non-refereed review and exploratory articles of interest to the journal's readers will be considered for publication under sections marked as Communications or Viewpoints. While English is the preferred language of the journal, papers in Afrikaans will also be accepted. Typed manuscripts for review should be submitted in triplicate to the editor.

Form of Manuscript

Manuscripts for *review* should be prepared according to the following guidelines.

- Use wide margins and $1\frac{1}{2}$ or double spacing.
- The first page should include:
  - title (as brief as possible);
  - author's initials and surname;
  - author's affiliation and address;
  - an abstract of less than 200 words;
  - an appropriate keyword list;
  - a list of relevant Computing Review Categories.
- Tables and figures should be numbered and titled. Figures should be submitted as original line drawings/printouts, and not photocopies.
- References should be listed at the end of the text in alphabetic order of the (first) author's surname, and should be cited in the text in square brackets [1–3]. References should take the form shown at the end of these notes.

Manuscripts accepted for publication should comply with the above guidelines (except for the spacing requirements), and may be provided in one of the following formats (listed in order of preference):

1. As (a) LaTeX file(s), either on a diskette, or via e-mail/ftp – a LaTeX style file is available from the production editor;
2. As an ASCII file accompanied by a hard-copy showing formatting intentions:
   - Tables and figures should be on separate sheets of paper, clearly numbered on the back and ready for cutting and pasting. Figure titles should appear in the text where the figures are to be placed.
   - Mathematical and other symbols may be either handwritten or typed. Greek letters and unusual symbols should be identified in the margin, if they are not clear in the text.

   Further instructions on how to reduce page charges can be obtained from the production editor.
3. In camera-ready format – a detailed page specification is available from the production editor;
4. In a typed form, suitable for scanning.

Charges

Charges per final page will be levied on papers accepted for publication. They will be scaled to reflect scanning, typesetting, reproduction and other costs. Currently, the minimum rate is R30-00 per final page for LaTeX or camera-ready contributions and the maximum is R120-00 per page for contributions in typed format (charges include VAT).

These charges may be waived upon request of the author and at the discretion of the editor.

Proofs

Proofs of accepted papers in categories 2 and 4 above will be sent to the author to ensure that typesetting is correct, and not for addition of new material or major amendments to the text. Corrected proofs should be returned to the production editor within three days.

Note that, in the case of camera-ready submissions, it is the author's responsibility to ensure that such submissions are error-free. However, the editor may recommend minor typesetting changes to be made before publication.

Letters and Communications

Letters to the editor are welcomed. They should be signed, and should be limited to less than about 500 words.

Announcements and communications of interest to the readership will be considered for publication in a separate section of the journal. Communications may also reflect minor research contributions. However, such communications will not be refereed and will not be deemed as fully-fledged publications for state subsidy purposes.

Book reviews

Contributions in this regard will be welcomed. Views and opinions expressed in such reviews should, however, be regarded as those of the reviewer alone.

Advertisement

Placement of advertisements at R1000-00 per full page per issue and R500-00 per half page per issue will be considered. These charges exclude specialized production costs which will be borne by the advertiser. Enquiries should be directed to the editor.

References

1. E Ashcroft and Z Manna. 'The translation of 'goto' programs to 'while' programs'. In *Proceedings of IFIP Congress 71*, pp. 250–255, Amsterdam, (1972). North-Holland.
2. C Bohm and G Jacopini. 'Flow diagrams, turing machines and languages with only two formation rules'. *Communications of the ACM*, **9**:366–371, (1966).
3. S Ginsburg. *Mathematical theory of context free languages*. McGraw Hill, New York, 1966.

South African Computer Journal

Number 11, May 1994
ISSN 1015-7999

Suid-Afrikaanse Rekenaar-tydskrif

Nommer 11, Mei 1994
ISSN 1015-7999

Contents