SCHEDULER: A KNOWLEDGE-BASED SYSTEM FOR PERSONAL SCHEDULING

Wenran Zhang and Michael N. Huhns

Center for Machine Intelligence
Department of Electrical and Computer Engineering
University of South Carolina, Columbia, SC 29208

ABSTRACT

The design and implementation of an expert system for personal scheduling is presented. In this system, a user can schedule activities which are daily periodic, weekly periodic, and nonperiodic. In addition, holidays and deadlines can be scheduled. Tasks which have a deadline are scheduled with increasing frequency by the system until it is informed that the tasks have been completed. The system uses multiple knowledge representations for different situations. Frames are used for declarative knowledge about schedules; production rules are used for control knowledge. Both rote learning and inductive learning are used by the system to accumulate event knowledge and to provide defaults. SCHEDULER also provides a real-time scheduling environment and a complete calendar.

I. INTRODUCTION

Several systems for time reasoning and scheduling have been reported to date. However, few have been designed for scheduling the activities of individuals in a variety of occupations, also, few investigations into the learning of personal habits and characteristics by intelligent computer assistants have been conducted. According to a proverb, "Time is life." Obviously, a system, which is able to help a user make long or short term time arrangements and bring all personal time and activities under an easily observable and controllable condition, is badly needed. Moreover, since personal scheduling may be an expensive secretarial function, it is worthwhile to construct a knowledge-based system in this application domain.

This paper presents work on the design and implementation of an expert system for personal scheduling, called SCHEDULER. The approach draws from studies in database engineering, knowledge engineering, software engineering, and computer learning. There are three major motivations for this research. The first is to analyze a scheduling system which could be used by individuals in a variety of occupations. The second is to do an experiment in using multiple knowledge representations to cope with different situations in one system. The third is to provide a foundation and some leads for later extension and research in the areas of computer learning, computer advisement, natural language understanding, and office automation.

II. DESIGN AND IMPLEMENTATION

In SCHEDULER, a user's activities to be scheduled are partitioned into three primary groups: daily periodic, weekly periodic, and nonperiodic. Two additional groups are special days, used for scheduling holidays, and deadlines. This partitioning provides the basis for making a user's time arrangements controllable and observable.

The system is designed as seven major subsystems, as shown in Figure 1. The natural language understanding subsystem (not yet implemented) is used to parse and understand a user's queries. The performance subsystem is used for updating and retrieving schedules. The advising element gives advice to the user. The computer system clock connected to SCHEDULER provides the system with a real-time scheduling environment.

The database subsystem consists of the daily-periodic schedule frames, the weekly-periodic schedule frames, the nonperiodic schedule frames, the deadline frames, and the special day frames. All these frames have two levels. The

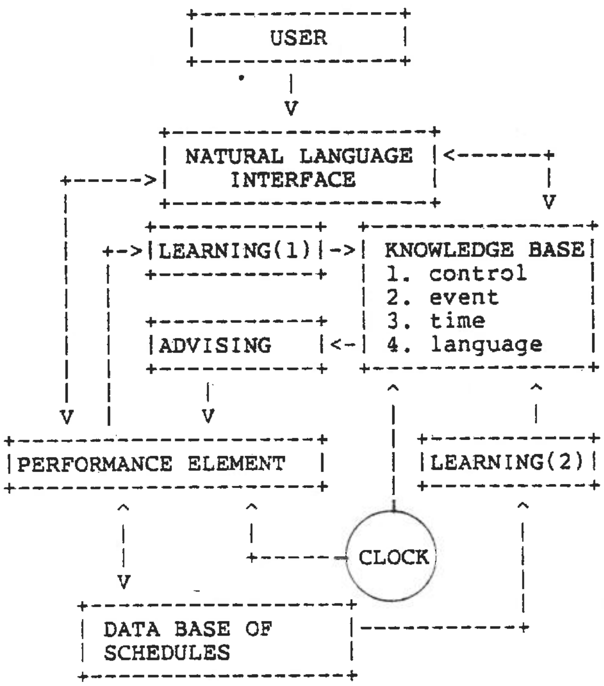

Figure 1. The Flow of Control and Flow of Information in SCHEDULER

first-level frames store time and activity arrangements, from which time knowledge can be calculated through a fetching and sorting procedure. The second-level frames store detailed information for different events. It is assumed that a user is free of daily periodic activities on a scheduled holiday and that a holiday may also be scheduled as nonperiodic, weekly periodic, or yearly periodic. An active cycle period may be specified for any periodic activity.

The knowledge base consists of control knowledge, event (task, special day, or deadline) knowledge, time knowledge, and natural language understanding knowledge. Time knowledge is directly calculated from the data base. Event knowledge learned by the two learning elements are also represented as frames similar to the second-level data base.

The learning (1) module uses rote learning to get knowledge directly from the user. The learning (2) module uses induction to learn time knowledge and event knowledge from the database environment. Two inductive learning algorithms for event knowledge are investigated. The idea of the first one is to store a second-level frame as an initial event sample for later reference, which is updated when another event of the same activity is scheduled. In this way, the knowledge for an event can be refined and accumulated (see Figure 2. (1)). The idea of another algorithm is to use a new event for an activity as a filter frame (or transfer frame) to the former sample frame, the identical values and slots between the two frames are learned as knowledge(see Figure 2. (2)) [1]. Both algorithms belong to the class of "learning from examples" [5]. The first one is implemented in this system.

```
old sample      new event       new sample
for activity1   for activity1   for activity1
+----------+    +----------+    +----------+
|slot|value|    |slot|value|    |slot|value|
|----------|    |----------|    |----------|
| x  |v1   |  + | x  | v1  |  = | x  | v1  |
|----------|    |----------|    |----------|
| y  |v2   |    | y  | v3  |    | y  |v3/v2|
+----------+    |----------|    |----------|
                | z  | v4  |    | z  | v4  |
                +----------+    +----------+

                     (1)

old sample      new event       new sample
for activity1   for activity1   for activity1
+----------+    +----------+    +----------+
|slot|value|    |slot|value|    |slot|value|
|----------|    |----------|    |----------|
| x  |v1   |  + | x  | v1  |  = | x  | v1  |
|----------|    |----------|    |----------|
| y  |v2   |    | y  | v3  |    | y  | ?   |
+----------+    |----------|    +----------+
                | z  | v4  |
                +----------+

                     (2)
```

Figure 2. Learning-from-examples events

The system is implemented as one process. The implementation environment is Franz Lisp and OPS5 languages executing on a VAX-11/780 under the VMS operating system. It is tested by both top-down and bottom-up techniques.

In the implementation, the program is organized into three parts. The first part consists of the data base, the event and time knowledge base, and a garbage collector, all of which are implemented as frames in Franz Lisp. The second part consists of a set of procedures which are called by the third part to perform different functions. This part is also implemented in Franz Lisp. The third part consists of control knowledge (also language knowledge for further extension) in the form of production rules grouped into modules implemented in OPS5.

The system, in its present version, can perform the following functions:

1. Understanding informal time phrases, such as "today", "tomorrow", "next Monday", etc. in a real time environment;
2. Finding appropriate periodic or nonperiodic time slots and reporting conflicts;
3. Accepting formal queries and updating schedules in terms of long or short period time arrangements. These updating tasks include scheduling daily periodic, weekly periodic, and nonperiodic activities, scheduling deadlines, scheduling holidays on any date in any detail, and changing the time for any activity on any date.
4. Learning and providing knowledge about a event when updating;
5. Showing time arrangements and specific tasks for any date in any detail;
6. Showing current time and scheduled tasks;
7. Providing reminders of scheduled deadlines and special days beforehand;
8. Being polite to the user by making greetings according to different time;
9. Fetching and saving old schedules as personal records.

The following shows some frame representations and how the system works for some sample queries.

<EXAMPLE 1: Show current time and tasks.>

```
--> (SCHEDULER!)

[load 'frames1]

[load 'frames2]

------------------------------------------

          GOOD MORNING!
          THIS IS SCHEDULER.

------------------------------------------

Enter a character to continue >> c

START
-----
Please specify one of the following,

         0. EXIT
         1. show current time
            and task
         2. show calendar
         3. update schedule
         4. show schedule
         5. save old schedule

here >> 1

------------------------------------------
    TIME:  Sun  Apr  1  7:07:03 1984
------------------------------------------
Current task:
 7:00 --  7:30 breakfast
       daily Jan 1 1984 to May 31 1984
Next task:
 7:30 --  8:30 go church
       weekly Jan 1 1984 to Dec 31 2000
------------------------------------------
```

<EXAMPLE 2: Show one day schedules.>

```
SHOW ONE DAY SCHEDULE
---------------------
Here 'next' means two days later if in
the same week, otherwise in the next week.
Specify one of the following,

  0. LEAVE             6. next Wednesday
  1. today             7. next Thursday
  2. tomorrow          8. next Friday
  3. the day after tomorrow
  4. next Monday       9. next Saturday
  5. next Tuesday     10. next Sunday

or you may specify any date as,
  e.g., "(Apr 22 1984)"

>> tomorrow

   (M Apr 2 1984)

 0:00 --  7:00 sleep
       daily Jan 1 1984 to May 31 1984
 7:00 --  7:30 breakfast
       daily Jan 1 1984 to May 31 1984
 8:00 --  8:55 E873
       weekly Jan 16 1984 to May 12 1984
 9:15 -- 10:00 see Jack
12:00 -- 13:00 lunch
       daily Jan 1 1984 to May 31 1984
14:00 -- 15:15 E525
       weekly Jan 16 1984 to May 12 1984
16:30 -- 17:30 jogging
       daily Jan 1 1984 to May 31 1984
```

18:00 -- 18:30 supper
daily Jan 1 1984 to May 31 1984
20:00 -- 24:00 programming
daily Mar 1 1984 to Apr 30 1984

DEADLINES IN 30 DAYS

application for admission
_____ May 1 1984

Detailed information may be available. If you need any, please specify the activity or enter "n" to continue.

Specify here >> E873

(M Apr 2 1984) E873:

is_a	:	class
place	:	E304
topic	:	image processing
by	:	Dr. Huhns
attendees	:	graduate students

III. DISCUSSION

Among the reported systems, some are for jobshop scheduling and some are for assisting a manager in scheduling a team [2], while the domain for SCHEDULER is different. It is for personal scheduling, the users of which are people from different facets of society. Since different users have different activities and the activities of each are often subject to change, the situations faced by SCHEDULER are more informal, general, and irregular than faced by the other reported systems. However, the selected domain is limited enough for a knowledge-based system approach because it is only used by one individual. On the other hand, since personal scheduling for many people is an expensive secretarial job, a knowledge-based system and computer learning approach for this domain is suitable and worthwhile.

After the application domain is given, selection of an appropriate representation scheme is a key. It is commonly agreed that a sound representation is the foundation for success in Artificial Intelligence, be it in knowledge-based expert system building or computer learning. In other words, some representation can always be found better than the others to represent a kind of knowledge in a specific situation although in some uninteresting theoretical sense all computer-based representations are equivalent [4]. Moreover, recent research has confirmed that a single representation is usually not sufficient for a knowledge-based system, and the use of an appropriate combination of different representations may provide the system with more advantages [3]. Based on these points of view, a combination of multiple knowledge representations have been selected for building SCHEDULER: frames are used for both the data base and event knowledge base as a major representaion (refer to II), while production rules and procedures are used for control as another major representation.

Frame theory contends that 1) intelligence arises from the application of large amounts of highly specific knowledge, as opposed to a few general inferencing mechanisms, and 2) this is accomplished through the use of a library of frames, packets of knowledge that provide descriptions of typical objects and events [2]. These descriptions contain both an abstract template providing a skeleton for describing any instance and a set of defaults for typical members of the class. The defaults allow the information system to supply missing detail, maintain expectations, and notice anomalies. These features make frames a natural representation for scheduling.

On the other hand, since personal scheduling faces a variety of users and activities, the system control must be flexible. This makes production rule representation a best candidate for control knowledge (also for language knowledge). As production rules are modular, rules can be added, deleted, or modified without directly affecting other rules. Also, production rules are uniform in structure and easy to understand. The characteristics of modularity and uniformity of rules gain the system two advantages. First, the control structure may be changed easily. Second, an arbitrary number of procedures may be called by adding more rules. So using rules as a control knowledge representation provides the system with maximum flexibility and extensibility.

The information flow and representation scheme used in SCHEDULER are abstracted as Figure 3, which may be generalized as "using rules to call different procedures to handle different frames". SCHEDULER confirms that this representation scheme is able to combine the best qualities of production rules and frames as well as eliminating the disadvantages of either.

```
+---------------------------+
|      user's queries       |
+---------------------------+
     | ^            | ^
     v |            v |
+-----------+    +-----------+    +---------+
|production|--->|procedures|--->| frames  |
|  rules   |<---|          |<---|         |
+-----------+    +-----------+    +---------+
```

Figure 3. Abstract model for control

IV. CONCLUSIONS

The presented system SCHEDULER provides an easier way for personal scheduling. Computer learning approaches are made in the selected domain. Different representations used in a combination in the system yield more advantages than would a single representation. This work presents a model of an intelligent system used for personal scheduling and also provides leads for further study on using a knowledge-based expert system to replace secretarial functions in the office automation domain.

The system is time efficient for performing most scheduling functions and responds in approximately one second. The major limitation is that its intelligence can only be used to help the user to make time arrangements and provide knowledge about events. The system is presently being extended to provide

1. a natural language interface so that it can accept informal queries;
2. categories of event knowledge so that the information provided is more reasonable and accurate;
3. a learning capability for a user's habits in order to give better scheduling advice.

BIBLIOGRAPHY

1. P. H. Winston, "Learning by Creating and Justifying Transfer Frames", Artificial Intelligence, vol. 19, no. 3, 1982.
2. I. P. Golstein, "Using Frames in Scheduling", Proc. of the 5th Int. Joint Conf. on AI, August, 1977.
3. J. S. Aikins, "Prototypical Knowledge for Expert Systems", Artificial Intelligence, Vol. 20, No. 2, February. 1983.
4. P. H. Winston, Artificial Intelligence, Adison-Wesley, 1984, Chapter 2.
5. R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, Machine Learning, Tioga, 1983, Chapter 1.
6. W-R. Zhang, SCHEDULER--A Knowledge-Based System for Personal Scheduling, M.S. Thesis, Department of Electrical and Computer Engineering of University of South Carolina, 1984.

Mr. Zhang! Please respond to this.
Dr. [illegible]

17th Southeastern Symposium on System Theory March 24-26, 1985

SSST

Conference Committee

GENERAL CHAIRMAN
Dr. Charles A. Gross
Electrical Engineering
Auburn University, AL 36849
(205) 826-4330

PROGRAM CHAIRMAN
Dr. Thomas H. Shumpert
Electrical Engineering
Auburn University, AL 36849
(205) 826-4330

LOCAL ARRANGEMENTS
Mr. James R. Wilbanks
Engineering Extension Service
Auburn University, AL 36849
(205) 826-4370

PUBLICATION & PUBLICITY
Dr. J. David Irwin
Electrical Engineering
Auburn University, AL 36849
(205) 826-4330

Dr. Sidney N. James
Electrical Engineering
Auburn University, AL 36849
(205) 826-4330

SOCIAL ACTIVITIES
Dr. Dallas W. Russell
Electrical Engineering
Auburn University, AL 36849
(205) 826-4330

Preliminary Announcement and Call for Papers

Sponsored by: Auburn University Electrical Engineering Department, Engineering Extension Service, and Computer Science and Engineering Department, the Alabama Section and the Montgomery Subsection of IEEE, and the Auburn Chapter of ASPE

In addition to invited sessions on specific topics, a partial list of topics includes:

- Linear Systems
- Nonlinear Systems
- Time-varying Systems
- Adaptive Control Systems
- Modeling and Simulation
- System Identification
- Interactive Design
- Pattern Recognition
- Instrumentation
- Power Systems
- Control Systems
- Digital Systems
- Computer Systems
- Microprocessor Systems
- Multiprocessor Systems
- Networks
- Digital Signal Processing
- Radar Systems
- Electromagnetics
- Communications Systems
- Social and Economic Systems
- Robotics

Original papers from all areas of systems theory, design, and applications are encouraged. In keeping with one of the fundamental aims of these SSST conferences, graduate students are strongly encouraged to present their theses and dissertations research. In particular, papers of interest to the members of the following IEEE Societies are being solicited: Computer; Circuits and Systems; Control Systems; Power Engineering; and Systems, Man, and Cybernetics. All papers will be included in the proceedings which will be available at the time of registration.

Author's Schedule:

	Mail no later than:
Summary (300-500 words)	November 1, 1984
Notification of Acceptance	December 1, 1984
Full Camera Ready Manuscript	January 15, 1985
Presentation	March 24-26, 1985

Prospective authors are invited to submit 3 copies of a 300-500 word summary. Please indicate your paper's appropriate classification (theory, design, applications, tutorial/review), and its general subject matter via keywords (systems, identification, CAD...) to help in identifying the best session for your paper. Please send these to the Program Chairman, Dr. Thomas H. Shumpert, Electrical Engineering, Auburn University, AL 36849-3501.

— PLEASE POST —