Planning and Scheduling for workflow domains

María D. R-Moreno [1], Daniel Borrajo [2] Angelo Oddi [3], Amedeo Cesta [3], and Daniel Meziat [1]

[1] *Departamento de Automática. Universidad de Alcalá.*
Carretera Madrid-Barcelona, Km. 33,600. 28871 Alcalá de Henares (Madrid), Spain.
{mdolores, meziat}@aut.uah.es

[2] Departamento de Informática. Universidad Carlos III de Madrid.
Avda. de la Universidad, 30. 28911 Leganés (Madrid), Spain.
dborrajo@ia.uc3m.es

[3] ISTC-CNR-Italian National Research Council
Viale Marx 15, I-00137 Rome, Italy.
{a.oddi, a.cesta}@istc.cnr.it

Abstract

One of the main obstacles in applying AI planning techniques to real problems is the difficulty to model the domains. Usually, this requires that the people that have developed the planning system carry out the modeling phase, since the representation depends very much on a deep knowledge of the internal working of the planning tools. Since, in Business Process Reengineering (BPR), there has already been some work on the definition of languages that allow non-experts to enter knowledge on processes into the tools, we propose here the use of one of such languages to enter knowledge on the organisation processes.

As instances of this domain, we will use the workflow modeling tool SHAMASH, where we have exploited its object oriented structure and rule-base approach to introduce the knowledge through its user-friendly interface. Then, we have used a translator to transform it into predicate logic terms. After this conversion, real models can be automatically generated using a planner that integrates Planning and Scheduling, IPSS. We present results in a simplified real workflow domain, the TELEPHONE INSTALLATION (TI) domain.

1 Introduction

Nowadays the efficiency of the internal workings of organisations strongly depends on the efficient management of company's resources and processes. Enhancing any organization processes is the main objective of BPR. Once the organization has been studied in depth from a process and resources perspective, corresponding models are generated in order to handle processes and resources computationally. Business processes are usually represented as workflow, that is, computerised models within which all the parameters needed for the completion of the processes can be defined: resources involved, orders, tasks, conditions, goals, quality criteria, information flow, etc. Workflow Management Systems have been widely deployed in sectors like insurance, banking, accounting, manufacturing, telecommunications, administration and customer service [10].

Although there have already been many approaches to the computer-aided design of processes, very few have focused on the automatic generation of process models that have in mind the organization resources as well as their capabilities and availability. Recently, there has been considerable interest in the application of AI techniques to Workflow Management (WM) Systems [3]. Most of these tools rely on process libraries as in [2, 9]. Instead, we will focus on how to model processes using AI P&S systems as automatic models generator instead of using processes generated by hand by the user and then saved them in libraries of processes.

In AI P&S some of the main concerns are how to represent the information and the problem solving techniques. The aims that we want to achieve with the integration proposed in this paper is two-fold: on one hand, given that the majority of BPR tools are based on objects and rules, we propose to translate this knowledge on models into the PDDL2.1 language, and on the other hand, we propose an integrated P&S system for solving the problems. We believe that systems that integrate AI P&S, as IPSS [14], are the best suitable candidates.

The paper is structured as follows: section 2 briefly describes the main concepts that Workflow Management (WM) and AI P&S systems share. Next, section 3 introduces two instances of both domains, SHAMASH and IPSS, explaining how the integration can be possible with a simple example of installing a new telephone line in a telecom company. Then, section 6 shows some results of IPSS against state of the art planners. Finally, section 7 outlines the conclusions and future work.

2 Workflow Management and AI Planning and Scheduling

The first step for the use of P&S for WM is to identify points in common by understanding the way WM and AI P&S work. The Process Model is the first stage in the adoption of a Workflow solution and involves the crucial task of revealing and recording all of the manual and automatic internal business processes of an organization. From there, the user designs, models, optimises and simulates the organisation's processes through user friendly interfaces. We include in this stage the design of the process templates that can be instantiated and enacted by a workflow system. Then comes the Process Planning where the activities required to achieve some user goals are instantiated, resources assigned, and a preliminary scheduling performed. This two stages are included in the Process Definition interface of the workflow architecture proposed by the Workflow Management Coalition (WFMC) [1]. Next, the enactment/execution stage, where agents (software or humans) carry out the activities, with the workflow system co-ordinating execution. The monitoring stage is conducted concurrently with Enactment/Execution. The system enacting the workflow is monitored, with status information being made available to human operators. Exceptions, such as deviation from the plan, and subsidiary processes are initiated to rectify problems.

In AI Planning systems the following phases can be identified: domain modelling, planning and scheduling, execution and monitoring.

Table 1 outlines at a high level the concepts that AI P&S share with the Workflow community (for a more detailed description we refer to the PLANET roadmap [3]).

[1] www.wfmc.org

Table 1: Concepts mapping between AI P&S and workflow.

AI P&S	Workflow
Modeling+Planning and Scheduling	Modeling and Scheduling
Execution	Enactment
Operators	Activities, tasks, ...
Initial State	Organisation, resources
Goals	Business goals, service provision, ...

3 The SHAMASH tool

Once the different stages between both areas have been identified, in this section we describe the features of a real workflow modeling tool, SHAMASH, and then, we make the connection between AI P&S and workflow modelling tools more precise, by describing first how to translate an organisation model described in terms of SHAMASH into a planning domain model for IPSS, how IPSS can produce the desired plan (model), and how this is translated back into SHAMASH.

3.1 The SHAMASH System

SHAMASH [1], a R&D project funded by the EU IV Esprit Program, generated a process modeling tool that allows simulation, modeling and optimisation of business processes taking into account a realistic model of the organisation.

This aspect combined with the features that the tool embodies, make SHAMASH a powerful tool for BPR. Very few tools allow its rich semantic model view of processes and resources. The richer modelling capabilities allow among others, modeling of organisation standards and rules of procedures, business goals, automatic validation and optimisation of the models, or generation of HTML/TEXT output, that allows people from the organisation to freely browse the processes in which they intervene. Basically the SHAMASH tool consists of 4 subsystems. The **Author Subsystem** that provides a user-friendly interface that enables definitions of three types of knowledge: on standards, on processes and on the organisation. The **Simulation and Optimisation Subsystem** checks the syntactic (statistically based simulation) as well as the semantic (rule-based) behaviour of the processes that the user creates. Also it is able to automatically perform an optimisation phase by which new better models are generated by searching the space of process models and using business goals as search metrics. The **Text Generation Subsystem** is responsible for maintaining coherence between the graphical and the text versions. Finally, the **Workflow Interface Subsystem** generates WPDL (Workflow Process Description Language, the standard generated by the WFMC as output of the process representation and be used as an input to any workflow engine.

Once the user defines all the organization knowledge (activities of processes, organisation structure, standards, ...) that will be part of the processes, s/he should specify how the activities are connected to each other. In the case of big processes, this stage is usually quite tedious and error prone for the user (not only from the syntactic point of view, but also from the semantic one). The focus of our work is to automatically generate the model, making sure that the established connections among activities conform a valid and efficient (according to resources usage and business goals) sequence of activities. After the model has been automa-

tically generated, the user can simulate and optimise the process using SHAMASH. Figure 1 shows a high level view of SHAMASH architecture together with the proposed integration with an AI P&S system, IPSS.

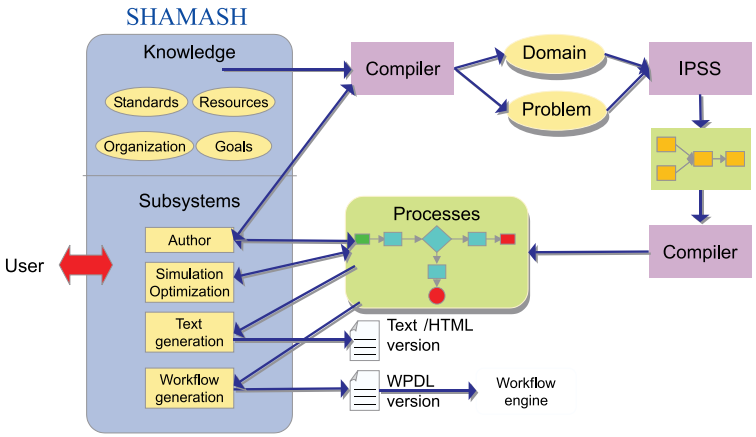

Figure 1: SHAMASH and IPSS integration.

3.2 The automatic model generation

To automate the process models generation, we need to translate the rich semantic representation of SHAMASH (objects and rules) into an AI planning language (operators, states and goals). The translation presented in this section could serve to any planner that uses logical formulae. A first step for the automatic process generation was to use the QPRODIGY planner [4]. But the disadvantages of using this planner were: the needs of domain dependent coded functions for the time reasoning, not explicit language for time and resources, and the output was given as a Total Order plan.

To overcome the drawbacks of the PRODIGY planner, we have used IPSS because it can reason about parallel actions, time and resources, or minimise the makespan at the same time as improving another quality metric, as these domains require. The automatic process generation using IPSS is as follows (see Figure 1): first, the system translates the SHAMASH objects and rules into the IPSS language. Then, IPSS produces a parallel plan of activities. And later, this sequence is translated back into SHAMASH to be presented graphically to the user. The translation has in mind the different semantics of BPR and AI planning concepts as well as their similarities.

4 IPSS: An integrated planning and scheduling system

In the present work we use the IPSS that is discussed in [14]. The planner extends the capabilities of the non-linear HSP metric planner QPRODIGY [4], integrating CSP-based scheduling abilities. IPSS works according to bi-partite architecture shown in Figure 2. Two main modules interact during problem solving: (a) IPSS-P that corresponds to the planning reasoner (it is composed of the QPRODIGY planner and a deordering algorithm that transforms the total ordered (TO) plan produced by QPRODIGY into a parallel plan the scheduling component reasons upon); (b) IPSS-S that corresponds to the scheduling reasoner (it is composed of a Temporal Constraint Network (TCN) that represent the current plan as a STP [12] and

a resource reasoner that analyzes resource conflicts according to the algorithm proposed in [16]).

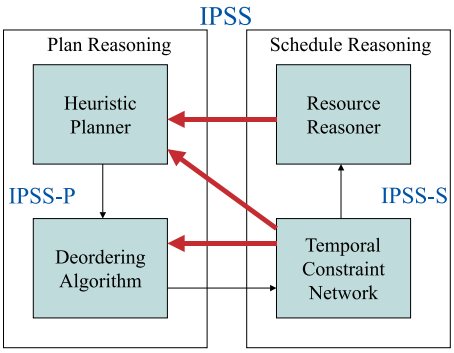

Figure 2: IPSS architecture.

The planning and the scheduling parts interleave information during the solving process. The planning module performs a bi-directional search: it begins by performing backward search from the goals, selecting respectively which goals to achieve, which operator to use to achieve the corresponding goal, and which bindings to use for its variables. Then, the deordering algorithm starts computing the link that satisfies the the preconditions of the operator that is going to be added to the TCN and must be supported by the effects of an operator that is already in the TCN. To compute the links, our algorithm starts searching from the first operator applied (origin) until the last one. The operator and the computed links are then added to the TCN.

The TCN layer represents the set of temporal constraints as a Simple Temporal Problem. Significant events, as start/end time of operators are represented as temporal variable tp_i called time points. And each temporal constraints has the form a $\leq$ tp_i-tp_j $\leq$ b, where tp_i - tp_j are time points and a and b are constants. The output from the deordering layer is trasformed in a STN that is checked for consistency. In case of inconsistency, feedback is given first to the deordering layer, to ask for a different deordering (for the sake of complexity the deordering algorithm is incomplete), then to the planner for rejecting the last choices.

The resource reasoner currently implement the conflict analysis for binary resources proposed by Smith and Cheng
citesmit-chen-93. The algorithm iteratively imposes ordering constraints for solving resource conflicts between activities that require the same resource. When there is more than one possible order it chooses as heuristic the link following the planner logical order. Again, in case of an unsolvable resource conflict (e.g., no precedence posting can solve the contention) a failure for resource inconsistency is sent to the planner that backtracks the last decision.

5 A workflow domain: installing a new telephone line

We have used in this paper a simplified version of a real workflow domain, the Telephone Installation domain, which comes from analysis of some BT processes [11]. In this domain, among other process related issues, a customer (e.g. Mary Thompson) contacts a telecom company customer service for a new telephone line. If the customer is asking a line for the first time, a spare pair of wires must be available from the house to make a connection from

the Distribution Point (DP, e.g.: telegraph pole). If no pair is available, then a new cable must be built. Then, it is necessary to check that there is a spare line card available in the exchange (in that case it is reserved/allocated). If none is available, its installation must be arranged. Installation involves making the connection at the DP (connecting a drop wire to the pair of wires that lead back to the exchange). Then, someone must: contact the customer to arrange a visit to the house to fit new network terminating equipment (NTE); arrange for an engineer to turn up on the right day/time to test the line end to end and install the NTE; allocate a telephone number to the new line and configure the exchange; update the exchange, line plant and customer records; and check with customer that s/he is happy with the service.

In this process there are several activities to perform: build a cable (BUILD-CABLE), set the spare pair of wires available (SET-SPARE-AVAILABLE), check if there is a spare line card available in the exchange (CHECK-LINE), contact the customer to fit the NTE (FIT-NTE), test the line (TEST-LINE), allocate a telephone number (ALLOCATE-NUMBER) and update customer data (UPDATE-DATA).

When the user specifies each activity in SHAMASH, s/he describes attributes specific to the activity (types of resources to be used, cost of the activity, ...), information and material elements used by the activity (data of the user, invoice, ...) as well as rules for describing pre-/post-conditions and its behaviour. An example of rule syntax in pseudocode is shown in Figure 3. This rule says that if there is a Spare Line with attributes *IsCable* equal to *No* and *Available* equal to *Yes*, and exist an Employee with *hours-left* greater or equal to three, then after the execution of the activity the field *IsCable* will be modified to *Yes* and the employee will have in its activities list to build the cable. Figure 4 shows the corresponding IPSS operator translated by the system automatically from the rule.

```
Rule Bill with properties ruleset behaviour
    If Exists a SpareLine of type MetaClassElement named var sp0
      with IsCable = ( FALSE )
      with Available = ( TRUE )
      with At-spare = ( Exists Zone of type MetaClassElement named var z0 )
    and
      Exists a Worker of type resource named var w0
          assigning to var z0 its At-worker
    Then Modify object sp0
      with IsCable is ( TRUE )
```

Figure 3: Example of a BUILD-CABLE activity rule in SHAMASH.

With respect to the problem definition, the initial conditions are generated automatically from the organisation information. This includes issues such as who works in each unit *(Assignedto Engineer-Unit Tom)*, the cost per hour of the employees belonging to a given category *(Cost Technician 548Euro)* and other information related to the details of the line (*(Building-cable 3 hours)*). In the problem, we also need to specify the goals that IPSS needs to accomplish. For example, to allocate the line with identifier number 3 *(Allocated Line3 TRUE)*. With these conditions IPSS generates a plan, with resources, units and roles needed in that process.

```
(OPERATOR BUILD-CABLE
   (params <sp0>)
   (preconds ((<w0> WORKER) (<sp0> SPARELINE) (<z0> ZONE))
      (and (IsCable <sp0> FALSE) (Available <sp0> TRUE)
           (At-spare <sp0> <z0>) (At-worker <w0> <z0>)))
   (effects ((del (IsCable <sp0> FALSE))
           (add (IsCable <sp0> TRUE))))
   (costs
           ((<unit> TIME)
            (<duration> (and (DURATION
                                  (cost-from-pred (BuildingCable <duration> <unit>)))))
           ((TIME <duration>)))))
```

Figure 4: IPSS operator corresponding to the BUILD-CABLE rule of Figure 3.

6 Experiments

In this section the different IPSS configurations are explained and the comparison against some state of the art planners in the TI domain will be presented.

6.1 IPSS *configurations used*

We have used in this paper 3 IPSS configurations. IPSS bases its search in the QPRODIGY search integrated with the algorithm that converts incrementally the partial TO plan generated by QPRODIGY during the planning process into a partial PO plan. Then, the *Ground*-CSP layer checks its consistency and provides the temporal information back to the search. IPSS-R is equal to the previous one, but including some resource leverage heuristic. The resource reasoning component (*Meta*-CSP) tells the planner what are the less used resources, and the planner selects these for assigning resources to operators. The -Q extension allows to search for more than one solution using a branch and bound procedure. The planner does not stop when it finds the first plan, but after the time bound has exceeded.

6.2 The TI *domain*

All the operations in this domain need to be executed by a human worker, and any worker can perform any action if s/he is not doing something else. There are some restrictions to consider: one worker cannot perform two operations at the same time; and a cable must be built if the spare card is not available. To perform the tasks, the worker should move to the zone where the spares or lines are, in order to accomplish the work that has to be done. This implies that there are two subtypes of planning problems within this domain: a path planning problem (moving to zones from others) and task planning (setting up lines). To relax the problem and exploit its potential parallelism, we will consider that in any case we have to allocate the line, so this can be done in parallel to any other activity instead of waiting that everything is done to allocate the line.

The modeling of the domain does not consider if the worker is or not occupied doing something else. So, apart from planners that provide serial solutions, as, for example, FF [8] or QPRODIGY [4], the solution given by other planners as LPG [6] or MIPS [5] is not correct. In order to compare not only TO planners against our approach, we have also coded this domain having in mind the availability of the chosen agent for performing the corresponding action. In this case, each action (operator) requires the worker not to be busy on performing some actions. After the execution of the action, the agent (worker) is freed by a dummy action (the *free-agent* action). This forces the planners that reason about parallel actions not to consider parallelizing two actions that require the same worker. We have called this domain the TI_OCCU domain and the original TI. The set of problems for the TI_OCCU domain is exactly the same as in the TI domain, but in all problems the availability of the worker must be explicitly set to *not-occupied*.

6.3 Experimental setup

We have generated 160 problems, distributed in four groups of forty problems. Each group has a fix number of workers: two (G2), five (G5), eight (G8) and eleven (G11). Each set of forty problems is subdivided in four subsets of ten problems each, with the following features: 4 lines and spare cards to allocate, in 4 different zones and from 2 to 4 goals; 14 lines and spare cards to allocate in 9 different zones and from 5 to 7 goals; 24 lines and spare cards to allocate, in 14 different zones and from 10 to 12 goals; and 34 lines and spare cards to allocate, in 19 different zones and from 15 to 17 goals.

Table 2 shows the planners used for TI domain. The top part of the table shows the planners that were run using the TI version (FF, and IPSS-R and QPRODIGY (QP), with and without control rules. The bottom part shows the planners that use the TI_OCCU version as they reason about activities in parallel (MIPS, IPSS and LPG). Since LPG is non-deterministic, we run it five times and then show the results of the best (LPG-MIN), worse (LPG-MAX), and median (LPG-MED).

Table 2: Number of problems solved by each planner in the TI domain with a time bound of 180 seconds.

Name	G2	G5	G8	G11	Problems solved	Percentage
IPSS-R	30	30	32	31	123	77%
QP	30	31	33	32	126	79%
FF	40	40	40	40	160	100%
IPSS	29	29	33	31	122	76%
MIPS	22	19	20	24	85	53%
LPG	40	40	40	40	160	100%

From this table we can see that FF and LPG solve all the problems. Then QP, IPSS-R and IPSS. Table 3 shows the accumulative makespan (MS) obtained for all the planners in problems solved by all in the given time limit (180 seconds). We see that IPSS-R finds the best global makespan for the 160 problems in the TI domain. Then LPG-MIN and the IPSS configurations. MIPS finds better solutions than the worse LPG run, and finally, the TO planners, as the makespan is equal to the number of operators: first FF and then the QP configurations.

If we let the planners (just LPG, QP and IPSS are able to) to search for more than one solution during 180 seconds, the results are shown in the second column of Table 3. Again,

Table 3: Accumulative Makespan (MS) and Time by each planner in the TI domain.

Name	MS No Q-version	MS Q-version	Total Time
IPSS-R	347	326	28,58
QP	699	685	11,49
FF	615	–	1,83
IPSS	425	413	25,82
MIPS	486	–	38,61
LPG-Min	346	331	3,99
LPG-Med	417	349	4,05
LPG-Max	515	351	4,14

IPSS-R-Q configuration finds the best solutions although very similar to the one found by LPG-Q.

Even if we do not show the results here, with regards to the number of operators in the solutions, FF finds the shortest solutions and IPSS configurations and LPG the longest. With respect to the execution time, FF is the faster as the third column of Table 3 shows. Due to the low percentage of problems solved by MIPS, there were some problems were IPSS-R found the solution faster than IPSS. In this table, these problems were not solved by MIPS. Because we have plotted only problems solved by all configurations, they are not considered so that is why IPSS-R obtains worse results than IPSS. But, in general, they show the superiority of the integration of the P&S approach against the rest. Also pointing out that from our experience in this kind of domains, it is better to obtain good solutions than fast and bad ones.

7 Conclusions and Future Work

In this paper, we have focused on P&S for real domains, in particular in workflow domains. We have described the advantages of using a planner that integrates P&S for modeling processes in SHAMASH, a Knowledge Based System that uses an object oriented and rule-based approach. The SHAMASH rules and objects are translated into types and operators to produce a plan that correspond to a process instance (tasks dependency). Each plan will vary depending on the resources available, elements that flow through the process and the different tasks that the organisations can introduce to adapt their processes to the market changes. With this approach the models generated are automatically validated avoiding inconsistencies in linking activities and saving time to the user.

Also outline what AI planners can gain with this approach. Generally, to specify the domain theory, a deep understanding of the AI planners terminology is needed. However, if we use a tool like SHAMASH, the description language is closer to the user and allows an automatic verification of the syntax through a friendly interface. In this integration we could have also used the PDDL2.1 language instead. Then, any planner that supports PDDL2.1 could exploit the advantages mentioned in the paper. Although the planner we have used has some features that makes it specially fit for real domains as to impose a makespan for the solutions or separate the resource reasoning from the causal reasoning (feature that is not available in most of the state of the art planners). This shows that it is a viable alternative for solving the modelling phase of workflow domains.

Acknowledgements

The SHAMASH project was funded by the EU as project number 25491 (IV Esprit programme). A complementary grant was given by the Spanish research commission, CICYT, under project number TIC98-1847-CE. We thank the partners of this project: UF, SAGE, SEMA GROUP, WIP, and EDP. We would specially like to thank all the UC3M team, the PLANET people and Paul Kearney. Through talks with him we have outlined many ideas. This work has also been partially funded by grant CICYT TAP1999-0535-C02-02 and TIC2002-04146-C05-05. Cesta and Oddi work is partially supported by ASI (Italian Space Agency) project ARISCOM.

References

[1] R. Aler, D. Borrajo, D. Camacho, and A. Sierra-Alonso. A knowledge-based approach for business process reengineering, SHAMASH. *Knowledge Based Systems*, 15(8):473–483, 2002.

[2] P. M. Berry and B. Drabble. SWIM: An AI-based System for Organizational Management. In *Procs. of the 2nd NASA Intl. workshop on Planning and Scheduling for Space. San Francisco, California.*, 2000.

[3] S. Biundo, R. Aylett, M. Beetz, D. Borrajo, A. Cesta, T. Grant, L. McCluskey, A. Milani, and G. Verfaille. *Technological Roadmap on AI Planning and Scheduling*. PLANET, 2003.

[4] D. Borrajo, S. Vegas, and M. Veloso. Quality-Based Learning for Planning. In *Working notes of the IJCAI'01 Workshop on Planning with Resources. IJCAI Press. Seattle, WA (USA)*, 2001.

[5] S. Edelkamp and M. Helmert. On the Implementation of MIPS. In *AIPS Workshop on Model-Theoretic Approaches to Planning.*, 2000.

[6] A. Gerevini, A. Saetti, and I. Serina. Planning through Stochastic Local Search and Temporal Action Graphs. *Jounal of Artificial Intelligence Research*, 20:239–290, 2003.

[7] M. Hammer and J. Champy. Reengineering the Corporation. In *Reengineering the Corporation. Harper Business Press, New York.*, 1993.

[8] J. Hoffmann. The Metric-FF Planning System: Translating Ignoring Delete Lists to Numerical State Variables. *Journal of Artificial Intelligence Research*, 2002.

[9] P. Jarvis, J. Moore, P. Chung, I. McBriar, J. Stader, M. Ravinranathan, and A. Macintosh. Applying Intelligent Workflow Management in the Chemicals Industries. In *The Workflow Handbook 2001, L. Fisher (ed), Published in association with the Workflow Management Coalition (WfMC)*, 2000.

[10] T. Lydiard, P. Jarvis, and B. Drabble. Realizing Real Commercial Benefits from Workflow: A Report from the Trenches. In *AAAI-99 Workshop on agent-Based Systems in The Business Context.*, 1999.

[11] MD. R-Moreno and P. Kearney. Integrating AI Planning with Workflow Management System. *International Journal of Knowledge-Based Systems. Elsevier*, 15:285–291, 2002.

[12] R. Dechter, and J. Pearl. Directed Constraint Networks: A Relational Framework for Casual Modelling *Procs. of IJCAI' 91.*, 1991.

[13] MD. R-Moreno, A. Oddi, D. Borrajo, A. Cesta and D. Meziat. IPSS: A Hybrid Reasoner for Planning and Scheduling. *The 16th European Conference on Artificial Intelligence, ECAI04*, 2004

[14] MD. R-Moreno. Representing and Planning Tasks with Time and Resources *PhD thesis. Dpto. de Automática, Universidad de Alcalá*, 2003.

[15] M. Veloso, J. Carbonell, A. Pérez, D. Borrajo, E. Fink, and J. Blythe. Integrating Planning and Learning: The PRODIGY Architecture. *Journal of Experimental and Theoretical AI*, 7:81–120, 1995.

[16] S. F. Smith, and C. Cheng. Slack-Based Heuristics for Constraint Satisfaction Scheduling. *Procs. of the 11th National Conference on AI (AAAI-93)*, 1993.