AIAA-94-1289-CP

# ALL FEASIBLE PLANS USING TEMPORAL REASONING

Debasis Mitra* Rasiah Loganantharaj†

{dm, logan}@cacs.usl.edu
Automated Reasoning Lab
Center for Advanced Computer Studies
University of Southwestern Louisiana
P.O. Box 44330
Lafayette, L.A. 70504

*Abstract*

In this paper we have discussed some of the issues involved in planning utilizing a temporal reasoning system. One of the advantages is that of being able to handle incomplete information. In these circumstances, there may be multiple plans available for achieving a task. Using an algorithm, designed recently by us, generating all feasible plans may be practicable in most of the instances, although the problem is NP-complete. The significance of having all feasible plans in a plan data base is quite important. We have also discussed these issues here.

## I. Introduction

Temporal reasoning is becoming an important tool for planning[4, 5, 11]. Although it is not very easy to represent planning problem as a temporal reasoning problem, the advantage of doing so is immense. In this paper we have discussed one of such advantages. Temporal reasoning allows one to represent incomplete information between operations and other primitives in a planning problem. This is a step forward from blocks world problem towards realistic planning. Under such uncertainty, there may be more than one plan which is feasible. Using an algorithm devised by us[8], all such feasible plans can be found out. The significance of the availability of such a complete plan data base is discussed in this article.

Interval based temporal reasoning scheme gives planning activity the advantage of having some parallely executable operations. In this paper we have used this interval-based temporal representation scheme. The price of having higher expressiveness in interval-based scheme is that the problem of handling incomplete information is NP-complete. Our algorithm is an efficient one, and under most of the circumstances it should be able to handle the problem in acceptable time, although the worst case growth rate remains exponential.

*Graduate Student
†Associate Professor

## II. Planning and Temporal Reasoning

Planning is the task of choosing a subset from a given finite set of operations and ordering them in time. Each operation has a set of precondition and postcondition states of the world. Preconditions are states required for an operation to become executable, and postconditions are states created by its execution. Given a set of *start* conditions and a set of *goal* conditions, the problem of planning is to see that chosen sequence of operations change the states of the world from start conditions to goal conditions. This perspective of planning is akin to a state change-based point of view in temporal reasoning, as in situation calculus. A different approach had been taken by Allen et al[2] about planning where they have tried to formalize planning with more explicit temporal reasoning. According to the conventional view, temporal identities, both operations and states of the world (or *fluents*), are related to each other in such a way that the end of one is the starting point of the other. In interval based temporal reasoning scheme this latter temporal relation is the 'meet' primitive[1]. There are 13 such other primitive relations feasible between two time intervals.

Classical planning, based on situation calculus, suffers from two shortcomings in their application in the real world. Firstly, it can not tackle a problem when two operations need to be performed at the same time to achieve an objective[3]. For example, to open a door one needs to turn door knob and push the door at the same time. This can not be easily represented in conventional planning schemes. Even under the circumstances where such parallel operations are not essential, such plans may execute faster (subject to the availability of sufficient resources for parallel execution of operations). Secondly, any incompleteness of information about the real world can not be represented in classical planning scheme. For example, one can make a phone call 'before' the class or 'after' the class, this flexibility of information can not be represented there. An interval based temporal reasoning scheme is capable of handling such situations.

In the interval-based temporal reasoning scheme, each of the fluents and operations are considered as an interval in time. They lie on a unique non-branching time line, and so, each of them is temporally related to all the other ones. A single primitive relation as a temporal relation between a pair of temporal entities indicates definite information between those two temporal assertions, whereas a disjunctive set of primitive temporal relations indicates incomplete information. In the next section, the scheme for interval based temporal reasoning will be discussed.

## II. Interval-based Temporal Reasoning

Suppose the following set of information is given in the context of petroleum exploration.
GS = gravity survey
SS = seismic survey
DA = drilling activity
PA = production activity
ER = enhanced recovery

Apart from these operations there are following fluents which are affected by the operations. [1] Surveys are done when there has not been any survey(NS) done on the area. Seismic survey produces subsurface geological knowledge(GK), it also produces huge noise(NO) caused by the artificial seismic explosions. Gravity survey can not be done during such noise. Drilling activity requires geological knowledge about the area, although it may make geological knowledge invalid. It also makes drilling wells available(DW) which are required by production activity. Production activity produces data about the reservoir(RD) which is necessary for enhanced recovery. Enhanced recovery is done for maximum production from a reservoir(MP). Actual temporal relations between them are given below.

1. NS –{finished-by}–> GS [2]
2. NS –{finished-by}–> SS
3. SS –{equal}–> NO
4. NO –{before, meet, after, met-by}–> GS
5. SS –{overlap}–> GK
6. GK –{during-inverse, finished-by, overlaps}–> DA
7. DA –{starts}–> DW
8. DW –{during-inverse, finished-by, overlaps}–> PA
9. PA –{overlaps, statrs, equal}–> RD
10. RD –{starts, equal, overlap, meet, before}–> ER
11. ER –{meets}–> MP

[1] The information given here about petroleum production is not necessarily realistic. Preconditions and postconditions are simplified to a great extent.

[2] For exact semantics of the 13 primitive temporal relations see[1].

Given these constraints, one may need to plan the activities for achieving maximum petroleum production from a reservoir, a state of MP, starting from a state of NS(no-srvey). For example, a trivial plan could be a serial ordering of the operations GS, SS, DA, PA and then ER.

Each of those operations or fluents is an interval in time. Some of the relations may not be consistent with respect to the others. A temporal reasoner's primary job is to find out whether any consistent scenario is feasible or not, by propagating above constraints all over the temporal data base. In the context of planning, this means whether there exists a plan or not. If there exists a temporally consistent scenario, or a *plan*, then finding one such instance would be the next task of a reasoner. Formalizing planning problem as an interval constraint propagation problem, is being addressed to in [2]. According to this formalism each interval is represented as node in a constraint graph, and disjunctive temporal relations as labels on the arcs between these nodes. The temporal constraint graph (TCN) is a complete graph because every temporal interval (we call it as *t-node*) is related to the other t-nodes, even if there is no specific information about how they are related. This complete lack of information is represented as disjunction of all 13 primitive relations (termed as *tautology*).

Any temporal constraint propagation algorithm systematically eliminates primitive relations from the labels on arcs, which are inconsistent. A global consistency algorithm[3] eliminates all such primitive relations which can not form a consistent scenario for the temporal assertions in the network. In such case, any primitive relation on any label can take part in at least one consistent scenario. Such a labelling is called *minimal labelling*. If during propagation any label gets all its primitive relations stripped off, having a *null* relation, then it implies that the two end nodes can not have any consistent labelling between them. This in turn implies the given constraints were inconsistent with respect to each other, and any consistent temporal scenario can not be formed.

[3] Global consistency algorithm checks for consistency of constraints all over the network, in contrast to a local consistent algorithm, which checks for consistency of every subnetworks of a fixed size.

Interval-based temporal reasoning increases expressiveness of planning. But the main problem with this scheme is that the problem of checking global consistency is NP-complete. There are approximate algorithms to address the problem and have inexact solutions[1, 10]. But generating a plan needs a temporally consistent model. Lack of this aspect was one of the weaknesses of the original work in this line by Allen et al[2]. In their scheme the arcs of the network will be left with *consistent* labelling, which is still a disjunctive set. Making a total order of the nodes is not feasible from such labelling, which is demanded by a planning problem. To manage the efficiency issue they have proposed a clustering approach. This would keep the network size under control so that run time of the algorithm is less affected by the increase in problem size. In this scheme all intervals should be clustered into a few groups based on some reference intervals, and propagation algorithm runs separately within each group. Although in some problem domains such hierarchy of reference intervals may be inherent, in many domains, like planning, this approximation may be impractical.

## III. Finding All Feasible Temporal Scenarios

A global consistency algorithm finds out a globally consistent models or all consistent models as a side effect while trying to determine global consistency of a network. Any trivial backtrack algorithm can do this work in exponential time. It is important to devise heuristics to make it efficient. First heuristic based global consistency algorithm was proposed by Valdes-Perez[12]. There, the algorithm was not written well enough to be efficiently implemented. Recently Ladkin et al[6] and we [8] have come up with two different heuristic based algorithms for solving global consistency problem.

Ladkin et al's algorithm randomly picks up a singleton relation from an arc and runs an approximate algorithm to update relations on the other arcs while checking for consistency of the picked up primitive relation. If picked up relation is found to be inconsistent the algorithm backtracks over the set of disjunctive relations on the current and previously picked up arcs, otherwise it goes ahead with the next arc. The algorithm terminates if a singleton labelling is found for all arcs, generating a consistent model. It may also terminate by backtracking up to the arc, which was picked up first, implying no consistent scenario is available. Experimental results, presented by them, provides us with a confidence that the global consistency is a not a very difficult problem in average case, although its worst case behavior is NP-complete[4].

**Table 1: Forward Pruning Algorithm**

```
For node number $i$=1 to $N$ do
   pickup an old singleton model of size $(i-1)$;
   for node number $j$=1 to $i-1$ do
      pickup singleton relation from label on
      arc $< i,j >$ and update labels on all arcs
      from $< i,j+1 >$ to $< i,i-1 >$ with respect to
      this singleton and arcs in old network;
      if updation fails on some arc
         if there is any more singleton left on this arc
            pickup next singleton and proceed
         otherwise backtrack to previous arc;
         if backtrack exhausts up to the first arc
            if a model was found
               save it in data base of partial models
               (of size $i$);
            otherwise return failure;
      if arc $< i,i-1 >$ is reached force backtrack;
```

The algorithm proposed by us[8] is based on a pruning heuristic where inconsistent relations are eliminated systematically. We call it *forward pruning* algorithm, because it prunes labels on all *forward* arcs in a preassigned order. A lose version of the algorithm is given in Table 1. There is reason to believe that such systematic working should improve the efficiency. Another feature in this algorithm is that the nodes are also systematically picked up one by one. At each stage of such addition of a node in the graph, a set of feasible models are generated. Then the next node is attempted for addition to each of these models. This model based approach makes it possible to generate all feasible consistent scenarios at a much lesser cost. In the worst case, the number of models itself may be exponentially growing with the number of nodes. Some preliminary experimental results with this algorithm have shown that in reality this growth rate is not very high, and manageable.

Efficiency of the forward pruning algorithm will also be aided with a fast preprocessing algorithm, which we have devised recently for incremental addition of nodes to temporal constraint graph[9]. The incremental and model-based approach will also make a *plan data base* available, in which new *operations* can be added one by one. Then new plans can be generated with respect to these newly added operations, or any newly added information. Significance of having all feasible plans will be discussed in the next section.

## IV. Significance of Having All Feasible Plans

Availability of all feasible plans allows one to have a choice of picking up the best plan according to some criteria. For example, in the previously mentioned case of petroleum exploration, one can find out a plan whose execution will take lesser time than most other plans. This can be done by choosing a plan where maximum number of 'overlap' relations occur (for parallely executing those operations)[5]. This procedure can also be automated by assigning some order of priority on the primitive relations on each of the labels on arcs. The resulting models, or plans, will then also be ordered by the system accordingly.

Normally a planning activity involves reaching a preassigned goal state. Most of the current planning systems are based on this objective. A temporal reasoning-based planning sys-

[4]A very good example of such cases where a NP-complete problem is addressed comfortably in most instances, is the case of linear programming using simplex algorithm.

[5]Thus, a *good* plan there, is SS after GS, SS overlapping DA, DA overlapping PA and PA overlapping ER.

tem, as described in this article, need not be goal driven only. Using a global consistency algorithm one can generate all possible scenarios, which allows one to do temporal projection. Then one can choose appropriate goal state and a corresponding plan, from those future scenarios. Thus, planning becomes a subproblem of temporal projection problem[7]. As far as we know the implication of this is yet to be studied.

In a dynamic situation where an agent is executing a plan unknown by another agent, if all feasible plans for the first agent are available, then the second agent can recognize the first one's currently executing plan by observing its operations executed so far. The plan recognition here becomes a problem of matching actions executed in past, or a partial plan, with the plans in the plan data base, and finding out the possible candidate plan(s), which the first agent might be executing currently.

There is a related problem of answering queries about plans. In our example, there may be a question whether *seismic survey* can be consistently done along with *enhanced recovery* under any plan. If the answer is 'yes', then the question would be what is the complete plan (or plans) under which that can be done. In the formalism described here, such questions can be answered using same pattern matching technique described in last paragraph, and it can also be done in real time, because of the availability of all plans. This may be an important advantage for an autonomous agent in any realistic environment. Isolating all possible scenarios also make it easy for updating plan data base with new information, which is a matter of adding or dropping consistent models.

In a time limited application, a time limit can be imposed when plans are generated, to create as many consistent plans as possible, rather than generating all plans. This will, of course, reduce some of the advantages discussed above. However, under some real life circumstances, that may be a better solution than to fail to generate any plan because sufficient time was not allocated.

## V. Conclusion

In this article we have discussed the feasibility of planning using interval-based temporal reasoning scheme. An advantage of using temporal representation is that of incorporating uncertainty of information about the temporal relation between different temporal entities in the domain. There can be many plans feasible under this circumstances. Additional advantage of interval based representation is of higher expressiveness, which will enable new types of planning not attempted before. The problem of interval based reasoning is that of its hardness. We have devised an efficient algorithm which can find out all feasible scenarios under incomplete information, in most of the circumstances. The implication of having all feasible plans available is multifold. They are also discussed here. So far nobody has taken any look into these aspects, although practicable algorithms for finding out consistent scenario is recently coming into fore.

## References

[1] J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):510–521, 1983.

[2] J. F. Allen and J. A. Koomen. Planning using a temporal world model. In *Proceedings of IJCAI-8*, 1983.

[3] James F. Allen, Henry A. Kautz, Richard N. Pellavin, and Josh D. Tenenberg. *Reasoning about Plans.* Morgan Kaufmann Publishers, Inc. (San Mateo, California), 1991.

[4] Mark Boddy. Classical nonlinear planning in complex domains. In *AAAI Symposium Note, the Spring Symposium on Foundations of Classical Planning*, pages 1–4, 1993.

[5] Alexander Horz. On the relation of classical and temporal planning. In *AAAI Symposium Note, the Spring Symposium on*

*Foundations of Classical Planning*, pages 48–52, 1993.

[6] Peter B. Ladkin and Alexander Reinefield. Effective solution of qualitative interval constraint problems. *Artficial Intelligence*, 57:105–124, 1992.

[7] Drew McDermott and Eugene Charniak. *Introduction to Artificial Intelligence.* Addison-Wesley Publishing Company, Menlo Park, California, 1985.

[8] Debasis Mitra and Rasiah Loganantharaj. Efficent exact algorithm for finding all consistent singleton labelled models. In *Proceedings of IEEE Robotics and Automation conference, Nice, France*, 1991.

[9] Debasis Mitra and Rasiah Loganantharaj. An efficient and approximate algorithm for temporal reasoning. Technical Report TR93-2-2, Center for Advanced Computer Studies, University of SW Louisiana, 1993.

[10] Debasis Mitra and Rasiah Loganantharaj. Incremental consistency algorithms for qualitative temporal reasoning. Technical Report TR93-2-5, Center for Advanced Computer Studies, University of SW Louisiana, 1993.

[11] J. Scott Penberthy and Daniel S. Weld. Temporal planning with constraints. In *AAAI Symposium Note, the Spring Symposium on Foundations of Classical Planning*, pages 112–116, 1993.

[12] Raul E. Valdes-Perez. The satisfiability of temporal constraint networks. In *Proceedings of AAAI*, 1987.