Comments on the Paper "Mission Planning System for Earth Observation Missions"

Pete Bonasso (r.p.bonasso@jsc.nasa.gov)

Texas Robotics and Automation Center Laboratories (TRACLABs)
1012 Hercules
Houston, TX 77059

Introduction

The paper in question essentially discusses two mission planning systems, one which uses at its core UC-POP, a result of several decades of AI research in automated plan generation, and the other which resorts to encoding human plan knowledge in the form of rules to be executed by an rule-base system (RBS). The orientation of the former system, the ATOS-4 planner, is as an experimental prototype, whereas the latter, the Envisat mission planning system, is in the context of a near fielded application.

The concern that leapt to my mind after reading this paper was simply how do the authors expect to carry out a real planning application without a planner? A rule-base system (RBS) can only guarantee that it will fire all rules that apply when they apply, but it can guarantee nothing about finding a plan in the space of action sequences. In my on line interchange with the first author, I came to realize that several issues bear on this question:

- existing software can limit what any planner can do
- the need to extend the planning approach
- the reality of human computer interfaces (HCI)
- the trade-off between a theoretical loss of power and an approach that will just get the job done for most intents and purposes

All four of these issues point to hard questions that need to be answered in the future, both for the AI community and for those who would apply AI results.

Existing Software

I learned in my interchange with the first author that apart from the basic logical model, the Envisat system was fully developed from scratch, yet some modules for the generation of primary activities for specific instruments had been developed by the space engineers as separate libraries. These procedures are not easily reduced to sequences of actions with start and end times, but can include arbitrary types of code. This makes it more difficult to model even simple properties that are needed by the planner, for instance minimum and maximum duration as a function of the arguments of the procedure.

In many applications, particularly control applications, one is usually constrained by the software that has been hand-coded for particular instruments. In a very large project we under took to control life support systems [Schreckenghost et al 98], we were constrained to build a hybrid control system "on top of" a suite of legacy control software. As a result we were not able to realize the theoretical efficiencies of the hybrid system.

Extending the Planner

The authors learned through their endeavors that the UC-POP system they might use from ATOS-4 for Envisat was limited in terms of the temporal constraints that exist between the depending activities in both environments. In UC-POP, secondary activities can only be created before or during the main activity. An attempt was made to analyze the dependencies existing between the Envisat mission activities; the temporal dependencies that can exist between them were far wider in scope. Basically a precondition for enabling an activity could be created by other activities anywhere in the plan, within constraints with respect to the initial activity. Many dependencies include fixed or variable delays (e.g., the secondary activity must have been executed at least one hour before the initial one). This capability would have to be added. The planning algorithm itself would have to be extended to handle threads to dependencies with future activities.

Here then we see a real limitation of a planning algorithm for a particular application. If the team were supported by a nearby research group, as is the case in several CMU projects where AI is applied in the nearby community, the Envisat team might have considered effecting the needed changes. But such was not the case, and the team had little experience in the use and modification of planners, and certainly not with the specific UC-POP code. As the first author put it in one exchange with me

> The technical constraints on the design in general were such that it was not clear that we could use the ATOS

> approach and still take advantage of it. The rule-based approach was considered safer.

The Reality of HCI

The Envisat team was more familiar with RBS techniques, than with planning software. This was especially true with regard to developing interactive human computer interfaces. It was not clear to them that they could extend the AI planning approach such that it would have accommodated the user interactive requirements and have it working on budget and schedule.

The HCI requirements in any application are as critical as any theoretical aspects associated with the type of problem. But here I suspect there might have been ways to build an HCI to UC-POP, and that the real issue was the limited budget. For instance, we have used an HTN planner at NASA-JSC since 1994 [Bonasso et al 97]. It came with an CLIM interface which was limited in what it could provide the user. Our solution was to make the HCI a separate application executable on a wider variety of workstations, and network it to the planner using a client-server message passing system.

Theory versus Practicality

There's no denying that UC-POP is sound and complete. It's not clear what one can say about a planner built in an RBS. My claim is that the UC-POP system will find a plan when the RBS planner may not. So it stands to reason that one should make an effort to "extend" the planning system rather than "settle" for the less complex, but fault prone RBS.

But it's not that simple. Although I have no personal experience with UC-POP, I understand that it is usually difficult to modify systems which emerge from an academic environment. As well, the ATOS team was not allowed to use the Lisp version, and had to recast the algorithm in C++. They did some of the extensions mentioned above for ATOS, but there was little time to effect all the needed changes for Envisat.

Further, there may be something to be said for what an experienced RBS team can do to provide an intelligent system on time and within budget, and still come up with a better than average reasoning system. The Envisat system basically schedules activities for satellites. The system will always produce a solution. It will simply de-scope requests and move activities around until it finds one. In the worst case this will be the degenerate solution: switching all the instruments to their safe mode. The team is wary of some of the pitfalls:

> We try to control things in the rule-based approach by splitting the rule set in modules that are activated separately. In practice, we only have to hope that our testing of these modules will ensure a correct functioning of the system, as per any other application. The modules are combined in finite sequences, so the risk is limited to individual modules. The risk actually exists for some modules only.

Conclusion

Nevertheless, my gut feel is that as earth observing satellites become more advanced, and as we begin to need to control more than one of them at a time, the task will quickly overcome an RBS approach. So what is to be done? I believe some answers lie within the AI research community but most must come from those who seek to apply AI technology.

Making major AI systems accessible via a remote HCI must have more of a priority in AI research. If we wish to see our solutions used in real world applications, we must make the development of APIs to get at function, value and representation an equal partner with efficiency and theoretical soundness. Of course that requires us to convince our funding groups of that need. Pointing out applications like Envisat can be a first step in that argument.

The rest of the issues seem to me to be resolvable if the engineering team makes the case for a budget which looks to the future. For instance, getting help in the form of consultation by universities might uncover solutions to extending a given planning algorithm. Or perhaps UC-POP isn't the planner for this problem. In my opinion, with the future in mind, it behooves the Envisat team to investigate other planning approaches. For example, the Envisat temporal constraints might be more readily handled with an interval planner such as the one used in Remote Agent [Pell et al 97]. Also, some planners exist that can accommodate user functions and legacy software. Two of these are SIPE as used in Cypress [Wilkins et al 95], and AP as used in 3T [Elsaesser and Slack 94]. AP in particular allows the integration of user functions in its operator preconditions, effects and task nets.

And finally, an effort ought to be made to convince management to include a person skilled in AI planning on the applications team. So many large applications require planning techniques that it should be worth the effort in the long run.

All of these suggestions require more time and resources. But I believe the answer is to convince the user of the importance of employing the very best technology has to offer rather than settling for good enough.

References

[Bonasso et al 97] R. Peter Bonasso, R. J. Firby, E. Gat, David Kortenkamp, D. Miller and M. Slack, "Experiences with an Architecture for Intelligent, Reactive Agents", Journal of Experimental and Theoretical Artificial Intelligence, 9(2), 1997, pp. 237-256.

[Elsaesser & Slack 94] Chris Elsaesser and Marc G. Slack. 1994, Integrating Deliberative Planning in a Robot Architecture, in Proceedings of the AIAA/NASA Conference on Intelligent Robots in Field, Factory, Service, and Space (CIRFFSS '94).

[Pell et al 97] Barney Pell, Erann Gat, Ron Keesing, Nicola Muscettola and Ben Smith. 1997. Robust Periodic Planning and Execution for Autonomous Spacecraft, in Proceedings of the 1997 International Joint Conference on Artificial Intelligence.

[Schreckenghost et al 1998] Schreckenghost, D; Bonasso, P.; Kortenkamp, D.; and Ryan, D. Three Tier Architecture for Controlling Space Life Support Systems . IEEE Symposium on Intelligence in Automation and Robotics. May,1998.

[Wilkins etal 95] David E. Wilkins and Karen L. Myers and John D. Lowrance and Leonard P. Wesley. 1994, "Planning and Reacting in Uncertain Dynamic Environments", Journal of Experimental an Theoretical AI (7).