Are We Comparing Dana and Fahiem or SHOP and TLPlan?
A Critique of the Knowledge-based Planning Track at ICP

Subbarao Kambhampati
Dept. of Computer Science and Engineering
Arizona State University
Tempe, AZ 85287-5406
http://rakaposhi.eas.asu.edu/rao.html rao@asu.edu

I'd rather learn from one bird how to sing
than teach ten thousand stars how not to dance
e.e. Cummings

Since AIPS 2000, the planning competitions started having a special track for "knowledge-based planners" (I shall henceforth refer to this track as the KB track). Despite the fact that fewer planners[1] competed in this track, the results of these tracks have attracted considerable attention. It could be said that the KB track brought more interest to knowledge-based planning than the many periodic polemics and laments about the lack of attention to domain knowledge in automated planning.

One of the important contributions of the planning competition has been in terms of the larger lessons the community gleaned from the results of those competitions. When IPP did better in the planning competition in 1998 instead of Blackbox, we learned that the exponential size of the compiled encodings can be a problem compared to the compact planning-graph based search. When heuristic search planners like FF, HSP and AltAlt did better in the competition in 2000, we learned that the reachability analysis that is the backbone of Graphplan can actually be better used as a basis for deriving heuristics. When LPG showed superlative performance in the 2002 competition, we learned that "reduced repair space local search" with sophisticated (planning-graph based) estimates of repair costs is a great idea.[2]

In contrast, one is hard-pressed to summarize the lessons of the KB track. When TLPlan (or TALPlan) "outperforms" SHOP in the competition (in terms of cpu time taken for solving a set of problems—arguably the least informative comparison measure for KB planners; see below) there are at least three ways of interpreting this:

- That TLPlan is a superior planning technology over SHOP.
- That the naturally available domain knowledge in the competition domains is easier to encode as linear temporal logic statements on state sequences than as procedures in the SHOP language.
- That Fahiem Bacchus and Jonas Kvarnstrom are way better at coming up with domain knowledge for blocks world (and other competition domains) than Dana Nau.

I would wager that the authors of the specific planners would prefer that the community take one of the first two lessons; but given the way the competition has been conducted, it seems to me that the third lesson is at least as equally valid. This, of course, is depressing for the authors[3] as well as the community.

I would like to take the position that the blame for this confusion lies squarely on the methodology that is currently being used to evaluate the planners in the KB track. I contend that the competition does not evaluate or address any of the questions that are *actually relevant* for charting the progress in knowledge based planning.

Questions worth asking: Unlike domain-independent planning, where there are essentially two roles–the user and the planner, in the KB planning there are in fact three roles. The user, the planner *and* the "Knowledge Engineer" who makes up, validates and writes the domain knowledge for the domain. Indeed, the ease of knowledge-engineering should be a primary criterion by which KB planners are judged. The aspects of a KB planner that affect the ease of knowledge engineering include such factors as:

- How easy/natural (for humans) is the language in which the planner accepts control knowledge?
- How easy is it to "validate" the control knowledge being input to the planner?
- Is the naturally available knowledge about a specific domain easily encoded in the language accepted by the planner?
- Does the planner allow "any expertise" behavior–solving the problems even without any control knowledge, but improving performance with added control knowledge? (or

[1] and more or less the same set of planners

[2] It is interesting to compare the evolution of LPG planner with the evolution of CISC vs. RISC computer architectures...

[3] Unless, of course, those authors had harbored some unrequited child-hood dream about conquering the blocks world.

is the control knowledge tightly intertwined with the domain physics?).

..and how they are not being asked: Given the way the competitions have been structured until now, I contend that *none* of these questions are being addressed even tangentially. Here are some of the reasons why:

- The role of the knowledge-engineer is played by the same person(s) who wrote the planner. So, the question of how natural the specific language is for third-party knowledge engineers is largely unaddressed.

  The question of which particular framework provides a better substrate for capturing domain knowledge is not unlike the question of deciding what particular specialized programming language provides the best support for writing a class of application programs. Imagine you were trying to decide whether LISP or C is the best language for writing AI programs. I would contend that you won't learn much by comparing the performance of Lisp programs written by Guy Steele to the C programs written by Kernighan. And yet, this is exactly what the competition does by comparing TLPlan control knowledge written by Bacchus to SHOP schemas written by Dana.[4]

  Another problem with control knowledge written by the authors of the planners is that it can even blur the tradeoffs between planner-specific search control knowledge[5] (of the form used by HSTS and RAX planners) vs. planner-independent search control knowledge.[6]

- No reasonable time limits are placed on coming up with the control knowledge. So, we don't learn much (or anything) about whether or not naturally available knowledge about a domain is easily representable in the language accepted by the planner.

  If it was possible to write and validate the requisite domain knowledge for a specific domain in half-a-day for the case of TLplan, and the same task were to require one day in the case of SHOP (or *vice versa*), we in the community would clearly want to know that. But, by giving the domains away significantly ahead of time, we are not able to get this information.[7][8]

Some directions for change: Based on the foregoing, I argue that a significant overhaul is needed if the KB-track of the competition were to provide any lessons for the community. Some ideas include:

- Recruit third-party volunteers who will play the role of knowledge engineers for the KB planners. Ideally, we would like to have the same people writing the control knowledge for a given domain for all the competing approaches (so one knowledge engineer per domain rather than one knowledge engineer per planner).
- (Alternative to above) Specify the control knowledge that is available, so all planners encode the same general knowledge. One idea might be to ask the designers of the domains (e.g. David Smith and his cohorts for the Satellite and Rovers domain) to provide, in english, what sort of control information they would like the planner to use.
- Measure the time taken to write and validate the control knowledge.
- Analyze the knowledge encoded by the different KB planners for the same domain, and characterize it in terms of (a) whether the knowledge is procedural or declarative and (b) how hard would it be to "learn" the same knowledge.

Of these, I think the first is the most important. Years ago, I had once asked students in the ASU planning seminar to write control knowledge for two different KB planners and was surprised at some non-trivial patterns that emerged. Done on a larger and more controlled scale, such studies will be quite helpful. I understand that organizing the competition this way will be harder (for one thing, the organizers would have to recruit third-party volunteers), but the lessons we could learn would be much more illuminating.

I realize that to a large extent, the methodological problems are not limited to the competition, but also pervade the evaluations of the KB planners presented in technical papers. I am often mystified by the tables reporting head-to-head comparisons between two KB planners solely in terms of the final cpu times, completely disregarding the knowledge acquisition effort. This is however all the more reason to overhaul the way the KB track is conducted in the competition, as it is more feasible to do a third-party evaluation in the competition setting than in individual conference papers.

In closing, lest I be misunderstood, let me say that I think KB planning is an important technology.[9] It is obviously silly to insist on rediscovering everything other than the domain physics in your own planner, if such knowledge is easily available and crying out to be used.[10] My reservations are not against KB planning, but the specific way KB-planning track is organized at the planning competitions.

[4]This of course would not be a problem if we find a way of downloading instances of Fahiem and Dana along with their planners. :-)

[5]The distinction I am trying to make here is whether the knowledge engineers is required to know only about the domain, or both about the domain and the internals of the planner.

[6]After all, Nicola and Kanna, who probably think like HSTS, might find it easier to write planner-specific search control knowledge for their planner than make up linear temporal logic assertions.

[7]It should be mentioned that the 1998 competition did have an "onsite" component (e.g. the "mystery domain"), but this component has reduced significantly by 2000, and seems to have disappeared all together by 2002.

[8]It should be mentioned that giving all the domains and problems away ahead of time also probably affects the domain-independent track to some extent, since there are several planners with many tunable parameters, and the authors might be tempted to manually tune the parameters on a domain-by-domain basis.

[9]And I have written my share of KB-planning papers!

[10]Indeed, researchers from OR and constraint programming communities often find the strong differentiation we make between "physics" and "control" knowledge as somewhat of an inexplicable idiosyncrasy!