Handbook of Evaluation for the ARPA/Rome Lab Planning Initiative

With Contributions From

**Paul Cohen, Tom Dean, Yolanda Gil,
Matt Ginsberg and Lou Hoebel**

Draft of February 16, 1994

1 Introduction

This document describes methods for evaluating research and development progress in automated planning. It is meant as a resource for the members of the research community participating in the planning initiative. The document attempts to explain the goals of evaluation and provide concrete examples of evaluation methods that are currently being used. It also serves as a source of ideas for designing new methods for evaluation and improving old ones.

The planning initiative is driven by a complex set of goals involving government officials, members of the scientific community, and users in the military and industry. Evaluation methods allow those involved in the planning initiative to measure progress toward these goals. This document is not meant to define once and for all what these goals should be, but, rather, to provide a basis for ongoing discussion to shape those goals and adapt them to circumstances.

In general, the goals of the initiative can be divided into two classes, programmatic and scientific. As a government program, the initiative is designed to serve the technology needs of a particular constituency, including but not limited to the transportation command. The initiative is a programmatic success insofar as the needs, both short and long term, of this constituency are well served. Some measure of programmatic success is achieved when users are able to perform existing work more efficiently or when an agency is able to provide services that it could not have beforehand. The initiative is a scientific success insofar as it formulates and answers questions of interest to the scientific community. Programmatic and scientific progress are combined when the answers to scientific questions give rise to technology that serves to forward programmatic goals.

Much of the technology developed in the initiative is evolutionary in the sense that it serves to improve current methods of solving problems. However, answers to scientific questions can point the way to revolutionary changes in the way that people approach problems. Revolutionary changes have to be carefully introduced and evaluated to convince users that the costs of adopting a new technology is worth the long term gains.

The Planning Initiative has developed an environment in which scientific and programmatic goals can be achieved. For example, domain packages have been constructed, and a common protoyping environment is available. What seems to be needed, now, is for more participants in the planning initiative to put this infrastructure to use.

Because we are all constrained to a greater or lesser extent by two sets of evaluation criteria—programmatic and scientific—this document discusses them in turn, as exclusive activities. That is, we ask, "What makes for a good programmatic evaluation," and then we ask, "How do you convince the scientific community that something good has been accomplished." Of course, we want good science with good programmatic impact, both; so the final section of this report will focus on how to achieve them, simultaneously.

2 Programmatic Evaluation

The planning initiative is designed to serve U.S. technology needs. Program managers seek out research and development work that they believe is likely to provide useful technology to their sponsors in the military and government. They have to convince these sponsors that their money is well spent. The evidence used by program managers runs the gamut from identifying satisfied customers and bottom line savings to claims that new technologies make us more competitive in strategic markets.

The future of the initiative depends on making the sponsors happy, understanding their goals and carrying out research and development that meets those goals. In some cases, the program managers can articulate the goals of the sponsors to the research community, but more often the research community has to work with the user community that the sponsor represents to elicit goals and define problems. If the end users are happy, then the sponsors are happy as well. Programmatic evaluations are ultimately based on how research and development influences the sponsors.

If you design a transportation scheduling algorithm that computes better schedules faster *and* you convince an end user to adopt your algorithm *and* that end user is pleased with the final result *and* conveys his pleasure to an appropriate sponsor, then you have contributed directly to the success of the program. However, it is unlikely that everyone working in the initiative will be able to directly influence sponsors.

A more likely scenario is that an individual will work to develop some piece of technology what will ultimately be used by some end user. Does this mean that individuals need not concern themselves with end users and sponsors? No! Quite the contrary, it is the responsibility of everyone in the initiative understand and educate both end users and sponsors.

If the value of your research is immediately recognized and integrated into products, then you are most fortunate. In most cases, you will have to convince others of the value of your research by applying your results to solve their problems, developing a prototype system, or whatever else is required for them to adopt your technology. Programmatic evaluation criteria are often open-ended and you are encouraged to be creative in pursuing programmatic goals.

The Planning Initiative is organized in three tiers corresponding to basic research, prototype applications, and technology demonstrations, respectively. Researchers can participate in all three tiers, as the following example from Brown University shows:

2.1 An Example: Time-Critical Planning at Brown University

Tier 1 to 2: Helping to transition previous research from Tier 1 to Tier 2. In particular, working with Honeywell SRC to develop and refine the temporal database technology developed at Brown. Providing specific algorithmic expertise and general understanding of planning initiative needs to direct the development of a production temporal reasoning tool.

Tier 1 to 3: Working with domain experts at Scott Air Force Base to translate real transportation scheduling problems into precise specifications amenable to algorithmic analyses. This also required distinguishing the problems that they currently solve because they are stuck with particular methods from the problems that they would like to solve if they had better computational methods and easier access to data.

Tier 1 to 3: Working with domain experts to convert available data into stochastic models that could be used by a variety of decision theoretic planning and scheduling systems. Prelude to this effort is developing a representational framework that can compactly encode dynamical systems involving high-dimensional state spaces.

Spanning all three tiers: Developing a formal description of transportation scheduling problems from static to dynamic, centralized to distributed. Providing a common basis for discussing problems, suggesting experiments and research directions, and considering both evolutionary and revolutionary approaches to changing the way that the military currently does business.

3 Scientific Evaluation

A catalog of empirical studies for the Planning Initiative can be organized by many dimensions. The following ones were selected because they bring into focus the goals and needs of the participants in the Initiative.

3.1 Technology

The visionary demonstrations for the Planning Initiative include many kinds of technology: several kinds of planners, schedulers, temporal reasoners, good interfaces, plan steering, integration tools, and instrumentation to measure how it all works. Evaluation metrics and experiment designs for one technology don't always make sense for another.

3.2 How applied is the technology?

Some work in the planning initiative is basic research and is unlikely to be implemented in fielded systems in the near term. Some work is ready for Technology Integration Experiments (TIEs), and some is the basis for Integrated Feasibility Demonstrations (IFDs). Different kinds of evaluation are appropriate at these levels.

3.3 The goals of the study

Some studies are exploratory, others test specific hypotheses. Some are designed to provide calibration data for other experiments, and others fit parameters of models. Some test whether a fielded system is acceptable to its end users, others test the sensitivity of the

system to small adjustments in parameters or aspects of scenarios. Some assess the performance of an individual system or technology, others compare systems or technologies. One particular kind of study merits special mention: the goal of *demonstrations* is to show that something can be made to work. Demonstrations are not necessarily exploratory, nor do they test hypotheses (besides the hypothesis that something can be made to work) or estimate parameters, or achieve any of the other goals listed above.

Clearly, the previous two dimensions are not independent: the goals of experiments with applied technology tend to be very different from the goals of experiments in basic research.

3.4 What is the task in the study?

Broadly, tasks are things like planning and temporal reasoning. To be more specific we need to consider the goals of the study. For example, if the goal is exploratory—finding out the conditions under which something works well and poorly—then the task specification includes: the aspects of the system that are being tested, the source and characteristics of test problems, and the experiment procedure. For example: to test a scheduling algorithm, identify several features of scheduling problems that probably affect the algorithm's performance, and generate scheduling problems that differ on these features. The procedure is to present the algorithm with a problem, have it generate a schedule, and then run a simulator to see how well the schedule works.

3.5 Where do we get test problems?

The Planning Initiative has developed a small number of very large, complex, realistic scenarios that are suitable for demonstrations. They are somewhat less useful to those engaged in basic research, for reasons we'll discuss in section 5 Many members of the Planning Initiative have faced independently the question, where do we get test problems? Four general answers are:

Real applications. Some researchers have developed planning technology for applications that share important features with transportation planning or crisis-action planning. The advantage of letting an application present problems is that the problems are representative of the application. This is also a disadvantage if we want to claim that our system applies more generally.

Realistic simulators. Along the same lines, some researchers develop technology to solve planning problems in simulations. For example, the Phoenix simulator generates problems that have many characteristics we're interested in: real time, limited information, sensor error, limited resources, multiple simultaneous goals, a map with realistic terrain features, weather, etc. The advantage of having a simulator is control: we can set up problems with known characteristics. The disadvantage is that we never know whether a planner that works in a simulated environment will work

in a real one; it's not clear whether the characteristics of simulated forest fires (and bulldozers, fuel carriers, weather, etc.) are similar enough to the characteristics of real wildland fires for us to claim that the Phoenix planner could handle wildland fires.

Artificial problem generators. This is the approach taken by several researcher projects, described later. The advantage of problem generators is we get to control the kinds of problems that are generated, so we can tie our results to controlled problem characteristics. This is also a disadvantage if we want to claim that our system will work in a particular application but we don't know the characteristics of the application.

Corpora The community might gather a corpus of real problems from real applications. This is in many respects the best approach because it satisfies both our needs: lots of problems and real problems. The machine learning community has taken this approach, and so has the MUC effort. In both cases, people worry about whether the corpus of problems is representative of problems in the real world. In machine learning, at least, there are active efforts to characterize the problems in the Irvine repository to understand why some are easier to solve than others.

The Planning Initiative could certainly use a large repository of real problems. But if they are all as complex as, say, the Tunisia Scenario, then the Initiative won't be able to afford to develop them, and researchers probably won't use them.

3.6 What is measured?

Objective "classical" measures include run time and space requirements. Beyond these, what's measured depends on the previous distinctions: what is the task, what is the technology, how applied is it, and what are the goals of the study? If the technology is, say, scheduling, we will measure throughput and tardiness, resource utilization and bottlenecks, and so on. If the technology is fairly applied (and especially if it is integrated with other technologies) we might assess "black box" measures, only: did the system come up with a good schedule in an acceptable time? But if the level of research is more basic, we will also look inside the system ("glass box" measures) to capture data that explains the performance figures. We will say more about measures, below. For now, note some distinctions:

Objective and expert measures We can assess which of two planners ran more quickly, but it's more difficult to assess which produced the better plan. Expert human planners can explain some of their criteria, easily, but others are more difficult to cast in objective, measurable terms. For example, it's difficult to assess whether a plan has a "surprise" component.

Black-box and Glass-box Measures Glass box measures tell us what is happening inside a system, black box measures look only at outputs. For example, the number of goals achieved by a plan is a black-box measure, and the number of ordering

violations repaired by critics is a glass-box measure. In general, it's difficult to explain performance without glass-box measures. The Planning Initiative developed an instrumentation package, CLIPS, to make it easy to acquire glass-box measures. Another facility, METERS, provides similar functionality but across distributed system components.

Indicators and Behaviors A *behavior trace* is a vector of measures tracked over time. For instance, a vector might be the current goal, heading and cargo of a Tileworld agent, and a behavior trace is this vector over time. A *behavior* is a subsequence of a behavior trace that is interesting for some reason, perhaps mere frequency; for example, Adele Howe discovered particular subsequences of failures were very common in traces of the Phoenix planner. *Indicators*, on the other hand, collapse over the time dimension; for example, the frequency distribution of types of failures is an indicator, and so is the mean latency to repair a failure; whereas the sequence of failures, or the sequence of latencies required to repair each, are behavior traces.

Samples and Measures It is a source of concern that some measures are assessed on very small samples. For instance, there is just one Tunisia Scenario. In general, if our goal is to obtain estimates of the value of a measure on cases we haven't seen, then larger samples provide more precise estimates. In statistical terms, if our goal is parameter estimation, then bigger samples are better. (Note, however, that when our goal is hypothesis testing (e.g., does scheduler A produce schedules with *significantly* less idle time than scheduler B?) we must be wary of tiny differences boosted to significance by huge samples.)

3.7 What is the Standard of Comparison?

Many experiments involve comparing performance to a standard. Where can we find the standards? Six answers are:

- "default rules" such as random performance,
- theories of optimal performance,
- the performance of other programs,
- objective standards set by humans,
- human performance on identical or comparable problems, and
- post hoc judgments by humans.

A program can meet one of these standards but not others; for example, a program might perform optimally but be judged poor by a human expert, and vice versa. More importantly for the Planning Initiative, it is difficult to get access to some of these standards, particularly human experts. This issue is discussed more fully in Section 5.

4 Some Experiments

The following catalog is organized by experiment goals, as described earlier. The other dimensions of the space of empirical studies are treated in the context of the catalog entries.

4.1 Exploratory Studies

With no particular hypothesis in mind, exploratory studies are designed to get a "feel" for how a system works in different conditions. Generally, we have some idea of which factors are important and which we can safely count as "noise" or variance, but exploratory studies often change these preconceptions. When we look at data from exploratory studies, our goal is to notice patterns or structures, regularities that suggest hypotheses. Here are some examples; see [Cohen, Ch. 2] for others:

Example 1. Steve Smith described how he assessed the performance of a scheduler in an exploratory mode: As intimated earlier, getting test cases was particularly difficult. Smith's solution was to build a problem generator, ensuring an inexhaustible supply of test cases. Then he explored the performance of his scheduler in a variety of conditions by manipulating factors such as load, slack time, loose and tight temporal coupling, and so on; and collecting several performance measures. In this way it's possible to show that the scheduler is "general" with respect to the space of problems that can be generated by the problem generator. It's not clear whether this is "general enough," nor, for that matter, a demonstration that the scheduler is good for a specific class of problems. On the other hand, as an exploratory technique, it's informative to run a system on a lot of problems that have different characteristics. The advantage of problem generators is they provide lots of probalems and control over their characteristics.

Example 2. Tim Oates recently developed a *plan steering* assistant for transportation networks. As ships move around a simulated shipping network, *demons* predict bottlenecks at ports. When a bottleneck is predicted, a plan steering agent advises that one or more ships should be diverted from the port that is predicted to be overfull, thus alleviating pressure on the port and, hopefully, avoiding (or "steering around") the bottleneck. Oates ran two sets of experiments: In the first, he determined that several factors affect the accuracy of demon predictions; for example, predictions are less accurate when ship travel time is very variable. In the second, Oates determined that cargo throughput would be enhanced by following the advice of the plan steering agent. That is, by diverting ships before bottlenecks occur, more ships and cargo could get through the network. Because this was an exploratory study, Oates measured throughput (and a lot of other things) in many different conditions, and he found something quite surprising: Even in conditions where the demons made bad predictions, throughput was enhanced by following the advice of the plan steering agent! This led to the following hypothesis: Random diversions of ships from their intended destinations to other destinations will increase throughput. In a confirmatory experiment, Oates determined that this was indeed the case, but random diversions were not as good as those suggested by a plan steering agent based on accurate demon predictions.

Example 3. When the subject of a study is a technology that has been around for a while—especially a technology for doing things that humans currently do— the what-to-vary and what-to-measure questions can be answered by consulting precedent. A more difficult problem is described by Tom Dean, who studies a class of problems that's "unrecognized" in the sense that they formalize aspects of the planning task in terms that human expert planners don't necessarily recognize. These problems involve the relationship between planning and control in conditions of stochastic uncertainty. Dean thinks this class of problems is important to the Planning Initiative, that is, learning to solve them will have technological benefits. But, as yet, the community of human expert planners does not describe its task in Dean's terms, so they cannot tell Dean what to vary and what to measure in his experiments. Dean's solution is to vary and measure *formal* parameters of his theory of planning and control. He uses a problem generator, just as Steve Smith did, and runs his studies in much the same way: Vary parameters and observe the effects on measures.

Needless to say, it isn't only in exploratory studies that formal models provide answers to the what-to-vary and what-to-measure questions. Formal models are rich sources of hypotheses, the subject of the next section.

Schematically, exploratory studies are simple:

Type of study: Exploratory

Goals: To get a sense of the factors that affect system performance. To notice patterns in data that hint at interactions between factors or sensitivity to particular levels of factors. To get baseline measures of performance for later comparisons.

Tasks: Exploratory studies look at the behavior of systems in a wide range of conditions—after all, one goal is to see how the system responds to different levels of factors. A good exploratory study will require a system to solve lots of problems. Where can we find lots of problems? This issue was discussed in Section 3.5. As far as we know, all exploratory studies in the Planning Initiative have been conducted with realistic simulators or artificial problem generators.

Measurements: Exploratory studies should collect many kinds of data because many interesting phenomena are discovered by cross-referencing results.For example, Tim Oates' hypothesis about random diversions of ships (above) was suggested only because he thought to measure the quality of demon predictions.

Procedure: Basically, run the system and collect data. If one is looking at a lot of factors that have many values, it's better to proceed incrementally, finding combinations of feature values that are interesting, rather than running an enormous factorial experiment.

4.2 Hypothesis-testing Studies

The purpose of hypothesis-testing studies is to answer questions, typically binary questions. Here are some examples:

Example 1. Consider an agent that executes plans that fail until it finds one that works; for instance, a robot delivering packages might try several routes, and find each blocked, until it finds a clear route. Simon and Kadane developed a theory of the order in which such an agent should try things. It related the order to the probability and cost of success. In 1990, Howe and Cohen tested this theory in the context of failure-recovery in the Phoenix planner. Briefly, plans fail in Phoenix and when they do, methods are applied to repair them, but the methods sometimes don't work. Some methods are likely to succeed but they are expensive, others are less likely to succeed, but cheap. Simon and Kadane's theory specified the order in which to try methods. Compared with control conditions in which other orders were tried, the orders that were predicted to be superior were, in fact, cheaper.

Example 2. Another study (ref. Oates and Cohen, 1994) tests the hypothesis that a *plan steering* agent helps humans keep traffic flowing smoothly through a transportation network. Experiments like this can be tricky because it's so easy to make humans perform badly in the control condition (where they get no assistance from the agent), and, thus, they appear to be helped by the agent. The study has three conditions:

1. humans get no advice from the agent and must take plan steering actions (e.g., rerouting ships) unassisted;
2. the agent follows its own advice and steers plans by itself, so the human is "out of the loop," altogether;
3. the human is offered advice by the agent and decides whether to follow the advice or do something else, or do nothing.

If performance is best in condition 3, then some sort of symbiotic relationship holds between the agent and the human subjects. If performance in condition 3 is not superior, we will look at all the decisions made by the human subjects and classify them into one of six categories: at each decision point in a trial the agent might have recommended an action or no action, and the human might have accepted the recommendation or taken some other action. We will score the state of the transportation network after each decision, and sort the scores into the six categories. Thus, we will be able to tell whether performance in condition 3 is poor because, for instance, the human didn't follow the agent's advice (or because the human did!).

The experiment procedure involves training humans to manage the flow of traffic in networks without assistance from an agent. This step ensures that the human performance is as good as it can be before we test the efficacy of the agent. Once trained, we will test the humans with and without assistance. To avoid further practice effects, we will

counterbalance the order of these conditions: half the humans will be tested first without assistance, half will be tested first with assistance.

Schematically, hypothesis-testing studies are simple—they are the classic "controlled experiments" we all learned in school.

Type of study: Hypothesis testing

Goals: To answer a specific question, often a binary one.

Tasks: An important aspect of hypothesis-testing studies is the design of control conditions. Therefore a task must be one that can be run in at least two conditions, and the structure of the task must not introduce any biases (see Cohen, Ch. 3). As an example of a bias, consider this: when running experiments with simulators, we typically terminate trials that run fora long time without solving a problem (e.g., if the Phoenix planner cannot contain a fire in 120 simulation-time hours, we quit trying). We must then decide whether or not to use the data from these trials. It's easy to show that either way, we can bias our results.

Measurements: In general, more is better. In particular, it's worthwhile to collect measures that not only demonstrate an effect but also help to explain the effect.

Procedure: Much has been written on procedures for hypothesis- testing studies [refs. Cook and Campbell, Leary, Cohen...]. Experiments with programs that do not learn have very simple procedures; experiments with people and programs that learn sometimes require training or counterbalancing, as in the Oates and Cohen example.

4.3 Sensitivity Studies

Sensitivity studies find out whether small changes in inputs or system parameters will have large effects on output. Here are some hypothetical examples (we don't know of any examples in the Planning Initiative as yet):

- A scheduler performs very well when tasks are "padded" with a small amount of slack time. What is the performance profile as the amount of slack time is reduced?
- A planner is able to modify a plan when environmental changes require it. As the environment changes more and more dramatically, do we reach a point at which it would be better to abandon a plan altogether, rather than trying to modify it?
- A projection algorithm uses beam search for efficiency. Does the accuracy of projections degrade gracefully as the beam width is varied, or are the projections pretty accurate until beam width falls below some level and then they go haywire?
- Will the time required to produce a plan increase gracefully as the number of goals increases?

The general schema for sensitivity studies is this:

Type of study: Sensitivity

Goals: To find the performance profile of a system as inputs or internal parameters are varied, and specifically to find whether the profile is "smooth" or whether it increases or decreases wildly after some level of input or internal parameters is exceeded.

Tasks: An interesting aspect of sensitivity studies is that they do not require a wide range of tasks, as exploratory studies and hypothesis testing studies (if one wants generality) do. In fact, one can look at sensitivity within a single task, provided one can manipulate internal parameters or environmental (not task) parameters. This makes sensitivity studies an ideal vehicle for experimental work in the Planning Initiative, particularly for experiments with large, complex scenarios such as the Tunisia Scenario.

Measurements: As with other kinds of studies, it is better to collect a lot of performance measures rather than just one. This way, one can see several performance profiles change simultaneously, and one can strive for a parameterization of a system that does "pretty well" on all the important ones.

Procedure: The procedure is pretty basic: step through levels of the factor that the system is supposed to be sensitive to, and measure performance. The statistics are more powerful if one runs the system on the same set of problems at each level of the factor one is varying, but this isn't essential.

4.4 Comparison Studies

The basic form of a comparison study involves solving a corpus of problems in two or more ways (at least one of which is one's own system, the subject of the study) and comparing performance. Importantly, performance of one's system is judged relative to a standard (the other problem-solvers in the study). Comparison studies are common when we lack absolute standards, or when we have absolute standards but we don't know *practical* maximum or minimum levels of performance. For example, on an absolute scale a planner might satisfy between zero percent and one hundred percent of its goals, but will either number ever arise in an experiment? And if 100% will never arise, is 80% good? We don't know until we know what's a practical maximum (Cohen, Ch. 3), and for this purpose, it often helps to compare a system's performance with other systems.

Comparison studies can be grouped by what they use as standards of comparison. Mature work in a mature field might be compared with other, related techniques, whereas new, revolutionary work might stand alone. Four distinctions along these lines are:

Paradigmatic comparisons. No specific comparison is made. Rather, promises of future comparisons are made. This is typically the case when a radically new methodology

is proposed for solving a class of problems. The claims are typically equally radical, involving a change in the average case complexity of the task being considered or the enabling of substantial new functionality of recognized importance. Paradigmatic evaluation only makes sense for work that corresponds to a paradigm shift in the field itself.

Alchemical comparisons. Comparisons are made only to previous work of the researcher who runs the comparisons. In some cases, there is no relevant work *other* than earlier versions of the system being considered. This will generally be the case as new paradigms are explored, so alchemical evaluation can be expected to follow paradigmatic evaluation. Alternatively, systems yielding new functionality are often developed by modifying their predecessors. Comparing this sort of a new system to *other* systems is premature until the new functionality is well understood, since the system will typically incur constant factor costs related to the new functionality itself. In this case, it may make the most sense to compare the system's performance to its performance before the new functionality was added.

The reason this is called "alchemical" is that it is easy to believe that you are personally on the track of turning lead into gold and that the most effective measure of your future progress is relative to your previous work. This is a trap: The purpose of alchemical evaluation ought to be to bridge a short-term gap during which it is inappropriate to perform more scientific experiments. What is gained if someone, Matt Ginsberg, say, reports that adding defaults made his planner 137.2 times faster? Nothing, really. The gain comes when Ginsberg shows that defaults make *someone else's* planner faster.

Scientific comparisons. Comparisons are made to other researchers in the field. It is most appropriate in a mature discipline doing business as usual. To evaluate a technology, dynamic backtracking, for example, one should not write a search engine, test it, add dynamic backtracking and test it again. One should find someone else's search engine, and add dynamic backtracking to that. This is why Ginsberg evaluated dynamic backtracking by incorporating it into TABLEAU and GSAT, the best satisfiability engines around at the moment. To do less would risk falling into the trap of alchemical evaluation.

In general, scientific comparisons ought to return more than one bit of information; they ought to tell us more than, "Algorithm A beat algorithm B." Indeed, it is difficult to imagine a single evaluation criterion for any piece of work: it may be that dynamic backtracking helps on one class of problems but hurts on others. There is nothing wrong with a scientific evaluation describing the class of problems for which a particular technique can help – provided that help is relative to the best other techniques available at the time, and not merely relative to another implementation by the same researcher.

Commercial comparisons. Eventually, work will be evaluated by end users, who will in effect do some "comparison shopping" among alternative technologies. Since re-

searchers are unlikely to be on hand during this process, it is the researcher's responsibility to make sure his or her techniques are presented in their best light.

Clearly, one reason for comparison experiments is to see who's techniques are better for end users, but it's equally clear that other kinds of comparisons, yielding different kinds of information, are necessary. Thus, the Planning Initiative is right not to have made "bakeoffs" the first or only kind of evaluation. Although the MUC competition is sometimes suggested as a model for the Planning Initiative, it raises some methodological issues:

- Special-purpose solutions will usually outperform general ones, so in this sense comparison studies discourage the development of general results and techniques.
- Little can be learned from comparison studies if the test problems are known to participants ahead of time.
- If system A's functionality is a subset of system B's, then they can be compared only on problems both can solve; yet these may be relatively unimportant, uncommon, or easy problems. Should system A outperform system B on these problems and be declared the "winner," we would have some sympathy for the losers.
- MUC returned only two bits of information for each comparison between systems, and, from a statistical standpoint, these conclusions were in any case dubious. In other words, not much was learned from the comparisons themselves.

All these problems can produce misleading results, and the danger is greater in the Planning Initiative than, say, MUC, because so few scenarios are available for testing applied systems, and these scenarios are documented in considerable detail before the tests are run. In contrast, MUC tested systems on 100 problems or more, and these were distinct from problems released for development purposes, and they were kept secret until the tests were run.

With these preliminaries, let's consider some examples of comparison studies.

Example 1. Ginsberg evaluated his dynamic backtracking algorithm by comparing it with an older version of dynamic backtracking and to WSAT (a nonsystematic method). Ginsberg tried to conform to the standard for evaluation in the Boolean satisfiability community: Randomly generated problems are produced from a fixed distribution (e.g., random 3-SAT). Theoretical papers paying lip service to experimentation typically run experiments on hundreds of such problems with search spaces of size around 10^{30}. Experimental papers typically run experiments on approximately 100,000 problems with search spaces of size up to 10^{700}. The fact that the search space is of size 10^x does *not* mean that many nodes are examined, of course, since it's possible to prune large portions of the search space using any of a variety of heuristics. Ginsberg's comparison to an older version of dynamic backtracking involved 400 trials of sizes 10^3 to 10^{30} (and was limited by the previous version of DB). Comparison to WSAT involved 600 trials of sizes 10^3 to 10^{30}.

Example 2. Causal induction means figuring out causal relationships without the benefit of controlled experiments. Hart and Cohen [ref.] applied a causal induction method to build a causal model of the Phoenix planner. More recently, Cohen's group compared the performance of their causal induction algorithm (called FBD) with Judea Pearl's IC algorithm. The study was simple at one level: run FBD and IC on a bunch of problems and see which performs best. The tricky parts were finding problems and measuring performance:

- Problem generators were used to a) construct "target models," b) generate sets of data that are consistent with the target model. Consistent means "the target model is usually the best explanation of the data." Each such dataset is a "problem." Because different problem generators might produce datasets that are somehow biased toward the FBD or IC algorithms, two generators were used. One was written by a group not affiliated with Cohen or Pearl.
- Real data were also used, but sparingly because they are hard to come by.
- The FBD and IC algorithms have some characteristics in common, but FBD is based on multiple regression and so is designed to produce *predictive* models, while IC is designed to produce models that are consistent with conditional independence constraints. Thus it was important to evaluate both algorithms on both kinds of criteria, to see how faithful FBD is to conditional independence, and how faithful IC is to covariance data.
- It was important to understand why IC and FBD performed as they did. Over thirty measurements were made on each trial; most were not performance measurements but were intended to explain performance. For example, it helps to know that when FBD fails to discover a direct causal pathway from A to B, it usually discovers one from A through another variable to B.
- Random target models (and datasets) are relatively easy to produce but they don't *mean* anything. A qualitative analysis of IC and FBD on a real dataset is an essential complement to the previous analyses. Nothing is measured, no winner is declared. Instead, the models produced by IC and FBD are examined, link by link, to see whether they make sense, given what's known about the source of the dataset.

In sum, the purpose of the comparison is not to see who won but to understand why each algorithm performed well or poorly according to particular measures. The measures were designed to see whether (and why) each algorithm did what it was designed for, but each algorithm was evaluated on all the measures.

Example 3. This is an oldie-but-goodie, not a planning system, but a famous expert system, MYCIN. We include it here, even though it isn't a planning system, because of what it tells us about how to evaluate by comparison to expert standards.

- The MYCIN team assessed performance two ways: was the diagnosis correct and was the therapy recommendation correct?

- Test cases were real problems from medical archives. Each problem had been solved by a physician, who we'll call the "original" physician.
- Lacking objective standards, the MYCIN team had a couple of options: To compare MYCIN's performance to that of the original physician, or to ask nationally-recognized experts for their opinions of MYCIN's performance.
- To guard against the possibility that the problems taken from the archives were very easy, the MYCIN team asked four physicians, an internist, a resident, and a medical school student to solve them. The team then had seven standards (including the original physician) against which to compare MYCIN. Ten cases were solved by each of these standards and by MYCIN, and the resulting recommendations were judged by a team of nationally-recognized experts.
- To guard against the possibility that the nationally-recognized experts were biased for or against MYCIN, the team "blinded" them by not telling them whether recommendations were generated by MYCIN or by the seven standards.

Blinding is an essential tactic whenever human judges are involved, as they are in the Planning Initiative because we don't have objective standards. Blinding will be hard to do with Planning Initiative technology, but if we don't do it, then we must live with the likelihood that judges are influenced in part by how the technology performs and in part by biases for and against the technology. Blinding requires test problems to be solved by at least two agents—say, your system and a human—and the solutions are then shown to an expert without attribution. Another excellent reason to have at least two agents solve problems is to control for the possibility that the problems are too easy or too difficult. For instance, we might find that a system cannot handle the Tunisia Scenario, but can *anyone*? Alternatively, we might find that a scheduling system keeps traffic flowing smoothly through a network, but as we described earlier, sometimes a random rule will perform almost as well.

The general schema for comparison studies is this:

Type of study: Comparison

Goals: To assess the performance of a system relative to a standard and, ideally, to explain deviations from the standard.

Tasks: As noted, tasks can come from problem generators or from real problems.

Measurements: More are better; when comparing two or more systems it's a good idea to evaluate them on a wide variety of measures because they were probably designed to do slightly different things.

Procedure: Some variants are:

- Your system and another system are compared on a bunch of objective measures.

- Your system is compared to human performance on objective measures.
- Your system and a human are compared by human judges who are blinded.
- Your system is compared to a default rule or a known optimal approach.

4.5 Demonstration Studies

This section is incomplete.

5 Merging Programmatic and Scientific Evaluation

The experimenter gets to choose tasks and measures—the organizers of the Planning Initiative pointedly decline to prescribe tasks and measures. Still, we want to work on tasks that are relevant to the programmatic goals of the Planning Initiative, and we want to assess the performance of our programs in many ways, including those that are related to the criteria of the end-users of our technology. It must be said that the Planning Initiative boasts few examples of work that satisfies both scientific and programmatic criteria. In this section we focus on how to rectify this state of affairs. We will begin with a list of reasons it exists, an attempt to explain why it has been difficult to satisfy both kinds of criteria. This is an attempt to understand a problem before fixing it. This done, we will suggest some minor modifications to the Planning Initiative that should make it easier to achieve both programmatic and scientific goals.

Ideally, good science should lead to good applications. Let us suppose the science in the Planning Initiative is good. Why has it produced few good applications? It oversimplifies matters (but not by much) to say that the science community is working on different problems than the operational community needs solved. To some extent this is because the two communities have different agendas, but let us suppose the research community genuinely wants to be instrumental in developing good applications. What stands in the way?

One issue is the *accessibility* of the task domain. Realistic scenarios are large and complicated, and only a handful exist. It takes a long time to learn them (and to make graduate students learn them). And besides, most researchers are working on component technology, and it's not clear how to make it "slot in" to an application that solves an entire NEO problem, for example. Now, to a great extent, these impediments have been partly cleared. We have domain packages to help us understand the domain and the CPE as an environment in which our components can be tested.

Another issue, which is particularly germane to a Handbook on Evaluation, is how to demonstrate success in programmatic terms. What can we say about a scheduler, for instance, that will entice an end user to adopt it? The language of the research community is not accessible to users.

So much for impediments. What can be done about them? Let us take as given that it's the responsibility of the research community to clear them away; for instance, it is

our responsibility to describe our accomplishments in domain terms rather than abstruse AI terms. But the operational community has some responsibility, too. It must provide programmatic evaluation criteria; for example, it must say, "We require a scheduler to handle 10,000 objects of 300 classes. We require real-time performance, which means one-hour turnaround for such-and-such problems." And so on. All of this might be implicit in the domain packages, but there's a great advantage to making tasks and evaluation criteria explicit.

To this end, here is a proposed "compact" between the operational and scientific communities. Its central feature is a "Rosetta Stone" of evaluation criteria:

- The criteria should state in domain terms and AI terms what problem is to be solved, and what are the target levels of performance. This way, if a researcher decides to work on a *different* domain problem, he or she can tell whether it is an *equivalent* problem.
- These criteria should be provided by experts and shaped into information-processing terms by the researchers. The experts should make sure that the criteria are realistic and reflect as much of their decision- making as can be formalized in current KR tools and languages. The researchers should make sure that these formalizations capture the expert's criteria, and reach similar conclusions when given similar data.
- The compilation of these criteria should be centralized, so experts and researchers go through the painstaking knowledge acquisition effort only once.
- These criteria should be publicly available and widely distributed in two forms: domain-specific informal descriptions, and their corresponding algorithmic formalizations. This would allow both experts and scientists to communicate.
- There should be a brief document that explains to non-experts what they need to know about the domain in order to understand these criteria. This information should be sufficient to work on the application.
- The criteria can change and be updated (say once a year). This makes the experts happy, because they can fine-tune the evaluations. It also makes scientists happy, because it is harder to keep up with systems that have the criteria hard coded.
- The criteria can be grouped into several sets, each set producing a different evaluation of the same plan but all criteria sharing the same underlying domain terms. Each set of criteria may correspond to a different domain expert.
- Last, each set of criteria should be characterized by researchers in terms of how they affect the performance of planning systems. For example, a domain-specific criterion could be "prefer plans that use both air and sea locations." A characterization of a criterion could be "needs to be handled as multiple positive goal interactions."

Something like this was tried in the MUC competitions, although, as noted above, the evaluation criteria were information-poor. Still, everyone knew what problem had to be solved and they knew the rules of the game.

Something like it was the cornerstone of the ARPA Speech Understanding Initiative in the 1970's. An explicit list of criteria was provided at the outset (e.g., unrestricted speech with a vocabulary of approximately 1000 words, with a grammar ABF of 30, in a moderately noisy environment ...).

Another example that is not so well known is the Sisyphus conferences. Each year, the conference organizers distribute in the call for papers a real problem in the form of a document. The participants submit papers describing how their systems address the problem (or parts of it). This year's meeting was last week, and the problem handed out was taken from the SALT application of Sandy Marcus for configuring elevators. The problem was summarized in a document by Yost (DEC) and an on-line ontology provided by Gruber (Stanford). In previous years, a 4-page description of an office assignment problem was used which was still real but more concise. It is remarkable that this knowledge-based systems community makes a huge investment in this effort to compare their systems.

Within the Planning Initiative, "The PRECiS document" by Reece, Tate, Brown and Hoffman is an effort to summarize and share data in a scenario that is both realistic and manageable. This document sets good grounds to compare planning systems by providing a common scenario (that can be publicly distributed outside the initiative, which is a plus). This document, however, focuses on plan generation and does not get into evaluation criteria, i.e., how to compare the quality of the alternative plans that may be generated. Evaluation criteria are the subject of another document by Tate, Hoffman, and Gil.

In sum, the "compact" between the research and operational communities is simple: Both communities agree on the problems that need to be solved, and they describe the problems in operational and information-processing terms. Both communities agree on what constitutes success, and they describe these evaluation criteria, also, in operational and information-processing terms. This done, the research community can see what's required, and the operational community can see what's been accomplished.

6 Suggested Readings

Chinchor, N. Hirschman, L., and Lewis, D. ...

Cohen, P. R. 1994. Empirical Methods for Artificial Intelligence. MIT Press. Forthcoming.

Cook and Campbell

Hanks, S., Pollack, M. and Cohen, P.R. .....

Hirschman, L ...

Howe, A. E. and Cohen, P. R., 19XX

Howe, A. E., 19XX

Leary

Oates, T. and Cohen, P.R., ...

Weems. C. ...

Wilkins, and ?.