SRI International

Technical Note 489 • May 1990

Introducing the Tileworld: Experimentally Evaluating Agent Architectures

Prepared by:

Martha E. Pollack
Artificial Intelligence Center
Center for the Study of Language and Information
and
Marc Ringuette
School of Computer Science
Carnegie Mellon University

This paper will appear in the *Proceedings of the National Conference on Artificial Intelligence* (AAAI-90), Boston, Massachussetts, August, 1990.

This research was supported by the Office of Naval Research Center under Contract No. N00014-89-C-0095, by a contract with the Nippon Telegraph and Telephone Corporation and by a gift from the System Development Foundation

SRI International®

333 Ravenswood Ave. • Menlo Park, CA 94025
(415) 326-6200 • TWX: 910-373-2046 • Telex: 334-486

Introducing the Tileworld: Experimentally Evaluating Agent Architectures *

Martha E. Pollack
Artificial Intelligence Center *and*
Center for the Study of Language and Information
SRI International
333 Ravenswood Ave.
Menlo Park, California 94025
pollack@ai.sri.com

Marc Ringuette
School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania 15123
mnr@cs.cmu.edu

Abstract

We describe a system called Tileworld, which consists of a simulated robot agent and a simulated environment which is both dynamic and unpredictable. Both the agent and the environment are highly parameterized, enabling one to control certain characteristics of each. We can thus experimentally investigate the behavior of various meta-level reasoning strategies by tuning the parameters of the agent, and can assess the success of alternative strategies in different environments by tuning the environmental parameters. Our hypothesis is that the appropriateness of a particular meta-level reasoning strategy will depend in large part upon the characteristics of the environment in which the agent incorporating that strategy is situated. We describe our initial experiments using Tileworld, in which we have been evaluating a version of the meta-level reasoning strategy proposed in earlier work by one of the authors [Bratman *et al.*, 1988].

Introduction

Recently there has been a surge of interest in systems that are capable of intelligent behavior in dynamic, unpredictable environments. Because agents inevitably have bounded computational resources, their deliberations about what to do take time, and so, in dynamic environments, they run the risk that things will change while they reason. Indeed, things may change in ways that undermine the very assumptions upon which the reasoning is proceeding. The agent may begin a deliberation problem with a particular set of available options, but, in a dynamic environment, new options may arise, and formerly existing options disappear, during the course of the deliberation. An agent that blindly pushes forward with the original deliberation problem, without regard to the amount of time it is taking or the changes meanwhile going on, is not likely to make rational decisions.

One solution that has been proposed eliminates explicit execution-time reasoning by compiling into the agent all decisions about what to do in particular situations [Agre and Chapman, 1987, Brooks, 1987, Kaelbling, 1988]. This is an interesting endeavor, but its ultimate feasibility for complex domains remains an open question.

An alternative is to design agents that perform explicit reasoning at execution time, but manage that reasoning by engaging in *meta-level reasoning*. Within the past few years, researchers in AI have provided theoretical analyses of meta-level reasoning, often applying decision-theoretic notions to it [Boddy and Dean, 1989, Horvitz, 1987, Russell and Wefald, 1989]. In addition, architectural specifications for agents performing meta-level reasoning have been developed [Bratman *et al.*, 1988], and prototype systems that engage in meta-level reasoning have been implemented [Cohen *et al.*, 1989, Georgeff and Ingrand, 1989]. The project we describe in this paper involves the implementation of a system for experimentally evaluating competing theoretical and architectural proposals.

More specifically, we have been constructing a system called Tileworld, which consists of a simulated robot agent and a simulated environment which is both dynamic and unpredictable. Both the agent and the environment are highly parameterized, enabling one to control certain characteristics of each. We can thus experimentally investigate the behavior of various meta-level reasoning strategies by tuning the parameters of the agent, and can assess the success of alternative strategies in different environments by tuning the environmental parameters. Our hypothesis is that the appropriateness of a particular meta-level reasoning strategy will depend in large part upon the characteristics of the environment in which the agent incorporating that strategy is situated. We shall describe below how the parameters of our simulated environment correspond to interesting characteristics of real, dynamic environments.

*This research was supported by the Office of Naval Research under Contract No. N00014-89-C-0095, by a contract with the Nippon Telegraph and Telephone Corporation and by a gift from the System Development Foundation.

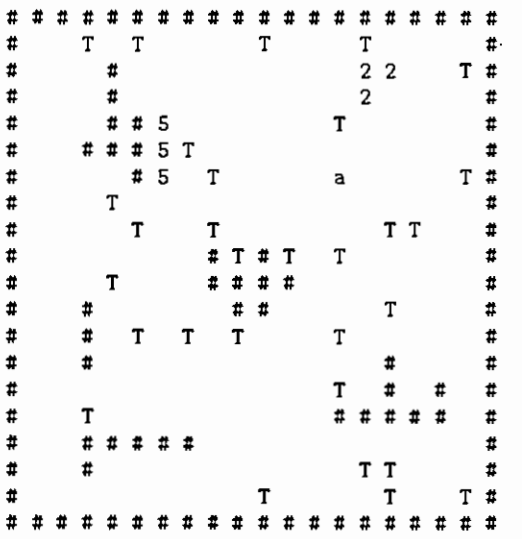


Figure 1: A Typical Tileworld Starting State

In our initial experiments using Tileworld, we have been evaluating a version of the meta-level reasoning strategy proposed in earlier work by one of the authors [Bratman *et al.*, 1988]. However, the Tileworld can be used to evaluate a range of competing proposals, such as the ones mentioned above: agents instantiating many alternative proposals can readily be imported into the Tileworld environment.

The Tileworld Environment

The Tileworld is a chessboard-like grid on which there are agents, tiles, obstacles, and holes. An agent is a unit square which is able to move up, down, left, or right, one cell at a time, and can, in so doing, move tiles. A tile is a unit square which "slides": rows of tiles can be pushed by the agent. An obstacle is a group of grid cells which are immovable. A hole is a group of grid cells, each of which can be "filled in" by a tile when the tile is moved on top of the hole cell; the tile and particular hole cell disappear, leaving a blank cell. When all the cells in a hole are filled in, the agent gets points for filling the hole. The agent knows ahead of time how valuable the hole is; its overall goal is to get as many points as possible by filling in holes.

Figure 1 depicts a typical Tileworld starting state. A Tileworld simulation takes place dynamically: it begins in a state which is randomly generated by the simulator according to a set of parameters, and changes continually over time. Objects (holes, tiles, and obstacles) appear and disappear at rates determined by parameters set by the experimenter, while at the same time the agent moves around and pushes tiles into holes. The dynamic aspect of a Tileworld simulation distinguishes it from many earlier domains that have been used for studying AI planning, such as blocks-world.

The Tileworld can be viewed a rough abstraction of the Robot Delivery Domain, in which a mobile robot roams the halls of an office delivering messages and objects in response to human requests. We have been able to draw a fairly close correspondence between the two domains (i.e., the appearance of a hole corresponds to a request, the hole itself corresponds to a delivery location, tiles correspond to messages or objects, the agent to the robot, the grid to hallways, and the simulator time to real time).

Features of the domain put a variety of demands on the agent. Its spatial complexity is nontrivial: a simple hill-climbing strategy can have modest success, but when efficient action is needed, more extensive reasoning is necessary. But the time spent in reasoning has an associated cost, both in lost opportunities and in unexpected changes to the world; thus the agent must make trade-offs between speed and accuracy, and must monitor the execution of its plans to ensure success. Time pressures also become significant as multiple goals vie for the agent's attention.

Of course, a single Tileworld simulation, however interesting, will give only one data point in the design space of robot agents. To explore the space more vigorously, we must be able to vary the challenges that the domain presents to the agent. We have therefore parameterized the domain, and provided "knobs" which can be adjusted to set the values of those parameters.

The knob settings control the evolution of a Tileworld simulation. Some of the knobs were alluded to earlier, for instance, those that control the frequency of appearance and disappearance of each object type. Other knobs control the number and average size of each object type. Still other knobs are used to control factors such as the shape of the distribution of scores associated with holes, or the choice between the instantaneous disappearance of a hole and a slow decrease in value (a hard bound versus a soft bound). For each set of parameter settings, an agent can be tested on tens or hundreds of randomly generated runs automatically. Agents can be compared by running them on the same set of pseudo-random worlds; the simulator is designed to minimize noise and preserve fine distinctions in performance.

The Tileworld environment is intended to provide a testbed for studying a wide range of dynamic domains and tasks to be performed in them. It exhibits spatial complexity, a central feature of many such domains; and it includes tasks of varying degrees of importance and difficulty. It is generic: although we have explored connections between Tileworld and tasks involving robot delivery, Tileworld is not tightly coupled to any particular application domain, but instead allows an experimenter to study key characteristics of whatever domain he or she is interested in, by varying parameter settings. For example, the experimenter can focus on domains in which the central characteristic is

a wide distribution of task values (simulated in Tileworld by hole scores), or of task difficulty (simulated by hole size). In this regard, Tileworld differs from the Phoenix simulator [Cohen *et al.*, 1989], which is more closely tied to a particular application. Instead, the goals of the Tileworld project are closer to those of the MICE simulator [Durfee and Montgomery, 1990]. However, Tileworld is a more highly dynamic environment than MICE. Also, where MICE is used to focus on issues of real-time inter-agent coordination, Tileworld is intended as a framework for the more general investigation of intelligent behavior in dynamic environments.

Using Plans to Constrain Reasoning

The agent we have implemented and used in our experiments instantiates IRMA—the Intelligent Resource-Bounded Machine Architecture [Bratman *et al.*, 1988]. IRMA builds on observations made by Bratman [Bratman, 1987] that agents who are situated in dynamic environments benefit from having plans because their plans can constrain the amount of subsequent reasoning they need to perform. Two constraining roles of plans concern us here:

- An agent's plans focus subsequent means-end reasoning so that the agent can, in general, concentrate on elaborating its existing plans, rather than on computing all possible courses of action that might be undertaken.
- An agent's plans restrict the set of further potential courses of action to which it needs to give full consideration, by filtering out options that are inconsistent with the performance of what the agent already plans to do.

The first role of plans has always been at least implicit in the standard models of AI planning: AI planners compute means to goals that the agent already has. The second has a more dramatic effect on the architecture we are investigating: it leads to the introduction of a *filtering mechanism*, which manages execution-time reasoning by restricting deliberation, in general, to options that are compatible with the performance of already intended actions. (To have the desired effect of lessening the amount of reasoning needed, the filtering mechanism must be computationally inexpensive, relative to the cost of deliberation.)

Of course, a rational agent cannot *always* remain committed to its existing plans. Sometimes plans may be subject to reconsideration or abandonment in light of changes in belief. But if an agent constantly reconsiders its plans, they will not limit deliberation in the way they need to. Thus, an agent's plans should be reasonably stable.

To achieve stability while at the same time allowing for reconsideration of plans when necessary, the filtering mechanism should have two components. The first checks a new option for compatibility with the existing plans. The second, an override mechanism, encodes the conditions under which some portion of the existing plans is to be suspended and weighed against some other option. The filter override mechanism operates in parallel with the compatibility filter. For a new option to pass through the filter, it must either pass the compatibility check or else trigger an override by matching one of the conditions in the override mechanism. A critical task for the designer of an IRMA-agent is to construct a filter override mechanism so that it embodies the right degree of sensitivity to the problems and opportunities of the agent's environment.

The options that pass through the filter are subject to deliberation. The deliberation process is what actually selects the actions the agent will form intentions towards. In other words, it is the deliberation process that performs the type of decision-making that is the focus of traditional decision theory. The filtering mechanism thus serves to frame particular decision problems, which the deliberation process then solves.

The process of deliberation is different from means-ends reasoning in our view, and this distinction is worth discussing fnrther. As we see it, deliberation is deciding *which* of a set of options to pursue, while means-ends reasoning is more a process of determining *how* to achieve a given goal. We see means-ends reasoning producing *options* (candidate plans to achieve a goal), which can then be the subject of deliberation.

This may be a surprising distinction to those familiar with the standard AI planning paradigm, in which the job of a planner is usually to produce the single best plan according to some set of criteria. Any deliberation which is to be done in such a system is done by the planner, and it might be argued that a planner is the best place for such reasoning. Certainly some pruning of alternatives must be done by a planner; however, there are reasons to believe that some deliberation belongs outside the planner. In some situations it is appropriate to have several means-ends reasoners with differences in solution quality and time required; these must be invoked appropriately and a single solution chosen. In other circumstances it is desirable to engage in a decision-theoretic analysis of competing alternatives. Consequently, we have maintained the distinction between deliberation and means-ends reasoning in our system.

The Tileworld Agent

In implementing an IRMA-agent for the Tileworld, we adopted a model of a robot with two sets of processing hardware. One processor executes a short control cycle (the *act cycle*), acting on previously formulated plans and monitoring the world for changes. The second processor executes a longer cycle (the *reasoning cycle*), which permits computations with lengths of up to several seconds.

The act cycle is straightforward; the agent performs

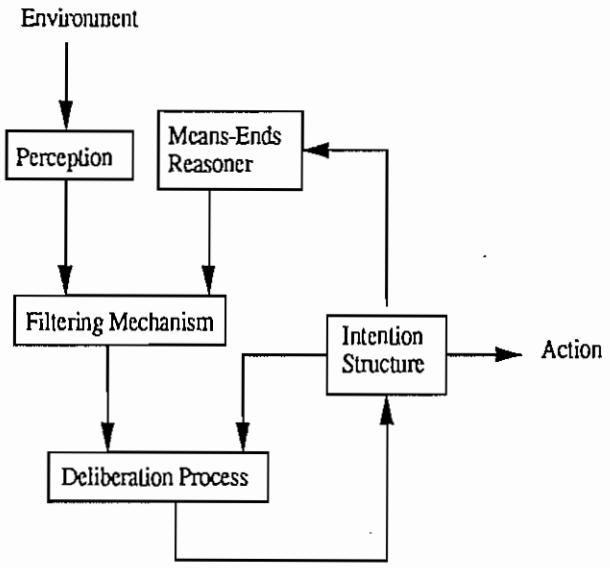


Figure 2: Tileworld Agent Architecture

those acts that have been identified during the previous reasoning cycle, monitoring for limited kinds of failures. Perception also occurs during the act cycle: the agent can access a global map of the world that indicates the locations of all objects, as well as the score and time remaining to timeout for all holes.

The reasoning cycle makes decisions about what goals to pursue and how to pursue them. The portion of the agent architecture that controls reasoning is depicted in Figure 2. Processing is aimed at maintaining the *intention structure*, a time-ordered set of tree-structured plans that represents that agent's current intentions. During any given reasoning cycle, one of two things can happen:

- Potential additions to the intention structure, called *options*, can be considered by the filtering and deliberation processes. These options can come from two sources. One, the agent may perceive environmental changes that suggest new options—in Tileworld, this occurs when new holes or tiles appear. Alternatively, options may be suggested by the means-end reasoner.
- Means-ends reasoning can be performed to produce new options that can serve as means to current intentions. The bulk of our means-ends reasoner is a special-purpose route planner.

We will concentrate here on the filtering and deliberation mechanisms. All options are in principle subject to filtering and deliberation; so far, however, we have confined such reasoning to top-level options, i.e., options to fill a particular hole.

Recall that the IRMA filtering mechanism has two parts: the compatibility filter and the filter override. An option passes the filter if it is either compatible with all existing intentions, or if it triggers an override.

Compatibility checking of top-level options, as implemented, is straightforward. A top-level option is either to fill a hole *now* or *later;* if the agent already has a current intention to fill a particular hole *now*, then an option to fill some other hole *now* is incompatible. All intentions to fill a hole *later* are compatible with each other.

The filter override must identify options that are potentially valuable enough that they warrant deliberation even if they fail the compatibility test. The simplest override mechanism compares the score of a hole being considered as an option to that of the hole currently being filled. If the difference between them equals or exceeds some threshold value v, then the new option passes the filter. The threshold value is set by a Tileworld parameter. Sometimes it may sensibly be set to a negative value: in that case, a new option could be subject to deliberation even if it involved filling a hole with a lower score than the hole currently being filled. This might be reasonable, since the new hole may, for instance, be much easier to fill. Setting the threshold value to $-\infty$ results in all options being subject to deliberation.

Recall that an option's passing the filter does not lead directly to its introduction into the intention structure: instead, it is passed to the deliberation process for more detailed consideration and comparison with the current intention. Deliberation may involve extensive analysis; deliberation strategies can be chosen in the Tileworld agent by the setting of a parameter. We currently have implemented two deliberation strategies.

The simpler deliberation module evaluates competing top-level options by selecting the one with the higher score. When there is a nonnegative threshold value in the filter, this mode of deliberation always selects the new option; with a negative threshold value, it instead always maintains the current intention. This illustrates a general point: if deliberation is extremely simple, it may be redundant to posit separate deliberation and filtering processes.

A more sophisticated deliberation strategy computes the likely value (LV) of a top-level goal. LV is an estimate of expected utility, combining information about reward (score) with information about likelihood of success. For a given option to fill a hole h, LV is computed as

$$LV(h) = \frac{score(h)}{dist(a,h) + \sum_{i=1}^{n} 2 * dist(h,t_i)}$$

where $score(h)$ is the reward for filling the hole, $dist(a,h)$ is the distance between the agent and the hole, n is the number of tiles needed to fill the hole, and $dist(h,t_i)$ is the distance from the hole to the i^{th} closest tile. The factor of 2 occurs because the agent must traverse the interval in both directions, i.e., it must make a "round trip". If there are fewer than n

tiles available, $LV(h)$ is zero.

We intend to design additional deliberation modules, including one that performs complete means-end reasoning for all options under consideration before making its decision. Such a deliberator must not be invoked carelessly; we expect our filtering mechanism to be increasingly useful as we add more sophisticated and time-consuming deliberation components.

Preliminary Experiments

With both the simulator and the agent in place, we are in a position to conduct experimental studies of the the performance of the agent. By adjusting the Tileworld "knobs", we can control a number of domain characteristics. We can vary what we call ***dynamism*** (the rate at which new holes appear), ***hostility*** (the rate at which obstacles appear), ***variability of utility*** (differences in hole scores), ***variability of difficulty*** (differences in hole sizes and distances from tiles), and ***hard/soft bounds*** (holes having either hard timeouts or gradually decaying in value). There are also variables we can adjust in the agent: ***act/think rate*** (the relative speeds of acting and thinking), the filter's ***threshold level***, and the ***sophistication of the deliberation mechanism***.

Experiment 1

To begin with, we set all of these parameters to provide a baseline environment which is dynamic, variable, and moderately paced. In this environment, a competent agent can achieve reasonable scores, but is penalized for wasting time or making poor choices. We will start by comparing the simple deliberation mechanism, based on score value, with the LV evaluator, which provides a better estimate of marginal utility. For orientation, we have also included the results of a human playing the role of the agent in the same simulation; and to gain an idea of the benefit of acting in parallel with reasoning, we have included results for an agent that that acts and reasons serially.

All of these agents were tested in the baseline environment and in a similar but more rapidly changing one. In the faster environment, objects appear and disappear on the average ten times more quickly, but the agent can also move ten times more quickly. However, the agent's reasoning takes place at the same rate of speed as in the baseline case, so the opportunity cost of reasoning is correspondingly greater in the faster environment. The agents were all evaluated by taking the average score from 30 trials; the human performed 10. Each trial is a self-contained simulation with a duration of 5000 ticks of the clock, where the agent can move once per clock tick.

Speed	Score for Agent				
	LV	LV/ Serial	Simple	Simple/ Serial	Human
Normal	396	353	347	291	468
Fast	256	234	183	152	3

Experiment #1

The differences here are quite apparent. In the normal speed environment, the human subject performed best, because he had more-sophisticated planning capabilities than the robot. But in the faster environment, the human's response speed was insufficient to allow him to keep up with the pace of change.

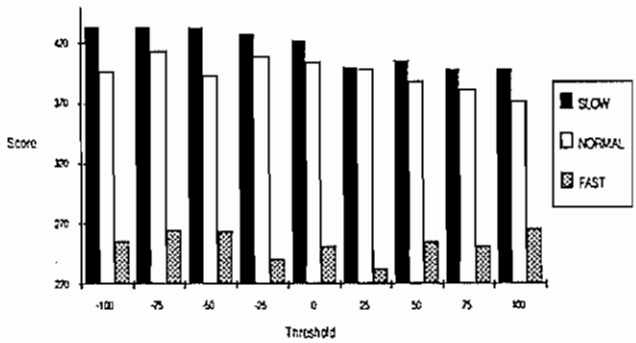
The robot agents were better able to adjust to the more rapidly changing environments, but it is clear that the cost of reasoning is still significant for them. This is evident both from an overall decrease in score in the high-speed environment, and from the superiority of the robot agents that could reason and act in parallel.

The other distinction of note is that the LV evaluator performs better than the simple evaluator, as expected.

Experiment 2

We now move on to our initial experiments directed at understanding some of the design trade-offs in our agent. The use of Tileworld to experimentally evaluate our agent architecture is an ongoing project, and these are early results. We stress that the hypotheses presented below are preliminary; significantly more experimentation and statistical analysis of the results need to take place before we can make strong claims about the relative appropriateness of any particular agent-design strategy.

In Experiment 2, we attempt to test the usefulness of the filtering mechanism in our agent as implemented, using the LV evaluator as the deliberation component, and using the most quickly computed evaluation metric, thresholding on the score value, as the filter override mechanism. We vary the threshold from -100 to 100. Since the score for each hole ranges from 1 to 100, a threshold setting of -100 means that every new option is subject to deliberation, while a setting of 100 means that no new option will ever be considered until the currently executing plan is complete. The resulting scores are summarized in the following chart, where each value represents an average over 30 trials.



Experiment #2

At the slowest speed setting, 100 times slower than our "normal" setting, it is better to do no filtering at all. The scores achieved at this speed decrease consistently as the threshold is increased. At the normal

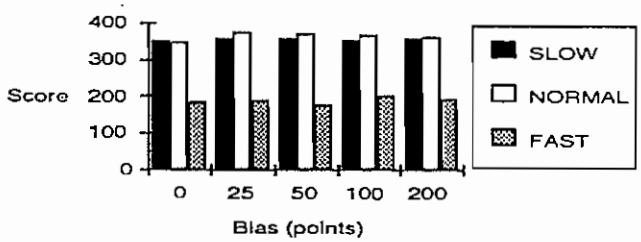
speed setting, the effect of increased filtering still appears to be negative, but less markedly so. At a setting 10 times faster than the normal one, there seems to be little correlation between threshold level and performance, although the uncertainty in the results, which appears to be in the range of 10-20 points, prevents a sure determination. We hope, in the future, to be able to make even these relatively subtle determinations; the noise in the data comes, we believe, largely from our decision to use actual CPU-time measurements to determine reasoning time. If we wish to get the cleanest trials possible, we may need to use a time estimate that does not depend on the vagaries of the underlying machine and Lisp system. Failing that, we will need to model the uncertainty involved, and run larger trial sets.

To sum up the results of this experiment, we see that filtering is harmful at slow speeds, and even at high speeds does not give a net benefit. Our hypothesis is that the time cost of the LV evaluator is not very high, and consequently, it is usually worth taking the time to engage in extra deliberation about new opportunities. The fact that filtering is less detrimental in the faster environment leads us to hypothesize that there may be a break-even point at even faster speeds, above which filtering is useful; we intend to test for such a point. We also intend to implement more accurate (and costly) deliberation mechanisms in the near future. For these, filtering may be much more valuable; perhaps the LV-estimator is efficient enough that it can itself be used as the filter override mechanism for the more complex deliberation components.

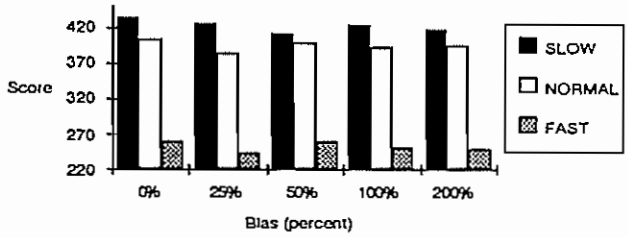
Experiment 3

In our third experiment, we attempt to test a conjecture that the LV evaluator as described is deficient in an important way: it does not consider the time cost of means-end reasoning already performed. We modify the deliberation functions by adding a bias in favor of existing intentions, since typically at deliberation time, some means-end reasoning about how to achieve these has already taken place. This is distinct from Experiment 2, in which we adjusted the *filtering* mechanism in an attempt to save deliberation time; here we investigate a bias in the deliberation process itself with the intent of reducing the time cost of *means-end reasoning*.

We consider two cases. In the first, deliberation is done by the simple evaluator, and we apply a bias towards existing intentions equal to a fixed number of points. In the second, deliberation is done by the LV evaluator, and we apply a bias equal to a fraction of the current LV. Thus, for example, with a 100 percent bias, a newly appearing hole must have double the LV of the current one to be adopted as a new intention. The environment settings and simulation sizes are the same as for Experiment 2.



Experiment #3: Simple Evaluator



Experiment #3: LV Evaluator

As shown by the experimental results, bias in the deliberator does not appear to have a clear effect on total performance. For the simple evaluator, this isn't terribly surprising; it provides a fairly weak assessment of a hole's actual potential value in any case. We expected to see much more effect of bias on the LV evaluator, however. Two hypotheses are available to explain this. First, our test environment may have too many opportunities available, minimizing the potential cost of high bias: if the agent spends most of its time doing something with high utility, a few missed opportunities will not have a significant impact on the final score. This hypothesis can be tested in a less favorable environment. Second, it may be that means-end reasoning in the current implementation is too inexpensive, minimizing the potential benefit of high bias. This hypothesis can be tested by increasing the *size* of the environment to increase the planning time required; the addition of more complex planning routines would also provide situations in which there is a higher time cost associated with planning.

Conclusion

The experiments we have run to date have included some important milestones in the Tileworld effort. The Tileworld domain has been demonstrated, and has been shown to be a viable system for evaluating agent architectures. The Tileworld agent was demonstrated and used to test differing deliberation and filtering strategies as described in [Bratman *et al.*, 1988].

The Tileworld project is ongoing. There are a number of specific research tasks that we intend to pursue in the near future. Perhaps most importantly, we will be continuing our experimental efforts. The hypotheses we drew from our preliminary experiments

suggested several obvious follow-ons, as described in the preceding section. It will be particularly useful to vary parameters other than those that control speed, for example, size of the overall space, distribution of task value and difficulty, and availability of limited resources such as tiles.

We will also implement more sophisticated deliberation algorithms, and, having done so, will attempt to identify better the principles separating the processing that is done in the filtering mechanism from that done in the deliberation procedure. In addition, we plan to implement a foveated perceptual scheme, in which the agent has access to detailed, precise information about its immediate surroundings and has only increasingly abstract, incomplete, and uncertain information about about more distant locations in its environment. Another possibility is to add learning to the system: two areas of potential benefit are in the means-ends reasoner (e.g., explanation-based learning of control rules) and in evaluations of marginal utility (e.g., empirical improvement of utility evaluations). Finally, we hope to extend the architecture to handle more difficult questions involving intention coordination. We expect that both means-end reasoning and deliberation will become much more difficult, and hence filtering much more important, when the intention structure involves more complex interactions among intentions.

More generally, we continue to investigate the larger question of how an agent should structure and control its computational effort. We believe that the architecture discussed here is a special case of a more general framework, and we are working towards a definition of that framework and its verification in our domain. We also see the Tileworld testbed as a good basis for comparison of other agent architectures proposed in the literature, and we strongly encourage other researchers to demonstrate their agents in our domain.[1]

The overall goal of our project is an improved understanding of the relation between agent design and environmental factors. In the future, when faced with a performance domain for an agent, one should be able to draw on such an understanding to choose more wisely from the wide range of implementation possibilities available.

References

[Agre and Chapman, 1987] P. E. Agre and D. Chapman. Pengi: An implementation of a theory of activity. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, Seattle, WA, 1987.

[Boddy and Dean, 1989] M. Boddy and T. Dean. Solving time-dependent planning problems. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, Detroit, MI, 1989.

[Bratman *et al.*, 1988] M. E. Bratman, D. J. Israel, and M. E. Pollack. Plans and resource-bounded practical reasoning. *Computational Intelligence*, 4(4), 1988.

[Bratman, 1987] M. E. Bratman. *Intention, Plans and Practical Reason*. Harvard University Press, Cambridge, MA, 1987.

[Brooks, 1987] R. A. Brooks. Planning is just a way of avoiding figuring out what to do next. Technical Report 303, MIT, Cambridge, MA, 1987.

[Cohen *et al.*, 1989] P. R. Cohen, M. L Greenberg, D. M. Hart, and A. E. Howe. Real-time problem solving in the phoenix environment. In *Proceedings of the Workshop on Real-Time Artificial Intelligence Problems*, Detroit, MI, 1989.

[Durfee and Montgomery, 1990] E. H. Durfee and T. A. Montgomery. MICE: A flexible testbed for intelligent coordination experiments. In L. Erman, editor, *Intelligent Real-Time Problem Solving: Workshop Report*, Palo Alto, CA, 1990. Cimflex Teknowledge Corp.

[Georgeff and Ingrand, 1989] M. P. Georgeff and F. F. Ingrand. Decision-making in an embedded reasoning system. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, Detroit, MI, 1989.

[Horvitz, 1987] E. J. Horvitz. Reasoning about beliefs and actions under computational resource constraints. In *Proceedings of the 1987 Workshop on Uncertainty in Artificial Intelligence*, Seattle, WA, 1987.

[Kaelbling, 1988] L.P. Kaelbling. Goals as parallel program specifications. In *Proceedings of the Seventh National Conference on Artificial Intelligence*,St. Paul, MI, 1988.

[Russell and Wefald, 1989] S. J. Russell and E. H. Wefald. Principles of metareasoning. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, Toronto, 1989.

[1] The Tileworld domain is relatively clean and portable, is written in CommonLisp, and is available electronically over the Internet from Marc Ringuette by sending electronic mail to `mnr@cs.cmu.edu`.