Report No. UIUCDCS-R-88-1452

Introducing Actions into Qualitative Simulation

by

Kenneth D. Forbus

August 1988

Qualitative Reasoning Group
Department of Computer Science
University of Illinois
1304 W. Springfield Avenue
Urbana, Illinois, 61801, USA

Supported in part by the Office of Naval Research
under grant N00014-85-K-0225

Introducing Actions into Qualitative Simulation

Abstract

Many potential uses of qualitative physics, such as robot planning and intelligent computer-aided engineering, require integrating physics with actions taken by agents. This paper proposes to augment qualitative simulation to include the effects of actions to form *action-augmented envisionments.* The action-augmented envisionment incorporates both the effects of an agent's actions and what will happen in the physical world whether or not the agent does something. Consequently, it should provide a richer basis for planning and procedure generation than any previous representation. This paper defines action-augmented envisionments and an algorithm for directly computing them, along with an analysis of its complexity and suitability for different kinds of problems. We describe our initial implementation and discuss potential extensions, including incremental algorithms.

Keywords: Qualitative reasoning, planning, artificial intelligence.

Presented at the 2nd Qualitative Physics Workshop
Paris, France
July, 1988

1 Introduction

Many tasks require predicting both the effects of changes in the physical world and the consequences of taking actions. A robot which makes tea, for instance, must enlist physical processes such as liquid flow and boiling to carry out its plan. An intelligent CAD tool for power plant design must be able to reason about the effects of actions taken by operators of the plant in order to design a safe system. Yet little work to date has addressed the issue of integrating a qualitative physics with representations of actions.

One way to integrate physics with action is to move the physics into the planner. Hogge [12] has developed an *domain compiler* that takes QP domain models and produces rules and operators suitable for a temporal planner. Given a goal like "Increase the water level in this container", his planner can use the knowledge of actions, combined with rules and "operators" representing what the physical world will do derived from the QP model, to figure out that it should place the container under a faucet and turn on the tap. Unfortunately, many oversimplifications are required to keep the compilation tractable[1]. Furthermore, adding more run-time knowledge to overcome this and other problems makes the temporal planner bog down [13,14]. For example, Hogge's original planner could figure out how to get water into an empty pot and how to make water in a full pot boil, but without adding run-time transitivity rules it could not compose these plans to fill an empty pot with boiling water. With transitivity rules, the planner would exceed machine resource limitations before finding the solution. While worth continued exploration, the difficulty of reconstructing the entire framework of qualitative simulation into rules suitable for efficient planning makes exploring alternatives worthwhile.

Here we explore the dual approach: Moving actions into the physics. The next section introduces a new representation, the *action-augmented envisionment* (or $\mathcal{AE}$), which integrates the results of qualitative simulation with the effects of an agent's actions. Section 3 describes an algorithm for computing $\mathcal{AE}$'s directly, and Section 4 analyzes its complexity and potential suitability for two tasks, robot planning and procedure generation in engineered systems. Section 5 describes the state of our implementation. Finally, we describe our plans for future work.

2 Action-Augmented Envisionments

Let us re-consider what qualitative states are. We will use Hogge's problem of figuring out from first principles how to boil water as an example in the rest of this paper. Informally, a qualitative state describes a class of particular behaviors for a physical system. Qualitative states are linked by transitions, which describe how these gross behaviors can change. For instance, if we have a pot of water on an operating stove, one state is that the water is heating up, and another state is that the water is boiling. These states have a transition

[1]For instance, the compiler assumes that if you influence a quantity in a particular direction it will actually change that way. Thus the planner would believe that one could bail out a sinking ocean liner with a teaspoon.

Figure 1: QP descriptions can be sensitive to the effects of actions
In the description of heat flow below, the truth of HEAT-ALIGNED depends on the location of the objects involved. Actions which change location indirectly affect whether or not this process can occur, as illustrated by the laws on the bottom.

```
Process Heat-Flow(?src,?dst,?path)
    Individuals:  ?src, Quantity(heat(?src))
                  ?dst, Quantity(heat(?dst))
                  ?path, Heat-Path(?src,?dst)
    Preconditions:  Heat-Aligned(?path)
    Quantity Conditions:  A[T(?src)] > A[T(?dst)]
    Relations:  Quantity(flow-rate)
                flow-rate = T(?src) - T(?dst)
    Influences:  I+(heat(?dst),A[flow-rate])
                 I-(heat(?src),A[flow-rate])

∀ ?c,?s Contained-Stuff(?c) ∧ Location(Container(?c))=On(?s) ⇒ Heat-Path(?s,?c)
∀ ?c Contained-Stuff(?c) ⇒ Heat-Aligned(?c)
Knob(Stove)=ON ⇒ Heat-Aligned(Stove)
∀ ?x,?y Heat-Path(?x,?y) ∧ Heat-Aligned(?x) ∧ Heat-Aligned(?y)
                          ⇒ Heat-Aligned(Heat-Path(?x,?y))
```

between them, whose condition is that the temperature of the water reaches its boiling temperature. Qualitative simulation consists of computing these states and transitions.

Every qualitative simulation leaves some "background information" unchanged. We do not, for example, consider what the world would look like if the stove suddenly vanished in the scenario above. In fact, qualitative simulations focus on just those changes predictable solely within whatever physics is being used. The complete set of states and transitions for some fixed set of background assumptions is the envisionment for that scenario. (When needed, we will use the conventions of [10,3,16,17] to describe envisionments and their components.) To capture the effects of actions, we must allow at least some of the background assumptions to vary. In the scenario above, for instance, we would like to capture the fact that switching the stove on will initiate the heat flow, and that moving the pot to a table will break thermal contact, and thus end the heat flow.

A necessary prerequisite for this extension is that the qualitative physics be sensitive to changes in background assumptions. Qualitative Process theory [6] provides two forms of explicit representation of, and hence dependence on, background assumptions. First, QP descriptions specify the kinds of individuals they apply to. Heat flow, for instance, can occur between any two objects which are modeled as having thermal properties, and which have some kind of thermal path connecting them. Second, QP descriptions can depend upon explicit *preconditions* that further restrict their applicability. For instance, one might consider the burner of a stove to be a heat path which is usable only when the stove is switched on (see Figure 1). Consequently, we use QP theory as our basis.

As indicated above, we do not typically consider every possible change in background assumptions. Call the set of background assumptions for a scenario $\mathcal{P}$. The subset of $\mathcal{P}$

which should be varied is exactly that part which could be changed, directly or indirectly, by the action of some agent. We call these the *manipulable assumptions* of $\mathcal{P}$, or $\mathcal{P}_m$. Clearly, $\mathcal{P}_m$ will depend on the set of operators used to model an agent's actions, and the laws which allow the effects of those operators to be inferred.

In typical qualitative physics systems, $\mathcal{P}_m$ is empty. That is, the envisionment $\mathcal{E}$ is taken with respect to fixed $\mathcal{P}$, which we can denote $\mathcal{E}(\mathcal{P})$. Let $\mathcal{P}_m^\star$ be the consistent combinations of $\mathcal{P}_m$, and $\mathcal{P}_f$ be the set of fixed background assumptions (i.e., $\mathcal{P} - \mathcal{P}_m$). Then the set of states in an action-augmented envisionment $\mathcal{AE}$ is just

$$States(\mathcal{AE}) = \bigcup_{p \in \mathcal{P}_m^\star} \mathcal{E}(p \cup \mathcal{P}_f)$$

Since standard envisionments contain transitions due to changes predicted within the physics, we assume the collection $States(\mathcal{AE})$ constructed so far inherits them intact. Now we must extend the set of transitions to include the occurrences of actions. We do this by analogy to the QP definition of state transitions. In QP theory, state transitions are represented as instances of *limit hypotheses*, potential changes in inequality relationships brought about by the direct and indirect effects of physical processes. The hypothesis that, for instance, the temperature of the water in the pot might reach its boiling temperature would be applicable to any situation where the water is being heated, regardless of the heat source involved. Similarly, we call an *action hypothesis* the conjecture that a particular action occurs. In thinking about the stove, for instance, the operator instance `Move-to(Pot1,On(Stove))` might occur in a number of states. Each conjectured occurrence is an action hypothesis.

We place several restrictions and constraints on transitions caused by action hypotheses. For simplicity, we make the following restrictions:

1. *Single action assumption:* At most one action can be taken at a time.

2. *Seperation assumption:* Actions do not coincide with state transitions introduced by the physics.

The single action assumption loses no generality, since the vocabulary of operators could always include compound operations. An important consequence of the seperation assumption is that actions cannot occur in states which the dynamics predicts will only last an instant. Unfortunately, it does potentially restrict the expressibility of $\mathcal{AE}$s. To hold, actions must be considered to occur quickly, relative to physical changes. For many circumstances this assumption is not onerous; for instance, the temperture of water in a kettle doesn't drop appreciably in the time it takes to move the kettle from the stove to a teapot. Also, in many cases where actions do take appreciable time (such as slowly opening a valve in a heating system) this limitation could be surmounted by modeling the action as a sequence of instantaneous actions or reifing it as a continuous changes in the physics triggered by actions.

An action hypothesis $\mathcal{A}_\mathcal{H}$ can be viewed as a function whose domain and range are $\mathcal{P}_m^\star$. Let $\mathcal{P}_m^2 = \mathcal{A}_\mathcal{H}(\mathcal{P}_m^1)$. Given a qualitative state $\mathcal{S}_1$ in which $\mathcal{P}_m^1$ holds, for $\mathcal{S}_2$ to be a possible result of $\mathcal{A}_\mathcal{H}$ on $\mathcal{S}_1$, it must satisfy the following restrictions:

1. *Consistency:* $\mathcal{P}_m^2$ holds in $\mathcal{S}_2$.
2. *Continuity:* When possible, no violations of continuity occur between $\mathcal{S}_1$ and $\mathcal{S}_2$.
3. *Closeness:* No state also considered to be a possible result of $\mathcal{A}_{\mathcal{Y}}$ on $\mathcal{S}_1$ has more in common with $\mathcal{S}_1$ than $\mathcal{S}_2$ does.

The consistency restriction is obvious. The continuity and closeness restrictions express the desiderata that, besides the necessary indirect consequences of the action, nothing else should change as a result of it.[2] All reasonable interpretations of closeness imply continuity, but it is worth mentioning explicitly because it is a useful filter. Unfortunately, continuity cannot always be satisfied. Consider a situation where the pressure in a boiler is rising dangerously, and safety valve is popped open to bleed off excess steam. The result of opening the valve can most easily be modeled by a discontinuous change where the pressure in the boiler is dropping[3]. There are several ways to define closeness, depending on the details of the qualitative physics and simulation strategy. A particular measure of closeness for envisioning in QP theory is described in the next section.

3 An Algorithm for constructing $\mathcal{A}\mathcal{E}$s

This algorithm for computing $\mathcal{A}\mathcal{E}$s is based on the representations used in the Qualitative Process Engine (`QPE`) [9]. Since `QPE` is based on an ATMS [4], and the formulation of $\mathcal{A}\mathcal{E}$s is based on describing different sets of assumptions, our algorithm will be particularly simple. In what follows we exploit the fact that a particular *situation* in `QPE` is defined by a set of assumptions $\mathcal{S}_A = \mathcal{Q}_s \cup \mathcal{P}_s$, where $\mathcal{Q}_s$ are drawn from the set of possible inequality assumptions and $\mathcal{P}_s$ are drawn from $\mathcal{P}$. Explicit temporal notations, such as situation markers or slices, are not used. Rather, the temporal scoping of facts is implicit in their ATMS label. Thus we would determine if `Location(Pot1) = On(Stove)` held in a situation by checking to see if that fact was implied by the assumptions defining the situation. This allows us to compactly represent a large number of situations, and apply consequences of rules as widely as possible.

For simplicity we choose the STRIPS representation for actions. While less expressive than other action representations (e.g. [1,18,19]), it easily satisfies the single action and seperation assumptions. To adapt this representation to `QPE`, we require all facts mentioned in the add lists and delete lists of operator instances to be in $\mathcal{P}_m$.

Given a domain model, which specifies the particular physical theory to be used, and a scenario, specified by $\mathcal{P}_f$, `QPE` expands the scenario by applying the abstractions of the domain model. This creates instances of views, processes, and derived objects (such as "water in the pot"). It is easy to extend this process to include finding operator instances, and to automatically accumulate $\mathcal{P}_m$. Since `QPE` can search variations in $\mathcal{P}_s$ as well as $\mathcal{Q}_s$, *States*($\mathcal{A}\mathcal{E}$)is computed via the standard envisioning procedure. Furthermore, since

[2]The existence of indirect consequences of actions are why generally $\mathcal{S}_2 - \mathcal{P}_m^2 \neq \mathcal{S}_1 - \mathcal{P}_m^1$.

[3]It would be possible to preserve continuity by introducing another state where `Ds[P] = 0`, but since this state serves no other purpose it seems inefficient to do so.

Figure 2: Operators for the kitchen domain

```
defOperator Move-To(?from,?thing,?to)
    Individuals:  ?from, Place(?from)
                  ?thing, Mobile(?thing)
                  ?to, Place(?thing) ∧ ?from ≠ ?to
    Delete-List:  Location(?thing) = ?from
                  ¬ Clear(?from)
                  Clear(?to)
    Add-List:     Location(?thing) = ?to
                  ¬ Clear(?to)
                  Clear(?from)

defOperator Flip(?switch,?from,?to)
    Individuals:  ?switch, Switch(?from)
                  ?from, Has-Setting(?switch,?from)
                  ?to, Has-Setting(?switch,?to) ∧ ?from ≠ ?to
    Delete-List:  ?switch = ?from
    Add-List:     ?switch = ?to
```

we have the operator instances we can create the set of $\mathcal{A}_{\mathcal{Y}}$'s. All that remains is (1) to ascertain when these $\mathcal{A}_{\mathcal{Y}}$'s are applicable and (2) to determine their effect in each case.

Consider again the operators in Figure 2. We will refer to the assumptions corresponding to the delete list and add list of an operator instance as $\mathcal{A}^s_{\mathcal{Y}}$and $\mathcal{A}^e_{\mathcal{Y}}$, respectively. To determine if an operator instance $\mathcal{O}_i$ can apply to a situation $\mathcal{S}_1$,

1. Unless $Individuals(\mathcal{O}_i)$ are implied by $\mathcal{S}_1$, fail.

2. Let $\mathcal{P}'_s$= ($\mathcal{P}_s$- $\mathcal{A}^s_{\mathcal{Y}}$) + $\mathcal{A}^e_{\mathcal{Y}}$. If $\mathcal{P}'_s$ is inconsistent, fail.

To complete the test we must find out if $\mathcal{P}'_s$ can be extended into a consistent situation $\mathcal{S}_2 \in States(\mathcal{AE})$. Otherwise, we consider the action inapplicable.

Since we already have $States(\mathcal{AE})$, finding the results of $\mathcal{O}_i$ on $\mathcal{S}_1$ can be viewed as a filtering problem whose result is the set $\mathcal{C}$. One algorithm is:

1. Let $\mathcal{C}_1 = \{\mathcal{S}_j \in States(\mathcal{AE}) \mid \mathcal{P}'_s \subseteq \mathcal{S}_j\}$

2. Let $\mathcal{C}_2$ be the subset of $\mathcal{C}_1$ which do not violate continuity, when viewed with respect to $\mathcal{S}_1$. If $\mathcal{C}_2$ is empty, $\mathcal{C}_2$= $\mathcal{C}_1$.

3. Let $\mathcal{C} = \{\mathcal{S}_j \in \mathcal{C}_2 \mid \nexists \mathcal{S}_k \in \mathcal{C}_2 \ s.t. \mid \mathcal{S}_k \cap \mathcal{S}_i \mid > \mid \mathcal{S}_j \cap \mathcal{S}_i \mid\}$

The first step provides our initial candidates by enforcing consistency, and the second uses the same continuity pruning used for limit analysis in QPE (see [9] for details). The final step provides a precise definition for the notion of closeness described before – in this algorithm, it is literally the number of assumptions shared with the previous situation. For each $\mathcal{S}_j \in \mathcal{C}$, the set $Transitions(\mathcal{AE})$ is extended to include a transition from $\mathcal{S}_i$ to $\mathcal{S}_j$, justified by $\mathcal{O}_i$.

Typically an action will result in a unique next state. Unfortunately, this will not always be the case, due to the ambiguity of qualitative representations. Consider again the boiler with relief valve. Once the relief valve blows there will be ambiguous influences on the pressure – the flow out through the relief valve will act to decrease it, while the generation of more steam will act to increase it. Consequently, the pressure could continue to increase, decrease, or remain constant, and so unless extra knowledge can disambiguate them, each is a legitimate consequence of that action.

4 Analysis

Our analysis addresses three questions: (1) What is the complexity of explicitly generating $\mathcal{AE}$? (2) Under what circumstances would explicit generation make sense? (3) Could $\mathcal{AE}$ be generated incrementally?

4.1 Complexity of the algorithm

The first question can be divided into two parts: (1) Given that the standard envisioning process can compute $States(\mathcal{AE})$, how complicated is the additional step of generating $\mathcal{A}_{\mathcal{H}}$'s and determining their consequences, and (2) How much does it cost to generate $States(\mathcal{AE})$ relative to a standard envisionment $\mathcal{E}$ (i.e., where $\mathcal{P}_m$ is empty).

Finding operator instances is easy. The worst-case complexity for each operator is $\mathcal{O}(i^t)$, where t is the number of specifications in the operator's individual field and i is the number of individuals in the scenario. Clever indexing on individual types, along with the fact that t tends to be small (around 3), makes this step trivial. Notice that this result does not depend on the number of situations, an advantage conferred by the implicit temporal reference scheme.

We assume the ATMS is arranged so that tests for logical implication and consistency of a set of assumptions are constant-time operations (they typically are). Let $n = \mid States(\mathcal{AE}) \mid$. Then finding whether or not operator instance $\mathcal{O}_i$ may apply in a particular situation takes a constant-time test for implication, and time linear in the number of assumptions to produce $\mathcal{P}'_s$. This cost is roughly constant over all situations, and depends on the average number of assumptions in a situation. This number is relatively small compared to the number of situations which can be generated from them, hence we consider it constant. Finding the initial candidate set $\mathcal{C}_1$ is also linear in n, so we are now bounded by $\mathcal{O}(n^2)$. The continuity computation and the computation of $\mathcal{C}$ from $\mathcal{C}_2$ are combinations of linear time operations, so the cost of adding action transitions is $\mathcal{O}(n^2)$.

What is the cost of computing $States(\mathcal{AE})$ relative to $\mathcal{E}$? The complexity of QPE's algorithm is still being established, so we must content ourselves with asking about the size of $States(\mathcal{AE})$ relative to $States(\mathcal{E})$, which depends on the size of $\mathcal{P}^{\star}_m$. Suppose $\mathcal{P}_m$ consists of pairs of propositions p and $\neg p$, and these assumptions are independent. Then an upper bound for the worst-case increase in the number of states is a factor of $2^{\mid \mathcal{P}_m \mid -1}$. So the signficant cost lies not in temporal inheritance, but in generating the states in the first place.

4.2 When would explicit generation make sense?

It is generally foolhardy, and typically impossible, to explicitly generate an entire problem space. Yet that is exactly what envisioning does, and the algorithm above relies on it. How close we come to worst-case performance depends on the interactions between the operator vocabulary and the rest of the domain model. If the operators are completely irrelevant to the domain model, then $\mid States(\mathcal{AE}) \mid = \mid \mathcal{P}_m^* \mid \times \mid States(\mathcal{E}) \mid$. But generally they interact, and only a small subset of the cross product is consistent. A domain where the dynamical behavior is complex, but the number of actions which can be taken is small, would be the best case.

The problem of *procedure generation* for engineered systems may be just such a problem. Consider a power plant (either stationary or onboard a ship). Its dynamical state can be complex, and a badly-timed action can result in disaster. But the kind of actions an operator can take are generally limited to flipping switches and opening or closing valves. Since an $\mathcal{AE}$ compactly represents the result of executing all possible plans, it could be useful in generating operating procedures and safety analyses. To deal with realistic systems will require the same decomposition stratagies as traditional engineering. For instance, procedures for a system are typically generated by combining procedures for subsystems. This suggests computing $\mathcal{AE}$'s for subsystems independently and combining their results.

4.3 Incremental generation

Typical "robot planning" domains are the worst kind of task for explicit $\mathcal{AE}$ generation, since $\mathcal{P}_m^*$ includes each location for each moving object, and thus could be huge. Incremental algorithms would be better, and appear both possible and feasible. The $\mathcal{AE}$ is just a problem space, whose operators are the actions which can be taken plus the set of limit hypotheses. Incremental temporal inheritance algorithms for QP theory exist [8,7], and could easily be extended to $\mathcal{A}_{\mathcal{H}}$s. The entire panoply of AI search strategies could potentially be used to generate plans. However, the fact that some transitions will occur whether or not the agent desires them changes the nature of the problem somewhat.

One way to view planning in the $\mathcal{AE}$ problem space is as playing a game with Nature. The agent controls actions, and Nature controls dynamics. This view is not exact, of course, since Nature is not generally held to have goals[4] and the "players" in this game don't take strict turns. Nevertheless, the metaphor is suggestive. For example, there is a "horizon effect" in this problem space which consists of dynamical changes undoing a state achieved by the agent. To assure that the intended effect of the action is maintained, qualitative simulation can be used to see if either (a) no dynamical transitions occur or (b) they take sufficiently long that the next action can be performed before they occur.

[4]Although Murphy's Law is tantamount to assuming that Nature is playing to win, i.e., to thwart the agent's goals if possible.

5 An implemenation

We have implemented the algorithm described in Section 3, and have tested it on one example at this writing. Here is the scenario: Consider a kitchen containing a table, a faucet, a stove, and a movable pot. The faucet is an infinite supply of water, and the stove is an infinite supply of heat, exactly when their respective knobs are `ON` (viz. Figure 1). Our goal is to have boiling water in the pot. The process vocabulary includes heat flow, liquid flow, and boiling [6], and the actions are those shown in Figure 2.

Here are the highlights of the $\mathcal{AE}$ created for this scenario. There are 244 situations, with 1054 transitions between them (78 due to dynamics, the rest due to actions). `QPE` automatically divides situations into equivalence classes for summarization, and in this description there are only 21 states, with 76 transitions between them (10 due to dynamics). The summaries for the standard envisionment, action transitions, and full $\mathcal{AE}$ are plotted in figure 3. In terms of the analysis presented earlier, this result is encouraging – this is a "worst-case" problem, after all. For this problem, $\mid States(\mathcal{E}(\mathcal{P}_f)) \mid = 25$, and there are 12 binary choice sets, and one set with three choices (i.e., the location of the pot). Thus the worst case would have been

$$\mid States(\mathcal{AE}) \mid = 25 \times 2^{12} \times 3 = 307,200$$

This ilustrates that simple combinatorial analyses can be misleading in highly constrained situations – the classic AI "small infinity" phenomena.

A simple graph-search planner was built to find plans given a start state, goal, and $\mathcal{AE}$. This $\mathcal{AE}$ does indeed contain many correct plans for boiling water (see figure 4), and careful examination has revealed no unexpected oddities. Currently we are developing a series of test cases to ensure our closeness requirement is sufficient. If it is not, then we suspect that *p-components* [6] will be required to narrow the subset of the situation in which violations of continuity are allowed.

6 Discussion

This paper proposes a method for integrating qualitative physics with models of action. It defines the *action-augmented envisionment*, which compactly represents all predicted changes due to a physics and possible actions within a scenario. An algorithm for explicitly generating augmented envisionments was presented, along with an analysis and a report on our initial implementation.

We believe this idea is an important step towards interfacing qualitative physics with planning. Such understanding could lead to important new applications, such as increased automation of procedure generation and safety analyses. We suspect that for some engineering applications, the cost of explicit generation of $\mathcal{AE}$'s may be offset by the increased confidence in the quality of the answer, particularly as we discover how to build multi-grain domain models [5]. However, even if $\mathcal{AE}$'s turn out to be infeasible to explicitly compute for all but the simplest systems, we expect the $\mathcal{AE}$ representation will be a useful theoretical foundation for developing incremental planning techniques.

Figure 3: An Augmented Envisionment for the Kitchen
The top left shows a summary of $States(\mathcal{AE})$ with only the limit hypotheses, and the top right shows the $\mathcal{A}_{\mathcal{H}}$s. The bottom shows the combination, with a reasonable plan marked.

Figure 4: A sample plan from the $\mathcal{AE}$
Here is a sample plan generated by graph search of $\mathcal{AE}$.

```
1.  MOVE-TO(ON(COUNTER1),POT,UNDER(FAUCET1)).
2.  FLIP(KNOB(FAUCET1),OFF,ON).
3.  Wait until A[AMOUNT-OF-IN(WATER,LIQUID,POT)]=ZERO ⟶ >.
4.  FLIP(KNOB(FAUCET1),ON,OFF).
5.  MOVE-TO(UNDER(FAUCET1),POT,ON(STOVE1)).
6.  FLIP(KNOB(STOVE1),OFF,ON).
7.  Wait until A[TBOIL(WATER,POT)]>A[TEMPERATURE(C-S(WATER,LIQUID,POT))] ⟶ =.
8.  Wait until A[AMOUNT-OF-IN(WATER,GAS,POT)]=ZERO ⟶ >.
9.  FLIP(KNOB(STOVE1),ON,OFF).
10.  MOVE-TO(ON(STOVE1),POT,ON(COUNTER1)).
```

6.1 Future Work

We are currently extending this work in several ways:

- We are testing our initial implementation with a variety of new examples, including subsystems from Navy propulsion plants (e.g., [15]) and NASA's planned Space Station.

- We are formalizing constraints for plan evaluation, including efficiency and safety. These constraints will be used to automatically select good plans via graph search from an $\mathcal{AE}$, and should be useful in formulating incremental generation techniques.

- A more sophisticated planner is needed. For example, the current representation of sensing steps assumes that the parameter involved in the transition can be directly sensed. To construct executable plans requires integrating the ideas of measurement from [11].

- We are investigating different representations for duration to reduce the ambiguity in plan-finding. An interesting possibility is to use partial quantiatitive information, as in Kuiper & Berleant's recent work [2].

- Incremental generation strategies require incremental qualitative simulation. Dennis DeCoste is designing an incremental version of QPE which could be harnessed for this purpose.

7 Acknowledgements

This research was supported by the Office of Naval Research, Contract No. N00014-85-K-0225, and an NSF Presidential Young Investigator award.

References

[1] Allen, J. and Koomen, J. "Planning using a temporal world model", Proceedings of IJCAI-83, August, 1983.

[2] Kuipers, B. and Berleant, D. "Using incomplete quantitative knowledge in qualitative reasoning" Proceedings of AAAI-88, August, 1988.

[3] de Kleer, J. and Brown, J. "A qualitative physics based on confluences", *Artificial Intelligence*, **24**, 1984

[4] de Kleer, J. "An assumption-based truth maintenance system", *Artificial Intelligence*, **28**, 1986

[5] Falkenhainer, B. and Forbus, K. "Setting up large-scale qualitative models", Proceedings of AAAI-88, August, 1988.

[6] Forbus, K. "Qualitative process theory" *Artificial Intelligence*, **24**, 1984

[7] Forbus, K. "The problem of existence", Proceedings of the Cognitive Science Society, 1985.

[8] Forbus, K. "Qualitative Process theory" MIT AI Lab Technical report No. 789, July, 1984.

[9] Forbus, K. "The Qualitative Process Engine", Technical Report No. UIUCDCS-R-86-1288, December, 1986. To appear, *International Journal of AI in Engineering*, 1988

[10] Forbus, K. "The Logic of Occurrence", Proceedings of IJCAI-87, August, 1987.

[11] Forbus, K. "Interpreting observations of physical systems", *IEEE Systems, Man, and Cybernetics*, Special issue on Causal and diagnostic reasoning, Vol SMC-17, Number 3, May/June, 1987.

[12] Hogge, J. "Compiling plan operators from domains expressed in Qualitative Process theory", Proceedings of AAAI-87, July, 1987.

[13] Hogge, J. "The compilation of planning operators from Qualitative Process theory models", Technical Report No. UIUCDCS-R-87-1368, September, 1987

[14] Hogge, J. "TPLAN: A temporal interval-based planner with novel extensions", Technical Report No. UIUCDCS-R-87-1367, September, 1987

[15] Hollan, J., Hutchins, E., and Weitzman, L., "STEAMER: An interactive inspectable simulation-based training system", AI Magazine, Summer, 1984.

[16] Kuipers, B. "Common sense Causality: Deriving behavior from Structure" *Artificial Intelligence*, **24**, 1984

[17] Kuipers, B. "Qualitative Simulation", *Artificial Intelligence*, **29**, September, 1986.

[18] McDermott, D. "A temporal logic for reasoning about processes and plans", *Cognitive Science*, Vol 6, 1982.

[19] Vere, S. "Planning in time: Windows and durations for activities and goals", *IEEE Trans. Patt. Anal. Mach. Intell.*, PAMI-5(3), May 1983.