Representing Knowledge within the Situation Calculus using Interval-valued Epistemic Fluents

John Funge
Microcomputer Research Lab
Intel Corporation
www.cs.toronto.edu/~funge

Abstract

The ability of interval arithmetic to provide a finite (and succinct) way to represent uncertainty about a large, possibly uncountable, set of alternatives turns out to be useful in building "intelligent" autonomous agents. In particular, consider the two important issues of reasoning and sensing in intelligent control for autonomous agents. Developing a principled way to combine the two raises complicated issues in knowledge representation. In this paper we describe a solution to the problem. The idea is to incorporate interval arithmetic into the situation calculus. The situation calculus is a well known formalism for describing changing worlds using sorted first-order logic. It can also be used to describe how an agent's knowledge of its world changes. Potentially, this provides a sound basis for incorporating sensing into logic programming. Previous work has relied on a possible worlds approach to knowledge. This leads to an elegant mathematical specification language. Unfortunately, there have been no proposals on how to implement the approach. This is because the number of possible worlds is potentially uncountable. We propose an alternative formalization of knowledge within the situation calculus. Our approach is based on intervals. The advantage is that it is straightforward to implement. Moreover, we can prove that it is sound and (sometimes) complete with respect to the previous possible worlds approach.

1 Introduction

The work described in this paper grew out of an attempt to build an "intelligent" autonomous agent. Inspired by new work in the area, we choose to use the situation calculus as the foundation of our approach. The situation calculus is a well-known formalism for representing changing worlds. Our intention was to use the situation calculus to represent the agent's knowledge. The logical basis of the situation calculus meant that the agent would be able to reason about the effect of its actions. It would, therefore, be able to choose a course of action that it believed would result in a desired effect. Unfortunately, the world the agent was to be situated in was highly complex. To some degree, the result of its actions would be unpredictable. It was clear that we needed to incorporate sensing into our framework.

Fortunately, the issue of sensing had already been looked at in the context of the situation calculus, but there was a problem. The previous approach to sensing was based on the idea of *possible worlds*, borrowed from modal logic. The theory was elegant and powerful, but it wasn't clear how to implement it. It seemed that in order to specify the agent's knowledge of its world we would be expected to list out an uncountable number of possible worlds!

One possible recourse would, perhaps, to have used a modal logic theorem prover. However, modal logic theorem provers are not widely available, or widely understood. In addition, we didn't want our approach to sensing to be inextricably bound up to the notion of theorem proving. That is, an important aim of ours was to make our approach to intelligent control simple and practical. We wanted our ideas to be useful to programmers whose job it is to build autonomous agents. In particular, we hoped to provide a rapid-prototyping tool that employed specifications that were amenable to gradual refinement to production code.

At this point, it occurred to us that what the notion of possible worlds was allowing us to express was the agent's *uncertainty* about aspects of its world. It struck us that interval arithmetic also provides a way to express uncertainty.

Moreover, intervals provide a finite (and succinct) way to represent uncertainty about a large, possibly uncountable, set of alternatives. The rest of our task (and the remainder of this paper) involved working out the details of how all this fits together. We begin with some necessary background on the situation calculus.

2 Background

The situation calculus [12, 16] is a way of describing change in sorted first-order logic. In this section we shall give the mathematical details required to understand the situation calculus and how we use it. We necessarily assume some familiarity with mathematical logic and the reader is referred to [4] for any additional background information required.

We shall use the following simple example, after [8], to illustrate various points about the situation calculus:

> Suppose we have two agents, called them Dognap and Jack. Let us suppose that Dognap is armed with a gun, and that Dognap wants to kill Jack. Let us further suppose that Jack is initially alive and that the gun is initially empty.

2.1 Sorts

A *situation*, of sort SITUATION, is a "snapshot" of the state of the world. A domain-independent constant S_0, of sort SITUATION, denotes the initial situation.

We want to use various number systems in our theory of action. Logical accounts of numbers are, however, problematic. Consequently, to avoid becoming embroiled in a task that would be at odds with the purpose of this paper we employ the following artifices to eschew the messy details:

- A collection of sorts for various number systems:

$$\begin{aligned} \mathbb{B} &\triangleq \text{Boolean numbers,} \\ \mathbb{N} &\triangleq \text{Natural numbers,} \\ \mathbb{Z} &\triangleq \text{Integer numbers,} \\ \mathbb{Q} &\triangleq \text{Rational numbers,} \\ \mathbb{R} &\triangleq \text{Real numbers.} \end{aligned}$$

  To avoid misunderstanding, we briefly clarify $\mathbb{B}$.[1] In particular, there are two constants of sort $\mathbb{B}$, namely 0 and 1. There is one unary function $\neg$, and two binary functions $\wedge, \vee$.

  Later on, we shall want to ensure that all our number systems have maximal and minimal elements. We shall indicate these augmented number systems with a $\star$, for example the extended real numbers: $\mathbb{R}^\star = \mathbb{R} \cup \{-\infty, \infty\}$. We shall also denote subsets of our number systems with appropriate designators, for example the non-negative reals $\mathbb{R}^+$.

- For each number system sort, we will only consider standard interpretations. That is, we shall work with interpreted theories in which the various functions and constants, associated with the sort, are fixed. There are functions and predicates corresponding to all the standard mathematical functions and predicates for the sort.

- For each number system sort, we assume the existence of an "oracle" that is capable of determining the truth or falsity of sentences about relationships between objects of that sort.[2]

All other objects are of sort OBJECT.

[1] See [9] for further detailed discussion on the subject.

[2] From a practical point of view, we might use mathematical software packages (such as Maple) to handle a wide range of useful queries.

2.2 Fluents

Any property of the world that can change over time is known as a fluent. A *fluent* is a function, with a situation term as (by convention) its last argument. We shall restrict fluents to taking on values in one of the number system sorts. For any functional fluents foo that take on values in $\mathbb{B}$, we shall adopt the standard abbreviation that $\mathsf{Foo}(s)$ is just shorthand for $\mathsf{foo}(s) = 1$. We may refer to such fluents as *relational fluents*.

Let us now introduce some fluents to capture the salient details of our example. This will enable us to formalize the scenario within the situation calculus.

$\mathsf{Alive}(s)$	–	Jack is alive in state s.
$\mathsf{Aimed}(s)$	–	The gun is aimed at Jack in state s.
$\mathsf{Loaded}(s)$	–	The gun is loaded in state s.

Actions, of sort ACTION, are the fundamental instrument of change in our ontology. The situation s' resulting from doing action a in situation s is given by the distinguished function do : ACTION $\times$ SITUATION $\rightarrow$ SITUATION, such that, $s' = do(a, s)$. In our example, we introduce the following actions:

load	–	Load the gun.
aim	–	Aim the gun at Jack.
shoot	–	Shoot the gun.

The possibility of performing action a in situation s is denoted by a distinguished predicate $Poss$: ACTION $\times$ SITUATION. Sentences that specify what the state of the world must be before performing some action are known as *precondition axioms*. We can give such axioms for the actions in our example:[3]

$Poss(load, s)$	–	The gun can always be loaded.
$Poss(aim, s)$	–	The gun can always be aimed at Jack.
$Poss(shoot, s) \Rightarrow \mathsf{Loaded}(s)$	–	The gun can only be shot if it's loaded.

2.3 The Qualification Problem

The *qualification problem* [11] is that of trying to infer when an action is possible. In our example, we only wrote down certain necessary conditions, we did not enumerate all the things that may prevent us from shooting the gun. For instance, we cannot shoot if the trigger is too stiff, or if the gun is encased in concrete, etc. By employing a *closed-world assumption*, we may obviate this problem and assume that our set of necessary conditions is also a sufficient set. For instance, under this assumption our precondition axiom for *shoot* now becomes:

$$Poss(shoot, s) \Leftrightarrow \mathsf{Loaded}(s).$$

In general, we have the following definition:

Definition 2.1 (Action precondition axioms). *Action precondition axioms give necessary and sufficient conditions* $\pi_a(\vec{x}, s)$ *for when an action* $a(\vec{x})$ *is possible. They are of the form:*

$$Poss(a(\vec{x}), s) \Leftrightarrow \pi_a(\vec{x}, s).$$

In [6, 10], some additional subtleties, including those that arise when we allow state constraints, are discussed.

2.4 Effect Axioms

Effect axioms give necessary conditions for a fluent to take on a given value after performing an action. We can use effect axioms to state the effects of the actions on the defined fluents in our example:

$\mathsf{Loaded}(do(load, s))$	–	The gun is loaded after loading it.
$\mathsf{Aimed}(do(aim, s))$	–	The gun is aimed at Jack after aiming it.
$Poss(shoot, s) \wedge \mathsf{Aimed}(s)$	$\Rightarrow$	$\neg\mathsf{Alive}(do(shoot, s))$
	–	If the gun is aimed at Jack and it can be shot then he is dead after shooting it.

[3] Throughout, all unbound variables are implicitly assumed to be universally quantified.

All that now remains to complete our first pass at formalizing our example is to specify the initial situation:

$$\begin{array}{lll} \mathsf{Alive}(\mathrm{S}_0) & - & \text{Initially Jack is alive.} \\ \neg\mathsf{Aimed}(\mathrm{S}_0) & - & \text{Initially the gun is not aimed at Jack.} \\ \neg\mathsf{Loaded}(\mathrm{S}_0) & - & \text{Initially the gun is not loaded.} \end{array}$$

2.5 The Frame Problem

Unfortunately, there are still some impediments to using the situation calculus in real applications. The most notable of these is the so called *frame problem* [12]. The frame problem is that of trying to infer what remains unchanged by an action. In our example, we only wrote down what changed after an action; we did not write down all the things that stayed the same. For instance, the gun stayed loaded after aiming it, or the gun did not turn into a horse after loading it, etc. In common-sense reasoning about actions, it seems essential to assume that, unless explicitly told otherwise, things stay the same. To formally state this "law of inertia", without changing our effect axioms, causes problems. In particular, if we have $\mathcal{A}$ actions and $\mathcal{F}$ fluents, then we must write down a set of $\mathcal{A} \times \mathcal{F}$ "frame" axioms. The problem is exacerbated by the planner having to reason efficiently in the presence of all these axioms.

At this point it is worth commenting on how the frame problem is dealt with in other fields that tackle related problems. Notably in control theory. In control theory we have the notion of a state vector. Each component of the state vector is similar to a fluent. The frame problem is tackled by simply assuming that the state vector completely characterizes the system in question and that values for all the components are explicitly specified. That is, if part of the state vector is unchanged then we must explicitly say so. Usually state vectors are chosen to be short so that this task is not too irksome. It does, however, put our approach into context. We do not want to be constrained to have to give our state vector at the outset. Moreover, we do not want (for reasons given in the preceding paragraph) to list out all the things that don't change. In our approach we can, at any point, mention some new fluent and have the system infer its value with respect to its value in the initial situation. Furthermore, if so desired, we can leave the initial situation underspecified. For example, suppose in the initial situation we say that the car is *either* blue *or* yellow. Now further suppose we perform no actions to affect the color. Then, after the action sequence, we will be able to infer that the car's color is still either blue, or yellow.

In [16], it is shown how we can avoid having to list out all the frame axioms. The idea is to assume that our effect axioms enumerate all the possible ways that the world can change. This closed world assumption provides the justification for replacing the effect axioms with *successor state* axioms. For instance, the successor state axiom for $\mathsf{Alive}(s)$ states that Jack is alive, if and only if, he was alive in the previous state and he was not just shot:

$$\mathit{Poss}(a, s) \Rightarrow [\mathsf{Alive}(\mathit{do}(a, s)) \Leftrightarrow \mathsf{Alive}(s) \wedge \neg(a = \mathit{shoot} \wedge \mathsf{Aimed}(s))]. \tag{1}$$

In general, we have the following definition:

Definition 2.2 (Successor state axioms). *Suppose $\gamma_f(\vec{y}, z, a, s)$ is a first-order formula whose free variables are among $\vec{y}, z, a, s$. Assume it states all the necessary conditions under which action a, if performed in s, results in $f(\vec{y}, s)$ becoming equal to z. Then, the corresponding successor state axiom, that assumes the given conditions are also sufficient ones, is of the form:*

$$\mathit{Poss}(a, s) \;\Rightarrow\; [(f(\vec{y}, \mathit{do}(a, s)) \;=\; z) \;\Leftrightarrow\; (\gamma_f(\vec{y}, z, a, s)) \vee (f(\vec{y}, s) \;=\; z \wedge \neg\exists z' \;\; \gamma_f(\vec{y}, z', a, s))]. \tag{2}$$

It is instructive to consider what this definition means for a relational fluent F. Let $\gamma_F^+(\vec{y}, a, s)$ be a disjunction of all the positive effects of the action a, and $\gamma_F^-(\vec{y}, a, s)$ be a disjunction of all the negative effects. Then the successor state axiom for F is:

$$\mathit{Poss}(a, s) \Rightarrow [F(\vec{y}, \mathit{do}(a, s)) \Leftrightarrow (\gamma_F^+(\vec{y}, a, s) \vee (F(\vec{y}, s) \wedge \neg\gamma_F^-(\vec{y}, a, s)))].$$

2.6 Exogenous Actions

It will often be the case that there are aspects of the domain that we can not, or do not want to, formalize. For example, suppose we are interested in the position of a ball floating in the ocean. It is all but impossible to attempt to formalize the motion of the waves, wind, etc. This will often be the case for phenomena that are outside the agent's ability to

control. We should like to simply define an action like $\mathit{moveBall}(x)$ and say that it is caused by mysterious external forces. Such actions are referred to as *exogenous* actions. While the cause of an exogenous action is difficult to state its effect need not be. For example, the $\mathit{moveBall}(x)$ simply moves the ball to the position x.

3 Knowledge producing actions

Up until now we have thought of actions as having effects on the world. We can, however, imagine actions whose only effect is to change what the agent knows about its world. A good example is an agent trying to make a phone call. The agent needs to know the number before dialing. The action of looking up the phone number has no effect on the world, but it changes the agent's knowledge. Sensing actions are therefore referred to as *knowledge producing actions*.

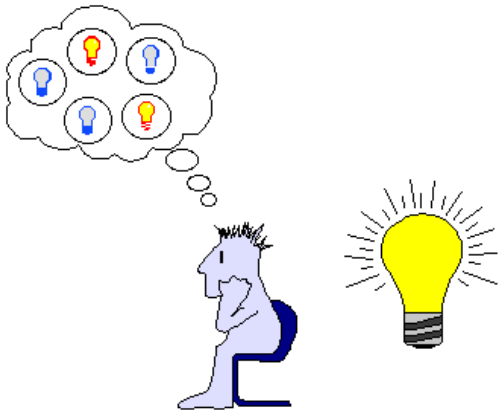

Figure 1: Before sensing, worlds where the light is on, or off, are possible.

In [17], an approach to incorporating knowledge producing actions into the situation calculus is described. The idea behind the approach is to define an epistemic fluent to keep track of all the worlds an agent thinks it might possibly be in. In figure 1, we depict an agent unable to decide which world it was in. That is, whether in its world the light is on or off. Figure 2 shows the agent turning around to see that the light is in fact turned on. The result of this sensing action is shown in the figure as the agent discarding some of the worlds it previously thought were possible. In particular, since it now knows that the light is on in its world, it must throw out all the worlds in which it thought the light was turned off. In this section we give the mathematical details of how this notion is modeled in the situation calculus.

3.1 An epistemic fluent

The way an agent keeps track of the possible worlds or, as the case may be, possible situations is to define an epistemic fluent K. The fluent keeps track of all the K-related worlds. These K-related worlds are precisely the ones in the bubbles above the agent's head in above mentioned figures. They are the situations that the agent thinks might be its current situation. So we write $\mathsf{K}(s', s)$ to mean that in situation s, as far as the agent can tell, it might be in the alternative situation s'. That is, the agent's knowledge is such that s and s' are indistinguishable. It can only find out if it is or not by sensing the value of certain terms, for example terms such as $\mathsf{light}(s)$.

When we say an agent *knows* the value of a term τ, in a situation s, is some constant c, we mean that τ has the value c in all the K-related worlds. For convenience, we introduce the following abbreviation:

$$\mathit{Knows}(\tau = c, s) \triangleq \forall s' \; \mathsf{K}(s', s) \Rightarrow \tau[s'] = c, \tag{3}$$

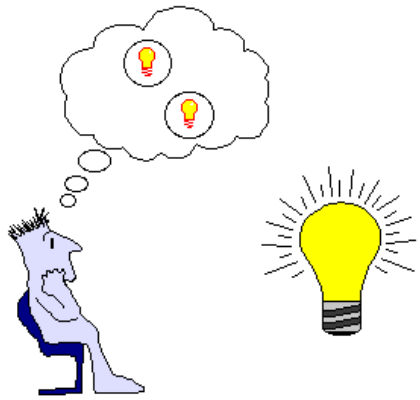

Figure 2: After sensing, only worlds where the light is on are possible.

where $\tau[s']$ is the term τ with the situation arguments inserted. For example, if $\tau = \mathsf{phoneNo}(\mathsf{Jack})$ then $\tau[s] = \mathsf{phoneNo}(\mathsf{Jack}, s)$. Note that for simplicity we are considering the case where we only have one agent. For more than one agent we simply need to make it clear which agent knows what. For example, $\mathit{Knows}(\mathsf{Dognap}, \tau = c, s)$ indicates that Dognap knows the value of τ.

When an agent knows the value of a term, but we do not necessarily know the value of the term, we use the notation $\mathit{Kref}(\tau, s)$ to say that the agent *knows the referent* of τ:

$$\mathit{Kref}(\tau, s) \triangleq \exists z\ \mathit{Knows}(\tau = z, s). \tag{4}$$

We now introduce some special notation for the case when τ takes on values in $\mathbb{B}$. In particular, since there are only two possibilities for the referent, we say we *know whether* τ is true or not:

$$\mathit{Kwhether}(\tau, s) \triangleq \mathit{Knows}(\tau = 1, s) \vee \mathit{Knows}(\tau = 0, s). \tag{5}$$

3.2 Sensing

As in [17], we shall make the simplifying assumption that for each term τ, whose value we are interested in sensing, we have a corresponding knowledge producing action sense_τ. In general, if there are n knowledge producing actions: sense_{τ_i}, $i = 0, \ldots, n-1$, then we shall assume there are n associated situation dependent terms: $\tau_0, \ldots, \tau_{n-1}$. The corresponding successor state axiom for K is then:

$$\begin{aligned}\mathit{Poss}(a, s) \Rightarrow [\mathsf{K}(s'', \mathit{do}(a, s)) \Leftrightarrow \\ \exists s'\ (\mathsf{K}(s', s) \wedge (s'' = \mathit{do}(a, s'))) \wedge \\ ((a \neq \mathit{sense}_{\tau_0} \wedge \cdots \wedge a \neq \mathit{sense}_{\tau_{n-1}}) \\ \vee (a = \mathit{sense}_{\tau_0} \wedge \tau_0(s') = \tau_0(s)) \\ \vdots \\ \vee (a = \mathit{sense}_{\tau_{n-1}} \wedge \tau_{n-1}(s') = \tau_{n-1}(s)))].\end{aligned} \tag{6}$$

The above successor state axiom captures the required notion of sensing and solves the frame problem for knowledge producing actions. We shall explain how it works through a simple example. In particular, let us consider the problem of sensing the current temperature. Firstly, we introduce a fluent $\mathsf{temp} : \textsc{situation} \rightarrow \mathbb{R}^+$, that corresponds to the temperature (in Kelvin) in the current situation. For now let us assume that the temperature remains constant:

$$\mathit{Poss}(a, s) \Rightarrow \mathsf{temp}(\mathit{do}(a, s)) = \mathsf{temp}(s). \tag{7}$$

We will have a single knowledge producing action *senseTemp*. This gives us the following successor-state axiom for K:

$$\mathit{Poss}(a,s) \Rightarrow [\mathsf{K}(s'', \mathit{do}(a,s)) \Leftrightarrow \exists s' \, (\mathsf{K}(s',s) \wedge (s'' = \mathit{do}(a,s'))) \wedge \\ ((a \neq \mathit{senseTemp}) \vee (a = \mathit{senseTemp} \wedge \mathsf{temp}(s') = \mathsf{temp}(s)))]. \quad (8)$$

The above axiom states that for any action other than *senseTemp* the set of K-related worlds is the set of images of the previous set of K-related worlds. That is, if s' was K-related to s, then the image $s'' = \mathit{do}(a,s')$, of s' after performing the action a is K-related to $\mathit{do}(a,s)$. Moreover, when the agent performs a *senseTemp* action, in some situation s, the effect is to restrict the set of K-related worlds to those in which the temperature agrees with the temperature in the situation s. In other words, *senseTemp* is the only knowledge producing action, and its effect is to make the temperature denotation known: $\mathit{Kref}(\mathsf{temp}, \mathit{do}(\mathit{senseTemp}, s))$. The reader is referred to [17] for any additional details, examples or theorems on any of the above.

3.3 Discussion

The formalization of knowledge within the situation calculus using the epistemic fluent K makes for an elegant mathematical specification language. It is also powerful. For example, suppose we have an effect axiom that states that if a gun is loaded then the agent is dead after shooting the gun:

$$\mathsf{Loaded}(s) \Rightarrow \mathsf{Dead}(\mathit{do}(\mathit{shoot}, s)).$$

Furthermore, suppose we know the gun is initially loaded $\mathit{Knows}(\mathsf{Loaded}, \mathrm{s}_0)$, then we can infer that we know the agent is dead after shooting the gun $\mathit{Knows}(\mathsf{Dead}(\mathit{do}(\mathit{shoot}, \mathrm{s}_0))$.

Unfortunately, there are some problems. One set of problems is associated with implementation, the second applies to reasoning about real numbers, both in theory and in practice.

3.3.1 Implementation

The implementation problems revolve around how to specify the initial situation. For example, if we choose an implementation language like Prolog, specifying the initial situation may involve having to list out an exponential number of possible worlds. For example, if we do not initially know if the gun is loaded then we might consider explicitly listing the two possible worlds `s_a`, and `s_b`, such that:

```
k(s_a,s0).
k(s_b,s0).
loaded(s_a).
```

As we add more relational fluents, that we want to be able to refer to our knowledge of, the situation gets worse. In general, if we have n such fluents, there will be 2^n initial possible worlds that we have to list out. Once we start using functional fluents, however, things get even worse: we cannot, by definition, list out the uncountably many possible worlds associated with not knowing the value of a fluent that takes on values in $\mathbb{R}$.

Intuitively, we need to be able to specify rules that characterize, without having to list them all out, the set of initial possible worlds. It may be possible to somehow coerce Prolog into such an achievement. Perhaps, more reasonably, we could consider using a modal, or full first-order logic theorem prover. However, such theorem provers are inefficient and experimental. In addition, in the introduction we advanced the idea that our approach can be used for rapid prototyping. This claim relies on the possibility of gradually removing the non-determinism from our specifications. In this way we might hope to eventually refine a specification so that it can be run without the need for an underlying theorem prover. This idea must, sadly, be forsaken if we are to ingrain the need for a theorem prover into our approach to sensing.

Ignoring all the above concerns let us assume that we can specify rules that characterize the set of initial possible worlds. For example, suppose that initially we know the temperature is between 10 and 50 Kelvin. We might express this using inequalities:

$$\forall s' \, \mathsf{K}(s', \mathrm{s}_0) \Rightarrow 10 \leqslant \mathsf{temp}(s') \leqslant 50.$$

This, however, brings us to our second set of problems related to reasoning about real numbers.

3.3.2 Real numbers

We just wrote down the formula that corresponds to:

$$\mathit{Knows}(10 \leqslant \mathsf{temp} \leqslant 50, \mathrm{s}_0). \tag{9}$$

Suppose, we are now interested in what this tells us about what we know about the value of the temperature squared. In general, if we know a term τ lies in the range $[u, v]$ we would like to be able to answer questions about what we know about some arbitrary function f of τ. Such questions take us into a mathematical minefield of reasoning about inequalities. Fortunately, a path through this minefield has already been charted by the field of interval arithmetic.

4 Interval arithmetic

To address the issues we raised in section 3.3 we turn our attention to interval arithmetic [13, 14, 18]. Some of the immediate advantages interval arithmetic affords us are listed below:

- Interval arithmetic enables us to move all the details of reasoning about inequalities into the rules for combining intervals under various mathematical operations.
- Interval arithmetic provides a finite (and succinct) way to represent uncertainty about a large, possibly uncountable, set of alternatives. Moreover, the representation remains finite after performing a series of operations of the intervals. In [15] interval arithmetic is compared to probability as a means of representing uncertainty.
- Writing a sound oracle for answering ground queries about interval arithmetic is a trivial task. Moreover, we can answer queries in time that is linear in the length of the query. Returning valid and optimal intervals is more challenging (see section 7). This should, however, be compared to the vastly unrealistic assumption we (and others) made earlier about the existence of oracles for answering queries about the real numbers.
- There is no discrepancy between the underlying theory of interval arithmetic, and the corresponding implementation. Thus we re-establish our claims about using our approach for rapid prototyping.

We construct interval arithmetics from our previously available number systems as follows:

- For each number system $\mathbb{X}$, we add a new number system sort $\mathcal{I}_{\mathbb{X}}$. The constants of $\mathcal{I}_{\mathbb{X}}$ are the set of pairs $\langle u, v\rangle$ such that $u, v \in \mathbb{X}$ and $u \leqslant v$. There are functions and predicates corresponding to all the functions and predicates of $\mathbb{X}$.
- For an interval $\boldsymbol{x} = \langle u, v\rangle$, we use the notation $\underline{\boldsymbol{x}} = u$ for the lower bound, and $\overline{\boldsymbol{x}} = v$ for the upper bound.
- The function width, returns the width of an interval $\boldsymbol{x}$, i.e. $\mathsf{width}(\boldsymbol{x}) = \overline{\boldsymbol{x}} - \underline{\boldsymbol{x}}$.
- When we have a number x and an interval $\boldsymbol{x} = \langle u, v\rangle$, such that $u \leqslant x \leqslant v$ we say that $\boldsymbol{x}$ contains x, we write $x \in \boldsymbol{x}$. Similarly for two intervals $\boldsymbol{x}$, $\boldsymbol{y}$ such that $\underline{\boldsymbol{y}} \leqslant \underline{\boldsymbol{x}}$ and $\overline{\boldsymbol{x}} \leqslant \overline{\boldsymbol{y}}$, we say that $\boldsymbol{y}$ contains $\boldsymbol{x}$, we write $\boldsymbol{x} \subseteq \boldsymbol{y}$.
- For two intervals $\boldsymbol{x}_0$, $\boldsymbol{x}_1$ we say that $\boldsymbol{x}_0 \leq \boldsymbol{x}_1$ if and only if $\overline{\boldsymbol{x}}_0 \leqslant \underline{\boldsymbol{x}}_1$.
- When we do not want to specify which particular number system we are using we will use $\bot$ and $\top$ to represent, respectively, the minimum and maximum elements. For example, in $\mathbb{R}^\star$, $\langle \bot, \top\rangle = \langle -\infty, \infty\rangle$.

As an example, consider the case of the number system $\mathcal{I}_{\mathbb{B}}$. There are three numbers in this number system: $\langle 0, 0\rangle$, $\langle 0, 1\rangle$ and $\langle 1, 1\rangle$. Note that we have $\langle 0, 0\rangle \leqslant \langle 0, 1\rangle \leqslant \langle 1, 1\rangle$, $\langle 0, 0\rangle \subset \langle 0, 1\rangle$, and $\langle 1, 1\rangle \subset \langle 0, 1\rangle$. In $\mathbb{B}$, 1 and 0 can be used to represent, respectively, "true" and "false". Similarly, $\langle 1, 1\rangle$, $\langle 0, 1\rangle$ and $\langle 0, 0\rangle$ in $\mathcal{I}_{\mathbb{B}}$ can be used to represent, respectively, "known to be true", "unknown", and "known to be false". We thus get what amounts to a *three-valued logic* which, by way of example, we develop further in section 5.

Complex numbers are also made up of a pair of (real) numbers, and operations on them are defined in terms of operations on the reals. However, it would lead to confusion, if when reading a text on complex analysis we could not comprehend complex numbers as a separate entity, distinct from pairs of real numbers. We therefore forewarn

the reader against making the same mistake for intervals. That is, although numbers in $\mathcal{I}_{\mathbb{X}}$ are made up of a pair of numbers from $\mathbb{X}$ it is important to treat them as "first-class" numbers in their own right.

Traditionally, interval arithmetic was used to address the innumerous problems with the ability of floating point arithmetic to accurately represent real arithmetic. For example, consider the real number $\sqrt{2}$. This real number cannot be represented exactly by any finite decimal. However, it can be represented by the *exact* interval $\langle 1.41, 1.42\rangle$. What this interval can be used to express is that the $\sqrt{2}$ lies somewhere between 1.41 and 1.42. That is, it expresses our uncertainty about the exact value of the $\sqrt{2}$ when expressed as a decimal. With modern computers our degree of uncertainty can be made miniscule and this is part of the appeal of interval arithmetic. Of course, the other part of interval arithmetic has to do with the arithmetic of intervals. We shall, however, delay any such discussion until section 7.

5 Interval-valued fluents

The epistemic K-fluent that we discussed previously allowed us to express an agent's uncertainty about the value of a fluent in its world. Unfortunately, in section 3.3 we saw there were implementation problems associated with trying to represent an agent's knowledge of the initial situation. Fortunately, in the previous section we saw that intervals also allow us to express uncertainty about a quantity. Moreover, they allow us to do so in a way that circumvents the problem of how to represent infinite quantities with a finite number of bits. It is, therefore, natural to ask whether we can also use intervals to replace the troublesome epistemic K-fluent.

The answer, as we shall seek to demonstrate in the remainder of this paper, is a resounding "yes". In particular, we shall introduce new epistemic fluents that will be interval-valued. They will be used to represent an agent's uncertainty about the value of certain non-epistemic fluents.

We have previously used functional fluents that take on values in any of the number systems: $\mathbb{B}$, $\mathbb{R}$, etc. There is nothing noteworthy about now allowing fluents that take on values in any of the interval numbers systems: $\mathcal{I}_{\mathbb{B}}$, $\mathcal{I}_{\mathbb{R}}$. Firstly, let us distinguish those regular fluents whose value maybe learned through a knowledge-producing action. We term such fluents *sensory fluents*. Now, for each sensory fluent f, we introduce a new corresponding interval-valued epistemic (IVE) fluent $\mathcal{I}_f$.

For example, we can introduce an IVE fluent $\mathcal{I}_{\mathsf{temp}} : \textsc{situation} \rightarrow \mathcal{I}_{\mathbb{R}^{*+}}$. We can now use the interval $\mathcal{I}_{\mathsf{temp}}(\mathrm{S}_0) = \langle 10, 50\rangle$ to state that the temperature is initially between 10 and 50 Kelvin. Similarly, we can even specify that the temperature is initially completely unknown: $\mathcal{I}_{\mathsf{temp}}(\mathrm{S}_0) = \langle 0, \infty\rangle$.

Our ultimate aim is that in an implementation we can use IVE fluents to completely replace the troublesome K-fluent. Nevertheless, within our mathematical theory, there is nothing to prevent our IVE fluents co-existing with our previous sole epistemic K-fluent. Indeed, if we define everything correctly then there are many important relationships that should hold between the two. These relationships take the form of state constraints and, as we shall show, can be used to express the notion of validity and optimality of our IVE fluents. If these state constraints are maintained as actions are performed then the IVE fluents completely subsume the troublesome K-fluent. This will turn out to be true until we consider knowledge of general terms. In which case we can maintain validity but may have to sacrifice our original notion of optimality (see section 8).

Seeking to make IVE fluent ubiquitous necessitates an alternative definition for *Knows* that does not mention the K-fluent. To this end, we introduce a new abbreviation, $\mathcal{I}_{\mathit{Knows}}$ such that for any term τ, $\mathcal{I}_{\mathit{Knows}}(\tau, s) = \langle u, v\rangle$ means that τ's *interval value* is $\langle u, v\rangle$. By "interval value" we mean the value we get by evaluating the expression according the set of rules that we shall discuss in section 8. For now, let us just consider the case when τ is some fluent f. When f is a sensory fluent then $\mathcal{I}_{\mathit{Knows}}$ is the value of the corresponding IVE fluent, otherwise it is completely unknown:

$$\mathcal{I}_{\mathit{Knows}}(f, s) = \begin{cases} \mathcal{I}_f(s) & \text{if } f \text{ is a sensory fluent,} \\ \langle \bot, \top\rangle & \text{otherwise.} \end{cases} \tag{10}$$

We now take the important step of redefining *Knows* to be the special case when $\mathcal{I}_{\mathit{Knows}}(\tau, s)$ has collapsed to a thin interval:

$$\mathit{Knows}'(\tau = c, s) \Leftrightarrow \mathcal{I}_{\mathit{Knows}}(\tau, s) = \langle c, c\rangle. \tag{11}$$

The definitions of *Kref*, and *Kwhether* are now in terms of the new definition for *Knows*′. As required, this new definition does not involve the problematic epistemic K-fluent.

We are now in a position to define what it means for an IVE fluent to be valid:

Definition 5.1 (Validity). *For every sensory fluent f, we say that the corresponding IVE fluent $\mathcal{I}_f$ is a* valid *interval if f's value in all of the* K*-related situations is contained within it:*

$$\forall s, s' \; \mathsf{K}(s', s) \Rightarrow f(s') \in \mathcal{I}_f(s).$$

Note that since we have a logic of knowledge (as opposed to belief) we have that every situation is K-related to itself: $\forall s \; \mathsf{K}(s, s)$. Thus, as an immediate consequence of definition 5.1, we have that if an IVE fluent $\mathcal{I}_f$ is valid then it contains the value of f: $\forall s \; f(s) \in \mathcal{I}_f(s)$.

The validity criterion is a state constraint that ensures the interval value of the IVE fluents is wide enough to contain all the possible values of the sensory fluents. It does not however prevent intervals from being excessively wide. For example, the interval $\langle -\infty, \infty \rangle$ is a valid interval for any IVE fluent that takes on values in $\mathcal{I}_{\mathbb{R}^\star}$. The notion of narrow intervals is captured in the definition of optimality:

Definition 5.2 (Optimality). *A valid IVE fluent $\mathcal{I}_f$ is also* optimal *if it is the smallest valid interval:*

$$\forall \, \boldsymbol{y}, s, s' \; \mathsf{K}(s', s) \Rightarrow (f(s') \in \boldsymbol{y} \Rightarrow \mathcal{I}_f(s) \subseteq \boldsymbol{y}).$$

6 Correctness

In this section we shall consider some of the consequences and applications of interval-valued fluents to formalizing sensing under various different assumptions. Our goal will be to show that we can maintain valid and optimal intervals as we perform actions. This leads to the soundness and completeness result given at the end of the section.

The first step will be to define successor state axioms for IVE fluents. This is done in much the same way as it was for regular fluents. For example, suppose we have a perfect sensor, then the following successor-state axiom states that after sensing, we "know" the temperature in the resulting situation

$$\begin{aligned} \mathsf{Poss}(a, s) \Rightarrow [\mathcal{I}_{\mathsf{temp}}(\mathsf{do}(a, s)) = \boldsymbol{y} \Leftrightarrow \\ (a = \mathsf{senseTemp} \wedge \overline{\boldsymbol{y}} = \underline{\boldsymbol{y}} = \mathsf{temp}(s)) \vee (a \neq \mathsf{senseTemp} \wedge \mathcal{I}_{\mathsf{temp}}(s) = \boldsymbol{y})]. \end{aligned} \quad (12)$$

Now let us consider the case in general. Firstly, we note that there is always an initial valid IVE fluent.

Lemma 6.1. *For any initial situation* s_0 *and sensory fluent f we have that $\mathcal{I}_f = \langle \bot, \top \rangle$ is a valid interval.*

Proof. The proof of the theorem is immediate from the fact that, by definition, $\langle \bot, \top \rangle$ bounds any possible value for f. So in particular it bounds all the values f can take in all the initial K-related situations. □

It is also reasonable to assume that there will also be an initial optimal interval. If there is no such initial optimal interval then all the correctness results in this section still hold, but the completeness results won't hold until after the first sensing action.

Lemma 6.2. *If the initial set of* K*-related situations is either completely unspecified or specified with inequalities that are as tight as possible (i.e. maximally restrictive) then we can find an initial optimal IVE fluent for each of the sensory fluents.*

Proof. **Case (i)** The initial set of K-related situations is completely unspecified. That is, we are initially completely ignorant of a sensory fluent f's value. Then, the maximal interval is also clearly optimal. That is, $\langle \bot, \top \rangle$ is the *only* interval that bounds all possible values for f in the initial K-related situations. Since it is the unique valid interval it must, by definition, be an optimal interval.

Case (ii) We have a specification such as

$$\begin{aligned} (\forall s' \; \mathsf{K}(s', \mathrm{s}_0) \Rightarrow u \leqslant f(s') \leqslant v) \wedge \\ \neg \exists \, u', v' \; [u < u' \wedge v' < v \wedge (\forall s' \; \mathsf{K}(s', \mathrm{s}_0) \Rightarrow u' \leqslant f(s') \leqslant v')] \end{aligned}$$

Then, consider $\mathcal{I}_f(\mathrm{s}_0) = \langle u, v \rangle$. As required, this is clearly the smallest valid interval.

□

In what follows we make the three following assumptions about all sensory fluents f:

1. The value of $\mathcal{I}_f$, in the initial situation, is optimal and valid. This assumption is justified by lemma 6.1 and 6.2.
2. The successor-state axiom for f is such that f remains constant:

$$\mathit{Poss}(a, s) \Rightarrow [f(\mathit{do}(a, s)) = f(s)]. \tag{13}$$

3. The successor-state axioms for each of the corresponding IVE fluents $\mathcal{I}_f$ are of the form:

$$\mathit{Poss}(a, s) \Rightarrow [\mathcal{I}_f(\mathit{do}(a, s)) = \boldsymbol{y} \Leftrightarrow (a = \mathit{sense}_f \wedge \overline{\boldsymbol{y}} = \underline{\boldsymbol{y}} = \mathsf{f}(s)) \vee (a \neq \mathit{sense}_f \wedge \mathcal{I}_f(s) = \boldsymbol{y})]. \tag{14}$$

In later sections, we will discuss how to relax some of these constraints. For now let us state our main correctness result.

Theorem 6.1. *With the above assumptions, for all situations s, and sensory fluents f, every IVE fluent $\mathcal{I}_f$ is valid and optimal.*

Proof. We shall prove the result by induction on s. We note that the base case follows by assumption 1. Therefore, we need only consider the case when $s^\star = \mathit{do}(a, s)$.

By induction we may assume that

$$\forall s' \; \mathsf{K}(s', s) \Rightarrow f(s') \in \mathcal{I}_f(s),$$

and that $\mathcal{I}_f(s)$ is optimal. We seek to prove that

$$\forall s'' \; \mathsf{K}(s'', s^\star) \Rightarrow f(s'') \in \mathcal{I}_f(s^\star), \tag{15}$$

and that $\mathcal{I}_f(s^\star)$ is optimal.

Case (i) Consider the case when $a \neq \mathit{sense}_f$. Let us fix a s'' such that $\mathsf{K}(s'', s^\star)$. Note that since K is reflexive we can be sure that such a s'' exists. Therefore, by the successor state axiom for K (equation 6) there is an s' such that

$$s'' = \mathit{do}(a, s') \wedge \mathsf{K}(s', s).$$

By induction we can thus infer that

$$f(s') \in \mathcal{I}_f(s).$$

Now by the successor state axiom for f (equation 13) we have that $f(s'') = f(s')$, which gives us that

$$f(s'') \in \mathcal{I}_f(s).$$

Then by the successor state axiom for $\mathcal{I}_f$ (equation 14) $\mathcal{I}_f(s^\star) = \mathcal{I}_f(s)$, we have that

$$f(s'') \in \mathcal{I}_f(s^\star),$$

as required for validity. This also shows that to be a valid interval for $\mathcal{I}_f(s^\star)$ the interval must also be a valid interval for $\mathcal{I}_f(s)$. Now by the assumption of optimality any interval narrower than $\mathcal{I}_f(s)$ would no longer be valid. Therefore, $\mathcal{I}_f(s)$ is also the narrowest valid interval for $\mathcal{I}_f(s^\star)$.

Case (ii) Similarly, when $a = \mathit{sense}_f$ then by the successor state axiom for K (equation 6), there is an s' such that

$$s'' = \mathit{do}(a, s') \wedge \mathsf{K}(s', s) \wedge f(s') = f(s).$$

Therefore,

$$f(s') \in \langle f(s), f(s) \rangle.$$

Now by the successor state axiom for f (equation 13) we have that $f(s'') = f(s')$, which gives us that

$$f(s'') \in \langle f(s), f(s) \rangle.$$

Then by the successor state axiom for $\mathcal{I}_f$ (equation 14) $\mathcal{I}_f(s^\star) = \langle f(s), f(s) \rangle$, we have that

$$f(s'') \in \mathcal{I}_f(s^\star).$$

as required for validity. To show optimality consider that the width of $\langle f(s), f(s) \rangle$ is 0. Therefore, there can be no narrower interval and so the interval must also be optimal.

□

As a corollary we have that the definition of *Knows* given in equation 3 is equivalent to the one given in equation 11.

Corollary 6.1. *For any sensory fluent f we have that:*

$$\mathit{Knows}(f = c, s) \Leftrightarrow \mathit{Knows}'(f = c, s).$$

Proof. Let us assume $\mathit{Knows}(f = c, s)$. By equation 3 this is equivalent to:

$$\forall s' \; \mathsf{K}(s', s) \Rightarrow f(s') = c.$$

Now by theorem 6.1, $\mathcal{I}_f$ is valid and optimal, therefore $\mathcal{I}_f(s) = \langle c, c \rangle$, which by equation 11 is the definition of $\mathit{Knows}'(f = c, s)$.

Now let us assume $\mathit{Knows}'(f = c, s)$, then by equation 11 we have that $\mathcal{I}_f(s) = \langle c, c \rangle$. Once again by applying theorem 6.1 we must have that

$$\forall s' \; \mathsf{K}(s', s) \Rightarrow f(s') \in \langle c, c \rangle.$$

Since $\langle c, c \rangle$ has width 0 we can re-write this as:

$$\forall s' \; \mathsf{K}(s', s) \Rightarrow f(s') = c,$$

which by equation 3 is the definition of $\mathit{Knows}(f = c, s)$, as required. □

In [17] a number of correctness results are proven for *Knows*. The above equivalence means that under the current set of assumptions the correctness results carry over for *Knows'*.

7 Operators for interval arithmetic

Back in section 3.3.2 one of our original motivations for introducing intervals was the promise of being able to conveniently calculate what we know about a term from our knowledge of its subcomponents. For example, suppose in a situation s we know the value of a fluent $f(s)$, what do we know about $(f(s))^2$?

The answer to this question leads us to the large and active research area of interval arithmetic. The fundamental principle used is that interval versions of a given function should be guaranteed to bound all possible values of the non-interval version. For example, let us consider a function $\phi : \mathbb{R} \rightarrow \mathbb{R}$. The interval version of this function is $\mathcal{I}_\phi : \mathcal{I}_\mathbb{R} \rightarrow \mathcal{I}_\mathbb{R}$. The result of applying $\mathcal{I}_\phi$ to some interval $\boldsymbol{x}$ is another interval $\boldsymbol{y} = \mathcal{I}_\phi(\boldsymbol{x})$. We say that the $\boldsymbol{y}$ is a *valid* interval if for every point $x \in \boldsymbol{x}$, we have that $\phi(x) \in \boldsymbol{y}$. Note also that for any valid interval $\boldsymbol{y}$, if $\boldsymbol{y} \subseteq \boldsymbol{y'}$ then, $\boldsymbol{y'}$ is also a valid interval. If, for every interval $\boldsymbol{x}$, $\mathcal{I}_\phi(\boldsymbol{x})$ gives a valid interval then we say that $\mathcal{I}_\phi$ is a *sound interval version* of ϕ.

As we might expect from our previous discussions defining a sound interval version of any function is trivial. In particular, we just let the interval version return the maximal interval of the relevant number system. For example, the function that, for any argument, returns $\langle -\infty, \infty \rangle$ is a sound interval version of any function $\phi : \mathbb{R} \rightarrow \mathbb{R}$.

Hence, we see that once again we also need to be concerned about returning intervals that are as narrow as possible. The *optimal interval version* of a function ϕ is thus defined to be the *sound interval version* that, for every argument, returns the smallest valid interval. Unfortunately, for most interesting functions, no such interval versions are known to exist. There are three basic approaches that have been found to address this shortcoming:

Special Forms Consider the expression $t + (50 - t)$. If we naïvely evaluate this expression for the interval $\langle 0, 50\rangle$ we get back the interval $\langle 0, 100\rangle$. It is clear, however, that the expression simplifies to 50 and the optimal interval is thus $\langle 50, 50\rangle$. Therefore, researchers have looked at various standard forms for expressions in an attempt to give better results when evaluating the expression using intervals. In general, however, not only is there no known optimal form but there is also no known single form that is always guaranteed to give the best result. The closest researchers have been able to do so far is the so called "centered forms" [1].

Subdivision The standard tool in the interval arithmetic arsenal is subdivision. Suppose we have an interval $\boldsymbol{x}$ and we evaluate $\mathcal{I}_\phi(\boldsymbol{x})$ to give us an interval that is too wide. Then we subdivide $\boldsymbol{x}$ into $\boldsymbol{x}_l$ and $\boldsymbol{x}_r$ such that $\boldsymbol{x} = \boldsymbol{x}_l \cup \boldsymbol{x}_r$. We then evaluate each half separately in the hope that $\mathcal{I}_\phi(\boldsymbol{x}_l) \cup \mathcal{I}_\phi(\boldsymbol{x}_r) \subset \mathcal{I}_\phi(\boldsymbol{x})$. In practice this usually works well although in theory the functions can be noncomputable in which case any hopes of refining our intervals vanish.

Linear intervals The final approach we mention is a new approach that was recently invented by Jeffrey Tupper [18]. The idea is that instead of using constants to bound an interval we use linear functions. Thus for linear expressions, such as $t + (50 - t)$, we can define operators that are guaranteed to return optimal intervals. Of course, we can then recreate similar problems by considering quadratic expressions but Tupper also shows how we can generalize interval arithmetic all the way up to intervals that use general Turing machines as bounds!

8 Knowledge of terms

Back in section 5 we introduced the abbreviation $\mathcal{I}_{\mathit{Knows}}$. In equation 10 we defined $\mathcal{I}_{\mathit{Knows}}$ for fluents and in what follows we shall show how to define $\mathcal{I}_{\mathit{Knows}}$ for general terms. We begin by stating what it means for our definitions to be valid.

Definition 8.1 (Validity for terms). *For every term τ, we say that the corresponding interval value of the term given by $\mathcal{I}_{\mathit{Knows}}(\tau, s)$ is a* valid *interval if τ's value in all of the* K*-related situations is contained within it:*

$$\forall s, s' \; \mathsf{K}(s', s) \Rightarrow \tau[s'] \in \mathcal{I}_{\mathit{Knows}}(\tau, s).$$

Fortunately, the general notion of soundness for interval arithmetic carries over into our notion of validity for a $\mathcal{I}_{\mathit{Knows}}$.

Theorem 8.1. *Suppose $\mathcal{I}_\phi$ is a sound interval version of an n-ary function $\phi : \mathbb{X}^n \to \mathbb{X}$. Furthermore, let $\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{n-1} \in \mathcal{I}_{\mathbb{X}}$ be, respectively, valid intervals for $\mathcal{I}_{\mathit{Knows}}(\tau_0, s), \ldots, \mathcal{I}_{\mathit{Knows}}(\tau_{n-1}, s)$. Then, $\mathcal{I}_\phi(\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{n-1})$ is a valid interval for $\mathcal{I}_{\mathit{Knows}}(\phi(\tau_0, \ldots, \tau_{n-1}), s)$.*

Proof. Suppose the theorem is false. Then $\mathcal{I}_\phi(\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{n-1})$ is not a valid interval for $\mathcal{I}_{\mathit{Knows}}(\phi(\tau_0, \ldots, \tau_{n-1}), s)$. That is,

$$\forall s' \; \mathsf{K}(s', s) \Rightarrow \phi(\tau_0[s'], \ldots, \tau_{n-1}[s']) \in \mathcal{I}_\phi(\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{n-1}),$$

is false. That is, for some K-related situation s' we have that

$$\phi(\tau_0[s'], \ldots, \tau_{n-1}[s']) \notin \mathcal{I}_\phi(\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{n-1}).$$

This, however, violates the assumption that $\mathcal{I}_\phi$ is a sound interval version of ϕ. □

The important consequence of this theorem is that our definition of $\mathcal{I}_{\mathit{Knows}}$ for terms can stand upon the shoulders of previous work in interval arithmetic. That is, we can define $\mathcal{I}_{\mathit{Knows}}$ recursively in terms of sound interval versions of functions. Assuming the same assumptions as in theorem 8.1 we have that

$$\mathcal{I}_{\mathit{Knows}}(\phi(\tau_0, \ldots, \tau_{n-1}), s) = \mathcal{I}_\phi(\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{n-1}).$$

Note that some of our functions may be written using *infix* notation, in which case we may refer to them as *operators*. The important aspect of this definition is that we do not have to redesign a plethora of operators for interval arithmetic and prove each of them sound. In the previous section we noted the difficulties associated with defining optimal versions of operators. We also noted that there are a number of ways to deal with the problem. Each of the methods we outlines maintains validity and is thus appropriate for us to use. Which particular method we choose to narrow our intervals can be thought of as an implementation issue for our approach.

9 Usefulness

For a long list of useful operators for interval arithmetic the reader could do no better than to consult [18]. By way of example, however, we shall list some useful operators for $\mathcal{I}_{\mathbb{B}}$. Interval versions of operators, and relations, are given in bold. Elsewhere, we rely on context to imply the intended meaning.

Definition 9.1 (Operators for $\mathcal{I}_{\mathbb{B}}$).

$$\begin{aligned}
\tau = \langle u, v\rangle &\Leftrightarrow \neg\tau = \langle \neg v, \neg u\rangle \\
\tau_0 = \langle u_0, v_0\rangle \wedge \tau_1 = \langle u_1, v_1\rangle &\Rightarrow \tau_0 \wedge \tau_1 \subseteq \langle u_0 \wedge u_1, v_0 \wedge v_1\rangle \\
\tau_0 \wedge \tau_1 = \langle u, v\rangle &\Rightarrow \tau_0 \subseteq \langle u, 1\rangle \\
\tau_0 = \langle u_0, v_0\rangle \wedge \tau_1 = \langle u_1, v_1\rangle &\Rightarrow \tau_0 \vee \tau_1 \subseteq \langle u_0 \vee u_1, v_0 \vee v_1\rangle \\
\tau_0 \vee \tau_1 = \langle u, v\rangle &\Rightarrow \tau_0 \subseteq \langle 0, v\rangle \\
[\exists x\, \tau(x)] = \langle u, v\rangle &\Rightarrow \tau(c) \subseteq \langle 0, v\rangle, \quad \textit{for any constant } c \\
\textit{For some constant } c, \quad \tau(c) = \langle u, v\rangle &\Rightarrow [\exists x\, \tau(x)] \subseteq \langle u, 1\rangle \\
[\forall x\, \tau(x)] = \langle u, v\rangle &\Rightarrow \tau(c) \subseteq \langle u, 1\rangle, \quad \textit{for any constant } c \\
\textit{For some constant } c, \quad \tau(c) = \langle u, v\rangle &\Rightarrow [\forall x\, \tau(x)] \subseteq \langle 0, v\rangle
\end{aligned}$$

These definitions enable us to evaluate $\mathcal{I}_{\mathit{Knows}}$ for terms taking on values in $\mathbb{B}$. Notice however that most of the definitions are in terms of $\subseteq$. This is because we can, in general, only guarantee valid results, not optimal ones. For example, if we assume $\tau = \langle 0, 1\rangle$ then we get that $\tau \vee \neg\tau \subseteq \langle 0, 1\rangle$. While this is valid, it is clearly not optimal. Since there are only two numbers in $\mathbb{B}$ we can subdivide to perform an exhaustive search for the optimal value. That is, let $\tau = \tau_0 \cup \tau_1$, where $\tau_0 = \langle 0, 0\rangle$, and $\tau_1 = \langle 1, 1\rangle$. Now we get that $\tau_0 \vee \neg\tau_0 = \langle 1, 1\rangle$, and $\tau_1 \vee \neg\tau_1 = \langle 1, 1\rangle$. With more variables the exhaustive search approach has worst case exponential complexity. In general it may be observed that if each variable occurs only once in an expression then evaluating it will yield an optimal result. Also if we start with thin intervals then we will also get an optimal result. Finally, for a propositional formula in Blake canonical form [3] evaluation with intervals *always* yields an optimal result [7]. Moreover, all propositional formulas can be converted to this form. Thus we can evaluate propositional formulas in linear time and get optimal results. The catch is that converting propositional formulas to Blake canonical form is NP-hard.

When we consider quantifiers the above rules would not form the basis of a particularly useful procedure for evaluating expressions. We recall that the simplest correct procedure would be the one that always just returns the interval $\langle 0, 1\rangle$. For queries containing quantifiers the procedure that follows from the above rules is almost as useless, except that it works adequately when tell it about a specific instance. It is important to bear in mind however that, in general, we are dealing with problems that are not even computable. Consequently for *any* finite set of rules there will always be problems for which we can do no better than return the maximal interval. One immediate consequence of this is that designers of interval arithmetic packages never need worry about unemployment! Regardless, the rules given above certainly suffice as a simple starting point.

Therefore, as we should expect, intervals do not provide us with a means to magically circumvent complexity problems. What they do provide, however, is the ability to track our progress in solving a problem. For the majority of real world problems, where exact knowledge is not imperative, this will often allow us to stop early once we have a "narrow enough" interval. At the very least we can give up early if convergence is too slow. This should be contrasted to other methods of evaluating expressions where we can never be sure whether the method is completely stuck, or is just about to return the solution.

Let us now consider some more examples in which our interval arithmetic approach can be shown to be useful and valid. We begin with a simple example. Suppose we have two relational fluents P, and Q, and that we know P is true or we know Q is true:

$$\mathit{Knows}(P, s) \vee \mathit{Knows}(Q, s).$$

Using the K-fluent it is not hard to see that this implies that we know P or Q:

$$\mathit{Knows}(P \vee Q, s).$$

Proof. The proof involves expanding out the definition of *Knows*:

$$\mathit{Knows}(P, s) \vee \mathit{Knows}(Q, s) \triangleq (\forall s' \; \mathsf{K}(s', s) \Rightarrow P(s')) \vee (\forall s'' \; \mathsf{K}(s'', s) \Rightarrow Q(s'')),$$

and then proceeding by case analysis. First consider the case when:

$$\forall s' \; \mathsf{K}(s', s) \Rightarrow P(s').$$

Then we can weaken the postcondition to give:

$$\forall s' \; \mathsf{K}(s', s) \Rightarrow P(s') \vee Q(s').$$

The other case is symmetrical, and the result follows from the definition of *Knows* given in equation 3. □

It is also not hard to see that the implication does *not* hold the other way around. As a counter example, consider the case when we have exactly two K-related situations: s_a and s_b, such that: $P(s_a)$, $\neg Q(s_a)$, $\neg P(s_b)$ and $Q(s_b)$.

Now consider the same example using interval-fluents. Once again we can easily prove that:

$$\mathit{Knows}'(P, s) \vee \mathit{Knows}'(Q, s) \Rightarrow \mathit{Knows}'(P \vee Q, s).$$

Proof. We begin by expanding out definitions:

$$\mathcal{I}_{\mathit{Knows}}(P, s) = \langle 1, 1\rangle \vee \mathcal{I}_{\mathit{Knows}}(Q, s) = \langle 1, 1\rangle,$$

and proceed by case analysis. When:

$$\mathcal{I}_{\mathit{Knows}}(P, s) = \langle 1, 1\rangle,$$

from definitions 9.1 we have that:

$$\mathcal{I}_{\mathit{Knows}}(P \vee Q, s) = \langle 1, 1\rangle.$$

The other case is symmetrical, and the result follows from the definition of *Knows'* given in equation 11. □

Conversely, if we start from the assumption:

$$\mathit{Knows}'(P \vee Q, s) \triangleq \mathcal{I}_{\mathit{Knows}}(P \vee Q, s) = \langle 1, 1\rangle.$$

Then, all the definitions 9.1 allow us to conclude is tautologies, namely that $\mathcal{I}_{\mathit{Knows}}(P, s) \subseteq \langle 0, 1\rangle$ and $\mathcal{I}_{\mathit{Knows}}(Q, s) \subseteq \langle 0, 1\rangle$. That is we can say nothing about our knowledge of P or our knowledge of Q. So, as we should hope, the implication does *not* hold the other way around.

Let us now consider some more examples. Consider knowing P to be false: $\mathit{Knows}(\neg P, s)$ versus not knowing P: $\neg\mathit{Knows}(P, s)$. Firstly, if we assume that K is reflexive, then we have that:

$$\mathit{Knows}(\neg P, s) \Rightarrow \neg\mathit{Knows}(P, s)$$

Proof. The proof is straightforward: We don't know P if in at least one of the K-related worlds P is false. So, if P is false in all the K-related worlds the result follows. We just have to be careful that there are any K-related worlds at all. This can be inferred from the fact that K is reflexive, so $\mathsf{K}(s, s)$. □

The implication clearly does not hold in the other direction.
Likewise, we have that:

$$\mathit{Knows}'(\neg P, s) \Rightarrow \neg\mathit{Knows}'(P, s)$$

Proof.

$$\begin{array}{rl} & \mathit{Knows}'(\neg P, s) \\ \triangleq & \mathcal{I}_{\mathit{Knows}}(\neg P, s) = \langle 1,1\rangle \\ \Rightarrow & \mathcal{I}_{\mathit{Knows}}(P, s) = \langle 0,0\rangle \\ \Rightarrow & \mathcal{I}_{\mathit{Knows}}(P, s) \neq \langle 1,1\rangle \\ \Rightarrow & \neg\mathcal{I}_{\mathit{Knows}}(P, s) = \langle 1,1\rangle \end{array}$$

□

And conversely

$$\begin{array}{rl} & \neg\mathit{Knows}'(P, s) \\ \triangleq & \neg\mathcal{I}_{\mathit{Knows}}(P, s) = \langle 1,1\rangle \\ \Rightarrow & \mathcal{I}_{\mathit{Knows}}(P, s) = \langle 0,0\rangle \vee \mathcal{I}_{\mathit{Knows}}(P, s) = \langle 0,1\rangle \end{array}$$

case (i)

$$\begin{array}{rl} & \mathcal{I}_{\mathit{Knows}}(P, s) = \langle 0,0\rangle \\ \Rightarrow & \mathcal{I}_{\mathit{Knows}}(\neg P, s) = \langle 1,1\rangle \end{array}$$

but for case (ii)

$$\begin{array}{rl} & \mathcal{I}_{\mathit{Knows}}(P, s) = \langle 0,1\rangle \\ \Rightarrow & \mathcal{I}_{\mathit{Knows}}(\neg P, s) = \langle 0,1\rangle \end{array}$$

so as we should hope the implication does not hold the other way around.

Now, consider the example of $\exists x\ \mathit{Knows}(P(x), s)$, versus $\mathit{Knows}(\exists x\ P(x), s)$.
Firstly we have that

$$\exists x\ \mathit{Knows}(P(x), s) \Rightarrow \mathit{Knows}(\exists x\ P(x), s)$$

Proof. $\mathit{Knows}(\exists x\ P(x), s)$ holds if in each K-related situation s' there is a constant $c_{s'}$ such that $P(c_{s'}, s')$ holds. Note, the constant $c_{s'}$ that makes $P(x, s)$ true can be a different constant in each s'. Our assumption, however, is that there is some constant c such that $P(c, s')$ holds in every K-related situation s'. Therefore, in each K-related situation s', we can simply set $c = c_{s'}$, and the result follows. □

The implication clearly does not hold in the other direction.
Now consider the same example using intervals. We also have that:

$$\exists x\ \mathit{Knows}'(P(x), s) \Rightarrow \mathit{Knows}'(\exists x\ P(x), s)$$

Proof.

$$\begin{array}{rl} & \exists x\ \mathit{Knows}'(P(x), s) \\ \triangleq & \exists x\, \mathcal{I}_{\mathit{Knows}}(P(x), s) = \langle 1,1\rangle \end{array}$$

Then, for some constant c, we have that

$$\begin{array}{rl} & \mathcal{I}_{\mathit{Knows}}(P(c), s) = \langle 1,1\rangle \\ \Rightarrow & \mathcal{I}_{\mathit{Knows}}(\exists x\ P(x), s) \subseteq \langle 1,1\rangle \\ \Rightarrow & \mathcal{I}_{\mathit{Knows}}(\exists x\ P(x), s) = \langle 1,1\rangle \end{array}$$

□

In the other direction we have that:

$$\begin{aligned} & \mathit{Knows}'(\exists x\, P(x), s) \\ \triangleq\ & \mathcal{I}_{\mathit{Knows}}(\exists x\, P(x), s) = \langle 1, 1\rangle \\ \Rightarrow\ & \mathcal{I}_{\mathit{Knows}}(P(c), s) \subseteq \langle 0, 1\rangle \end{aligned}$$

Which is a tautology, from which we can (rightly) conclude nothing.

Finally, in section 3.3 we saw that we could make deductions based on *modus ponens*. Fortunately, we can perform similar reasoning with intervals.

Theorem 9.1. *Let τ_0 and τ_1 be terms for that take on values in $\mathbb{B}$, such that $\langle u, v\rangle$ is a valid interval value for $\mathcal{I}_{\mathit{Knows}}(\tau_0, s)$, and $\tau_0[s] \Rightarrow \tau_1[s]$. Then, $\mathcal{I}_{\mathit{Knows}}(\tau_1, s) \subseteq \langle u, 1\rangle$.*

Proof. Since $\langle u, v\rangle$ is a valid value for $\mathcal{I}_{\mathit{Knows}}(\tau_0, s)$, by definition 8.1, we have that

$$\mathcal{I}_{\mathit{Knows}}(\tau_0, s) = \langle u, v\rangle \triangleq \forall s'\ \mathsf{K}(s', s) \Rightarrow \tau_0[s'] \in \langle u, v\rangle.$$

In particular, $u\tau_0[s']$. Also, by the assumption that $\tau_0[s] \Rightarrow \tau_1[s]$ we have that $\tau_0[s'] \leqslant \tau_1[s']$. Hence, $u \leqslant \tau_0[s']\tau_1[s'] \leqslant 1$, to give us that

$$\forall s'\ \mathsf{K}(s', s) \Rightarrow \tau_1[s'] \in \langle u, 1\rangle.$$

Therefore, by definition 8.1, $\langle u, 1\rangle$ is a valid interval for $\mathcal{I}_{\mathit{Knows}}(\tau_1, s)$, as required. □

10 Inaccurate Sensors

In [2], the K-fluent approach is extended to handle noisy sensors. It is worth noting that by redefining *Knows* we can also easily extend our approach to allow for inaccurate sensors. We may say that we know a fluent's value to within some Δ, if the width of the interval is less than twice Δ:

$$\mathit{Knows}(\Delta, f = z, s) \triangleq \mathcal{I}_f(s) \subseteq \langle z - \Delta, z + \Delta\rangle. \tag{16}$$

If we have a bound of $\pm\Delta$ on the greatest possible error for the sensor that recorded yesterday's temperature then we can state that the value sensed for the temperature is within $\pm\Delta$ of the actual value:

$$\begin{aligned} \mathit{Poss}(a, s) \Rightarrow [\mathcal{I}_{\mathsf{temp}}(\mathit{do}(a, s)) = \langle u, v\rangle \Leftrightarrow \\ (a = \mathit{senseTemp} \wedge u = \max(\underline{\mathcal{I}_{\mathsf{temp}}(s)}, \mathsf{temp}(s) - \Delta) \wedge v = \min(\mathsf{temp}(s) + \Delta, \overline{\mathcal{I}_{\mathsf{temp}}(s)})) \vee \\ (a \neq \mathit{senseTemp} \wedge \mathcal{I}_{\mathsf{temp}}(s) = \langle u, v\rangle)]. \end{aligned} \tag{17}$$

11 Sensing Changing Values

Until now, we only considered sensing fluents whose value remains constant. In [17] once a fluent becomes known then it stays known. That is, if the value of a known fluent changes then the agent will automatically know the fluents new value. In many cases this is somewhat counterintuitive. For example, if one has checked the temperature once then it is quite natural to assume that after a certain period of time the information may be out of date. That is, we would expect to have to sense the temperature periodically.

Using the epistemic K-fluent to model information becoming out of date corresponds to adding possible worlds back in. Unfortunately, the K-fluent keeps track of an agent's knowledge of all the sensory fluents all at once. It can therefore be hard to specify exactly which worlds the agent should be adding back into its consideration. In contrast, with intervals there is nothing noteworthy about allowing the particular relevant interval to expand. We must simply ensure that our axioms maintain the state constraint that the interval bounds the actual value of the fluent.

At the extreme we can extend our approach to handle fluents that are constantly changing in unpredictable ways. We can model this with exogenous actions. We assume that the current temperature changes in a completely erratic and unpredictable way, according to some exogenous action *setTemp*. Then, we can write a successor-state axiom for temp that simply states that the temperature is whatever it was set to:

$$\begin{aligned} \mathit{Poss}(a,s) \Rightarrow \mathsf{temp}(\mathit{do}(a,s)) = z \Leftrightarrow \\ [(a = \mathit{setTemp}(z)) \vee (a \neq \mathit{setTemp} \wedge \mathsf{temp}(s) = z)]. \end{aligned}$$

We can, also, write a successor state axiom for $\mathcal{I}_{\mathsf{temp}}$. In particular, if we again assume accurate sensors, we can state that the temperature is known after sensing it, otherwise, it is completely unknown:

$$\begin{aligned} \mathit{Poss}(a,s) \Rightarrow [\mathcal{I}_{\mathsf{temp}}(\mathit{do}(a,s)) = \langle u,v\rangle \Leftrightarrow \\ (a = \mathit{senseTemp} \wedge u = v = \mathsf{temp}(s)) \vee (a \neq \mathit{senseTemp} \wedge u = 0 \wedge v = \infty)]. \end{aligned} \quad (18)$$

Note that this definition works because, by definition, $\forall s\ \mathsf{temp}(s) \in \langle 0,\infty\rangle$. At first glance it may appear strange that we have, for example, $\mathcal{I}_{\mathsf{temp}}(\mathit{do}(\mathit{setTemp}(2),s)) = \langle 0,\infty\rangle$. Upon reflection, however, the reader will hopefully recall that our intention is to use the IVE fluents to model an agent's knowledge of its world. Therefore, until sensing, the agent rightly remains oblivious as to the effect of the exogenous action *setTemp*. For the fluent that keeps track of the temperature in the virtual world we of course get that $\mathsf{temp}(\mathit{do}(\mathit{setTemp}(2),s)) = 2$.

If we have a bound on the maximum rate of temperature change, per unit time, to be $\Delta\mathsf{temp}$, and we add the ability to track the time to our axiomatization, then we can do a lot better. Suppose we have an action *tick* that occurs once per unit of time. Moreover, we limit exogenous actions to only occurring directly before a tick action. Then we can have a successor-state axiom that states the temperature is known after sensing; or after a period of time it is known to have changed by less than some maximum amount; otherwise it is unchanged:

$$\begin{aligned} \mathit{Poss}(a,s) \Rightarrow [\mathcal{I}_{\mathsf{temp}}(\mathit{do}(a,s)) = \langle u,v\rangle \Leftrightarrow \\ (a = \mathit{senseTemp} \wedge u = v = \mathsf{temp}(s)) \vee \\ (a = \mathit{tick} \wedge \exists u_p, v_p\ \mathcal{I}_{\mathsf{temp}}(s) = \langle u_p, v_p\rangle \wedge \\ u = \mathsf{max}(0, u_p - \Delta\mathsf{temp}) \wedge v = v_p + \Delta\mathsf{temp}) \vee \\ (a \neq \mathit{senseTemp} \wedge a \neq \mathit{tick} \wedge \mathcal{I}_{\mathsf{temp}}(s) = \langle u,v\rangle]. \end{aligned} \quad (19)$$

This type of axiom can be used to "plan to replan". That is, the degradation in our knowledge level is predicatable and can be used as the basis for a replanning action.

12 Conclusion

In the introduction, we mentioned that what started us on the work we have described in this paper was a desire to build an "intelligent" autonomous agent. Happily, and thanks in large part to the work described herein, we have indeed successfully developed autonomous agents that can reason, act and perceive in changing, incompletely known, unpredictable environments. In particular, we have applied our approach to computer animation, computer games, and cinematography. Some frames from some corresponding animations can be found at `www.cs.toronto.edu/~funge`. Additional documentation on the work can be found in [5].

References

[1] G. Alefeld and Jurgen Herzberger. *Introduction to Interval Computations*. Academic Press, 1983.

[2] F. Bacchus, J.Y. Halpern, and H. Levesque. Reasoning about noisy sensors in the situation calculus. In C.S. Mellish, editor, *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI–95)*, pages 1933–1940, Montreal, August 1995.

[3] A. Blake. *Canonical Expressions in Boolean Algebra*. PhD thesis, University of Chicago, 1938. Published by University of Chicago Librarries, 1938.

[4] Herbert B. Enderton. *A Mathematical Introduction to Logic*. Academic Press, 1972.

[5] J. Funge. *Making Them Behave: Cognitive Models for Computer Animation*. PhD thesis, Department of Computer Science, University of Toronto, Toronto, Canada, 1998.

[6] M.L. Ginsberg and D.E. Smith. Reasoning about action ii: the qualification problem. *Artificial Intelligence*, 35:311–342, 1988.

[7] H. Levesque, University of Toronto. *Personal communication*, August 1997.

[8] S. Hanks and D. McDermott. Temporal reasoning and default logics. Technical report, Yale University, 1985. Computer Science Research Rept. No. 430.

[9] E.C.R. Hehner. Boolean formalism and explanations. In *International Conference on Algebraic Methods and Software Technology*, July 1996.

[10] Fangzhen Lin and Raymond Reiter. State constraints revisited. *Journal of Logic and Computation, Special Issue on Actions and Processes*, 4(5):655–678, 1994.

[11] J. McCarthy. Epistemological problems of artificial intelligence. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI–77)*, pages 1038–1044, Cambridge, MA, 1977.

[12] J. McCarthy and P. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4*, pages 463–502. Edinburgh University Press, Edinburgh, 1969.

[13] R. E. Moore. *Interval Analysis*. Prentice-Hall, 1966.

[14] R. E. Moore. *Methods and Applications of Interval Analysis*. SIAM, 1979.

[15] J. Pesonen and E. Hyvonen. Interval approach challanges monte carlo simulation. In *Scientific Computing, Computer Arithmetic and Validated Numerics*, 1995.

[16] R. Reiter. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In Vladimir Lifschitz, editor, *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honour of John McCarthy*, pages 359–380,418–420. Academic Press, 1991.

[17] R. Scherl and H. Levesque. The frame problem and knowledge-producing actions. In *Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI–93)*, Menlo Park, CA., 1993. AAAI Press.

[18] J. Tupper. *Graphing Equations with Generalized Interval Arithmetic*. MSc thesis, Department of Computer Science, University of Toronto, Toronto, Canada, January 1996.