Epistemic Semantics in Guarded String Models

Anonymous SCiL submission

Abstract

Constructive and computable multi-agent epistemic possible worlds models are defined, where possible worlds models are guarded string models in an epistemic extension of Kleene Algebra with Tests. The account is framed as a formal language Epik (Epistemic KAT) for defining such models. The language is implemented by translation into the finite state calculus, and alternatively by modeling propositions as lazy lists in Haskell. The syntax-semantics interface for a fragment of English is defined by a categorial grammar.

1 Introduction and Related Work

Linguistic semantics in the Montague tradition proceeds by assigning propositional *semantic values* to disambiguated sentences of a natural language. A proposition is a set or class of *possible worlds*. These are often assumed to be things with the same nature and complexity as the world we occupy (**?**). But alternatively, one can work with small idealized models, in order to illustrate and test ideas. The point of this paper is to scale up toy or idealized models to countable sets of worlds, and to constructive and computable modeling of epistemic alternatives for agents. We describe a certain systematic way of defining such models, and illustrate how to apply them in natural language semantics. The focus is on epistemic semantics and clausal embedding. The fundamental move is to identify possible worlds with strings of primitive events, so that propositions are sets of strings. An advantage in this is that it allows for a mathematical description of an algebra of propositions, coupled with a computational representation using either lazy lists of strings, or finite state machines that describe sets of strings.

The approach taken here synthesizes five antecedents in a certain way. John McCarthy's *Situation Calculus* is the source of the idea of constructing possible worlds as event sequences (**??**). The algebraic theory of *Kleene Algebra with Tests* characterizes algebras with elements corresponding to propositions and event types in our application (**?**). *Action models* in dynamic epistemic semantics introduce the technique of constructing epistemic models from primitive alternative relations on events, in order to capture the epistemic consequences of perceptual and communicative events (**?**). Literature on *finite state methods in linguistic semantics* has used event strings and sets of event strings to theorize about tense and aspect in natural language semantics (**???**) and to express intensions (**?**). Work on *finite state intensiononal semantics* has investigated how to do the semantics of intensional complementation in a setting where compositional semantics is expressed in a finite state calculus (**??**).

A running example of an event-sequence model is *The Concealed Coin*. Amy and Bob are seated at a table. There is a coin on the table under a cup, heads up. The coin could be heads or tails, and neither agent knows which it is. This initial situation is possible world w_1. Two additional worlds w_2 and w_3 are defined by sequencing events after the initial state, with events interpreted as in (1). The truth values for English sentences shown in (3) are observed.

(1) a_1 Amy peeks at heads, by tipping the cup. Bob sees she's peeking, but not what she sees.
b_1 Bob peeks at heads.
a_0 Amy peeks at tails.
b_0 Bob peeks at tails.

(2) $w_2 = w_1\, a_1 \quad w_3 = w_1\, a_1\, b_1$

(3)

w_1	w_2	w_3	Sentence
false	true	true	Amy knows that it's heads.
false	false	true	Bob knows thats it's heads.
false	false	true	Bob knows Amy knows it's heads.
false	true	true	Bob knows Amy knows whether it's heads or tails.

The events come with pre-conditions. Amy can peek at heads only if the coin is heads up, so a_1 has the precondition of the coin being heads up. Let h be the Boolean proposition that the coin is heads up. Then preconditions can be described by Boolean formulas, with h being the precondition of a_1 u. Events come as well with a relation between prior and following state, for instance with u incrementing the floor. This is expressed using an operator ":" (read "and next") that pairs Boolean formulas. The formula $h : h$ describes a_1 (Amy looking at heads) as happening only in an h state, and as not changing the state. Symmetrically, a_0 (Amy looking at tails) can happen only in a not-h state, and does not change the state ($\bar{h}$:$\bar{h}$).

2 Epistemic guarded string models

Suppose that in the coin example, we have an additional primitive stative proposition (or atomic test) t, interpreted as tails. A sequence such as $\bar{h}t$ can be viewed as a valuation of primitive propositions, which is used to describe world state. The primitives are listed in fixed order, and left unmarked (indicating true) or marked with the overbar (indicating false). Since a coin is heads or tails but not both, we want to allow the valuations $h\bar{t}$ and $\bar{h}t$, and disallow ht and $\bar{h}\bar{t}$. This is enforced by a *state formula*, which is a Boolean formula, in this case the one given on the second line of (4). Where B is a set of atomic tests and ϕ is a state constraint over B, $\mathcal{A}^{\phi}_{\mathsf{B}}$ is the set of valuations of B that make formula ϕ true. Valuations are called atoms, because they correspond to the atoms of a Boolean algebra of tests (**?**).

Formulas like the ones in (**??**) that describe pre- and post-conditions are *effect formulas*. They are interpreted as defining relations between atoms, as defined in Figure 1. The atoms they relate are constrained by the state formula as well. For the heads-tails example, let the state formula and the effect formula for a_1 (Amy peeking at heads) be as specified in (4). Then $\mathcal{A}^{\varphi}_{\mathsf{B}}$ and the relation on atoms for the event a_1 are as given at the bottom in (4).

$$
\begin{array}{lll}
\text{state formulas} & & (a \in \mathsf{B}) \\
\rho, \sigma, \varphi & ::= & a \mid 0 \mid 1 \mid \rho + \sigma \mid \rho\,\sigma \mid \bar{\rho} \\
\text{effect formulas} & & \\
\zeta, \eta & ::= & \rho : \sigma \mid \zeta + \eta \mid \zeta \,\&\, \eta \mid \bar{\zeta} \\
[\![\rho : \sigma]\!]^{\varphi} & = & \mathcal{A}^{\rho\varphi}_{\mathsf{B}} \times \mathcal{A}^{\sigma\varphi}_{\mathsf{B}} \\
[\![\zeta + \eta]\!]^{\varphi} & = & [\![\zeta]\!]^{\varphi} \cup [\![\eta]\!]^{\varphi} \\
[\![\zeta \,\&\, \eta]\!]^{\varphi} & = & [\![\zeta]\!]^{\varphi} \cap [\![\eta]\!]^{\varphi} \\
[\![\bar{\zeta}]\!]^{\varphi} & = & \mathcal{A}^{\varphi}_{\mathsf{B}} \times \mathcal{A}^{\varphi}_{\mathsf{B}} \setminus [\![\zeta]\!]^{\varphi}
\end{array}
$$

Figure 1: Syntax of state formulas and syntax and semantics of effect formulas. Effect formulas denote relations between atoms. In a state formula, juxtaposition $\rho\,\sigma$ is conjunction.

(4)

B	$\{h, t\}$
state formula φ	$h\bar{t} + \bar{h}t$
effect formula ζ for a_1	$h : h$
$\mathcal{A}^{\varphi}_{\mathsf{B}}$	$\{h\bar{t}, \bar{h}t\}$
$[\![\zeta]\!]^{\varphi}$	$\{\langle h\bar{t}, h\bar{t}\rangle\}$

Epik is a specification language for possible worlds models that includes declarations of events and states, state formulas, effect formulas, and additional information. Figure 2 shows an Epik program that describes a possible worlds model for two agents with information about one coin, and events of the agents semi-privately looking at the coin. The line beginning with `state` enumerates B. The line beginning with `restrict` gives the state formula. The lines beginning with `event` declare events and their effect formulas. Finally the lines beginning with `agent` define *event alternative* relations for agents. Each clause with an arrow has a single event symbol on the left, and a disjunction of alternative events on the right of the arrow. The interpretation of Amy's alternatives for b_1 (Bill peeks at heads), is that when b_1 happens, for Amy either b_1 or b_0 (Bill peeks at tails) could be happening.

This paper focuses on defining a concrete possible worlds model from an Epik specification. The models are an extension of guarded-string models for Kleene Algebra with Tests (KAT). This is an algebraic theory that has model classes including guarded string models, relational models, finite models, and matrix models. Our definitions and notation follow (**?**). We add syntax and semantics is included to cover multi-agent epistemic semantics.

Guarded strings over a finite alphabet P are like ordinary strings, but with atoms over a set B al-

```
                        agent aly
                          a1 -> a1
state h t                 a0 -> a0
restrict h!t + t!h        b1 -> b1 + b0
event a1 h:h              b0 -> b1 + b0
event a0 t:t            agent bob
event b1 h:h              b1 -> b1
event b0 t:t              b0 -> b0
                          a1 -> a1 + a0
                          a0 -> a1 + a0
```

Figure 2: Epik program describing a possible-worlds event sequence model for two agents with information about one coin, and events of the agents semi-privately looking at the coin.

ternating with the symbols from P. In the algebra described by Figure 2, P is the set of events $\{a_1, a_0, b_1, b_0\}$, and B is $\{h, t\}$.

In the coin example, as we already saw in (4), $\mathcal{A}^{\varphi}_{\mathsf{B}}$ is $\{h\bar{t}, \bar{h}t\}$, for which we use the shorthand $\{H, T\}$. A guarded string over P and B is a strings of events from P, alternating with atoms over B, and beginning and ending with atoms. (**??**) gives the encoding as guarded strings of the worlds in (**??**) and (1). The length of a guarded string p, written $|p|$ is the number of events in p. An atom such as H is a guarded string of length 0. Correspondingly $|w_1| = 0$, $|w_2| = 1$ and $|w_3| = 2$.

The discussion of (2) mentioned building worlds by incrementing worlds with events. This is accomplished in guarded string models with fusion product $\diamond$, a partial operation that combines two guarded strings, subject to the condition that the atom at the end of the the first argument is identical to the atom at the start of the second one. (5) gives some examples.

(5) $H\, b_1\, H \;\diamond\; H\, a_1\, H = H\, b_1\, H\, a_1\, H$
$H \;\diamond\; T\, a_1\, T = \textit{undefined}$

Rather than individual guarded strings, elements of a guarded string model for KAT are sets of guarded strings. In our application, these elements have the interpretation of propositions, which are sets of possible worlds. In a free guarded string model for KAT, any event can be adjacent to any atom in a guarded string that is an element of the underlying set for the algebra. We instead impose the constraints coming from the state and effect formulas. (6) defines the well-formed guarded strings determined by and Epik specification. Condition (i) says that each atom is consistent with the state constraint, and condition (ii) says that each constituent token event $\alpha_i e_i \alpha_{i+1}$ is consistent with the effect constraint on e_i.[1]

(6) Given P, B, a state formula φ, and an effect formula ζ_e for each event e in P, $\alpha_0 e_0 ... e_n \alpha_{n+1}$ is well-formed iff
(i) $\alpha_i \in \mathcal{A}^{\varphi}_{\mathsf{B}}$ $(0 \leq i \leq n)$, and
(ii) $\langle \alpha_i, \alpha_{i+1} \rangle \in [\![\zeta_{ei}]\!]^{\varphi}$, $(0 \leq i \leq n)$.

Well-formed guarded strings have the interpretation of worlds in the application to natural-language semantics. The set of possible worlds in the Kripke frame determined by an Epik specification is the set of well-formed guarded strings. At this point, we could say that any set of worlds is a proposition, so that the set of propositions is the power set of the set of worlds (**??**). We will instead define a more restrictive set of propositions corresponding to the regular sets of strings. This is deferred to the next section. Certain sets of well-formed guarded strings have the addional interpretation of event types. An event-type is something that can "happen" is different worlds. For example, a_1 has the event type $\{H\, a_1\, H\}$, and a_0 has the even type $\{T\, a_0\, T\}$. The event types for b_0 and b_1 are analogous.

The construction so far defines a set of worlds from an Epik specification. Normally the set is countably infinite, though some choices of effect formulas can result in a finite set of worlds. The next step is to define an alternative relation R_a on worlds for each agent a. This will result in a Kripke frame $\langle W, R_1, ..., R_n \rangle$ consisting of a set of worlds, and a world-alternative relation for each agent (**?**). An Epik specification defines an alternative relation on bare events for each agent a, which we notate as $\hat{R}_a$. This should be lifted to a relation R_a on worlds. The basic idea is that when a world w is incremented with an event e, in the resulting world $w \diamond e$, epistemic alternatives for agent a are of the form $w' \diamond e'$, where w' is an alternative to for a in w, and e' is and event-alternative to e for a.[2]

[1] An alternative is to define equations such as $\bar{\phi} = 0$ (from the state formula ϕ) and $a_1 = h a_1 h$ (from the effect formula $h : h$ for event a_1), and construct a quotient algebra from the equivalence relation generated by these equations. This results in equating sets of guarded strings in the free algebra that differ by guarded strings that are ill-formed according to the state and effect formulas. In the development in the text, we instead use a set of guarded strings that are well-formed according to the state and effect formulas as the representative of the equivalence class.

[2] In this it is important that the event-alternative relation for an agent is constant across worlds. We anticipate that the definition given here produces results equivalent to what is found in literature on event alternatives in dynamic epistemic

events	$e \in \mathsf{P}$	
states	σ as in Figure 1	
p, q	$::= e \mid \sigma \mid p+q \mid pq \mid p^* \mid \neg p \mid \Diamond_a p$	
$\Box_a p$	$\triangleq \neg\Diamond_a \neg p$	
$p \wedge q$	$\triangleq \neg(\neg p + \neg q)$	
$\bullet$	$\triangleq \sum_{e \in \mathsf{P}} e$	

Figure 3: The language of Epik terms and key derived operators.

This needs to be implemented in a way that takes account of pre- and post-conditions for events. For this, our approach is to refer the definition of well-formed guarded strings. (7) defines a relation on worlds from a relation on bare events.

(7) Let W be a set of guarded strings over events P and primitive tests B, and $\hat{R}$ be a relation on P. The corresponding relation R on W holds between a guarded string $\alpha_0 e_0 ... e_n \alpha_{n+1}$ in W and a guarded string q iff q is an element of W and is of the form $\alpha'_0 e'_0 ... e'_n \alpha'_{n+1}$,, where for $0 \leq n$, $\langle e_i, e'_i \rangle \in \hat{R}$.

This requires that in an alternative world, each constituent event e'_i is an alternative to the event e_i in the base world. Compatibilities between events in the alternative world are enforced by the requirement that the alternative world is an element of W, so that state and effect formulas are enforced.

Consider a scenario like the one from Figure 1, but with an additional agent Cal. The base world Tb_0Tc_0T is one where the coin is tails, and first Bob looks at tails, and them Cal looks at tails. The first event b_0 has the alternatives b_0 and b_1 for Amy, and the second event c_0 has the alternatives c_0 and c_1 for Amy. This results in four combinations b_0c_0, b_0c_1, b_1c_0, and b_1c_1. But these are filtered by post- and pre-conditions of events in the alternative world, so that the set of alernatives for Amy in Tb_0Tc_0T is $\{Tb_0Tc_0T, Hb_1Hc_1H\}$, with two world-alternatives instead of four.

3 The logical language of Epistemic KAT

The standard language for Kleene algebra with tests has the signature $\langle K, +, \cdot, *, ^-, 0, 1\rangle$ (?). In a guarded string model for KAT, K is a set of sets of guarded strings, $+$ is set union, the operation $\cdot$ is fusion product raised to sets, $*$ is Kleene star, the operation $^-$ is complement for tests, 0 the empty set, and 1 is the set of atoms.[3] To this we add a unary modal operation $\Diamond_a$ for each agent, and a unary complement operation $\neg$ on elements of K. Intuitively, $\Diamond_a p$ is the set of worlds where proposition p is epistemically possible for agent a. Propositional complement is included because natural languages have sentence negation. In addition, universal box modalities are defined as duals of existential diamond modalities.

With modalities and propositional negation added, the signature of n-agent epistemic KAT is $\langle K, +, \cdot, *, ^-, 0, 1, \neg, \Diamond_1 ... \Diamond_n\rangle$. Figure 3 defines the syntax of the language. Juxtaposition is used for product. Terms in this language are used to represent the proposositional semantic values of English sentences. (8) gives some examples. To explain the first one $\bullet$ as defined in Figure 3 is the disjunction of the primitive events. Since a world is a well-formed sequence of events, $\bullet^*$ is the set of worlds. Multiplying by the state symbol h in the term $\bullet^* h$ has the effect of conjoining h with the atom at the end of the world. So $\bullet^* h$ is the set of worlds where the coin is heads.[4].

(8)
	$\bullet^* t$	It's tails.
	$\bullet^* h$	It's heads.
	$\Box_a \bullet^* h$	Amy knows that it's heads.

$\Box_b(\Box_a \bullet^* t + \Box_a \bullet^* \neg t)$
Bob knows that Amy knows whether it's tails.

A term p of the logical language is interpreted as a set of guarded strings $[\![p]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta}$, where superscript captures dependence on an Epik specification. Figure 4 defines the interpretation. The interpretation $[\![1]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta}$ of the multiplicative identity 1 is the set of atoms that satisfy the state constraint φ. Where b is a primitive Boolean, $[\![b]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta}$ is the set of atoms that satisfy the state constraint and where b is true. Where e is a primitive event, $[\![e]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta}$ is the set of guarded strings that have the form of e flanked by compatible atoms, as determined by the event formula ζ_e. The product pq is interpreted with fusion product raised to sets of guarded strings.

semantics, though we have not verified this. That literature primarily focuses on mapping an epistemic model for a single time and situation to another, and uses general first-order models, rather than guarded string models. See **?**, **?**, and articles in **?**. Previous literature is motivated by epistemic logic and AI planning, rather computable possible worlds models in natural language semantics.

[3] 0 has the dual role the identity for $+$ (union), and as False for operations on tests. 1 has the dual role of the identity for product (fusion product raised to sets), and True for tests.

[4] A mapping from English sentences to logical terms in epistemic KAT is presented in Section 6

$$
\begin{array}{lcl}
[\![0]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta} & \triangleq & \emptyset \\
[\![1]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta} & \triangleq & \mathcal{A}^{\varphi}_{\mathsf{B}} \\
[\![b]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta} & \triangleq & \mathcal{A}^{b\varphi}_{\mathsf{B}} \\
[\![\bar{\sigma}]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta} & \triangleq & \mathcal{A}^{\varphi}_{\mathsf{B}} \setminus [\![\sigma]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta} \\
[\![e]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta} & \triangleq & \{\alpha e \beta | \alpha \zeta_e \beta\} \\
[\![p+q]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta} & \triangleq & [\![p]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta} \cup [\![q]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta} \\
[\![pq]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta} & \triangleq & \left\{ x \diamond y \,\middle|\, \begin{array}{l} x \in [\![p]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta} \\ y \in [\![q]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta} \\ x \diamond y \text{ is defined} \end{array} \right\} \\
[\![p^*]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta} & \triangleq & \bigcup_{n \geq 0} [\![p^n]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta} \\
[\![\neg p]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta} & \triangleq & [\![\bullet^*]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta} \setminus [\![p]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta} \\
[\![\Diamond_a\, p]\!] & \triangleq & \{x \mid \exists y . x \hat{R}_a y \wedge y \in [\![p]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta}\}
\end{array}
$$

Figure 4: Interpretation of Epik terms as sets of guarded strings

Kleene star is interpreted as the union of exponents (p^n is the n-times product of p with itself, with $p^0 = 1$). Propositional complement is complement relative to the set of worlds. The epistemic formula $\Diamond_a p$ is interpreted with Kripke semantics for epistemic modality, as the pre-image of the embedded proposition under the world-alternative relation R_a.

Summing up, given an Epik specification $\mathsf{B}, \mathsf{P}, \varphi, \zeta$, term p (as defined syntactically in Figure 3) is interpreted as a set of guarded strings $[\![p]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta}$. Let $K^{\mathsf{B},\mathsf{P},\varphi,\zeta}$ be the sets that are interpretations of terms. Then $\langle K^{\mathsf{B},\mathsf{P},\varphi,\zeta}, +, \cdot, *, \bar{\ }, 0, 1, \neg, \Diamond_{a_1}, ..., \Diamond_{a_n} \rangle$ is a concrete guarded string interpretation for the signature of epistemic KAT, with operations as in Figure 4 (e.g. the binary operation $+$ is union, and the unary operation $\Diamond_a$ is pre-image relative to $\hat{R}_a$). This provides a concrete n-agent Kripke frame $\langle [\![\bullet^*]\!]^{\mathsf{B},\mathsf{P},\varphi,\zeta}, \hat{R}_1, ..., \hat{R}_n \rangle$.[5] The frame consists of a set of worlds, and an epistemic-alternative relation for each agent. It is used as a target for natural-language interpretation in Section 6.

4 Translation into the finite state calculus

The finite state calculus is an algebra of regular sets of strings and regular relations between strings that was designed for use in computationonal phonology and morphographemics (**??**). Current implementations allow for the definition of functions that have the status of defined operations on regular sets and relations (**??**). Such definitions are used here

[5] The domain of the Kripke frame differs from the domain of the guarded string model, because the former is the set of worlds, while the latter is the set of propositions.

```
St
  Atomic Tests such as 0110.
UnequalStPair
  Sequence of two unequal tests such as 0110 0111.
define Wf0 ~[$ UnequalStPair];
  String that doesn't contain a non-matching test pair.
define Squash St -> 0 || St _;
  Rewrite relation deleting the second of two tests.
define Cn(X,Y)
    [[[X Y] & Wf0] .o. Squash].l;
  KAT product.
define Kpl(X)
    [[[X+] & Wf0] .o. Squash].l;
define Kst(X) St | Kpl(X);
  KAT Kleene plus and Kleene star. The Fst operation | is
  union.
```

Figure 5: Definition in Fst of KAT product and KAT Kleene star. Where `X` and `Y` are regular sets and `R` and `S` are regular relatiions, `X&Y` is the intersection of `X` and `Y`, `X|Y` is the union of `X` and `Y`, `~X` is the complement of `X`, `R.o.S` is the composition of `R` and `S`, `R.l` is the co-domain of `R`, and `$X` is the set of strings that have a substring in X.

to construct of a model for epistemic KAT inside the finite state calculus. The space of worlds is a set of ordinary strings. Bit sequences (sequences of 0's and 1's) are used for atoms, and these alternate with event symbols in the encoding of a world. (9) gives the encoding of worlds from the example. A string is a finite sequence of symbols, and `0`, `1`, `u`, and `d`, are symbols. `a0` and `b0` are multi-character symbols that are found in implementations of the finite state calculus. The multi-character symbol `a0` is an element of the alphabet that has no connection with the element `a`.

(9) Worlds coded as strings

World	*String*
w_1	`1 0`
w_2	`1 0 a1 1 0`
w_3	`1 0 a1 1 0 b1 1 0`

Terms in the finite state calculus are interpreted as sets of strings, or for relational terms, as relations between strings. Computationally, the sets and relations are represented by finite state acceptors. As used here, a program in the Fst language of the finite state calculus is a straight-line program that defines a sequence of constants naming sets, constants naming relations, and functions (defined as macros) mapping one or more regular sets or relations to a regular set or relation. The finite state calculus has a product operation of string concatenation raised to sets. Concatenation of strings with atoms (Boolean vectors) at both ends has the effect

```
define RelKpl(R)
  Squash.i .o. Wf0 .o. [R+] .o. Wf0 .o. Squash
```

$\underbrace{\texttt{Squash.i .o.}}_{c}\underbrace{\texttt{Wf0 .o.}}_{b}\underbrace{\texttt{[R+]}}_{a}\underbrace{\texttt{.o. Wf0}}_{b}\underbrace{\texttt{.o. Squash}}_{c}$

- a Relational Kleene plus in the string algebra
- b Constrain domain and co-domain to contain no unmatched tests.
- c Reduce doubled tests to a single test in the domain and co-domain.

```
define Kst(R) [St .x. St]|Kpl(X);
```

The Fst operation `.x.` is Cartesian product. `R.i` is the inverse of relation `R`.

Figure 6: Definition in Fst of the Kleene concatenation closure of a relation between guarded strings.

of doubling atoms at the juncture, and does not enforce matching of atoms at the juncture. Therefore KAT product can not be identified with product in the finite state calculus. Instead, KAT product and KAT Kleene star are defined operations, see Figure 5. The binary product operation `Cn` and the unary Kleene star operation `Kst` combine strings in the string algebra, remove strings with non-matching atoms, and then delete the second of two tests to create well-formed guarded strings. Matching of atoms is enforced with the set `Wf0`, which is the set of strings that do not contain non-matching Boolean vectors. The containment operator (expressed by the dollar sign) and complement (expressed by tilde) are operators of the finite state calculus. The set of non-matching sequences of atoms `UnequalStPair` is defined by a finite disjunction. Deletion is accomplished by a re-write rule in the finite state calculus, which is a notation for defining regular relations by contextually constrained substitutions. In this case, is `Squash` is a regular relation that deletes an atom (a sequence of 0's and 1's of a certain length) when it follows an atom.[6]

An event symbol such as a_1 (Amy looks at heads) is in the KAT algebra a set of bare events decorated with compatible tests on each side, $\{10a_110\}$ in this case. This is a unit set rather than a guarded string, because elements of the KAT algebra are sets. Worlds in the KAT algebra are defined by sequencing events using *Kst*. The operation enforces compatibility of states, so that $(a_1 + a_0)(b_1 + b_0)$ contains two worlds rather than four. The program in Figure 2 as interpreted in FST defines a countably infinite set of possible worlds by KAT Kleene closure as $\mathit{Kst}(a_1 + a_0 + b_1 + b_0)$.

It remains to define an epistemic alternative relation on worlds for each agent. The relevant information in Figure 2 is a relation between bare events for each agent. This determines a relation in the guarded string algebra a relation between bare events decorated with compatible tests. For agent Amy, this is the relation described in (10) as a set of ordered pairs.

$$(10)\left\{\begin{array}{l}\langle 10a_110, 10a_110\rangle, \langle 01a_001, 01a_001\rangle,\\ \langle 10b_110, 10b_110\rangle, \langle 10b_110, 01b_001\rangle,\\ \langle 01b_001, 10b_110\rangle, \langle 01b_001, 01b_001\rangle\end{array}\right\}$$

The relation on decorated events needs to be generalized to a relation of worlds. The principle for this is that an epistemic alternative to a world of the form we is a world of the form vd, where v is a world-alternative to w, d is an event-alternative to e, and vd is defined (i.e. the world alternative v satisfies the pre-conditions of the event alternative d). This principle is found in earlier literature (Moore 198x, Baltag, Moss and Solecki 20xx). In the construction in Fst, the definition of world alternatives takes a simple form. Where R_a is the relation on decorated events for agent a, the the corresponding relation on worlds in is the Kleene closure of R_a. Where R and S are relations, the concatenation product of R and S is the set of pairs of the form $\langle x_1x_2, y_1y_2\rangle$, where $\langle x_1, y_1\rangle$ is in relation R, and $\langle x_2, y_2\rangle$ is in relation S. The Kleene closure of relation R is $\cup_{n\geq 0}R^n$, where R^n is the n-times concatenation product of R with itself (the 0-times concatenation product is $[\![1]\!]^{\varphi,\mathcal{E}}$). This is an operation in the finite state calculus. Figure **??** defines the corresponding operation in the guarded string algebra. The epistemic alternative relation on worlds for an agent is then defined as the concatenation closure of the event alternative relation for the agent.

5 Bounded Lazy Interpretation

We also implement the semantics of Epik terms using lazy lists (**?**) in Haskell, rather than the direct interpretation as sets. Unfortunately, regular expressions, and hence also Epik, can denote infinite sets of strings, for example the term $\bullet^*$. Normally regular languages are represented using a finite coalgebra (). However the non-distributivity of $\diamond$ accross ; complicates this construction[7]. To

[6] This is a non-equal length regular relation. The finite state calculus includes such relations, and they can be used with relation composition and relation domain and co-domain. They are restricted in that the complement and set difference for non-equal length relations is not defined. The epistemic alterative relations that are defined in Figure – are equal-length relations.

[7] Axiomatic and Coalgebraic models for Epik are open questions

sidestep this, we parameterize the interpretation function on a positive integer n and only produce guarded strings of length n or less.

We translate the Epik terms into a Haskell algebraic datatype that represents terms with the same signature as described in Section 2. We then paramaterize the interpretation function on an integer n that describes the maximum length of a string we will produce.

The bounded interpretation into lists of strings is very similar to the unbounded interpretation into sets of strings, except for the bounds checking. The full details are shown in Figure **??**. First note that when $n = 0$, we simply return the empty list, denoted $[]$. Terms of the form 0, 1, e, and ψ have the same denotation as before, translated into a list (for a set X, $\lfloor X \rfloor$ is a list with the same elements as X arbitrarily ordered). We compute these lists using BBDs, a standard technique for concisely and cannoncially representing boolean functions (**?**).

We lift the remaining operators (except Kleene star) to their list equivalents: union becomes list append (written $+\!\!+$); concatenation becomes the fusion product (lifted to lists this time), negation is implemented using list difference ($\backslash$), and the modal operator lifts the alternative relation over lists of strings[8]. The only caveat to these direct interpretations is that we restrict the operators to have size $\leq n$, denoted as $l|_n$ for a list of guarded strings l.

The denotation of p^* uses the fact that p^* and $1 + p; p^*$ are equivalent, and decrements the size threshold on the recursive denotation of p^* by i, where i is the length of the longest (nonzero) string in the denotation of p, making sure to filter out guarded strings that are too long.

The lists we use here are *lazy* (as opposed to *strict*), which broadly means that computation is delayed until the value is needed. This allows us to avoid computing large, unnecessary iterations.

6 Syntax-semantics interface

English sentences are mapped to terms in the logical language via an interpreted grammar, and these terms are in turn interpreted as propositions (sets of possible worlds). The grammar is a semantically interpreted multi-modal categorial grammar, consisting of a lexicon of words, their categorial

$$
\begin{aligned}
\llparenthesis p \rrparenthesis_0^{\mathsf{B},\mathsf{P},\phi,\zeta} &\triangleq [] \\
\llparenthesis 0 \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta} &\triangleq [] \\
\llparenthesis 1 \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta} &\triangleq \lfloor \mathcal{A}_{\mathsf{B}}^{\varphi} \rfloor \\
\llparenthesis e \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta} &\triangleq [\alpha e \beta \mid \alpha \; \zeta_e \; \beta] \\
\llparenthesis b \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta} &\triangleq \lfloor \mathcal{A}_{\mathsf{B}}^{b\psi} \rfloor \\
\llparenthesis p+q \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta} &\triangleq \llparenthesis p \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta} +\!\!+ \llparenthesis q \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta} \\
\llparenthesis p; q \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta} &\triangleq (\llparenthesis p \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta} \diamond \llparenthesis q \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta})|_n \\
\llparenthesis p^* \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta} &\triangleq [] + (\llparenthesis p \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta} \diamond \llparenthesis p^* \rrparenthesis_{n-i}^{\mathsf{B},\mathsf{P},\phi,\zeta})|_n \\
\text{where} \quad i &= \max\{1, \min\{|g| \mid g \in \llparenthesis p \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta}\}\} \\
\llparenthesis \neg p \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta} &\triangleq \llparenthesis \bullet^* \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta} \setminus \llparenthesis p \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta} \\
\llparenthesis \Diamond_a p \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta} &\triangleq [g' \mid g' \, \hat{R}_a \, g, \texttt{ for } g \texttt{ in } \llparenthesis p \rrparenthesis_n^{\mathsf{B},\mathsf{P},\phi,\zeta}]
\end{aligned}
$$

Figure 7: Bounded interpretation using lazy lists

types, and interpretations in a logical lambda language. The grammar covers basic statives (it's heads), that- and whether-complements of *know*, sentence negation, and predicate and sentence conjunction. Figure 8 gives illustrative lexical entries.[9][10] The grammar and semantics are optimized for a simple fragment of English concerned with clausal complementation. The agent names *Amy* and *Bob* contribute the epistemic alternative relations for those agents, rather than individuals. This is possible because the agents are never arguments of extensional predicates. The root verb *know* contributes existential modal force. The complementizers *that* and *whether* are the heads of their dominating clauses, and assemble an alternative relation, modal force, and proposition contributed by the complement. These complementizers introduce the dual via two negations, in order to express universal modal force.

Multimodal categories such as $\backslash_D$ and $\backslash_M$ are used to control the derivation. The semantic translations in the third column of Figure 8 use the Epik term language, incremented with lambda. The body of $\lambda x. \bullet^* \, h$, which is the semantic lexical entry for *heads*, is a term denoting the set of all worlds where the coin is heads, expressed as the set of all guarded strings that end with a Boolean valuation where the primitive proposition h (it's heads) is true. The body of $\lambda p.\lambda R.\Diamond_R p$, which is the semantic lexical entry of *knows*, is an Epik term denoting the preimage of the world-alternative relation contributed by the subject. This is not the right semantics for *Amy knows that it's heads*, because it is an existen-

[8] Figure **??** depicts this using the list comprehension notation, which is analogous to set builder notation, except that it is written using square brackets. Element order is evoked by the keyword `for`, rather than using the unordered $\forall$.

[9] Category symbols use Lambek/Bar-Hillel notation for slashes, so that $(d\backslash t)/(d\backslash_D t)$ combines with $d\backslash_D t$ on the right to give a value that combines with d on the left to give t.

[10] Lambda abstractions with multiple parameters are written $\lambda x \; y. \; e$ rather than the more verbose $\lambda x.\lambda y.e$.

ITEM	TYPE	SEMANTICS
Amy	e	R_a
Bob	e	R_b
it	d	d
heads	$d\backslash_D t$	$\lambda x.\bullet^*\, h$
tails	$d\backslash_D t$	$\lambda x.\bullet^*\, t$
is	$(d\backslash t)/(d\backslash_D t)$	$\lambda P\, x.\, P\, x$
knows	$(e\backslash t)/_M t$	$\lambda p\, R.\, \Diamond_R p$
that	$(((e\backslash t)/_M t)\backslash(e\backslash t))/t$	$\lambda p\, m\, R.\neg(m\,(\neg p)\, R)$
whether	$(((e\backslash t)/_M t)\backslash(e\backslash t))/t$	$\lambda p\, m\, R.\neg(m\,(\neg p)\, R) + \neg(m\, p\, R)$

Figure 8: Partial categorial grammar lexicon. The first column has a word form. the second column a categorial type, and third column a semantic translation in a logical language that extends the Epik term language with lambda.

tial modality $\Diamond_R p$, rather than an universal modality $\Box_R p$. This is corrected by the complementizer *that* or *whether*, which introduces the dual.

Sentences are parsed with a chart parser for categorial grammar. The semantics for complex phrases are obtained by syntactic application of semantic translations, accompanied by beta reduction. In consequence, the semantic term translating a sentence is a term of the logical language language. Such a term designates a set of possible words (guarded strings) in the possible worlds model deterimened by an Epik specification. By way of example, (11a) is an English sentence with predicate conjunction and three levels of clausal embedding. Using the grammar and parser, the sentence is mapped to the Epik term in (11b).

(11) a. Amy knows that Bob knows that Amy knows whether it is heads and knows that Bob doesnt know that Amy knows that it is tails.

b. $\Box_{\text{Amy}}\Box_{\text{Bob}}(\Box_{\text{Amy}}\bar{h} + \Box_{\text{Amy}}\bar{t})$
$\cdot\ \Box_{\text{Amy}}\neg\Box_{\text{Bob}}\Box_{\text{Amy}}\bar{t}$

(11b) is compiled an Fst finite state machine with – nodes and – edges that accepts a countably infinite set of strings (worlds). Or similarly interpreted as a lazily computation of guarded strings. These are concrete computational representations of the proposition denoted by (11a). This contrasts with standard computational approaches to intensional semantics, sentences are translated into logical formulas that have a mathematical interpretation as a set of worlds, but not a computational one. Our representations can be used to check entailment between sentences (via a subset check), print random worlds that satisfy the sentence (Fst only), print all worlds shorter than some bound, and the like.

7 Conclusion

[ehc]: Insert happy words!