Lecture 4 – Temporal Logics in AI (II)

1. Introduction to Temporal Reasoning

2. Types of Logics and Techniques

   (a) Method of temporal arguments

   (b) Method of reification

3. Three Formalisms

   (a) BTK – Bacchus et. al.

   (b) STL – Shoham

   (c) Situation Calculus – McCarthy

Shoham's STL

– STL is a temporal logic which uses the method of reification and has a non-standard semantics.

– An attempt by Shoham to formally define reification techniques used in the logics of McDermott and Allen.

– Starting Point: "how does one represent temporal information?".

– Example: the color of HOUSE17 is red during the interval $\langle 1, 4 \rangle$.

– Temporal arguments method:

$$Color(1, 4, HOUSE17, Red).$$

Shoham is very much against such representations:

> This option is not acceptable from our standpoint, although there is nothing technically wrong with it. The problem is that if time is represented as an argument (or several arguments) to predicates, then there is nothing general you can say about the temporal aspects of assertions. For example, you cannot say that "effects cannot precede their causes"; at most you can say that about specific causes and effects. Indeed, this first option accords no special status to time – neither conceptual nor notational – which goes against the very spirit of our enterprise.([**?**], p. 90.)

Shoham's Method

– Separate atemporal component from the temporal component:

- **Temporal component:** Interval $\langle 1, 4 \rangle$
- **Atemporal component:** Propositional Type "HOUSE17 is red".

– The propositional type and the interval will appear as arguments to the "predicate" $TRUE$:

$$TRUE(1, 4, COLOR(HOUSE17, RED))$$

The immediate questions that arise are

1. What is the logical status of $COLOR(HOUSE17, RED)$? Is it a term or somthing else?
2. What sorts of entities are allowed as propositional types? Do we allow variables or quantifiers? If so, what is a well-formed formula?
3. What is the meaning of such formulas?

Two possible approaches:

1. Propositions can be reified, so that the propositional type $COLOR(HOUSE17, RED)$ has the logical status of a term in the language. (McCarthy)

2. Retain the propositional nature of the propositional types and at the same time the classical nature of the logic as a whole. (Shoham)

Shoham's Idea:

- Each primitive formula is an interval/assertion pair, $\langle\langle t_1, t_2\rangle, p\rangle$, where t_1 and t_2 are timepoints and p is a proposition $r(trm_1, \ldots, trm_n)$ where r is a relation symbol and the terms used as arguments are nontemporal.

- $TRUE(t_1, t_2, p)$ used as a visual aid for the primitive formula $\langle\langle t_1, t_2\rangle, p\rangle$.

- Introduce a nonstandard semantics for interpreting primitive formulas.

Syntax

In the 2-sorted case,

- the vocabulary $\Lambda = \langle C, F, R, d\rangle$, contains a mapping d defined on $C \cup F \cup R$;

- Let $C = TC \cup C'$ where $d(c) = t$ for $c \in TC$ and $d(c) = u$ for $c \in C'$;

- For each constant c, $d(c)$ is either t or u, representing the sort (t temporal, u atemporal). TC and C' are disjoint.

- Let $F = TF \cup F'$ where for $f \in TF$, $d(f) = \langle t_1, \ldots, t_n \rightarrow t\rangle$, and for $f \in F'$, $d(f) = \langle u_1, \ldots, u_n \rightarrow u\rangle$. The functions in TF are called *temporal functions* and those in F' are called *atemporal functions*. TF and F' are disjoint.

- Let $R = \{TRUE\} \cup R'$ where for $r \in R'$, $d(r) = \langle u_1, \ldots, u_n\rangle$, and for $r = TRUE$, $d(r) = \langle t_1, t_2, \langle u_1, \ldots, u_n\rangle\rangle$.

The list of variables is:

$$\xi^t_1, \xi^t_2, \xi^t_3, \ldots, \xi^u_1, \xi^u_2, \xi^u_3, \ldots$$

where $TV = \{\xi^t_1, \xi^t_2, \xi^t_3, \ldots\}$ and $V = \{\xi^u_1, \xi^u_2, \xi^u_3, \ldots\}$. In each case the superscript shows the sort of variable.

We can now define terms as follows:

Definition 1 (Temporal Terms) *Let $\Lambda = \langle C, F, R, d \rangle$ be a vocabulary of the 2-sorted logic. Then*

- *For $c \in TC$, c is a term of sort $d(c) = t$.*
- *The variable ξ_i^t is a term of sort t.*
- *If $g \in TF$ is a function symbol of Λ with $d(g) = \langle t_1, \ldots, t_n \rightarrow t \rangle$, and $\mu_1, \ldots, \mu_n$ are terms of Λ of sort t,then*
$$g(\mu_1, \ldots, \mu_n)$$
is a term of Λ of sort $d(g) = t$.

Definition 2 (Atemporal Terms) *Let $\Lambda = \langle C, F, R, d \rangle$ be a vocabulary of the 2-sorted logic. Then*

- *For $c \in C'$, c is a term of sort $d(c) = u$.*
- *The variable ξ_i^u is a term of sort u.*
- *If $g \in F'$ is a function symbol of Λ with $d(g) = \langle u_1, \ldots, u_n \rightarrow u \rangle$, and $\mu_1, \ldots, \mu_n$ are terms of Λ of sorts u,then*
$$g(\mu_1, \ldots, \mu_n)$$
is a term of Λ of sort $d(g) = u$.

Definition 3 (Terms) *The terms of the language are the union of the temporal and atemporal terms.*

Definition 4 (Atomic Formula) *An* atomic formula *of Λ is an expression of Λ of the form:**

$$TRUE(t_1, t_2, r(\mu_1, \ldots, \mu_n))$$

where t_1 and t_2 are temporal terms, $r \in R'$ is a relation symbol of Λ, $d(r) = \langle u_1, \ldots, u_n \rangle$, and $\mu_1, \ldots, \mu_n$ are terms of Λ of sort u.

Definition 5 (Formulas) *The set of* formulas *of Λ is similar to the the 1-sorted case with some modification and is defined inductively as follows:*

1. *The atomic formulas of Λ are formulas of Λ.*

2. *If t_1 and t_2 are temporal terms, then $t_1 = t_2$ and $t_1 \preceq t_2$ are formulas of Λ.*

3. *Clauses 2-4 in Davis, p. 33.*

*Recall that $TRUE(t_1, t_2, r(\mu_1, \ldots, \mu_n))$ is syntactic sugar for $\langle\langle t_1, t_2\rangle r(\mu_1, \ldots, \mu_n)\rangle$. "$TRUE$" is not a predicate or modal operator and "$r(\mu_1, \ldots, \mu_n)$" is not a term.

Semantics

An *interpretation* I of the 2-sorted vocabulary Λ is a tuple $I = \langle D_t, \leq, D_u, TFN, FN, RL, M \rangle$ where:

1. The domain D_t is a non-empty set of timepoints.

2. $\leq$ is a binary mapping $D_t \times D_t \mapsto \{0, 1\}$.

3. The domain D_u is a non-empty set of individuals disjoint from D_t.

4. TFN is a set of total functions $\bigcup_i \{D_t^1 \times, \ldots, D_t^i \mapsto D_t\}$.

5. FN is a set of total functions $\bigcup_i \{D_u^1 \times, \ldots, D_u^i \mapsto D_u\}$.

6. RL is a set of total functions $\bigcup_i \{D_u^1 \times, \ldots, D_u^i \mapsto \{1, 0\}\}$.

7. M is a meaning function $\langle M_1, M_2, M_3, M_4, M_5 \rangle$ where
   (a) M_1 is a mapping from $TC \mapsto D_t$.
   (b) M_2 is a mapping from $C' \mapsto D_u$.
   (c) M_3 is a mapping from $TF \mapsto TFN$.
   (d) M_4 is a mapping from $D_t \times D_t \times F' \mapsto FN$. The first two arguments of M_4 are commutative.
   (e) M_5 is a mapping from $D_t \times D_t \times R' \mapsto RL$. The first two arguments of M_5 are commutative.

Valuations:

A *valuation* v of an interpretation I is the sequence

$$v_j^t, v_j^u;\ j = 1, 2, 3, \ldots$$

where each $v_j^t \in D_t$ and each $v_j^u \in D_u$. We say that v_j^k is the value of the variable ξ_j^k with respect to the valuation v. If v and w are both valuations of I, we write $v \equiv_{k,j} w$ to mean that $v_i^k = w_i^k$ for all $i \neq j$.

Temporal Terms:

If υ is a temporal term of Λ, I is an interpretation of Λ with domains D_t and D_u, and v is a valuation of I, we define $VALUE(\upsilon, I, v) \in D_t$ as follows:

- Temporal Terms:
  1. If $\upsilon \in TC$ then $VALUE(\upsilon, I, v) = M_1(\upsilon)$.
  2. If $\upsilon \in TV$ is the variable ξ_j^t then $VALUE(\upsilon, I, v) = v_j^t$.
  3. If $\upsilon \in TF$ is the function $\upsilon = g(\mu_1, \ldots, \mu_n)$, and if

     $$VALUE(\mu_i, I, v) = d_i \in D_t, \text{ for } i = 1, 2, \ldots, n.$$

     then we set

     $$VALUE(\upsilon, I, v) = M_3(g)(d_1, d_2, \ldots d_n).$$

Atemporal Terms:

If υ is an atemporal term of Λ, I is an interpretation of Λ with domains D_t and D_u, and v is a valuation of I, we define $MVA(d_t, d'_t, \upsilon) \in D_u$ as follows:

- Atemporal Terms:
  1. If $\upsilon \in C'$ then $\forall d_t, d'_t \in D_t.MVA(d_t, d'_t, \upsilon) = M_2(\upsilon)$.
  2. If $\upsilon \in V'$ is the variable ξ^u_j, then $\forall d_t, d'_t \in D_t.MVA(d_t, d'_t, \upsilon) = v^u_j$.
  3. If $\upsilon \in F'$ is the function $\upsilon = g(\mu_1, \ldots, \mu_n)$, then for all $d_t, d'_t \in D_t$, if

     $$MVA(d_t, d'_t, \mu_i) = d_i \in D_u, \text{ for } i = 1, 2, \ldots, n$$

     then

     $$M_4(d_t, d'_t, g)(d_1, d_2, \ldots d_n).$$

If γ is a formula of Λ, and I, d, and v are as above, we define the $VALUE(\gamma, I, v) \in \{0, 1\}$ as follows:

0. If $\gamma = TRUE(t_1, t_2, r(\mu_1, \ldots, \mu_n))$ is an atomic formula and if

   $VALUE(t_1, I, v) = d_t \in D_t$ and $VALUE(t_2, I, v) = d'_t \in D_t$

   and

   $$MVA(d_t, d'_t, \mu_i) = d_i \in D_u, \text{ for } i = 1, 2, \ldots, n$$

   then

   $$VALUE(\gamma, I, v) = M_5(d_t, d'_t, r)(d_1, d_2, \ldots d_n).$$

1-5. Same as in Davis, pp36-37.

6. If $\gamma = (\exists \xi_j^k)\alpha$ then $VALUE(\gamma, I, v) = 1$ if there is a valuation w of I such that $w \equiv_{k,j} v$ and $VALUE(\alpha, I, w) = 1$; else $VALUE(\gamma, I, v) = 0$.

7. If $\gamma = (\forall \xi_j^k)\alpha$ then $VALUE(\gamma, I, v) = 0$ if there is a valuation w of I such that $w \equiv_{k,j} v$ and $VALUE(\alpha, I, w) = 0$; else $VALUE(\gamma, I, v) = 1$.

8. If $\gamma = (t_1 = t_2)$ then $VALUE(\gamma, I, v) = 1$ if $VALUE(t_1, I, v) = VALUE(t_2, I, v)$; otherwise $VALUE(\gamma, I, v) = 0$.

9. If $\gamma = (t_1 \preceq t_2)$ then $VALUE(\gamma, I, v) = 1$ if $VALUE(t_1, I, v) \leq VALUE(t_2, I, v)$; otherwise $VALUE(\gamma, I, v) = 0$.

Temporal Ontology

– Temporal structure is built in: $=$, $<$.

– Shoham's base logic permits reasoning with *temporal propositions*.

– Richer Fact Ontology is often required:

- *Properties* such as an object's color, or speed.
- *Actions* such as moving a block, or shooting a gun.
- *Processes* such as keeping a bucket filled with water, or running around a building.

– Rather than fixing an ontology and hard-wiring specific propositional types into the logic as as done by Allen with his properties/events/processes ontology, or McDermott with his facts/events ontology, Shoham's logic permits the construction of ontology via axiomatization.

– In other words, the axiom writer, or knowledge representer can add new propositional types to the formalism simply by writing axioms for the type. Different kinds of propositional type will be distinguished by specifying how the truth of a proposition over one interval is related to its truth over other intervals.

– **Example1**

"The robot travels less than two miles"

is a type of activity where if it occurs during an interval, it also occurs during all subintervals of the interval.

– Such a propositional type has the property of being *downward hereditary*.

Definition 6 *A proposition type x is considered* downward-hereditary *if whenever it holds over an interval it holds over all of its subintervals, possibly excluding the two end points.*

This property can be characterized in Shoham's logic using the following axiom:

$$\forall t_1, t_2, t_3, t_4.\ t_1 \leq t_3 \wedge t_4 \leq t_2 \wedge t_1 \neq t_4 \wedge t_3 \neq t_2 \wedge TRUE(t_1, t_2, x) \rightarrow TRUE(t_3, t_4, x). \tag{1}$$

– **<u>Example2</u>**

"The robot travelled at a speed of two miles per hour"

is a type of activity where if it occurs in all subintervals of an interval, it also occurs in that interval.

– Such a propositional type has the property of being *upward hereditary*.

Definition 7 *A proposition type x is considered* upward-hereditary *if whenever it holds for all proper subintervals of some nonpoint interval (except possibly at its endpoints) it also holds over the nonpoint interval itself.*

This property can be characterized in Shoham's logic using the following axiom:

$$\begin{aligned} \forall t_1, t_2 . t_1 < t_2 \wedge (\forall t_3, t_4 . t_1 \leq t_3 \leq t_4 \leq t_2 \wedge \\ (t_1 \neq t_3 \vee t_2 \neq t_4) \wedge t_1 \neq t_4 \wedge t_3 \neq t_2 \rightarrow TRUE(t_3, t_4, x)) \\ \rightarrow TRUE(t_1, t_2, x). \end{aligned} \quad (2)$$

STL Blocks World Example

– World: Blocks world where blocks can be moved and painted

We will use the following sorts:

1. timepoints (t),
2. individuals (u).

Our vocabulary, $\Lambda = \langle C, F, R, d \rangle$, contains the following:

1. $C = TC \cup C'$ where

$$TC = \{1, 2, \ldots\} \text{ and } \forall x \in TC.d(x) = t,$$

$$C' = \{Block_1, \ldots Block_n, Table, Red, White, Blue\}$$

$$\text{and } \forall x \in C'.d(x) = u.$$

2. $F = TF \cup F'$ where

$$TF = \{+\} \text{ and } d(+) = \langle t \times t \to t \rangle.$$

$$F' = \{top, move\} \text{ and } d(top) = \langle u \to u \rangle,$$

$$d(move) = \langle u \times u \to u \rangle.$$

3. $R = \{At, Color, Clear, Move, Paint, Location\}$
   $\cup\{Block, Trivial, Ab_x\}$ where

$$d(At) = d(Color) = d(Move) = d(Paint) = d(Eq)$$

$$= \langle u \times u \rangle,$$

$$d(Clear) = d(Trivial) = d(Ab_7) = d(Location)$$

$$= d(Block) = \langle u \rangle,$$

$$d(Ab_n) = \langle u \times u \rangle \text{ for n = 1...4},$$

The list of variables for sorts is:

$$\xi_1^t, \xi_2^t, \xi_3^t, \ldots, \xi_1^u, \xi_2^u, \xi_3^u, \ldots$$

To make the axioms more readable we will use the sort type (possibly subscripted) for variables:

$$t, t_1, t_2, t_3, \ldots, u, u_1, u_2, u_3, \ldots \text{etc.}$$

– **Time Structure:**

We will also use the simplifying assumption that the time structure is that of the (positive and negative)integers (discrete, linear, unbounded on both sides). In other words, we will only assume use of models with this particular structure.

The Axioms

Initial Situation

All blocks are on the table and are white.

$$\forall u.TRUE(1,1,Block(u)) \rightarrow TRUE(1,1,At(u,Table)). \quad (3)$$

$$\forall u.TRUE(1,1,Block(u)) \rightarrow TRUE(1,1,Color(u,White)). \quad (4)$$

Laws of Motion

Blocks change their location when moved and their color when painted.

$$\forall t,t',u,u'.t = t' \wedge TRUE(t,t',Move(u,u')) \wedge \neg TRUE(t,t',Ab_3(u,u')) \rightarrow TRUE(t+1,t'+1,At(u,u')). \quad (5)$$

$$\forall t,t',u,u'.t = t' \wedge TRUE(t,t',Paint(u,u')) \wedge \neg TRUE(t,t',Ab_4(u,u')) \rightarrow TRUE(t+1,t'+1,Color(u,u')). \quad (6)$$

Frame Axioms

Blocks change their location and color only if there is a reason to change.

$$\forall t, t', u, u'. t = t' \wedge TRUE(t, t', At(u, u')) \wedge \neg TRUE(t, t', Ab_1(u, u')) \rightarrow TRUE(t+1, t'+1, At(u, u')). \quad (7)$$

$$\forall t, t', u, u'. t = t' \wedge TRUE(t, t', Color(u, u')) \wedge \neg TRUE(t, t', Ab_2(u, u')) \rightarrow TRUE(t+1, t'+1, Color(u, u')). \quad (8)$$

$$\forall t, t', u, u'. t = t' \wedge TRUE(t, t', Move(u, u')) \rightarrow TRUE(t, t', Ab_1(u, u')). \quad (9)$$

$$\forall t, t', u, u'. t = t' \wedge TRUE(t, t', Paint(u, u')) \rightarrow TRUE(t, t', Ab_2(u, u')). \quad (10)$$

Qualification Axioms

Known conditions which prevent a block from being moved.

$$\forall t, t', u, u'. \neg TRUE(t, t', Block(u) \wedge Clear(top(u))) \vee \neg TRUE(t, t', Clear(u')) \vee TRUE(t, t', Eq(u', top(u))) \rightarrow TRUE(t, t', Ab_3(u, move(u, u'))). \quad (11)$$

State Constraints

Since we do not use types, we must identify our objects.

$$\forall t, t'.TRUE(t, t', Block(Block_1) \wedge \ldots \wedge Block(Block_n)). \tag{12}$$

$$\forall t, t', u.TRUE(t, t', Location(Table)). \tag{13}$$

$$\begin{aligned} &\forall t, t', u.TRUE(t, t', Block(u)) \rightarrow \\ &\quad TRUE(t, t', Location(top(u))). \end{aligned} \tag{14}$$

$$\begin{aligned} &\forall t, t'.TRUE(t, t', Color(Red) \wedge Color(White) \wedge \\ &\qquad Color(Blue)). \end{aligned} \tag{15}$$

A location is clear if there are no trivial objects on it.

$$\begin{aligned} &\forall t, t', u.TRUE(t, t', Clear(u)) \equiv \\ &\quad \neg\exists u'.\neg TRUE(t, t', Trivial(u')) \wedge \\ &\qquad TRUE(t, t', At(u', u)). \end{aligned} \tag{16}$$

$$\begin{aligned} &\forall t, t', u.\neg TRUE(t, t', Ab_7(u) \rightarrow \\ &\quad \neg TRUE(t, t', Trivial(u)) \end{aligned} \tag{17}$$

Querying the System

– To answer the question

"is Block1 White after it is moved on top of Block2?",

we would first add

$$TRUE(1, 1, Move(Block1, top(Block2)))$$

to our theory and then try to prove

$$TRUE(2, 2, Color(Block1, White)).$$

– Here is another query involving explanation.

"Given that Block1 is Red at time 5, can this be explained by the performance of a paint action?".

We would first add

$$TRUE(5, 5, Color(Block1, Red))$$

to our theory and then try to prove

$$\exists t. 1 \preceq t \prec 5 \wedge TRUE(t, t, Paint(Block1, Red)).$$

The Situation Calculus

– The SC is a temporal logic which generally uses the method of reification and is a changed-based formalism.

– No explicit notion of time, but a sequence or tree of situations in which facts are true.

– World persists in one state until an action is performed which changes the state to a new state.

– Temporal reasoning is done by reasoning about facts true in a situation as a result of actions performed in a sequence.

– In SC, only one event occurs at a time, time durations are irrelevant, actions are instantaneous, state of world during performance of an action is also irrelevant.

Situation Calculus:

- Sorted FOPC with sorts for facts, actions and situations.
- Facts will be called *fluents* .
- Reified Logic: "logic where propositions are made into objects (reified) by providing special propsitional terms to refer to them (fluents).

Essentials of the Situation Calculus

- A *Situation* is the complete state of the universe at an instant in time.

- A *fluent* is a function defined on situations.

  For example, the location of a block x at situation s is denoted $location(x, s)$.

- An *action* is a function or constant.

  For example, the action of moving block x to location l is denoted by $move(x, l)$.

- If s is a situation and a is an action then the situation resulting from performing action a in s is represented by the function $result$, where $result(a, s)$ refers to the resulting situation.

  For example,

  $$location(A, result(move(A, Table), S0)) = Table$$

  states that the location of A after performing the move to table operation in situation $S0$ is $Table$ and $result(move(A, Table), S0)$ refers to the situation in which this is the case.

- In order to assert that fluents have particular values in particular states, we use the predicate $Holds$, where if f is a fluent and s is a situation then $Holds(f, s)$ denotes that the fluent f is true in situation s.

  For example,

  $$Holds(dead(fred), result(shoot(fred), S0))$$

  states that fred is dead in the situation resulting from performing the action of shooting fred in situation $S0$.

Syntax
Semantics

The Syntax (see example) and Semantics of SC are those of an n-sorted FOPC. See BTK and extrapolate or Davis' article.

SC Temporal Ontology

– Situations are the basic temporal notion in SC, yet it is unclear just what a situation is.

– SC has been used more as a notation where different researchers have used different axiomatizations for it, including the notion of a situations

Some possible choices:

1 Situation domain forms a tree structure with intial situation $S0$ as root and where $Result$ is used to form successive descendents. Situations are then somewhat like tags where various facts are true or false and its situation is unique. In this case, the following axioms are true:

$$S0 \neq Result(A, s)$$

$$Result(A, s) = Result(A', s') \supset A = A' \wedge s = s'$$

This is referred to as Herbrand Situations.

2 A situation is a timeless state of the world, i.e. a mapping from properties to truth values. States are characterized by two properties:

(a) *Uniqueness property*:

$$\forall r \forall r'.[\forall p.[Holds(p,r) \equiv Holds(p,r')] \supset r = r']$$

(b) *Existence property*:

$$\forall r \forall p \exists r' [(Holds(p,r) \equiv \neg Holds(p,r')) \wedge \\ \forall p' [p \neq p' \wedge (Holds(p',r) \equiv Holds(p',r'))]] \quad (18)$$

– Two choices relating situations and states:

1. *Strict State Semantics* - Identifies situations and states:

2. *Weak State Semantics* - A state is associated with each situation, but several situations may be associated with the same state.

– Each model for the situation calculus contains the whole situations space, i.e. it assigns a truth-value to each property/situation pair. Several alternative histories are represented in each model.

– Multiple models for a scenario are obtained if $Holds(p, S0)$ is unspecified for some p, and if (for Herbrand situations) if some actions are non-deterministic.

SC Blocks World Example

– World: Blocks world where blocks can be moved and painted

We will use the following sorts:

1. actions (a)
2. situations (s)
3. truth-valued fluents (f)
4. blocks (b)
5. locations (l)
6. colors (c).

Our vocabulary, $\Lambda = \langle C, F, R, d \rangle$, contains the following:

1. $C = C_s \cup C_b \cup C_l \cup C_c$ where

$$C_s = \{S0\} \text{ and } \forall x \in C_s.d(x) = s,$$

$$C_b = \{Block_1, \ldots Block_n\} \text{ and } \forall x \in C_b.d(x) = b,$$

$$C_l = \{Table\} \text{ and } \forall x \in C_l.d(x) = l,$$

$$C_c = \{Red, White, Blue\} \text{ and } \forall x \in C_c.d(x) = c.$$

2. $F = \{top, at, color, clear, move, paint, result, trivial\}$ where

$$d(top) = \langle b \to l \rangle,$$

$$d(at) = \langle b \times l \to f \rangle,$$

$$d(color) = \langle b \times c \to f \rangle,$$

$$d(clear) = \langle l \to f \rangle,$$

$$d(move) = \langle b \times l \to a \rangle,$$

$$d(paint) = \langle b \times c \to a \rangle,$$

$$d(eql) = \langle l \times (b \to l) \to f \rangle,$$

$$d(result) = \langle a \times s \to s \rangle.$$

$$d(trivial) = \langle b \to f \rangle,$$

3. $R = \{Holds\}$ where

$$d(Holds) = \langle f \times s \rangle.$$

$$d(Ab_n) = \langle b \times a \times s \rangle \text{ for n} = 1 \ldots 4,$$

$$d(Ab_7) = \langle b \times s \rangle.$$

– The list of variables for sorts is:

$$\xi_1^a, \xi_2^a, \xi_3^a, \ldots, \xi_1^s, \xi_2^s, \xi_3^s, \ldots \text{etc.}$$

To make the axioms more readable we will use the sort type (possibly subscripted) for variables:

$$a, a_1, a_2, a_3, \ldots, s, s_1, s_2, s_3, \ldots \text{etc.}$$

The Axioms

Initial Situation

$$\forall b. Holds(at(b, Table), S0) \tag{19}$$

$$\forall b. Holds(color(b, White), S0) \tag{20}$$

Laws of Motion

Blocks change their location when moved and their color when painted.

$$\begin{gathered}\forall b, a, s. \neg Ab_3(b, a, s) \rightarrow \\ Holds(at(b, l), result(move(b, l), s))\end{gathered} \tag{21}$$

$$\begin{gathered}\forall b, a, s. \neg Ab_4(b, a, s) \rightarrow \\ Holds(color(b, c), result(paint(b, c), s))\end{gathered} \tag{22}$$

Frame Axioms

Blocks change their location and color only if there is a reason to change.

$$\forall b, a, s, l. Holds(at(b,l), s) \wedge \neg Ab_1(b,a,s) \rightarrow Holds(at(b,l), result(a,s)) \tag{23}$$

$$\forall b, a, s, c. Holds(color(b,c), s) \wedge \neg Ab_2(b,a,s) \rightarrow Holds(color(b,c), result(a,s)) \tag{24}$$

$$\forall b, l, s. Ab_1(b, move(b,l), s) \tag{25}$$

$$\forall b, c, s. Ab_2(b, paint(b,c), s) \tag{26}$$

Qualification Axioms

Known conditions which prevent a block from being moved.*

$$\forall b, l, s. \neg Holds(clear(top(b)), s) \vee \neg Holds(clear(l), s) \vee Holds(eql(l, top(b)), s) \rightarrow Ab_3(b, move(b,l), s) \tag{27}$$

*Note that eql is a very specific type of equality fluent, applying only to locations. It should be made more general.

State Constraints

A location is clear if there are no trivial objects on it.*

$$\forall l, s. Holds(clear(l), s) \equiv \\ \neg\exists b. \neg Holds(trivial(b), s) \wedge Holds(at(b, l), s) \quad (28)$$

$$\forall b, s. \neg Ab_7(b, s) \rightarrow \neg Holds(trivial(b), s) \quad (29)$$

*Trivial objects would be such things as specks of dust, etc. Unfortunately, we only have the sort blocks for the moment, all of which are non-trivial objects.

Querying the System

– For example, to answer the question

"is Block1 White after it is moved on top of Block2?",

we would try to prove

$$Holds(color(Block1, White), \\ result(move(Block1, top(Block2), S0))). \qquad (30)$$

–Suppose we would like a plan for painting Block1 Red, placing Block2 on Block3, and Block1 on Block2. We would try to prove the following constructively:

$$\exists s. Holds(color(Block1, Red), s) \wedge \\ Holds(at(Block1, top(Block2), s) \wedge \\ Holds(at(Block2, top(Block3), s) \qquad (31)$$

A reasonable answer might be:

$$result(move(Block1, top(Block2)), s2)$$

where

$$s1 = result(paint(Block1, Red), S0),$$

$$s2 = result(move(Block2, top(Block3)), s1).$$

Conclusions

- Comparing BTK, STL, SC
  - Syntactic Overhead
  - Semantic Overhead
  - Type and accessibility of temporal structure.
  - Type of actions, events etc.
  - Possibility of implementation
  - Expressivity