Specification and classification with Tarskian system executions

(Bakery Algorithms as an example)

Uri Abraham

Departments of Mathematics and CS

Ben Gurion University of the Negeuv

Be'er Sheva, Israel

Verification Day 2008.

Verification as a human activity

What are the most suitable formal frameworks for arguing about concurrency, expressing ideas about specifications of distributed systems, and conducting proofs at both the informal and formal levels in ways that researchers and learners of algorithms will find most natural?

Verification as a human activity

What are the most suitable formal frameworks for arguing about concurrency, expressing ideas about specifications of distributed systems, and conducting proofs at both the informal and formal levels in ways that researchers and learners of algorithms will find most natural?

I will argue that in many cases Tarskian structures are the natural answer.

Verification as a human activity

What are the most suitable formal frameworks for arguing about concurrency, expressing ideas about specifications of distributed systems, and conducting proofs at both the informal and formal levels in ways that researchers and learners of algorithms will find most natural?

I will argue that in many cases Tarskian structures are the natural answer.

I will support this claim by showing how two seemingly different algorithms, one using registers and the other channels, share the same high-level abstract proof of mutual exclusion.

Some history

Cantor discovered Infinity, and Tarski the Truth.

Some history

What was so difficult to prove at Cantor’s time (that the reals are equinumerous with the plane) became easy when the right framework was found. (Due to work by Tarski and others.)

What is a formal model? The world—the language—the models

We shall compare two kinds of models: the Tarskian models for predicate languages, and the state-history models for temporal logic.
Introduced by Pnueli (The temporal logic of programs. 1977), temporal logic is used to specify the behavior of reactive systems and to prove their properties. This is now an important and very active research area with industrial applications. Yet, for human reasoning, we believe that the older quantification languages are more suitable, and we want to make this point with a concrete example. We will try to define a class of Bakery Algorithms using first-order predicate languages.

What are Tarskian structures?

A Tarskian structure M is an interpretation of a predicate language; it consists of:

1. A universe, namely a set A of elements.
2. For each sort S (a type of elements) a set $S^M \subseteq A$ is associated which represents the members of M of sort S.
3. For every relation symbol R of arity k in the signature, a relation $R^M \subseteq A^k$.
4. For every function symbol F in the signature, of arity k, F^M is a function from A^k to A.
5. For every constant c of sort S in the signature, $c^M \in S^M$.

The state based history models

- A global state is an instantaneous description of the system. Some states are designed as "initial states".
- A step, or action, can atomically change a global state into another state.
- A history is a sequence of global states, beginning with an initial state and such that every state and its successor in the history form a step. A system is modeled by the set of all its possible histories. This is the interleaving semantics.
- A correctness condition is some Temporal logic sentence φ. If it holds in every history of the system, then the system is said to satisfy φ.

For mechanical correctness proofs the state/history approach is the best choice, but for human understanding a Tarskian approach seems more natural in many cases.

Lamport's informal notes

A new approach to proving the correctness of multiprocess programs. (1979)

> Like everyone else at the time, when I began studying concurrent algorithms, I reasoned about them behaviorally. Such reasoning typically involved arguments based on the order in which events occur. I discovered that proofs can be made simpler, more elegant, and more mathematical by reasoning about operations (which can be composed of multiple events)...
> [All behavioral reasoning is hard to make completely formal, and it collapses under the weight of a complex problem.] You should use assertional methods to reason about complex algorithms.

We claim however that:

Behavioral reasoning can be made completely formal. Tarskian structures can capture the essence of the behavior of an algorithm at the appropriate abstract level.

Simplistic Bakery: a global clock mutual exclusion

l_1 $\quad R_i :=$ *pick-a-number*$()$
l_2 $\quad$ **forall** $j \neq i$
$\qquad\qquad$ **repeat** $n := R_j$
$\qquad\qquad$ **until** $n > R_i$
l_3 $\quad$ Critical Section
l_4 $\quad R_i := \infty$.

Protocol for process P_i. Execution of line l_1 is assumed to be atomic, and later executions pick larger numbers. The simplistic protocol relies on several strong assumptions about the available hardware.

Crucial Properties of executions

Assume that X in P_j and Y in P_i where $i \neq j$ are protocol executions, and let $x = \textit{pick}(X)$, $y = \textit{pick}(Y)$.

1. Either x dominates y, or y dominates x (but not both).
2. If $y \prec x$, then x dominates y.
3. If Y is terminating and $\Omega(Y, j) \prec X$, then $y \prec x$.
4. Assume that Y is terminating and $\Omega(Y, j) = X$. Then
   (a) $x \prec cs(Y)$, and
   (b) x dominates y, or X is terminating and $cs(X) \prec cs(Y)$.

Theorem. *The high-level properties imply the mutual exclusion property.* In fact, we prove more. If X and Y are terminating executions such that $y = \textit{pick}(Y)$ dominates $x = \textit{pick}(X)$, then $cs(X) \prec cs(Y)$.

proof

If $y \prec x$, then x dominates y, which contradicts our assumption that y dominates x. Thus we get

(1) $$\neg(y \prec x).$$

X and $\Omega(Y, j)$ are in P_j. By seriality of processes, we have one (and only one) of the following three possibilities.
Case 1: $\Omega(Y, j) \prec X$. By Item 3, $y \prec x$. A contradiction.

Case 2: $X \prec \Omega(Y, j)$. Say, $Z = \Omega(Y, j)$. By 4(a) we have *pick*$(Z) \prec cs(Y)$. Since $X \prec Z$ and X and Z are both in P_j, X is terminating and hence $cs(X)$ is defined and $cs(X) \prec$ *pick*(Z). So $cs(X) \prec cs(Y)$ by transitivity of $\prec$, as required.

Case 3: $X = \Omega(Y, j)$. Then item 4(b) implies that x dominates y, or X is terminating and $cs(X) \prec cs(Y)$. But it is not the case that x dominates y, and hence $terminating(X)$ and $cs(X) \prec cs(Y)$.

The Ricart and Agrawala algorithm uses messages and is based on Lamport's causal ordering of events. It turns our that this algorithms satisfies the same Crucial Properties of executions that the Bakery Algorithm satisfies, when $\prec$ is interpreted as the causal ordering. The fact that two seemingly very different algorithms (one based on shared memory and the other on messages) share the same reasons for the mutual-exclusion property and that these properties can be expressed in a first-order language should be viewed as a support for the claim that Tarskian structures are adequate for this mission. For details ask for a preprint.