Infinite-State Systems

(Based on [ACJT 2000])

Yih-Kuen Tsay

Dept. of Information Management

National Taiwan University

Infinite-State Systems

- Where does infinity arise?
  - Infinite data domains: integers, reals, ...
  - Unbounded storage, message channels, ...
  - Unbounded replications of finite artifacts
  - Time and other natural phenomena (temperature, atmospheric pressure, ...)
- Popular computation models
  - Classic: push-down automata, Turing machines, ...
  - Petri nets and their variants
  - Transition systems with an infinite set of states
  - Indexed/parameterized finite-state systems
  - Timed automata, Hybrid automata, ...

Undecidablity and Decidablity

- The halting problem for Turing machines is *undecidable*.
- Rice's theorem:

  Any *non-trivial* problem about Turing machines is *undecidable*.
- When a computation model can simulate the Turing machine, there is no hope of fully automating its verification.
- To obtain decidability, some restrictions must be present:
  - limited operations on the storage/variables
  - certain regularity on the state transitions
  - lossy message channels
  - simpler test/enabling conditions

Well-Structured Systems: An Overview

- Well-structured systems embody general mathematical structures that are common of several popular infinite-state computation models.
- In such systems, the state space is equipped with a simulation preorder (quasi-order) in that “smaller” states can be simulated by “larger” states.
- The quasi-order, in addition, is a well quasi-ordering in that every infinite sequence of states contains a pair of states such that the earlier state is “smaller” than the latter state.
- With additional requirements, the following problems are decidable for well-structured systems: reachability, eventuality, and simulation.

Labeled Transition Systems

- As a general model of infinite-state systems, we adopt labeled transition systems.
- We assume a finite set Λ of *labels*. Each label $\lambda \in \Lambda$ represents an observable interaction with the environment.
- A (labeled) transition system $\mathcal{L}$ is a pair $\langle S, \delta \rangle$:
  - S is a set of *states*, formed as the cartesian product $Q \times D$ of a finite set Q of *control states* and a possibly infinite set D of *data values*.
  - $\delta \subseteq S \times \Lambda \times S$ is a set of *transitions*.

Labeled Transition Systems (cont.)

- We use $\langle q, d \rangle$ to denote the state whose control state is q and whose data value is d.
- We use $s \xrightarrow{\lambda} s'$ to denote that $\langle s, \lambda, s' \rangle \in \delta$. Intuitively, $s \xrightarrow{\lambda} s'$ means that the system can move from state s to state s' while performing the observable action λ.
- We let $s \longrightarrow s'$ denote that there is a λ such that $s \xrightarrow{\lambda} s'$, and let $\xrightarrow{*}$ denote the reflexive transitive closure of $\longrightarrow$.
- For $s \in S$ and $T \subseteq S$, we say that *T is reachable from s* (written $s \xrightarrow{*} T$) if there exists a state $s' \in T$ such that $s \xrightarrow{*} s'$.

Labeled Transition Systems (cont.)

- For $T \subseteq S$ and $\lambda \in \Lambda$, we define $pre_\lambda(T)$ to be the set $\left\{s' \mid \exists s \in T.\ s' \stackrel{\lambda}{\longrightarrow} s\right\}$.
- Analogously, we define $post_\lambda(T)$ as $\left\{s' \mid \exists s \in T.\ s \stackrel{\lambda}{\longrightarrow} s'\right\}$.
- By $pre(T)$ ($post(T)$) we mean $\cup_{\lambda \in \Lambda} pre_\lambda(T)$ ($\cup_{\lambda \in \Lambda} post_\lambda(T)$).
- Sometimes we write $pre(s)$ ($post(s)$) instead of $pre(\{s\})$ ($post(\{s\})$).
- A *computation from a state* s is a sequence of the form $s_0 s_1 \cdots s_n$, where $s_0 = s$, $s_i \longrightarrow s_{i+1}$, and either $n = \infty$ (i.e., the sequence is infinite) or there is no state s' such that $s_n \longrightarrow s'$.

Quasi-Orders (or Preorders)

- A *preorder* $\preceq$ is a reflexive and transitive (binary) relation on a set D.
- We say that $\preceq$ is *decidable* if there is a procedure which, given $a, b \in D$, decides whether $a \preceq b$.
- The relation $\preceq$ is a *well quasi-ordering* if, for every infinite sequence $a_0, a_1, a_2, \ldots$ in D, $a_i \preceq a_j$ for some $i < j$.
- A set M is said to be *canonical* if $a, b \in M$ implies $a \not\preceq b$.
- We say that $M \subseteq A$ is a *minor set* of A, if
  1. for all $a \in A$ there exists $b \in M$ such that $b \preceq a$, and
  2. M is canonical.

Partial Orders vs. Quasi-Orders

- Let P be a set.
- A *partial order*, or simply *order*, on P is a binary relation $\leq$ on P with the following properties:
  1. $\forall x \in P, x \leq x$. (reflexivity)
  2. $\forall x, y, z \in P, x \leq y \wedge y \leq z \rightarrow x \leq z$. (transitivity)
  3. $\forall x, y \in P, x \leq y \wedge y \leq x \rightarrow x = y$. (antisymmetry)
- A set P equipped with a partial order $\leq$, often written as $\langle P, \leq \rangle$, is called a *partially ordered set*, or simply *ordered set*, sometimes abbreviated as *poset*.
- A binary relation that is reflexive and transitive is called a *pre-order* or *quasi-order*.

Chain Conditions

- Let P be an ordered set.
- P satisfies the ascending chain condition (ACC), if given any sequence $x_1 \leq x_2 \leq \cdots \leq x_n \leq \cdots$ of elements in P, there exists $k \in N$ such that $x_k = x_{k+1} = \cdots$.
- Dually, P satisfies the descending chain condition (DCC), if given any sequence $x_1 \geq x_2 \geq \cdots \geq x_n \geq \cdots$ of elements in P, there exists $k \in N$ such that $x_k = x_{k+1} = \cdots$.

Quasi-Orders (cont.)

- A set $I \subseteq D$ is an *ideal* (in D) if $a \in I$, $b \in D$, and $a \preceq b$ imply $b \in I$, i.e., the set I is upward-closed with respect to the relation $\preceq$.
- We define the (upward) *closure* of a set $A \subseteq D$, denoted $\mathcal{C}(A)$, as the ideal $\{b \in D \mid \exists a \in A.\ a \preceq b\}$ which is generated by A.
- For sets A and B, we say that $A \equiv B$ if $\mathcal{C}(A) = \mathcal{C}(B)$.
- Observe that $A \equiv B$ if and only if for all $a \in A$ there is a $b \in B$ such that $b \preceq a$, and vice versa.

Quasi-Orders (cont.)

Lemma 3.1. If a preorder $\preceq$ on D is a well quasi-ordering, then for each subset A of D there exists at least one finite minor set of A.

- Assuming no finite minor set of A exists, we can find, for any sequence $a_0, a_1, a_2, \ldots, a_i$, an element a_{i+1} such that $a_j \not\preceq a_{i+1}$ for each $j, 0 \leq j \leq i$.
- Continuing this way, we can construct an infinite sequence $a_0, a_1, a_2, \ldots$ that violates the well quasi-ordering property.
- We use min to denote a function which, given a set A, returns a minor set of A.
- From Lemma 3.1 and that $\mathcal{C}(min(I)) = I$ for each ideal I, we can use $min(I)$ as a finite representation of I.

Quasi-Orders (cont.)

Lemma 3.2. For a preorder $\preceq$ on a set A, $\preceq$ is a well quasi-ordering iff for each infinite sequence $I_0 \subseteq I_1 \subseteq I_2 \subseteq \cdots$ of ideals in A there is a k such that $I_k = I_{k+1}$

- (only if) Suppose $I_0 \subset I_1 \subset I_2 \subset \cdots$. It follows that there is a sequence $a_0, a_1, a_2, \ldots$ of elements in A such that for all $k \geq 0$ we have $a_k \in I_k$ and $a_k \notin I_j$ for each $j < k$. This means $a_j \not\preceq a_k$ for $j < k$, otherwise $a_k \in I_j$, since I_j is an ideal.
- (if) Suppose there exists an infinite sequence $a_0, a_1, a_2, \ldots$ in A where $a_j \not\preceq a_k$ if $j < k$. We define an infinite sequence $I_0, I_1, I_2, \ldots$ of ideals where $I_k = \mathcal{C}(\{a_0, a_1, \ldots, a_k\})$. It is clear that $I_0 \subset I_1 \subset I_2 \subset \cdots$.

Monotonicity (Simulation)

- We require that the set D of data values is equipped with a decidable preorder $\preceq$.
- We assume that we are given a minor set of D which we henceforth call D_{min}.
- We extend the preorder $\preceq$ on D to a decidable preorder $\preceq$ on the set S of states defined by $\langle q, d \rangle \preceq \langle q', d' \rangle$ if and only if $q = q'$ and $d \preceq d'$.
- A transition system $\langle S, \delta \rangle$ is *monotonic* (with respect to $\preceq$) if

  for each $s_1, s_2, s_3 \in S$ and $\lambda \in \Lambda$, if $s_1 \preceq s_2$ and $s_1 \xrightarrow{\lambda} s_3$, then there exists s_4 such that $s_3 \preceq s_4$ and $s_2 \xrightarrow{\lambda} s_4$.

Monotonicity (cont.)

Lemma 3.3. A transition system $\langle S, \delta \rangle$ is monotonic iff the set of ideals in S is closed under the applications of both pre_λ and pre.

- (only if) Take any ideal I in S. Given $s_1 \in pre_\lambda(I)$ and $s_1 \preceq s_2$, we need to show that $s_2 \in pre_\lambda(I)$. There is $s_3 \in I$ such that $s_1 \xrightarrow{\lambda} s_3$. By monotonicity it follows that there is s_4 such that $s_3 \preceq s_4$ and $s_2 \xrightarrow{\lambda} s_4$. Since I is an ideal, we have $s_4 \in I$ and hence $s_2 \in pre_\lambda(I)$.
- (if) Assuming $\langle S, \delta \rangle$ is not monotonic, there are states s_1, s_2, and s_3, and $\lambda \in \Lambda$ such that $s_1 \preceq s_2$, $s_1 \xrightarrow{\lambda} s_3$, but there is no s_4 where $s_3 \preceq s_4$ and $s_2 \xrightarrow{\lambda} s_4$. Define the ideal $I = \mathcal{C}(\{s_3\})$. It is clear that $s_1 \in pre_\lambda(I)$ but $s_2 \notin pre_\lambda(I)$. This means that $pre_\lambda(I)$ is not an ideal.

Well-Structured Systems

- A transition system $\mathcal{L} = \langle S, \delta \rangle$, assuming a decidable preorder $\preceq$ on the set D of data values, is said to be *well-structured* if
  1. it is monotonic;
  2. $\preceq$ is a well quasi-ordering; and
  3. for each state $s \in S$ and $\lambda \in \Lambda$, the set $min(pre_\lambda(\mathcal{C}(\{s\})))$ is computable.
- Note that $min(pre_\lambda(\mathcal{C}(\{s\})))$ is finite if $\preceq$ is a well quasi-ordering.
- We define $minpre_\lambda(s)$ as notation for $min(pre_\lambda(\mathcal{C}(\{s\})))$.

Lattices and Complete Lattices

- Let P be a *non-empty* ordered set.
- P is called a *lattice* if $x \vee y$ and $x \wedge y$ exist for all $x, y \in P$.
- P is called a *complete lattice* if $\bigvee S$ and $\bigwedge S$ exist for all $S \subseteq P$.
- Every finite lattice is complete.

Directed Sets

- Let S be a *non-empty* subset of an ordered set.
- S is said to be *directed* if, for every pair of elements $x, y \in S$ there exists $z \in S$ such that $z \in \{x, y\}^u$.
- S is directed if and only if, for every finite subset F of S, there exists $z \in S$ such that $z \in F^u$.
- In an ordered set with the ACC, a set is directed if and only if it has a greatest element.
- When D is directed for which $\bigvee D$ exists, we write $\sqcup D$ in place of $\bigvee D$.

Complete Partial Orders (CPO)

- An ordered set P is called a *Complete Partial Order* (*CPO*) if
  1. P has a bottom element $\perp$ and
  2. $\sqcup D$ exists for each directed subset D of P.
- Alternatively, P is a CPO if each chain of P has a least upper bound in P.
- Any complete lattice is a CPO.
- For an ordered P satisfying Condition 2 above (called a pre-CPO), its lifting $P_\perp$ is a CPO.

Continuous Maps

- Let P and Q be CPOs.
- A map $\varphi : P \to Q$ is said to be continuous if, for every directed set D in P,
  1. the subset $\varphi(D)$ of Q is directed and
  2. $\varphi(\bigsqcup D) = \bigsqcup \varphi(D)$.
- A continuous map need not preserve bottoms, since by definition the empty set is not directed.
- A map $\varphi : P \to Q$ such that $\varphi(\bot) = \bot$ is called strict.

A Fixpoint Theorem for CPOs

- The n-fold composite F^n of $F : P \to P$ is defined as follows.
  1. F^0 is the identity.
  2. $F^n = F \circ F^{n-1}$ for $n \geq 1$.
- If F is order-preserving, so is F^n.

CPO Fixpoint Theorem I

Let P be a CPO and $F : P \to P$ an *order-preserving* map. Define $\alpha \triangleq \bigsqcup_{n \geq 0} F^n(\bot)$.

1. If $\alpha \in \mathrm{fix}(F)$, then $\alpha = \mu(F)$.
2. If F is continuous, then $\mu(F)$ exists and equals α.

Control State Reachability

- The problem: given a state s and a control state q, we want to check whether $\langle q, D \rangle$ is reachable from s.
- We will actually solve the more general problem of deciding whether an ideal I is reachable from a given state s.
- Since $\langle q, D \rangle$ is an ideal, the control state reachability problem is a special case of the reachability problem for ideals.
- To check the reachability of an ideal I, we perform a reachability analysis backwards.

Control State Reachability (cont.)

- Starting from I we define the sequence $I_0, I_1, I_2, \ldots$ of sets by $I_0 = I$ and $I_{j+1} = I \cup pre(I_j)$.
- Intuitively, I_j denotes the set of states from which I is reachable in at most j steps.
- Thus, if we define $pre^*(I)$ to be $\cup_{j \geq 0} I_j$, then I is reachable from s if and only if $s \in pre^*(I)$.
- Notice that $pre^*(I)$ is the least fixpoint $\mu X.\, I \cup pre(X)$.
- By Lemma 3.3, each I_j is an ideal in S.
- We know that $I_0 \subseteq I_1 \subseteq I_2 \subseteq \cdots$, and hence from Lemma 3.2, it follows that there is a k such that $I_k = I_{k+1}$.
- It can easily be seen that $I_\ell = I_k$ for all $\ell \geq k$ implying that $pre^*(I) = I_k$.

Control State Reachability (cont.)

- The method for deciding whether I is reachable is based on generating the preceding sequence $I_0, I_1, I_2, \ldots$ of ideals, and checking for convergence.
- This cannot be carried out directly since I_j is an infinite set.
- Instead, we represent each I_j by a canonical set $M_j = min(I_j)$.
- By Lemma 3.1, each minor set M_j is finite.
- It is straightforward to show that $M_{j+1} \equiv min(min(I) \cup minpre(M_j))$, which is computable as

$$M_{j+1} = min\left(min(I) \cup \bigcup_{s \in M_j} min(pre(\mathcal{C}(\{s\})))\right)$$

Control State Reachability (cont.)

Theorem 4.1 The control state reachability problem is decidable for well-structured systems.

- From the previous discussion we conclude that if we define $minpre^*(M_0)$ to be $\cup_{j \geq 0} M_j$, then there is a k such that $M_{k+1} \equiv M_k$, and $minpre^*(M_0) \equiv M_k$.
- This implies that $minpre^*(M)$ is computable for any minor set M of I and in fact $\mathcal{C}(minpre^*(M)) = pre^*(I)$.
- Given a state s and a control state q, we compute $minpre^*(\langle q, D_{min} \rangle)$.
- We then check whether there is an $s' \in minpre^*(\langle q, D_{min} \rangle)$ such that $s' \preceq s$.

Abstract Interpretation

- The above analysis algorithm can also be phrased in terms of abstract interpretation.
- We intend to compute the fixpoint $\mu X.\, I \cup pre(X)$ for a set $I \subseteq S$ by iteration.
- Instead of computing this fixpoint in the lattice $\langle 2^S, \subseteq \rangle$ of sets of states, we move to the abstract lattice $\langle \mathcal{M}, \sqsubseteq \rangle$, where $\mathcal{M}$ is the set of canonical subsets of S, and where $M \sqsubseteq M'$ if $\mathcal{C}(M) \subseteq \mathcal{C}(M')$.

Abstract Interpretation (cont.)

- The correspondence between the concrete lattice $\langle 2^S, \subseteq \rangle$ and the abstract lattice $\langle \mathcal{M}, \sqsubseteq \rangle$ is expressed by a pair $\langle \alpha, \gamma \rangle$ of functions as follows.
  - $\alpha : 2^S \mapsto \mathcal{M}$, defined by $\alpha(T) = min(T)$ maps each set of states in the concrete lattice to its abstract representation.
  - $\gamma : \mathcal{M} \mapsto 2^S$, defined by $\gamma(M) = \mathcal{C}(M)$ recovers the concrete meaning of an element in the abstract lattice.

Abstract Interpretation (cont.)

- The pair $\langle \alpha, \gamma \rangle$ forms a *Galois insertion* of $\langle \mathcal{M}, \sqsubseteq \rangle$ into $\langle 2^S, \subseteq \rangle$.
- Let $\langle Concr, \sqsubseteq_{Concr} \rangle$ be an ordered *concrete* domain, and let $\langle Abs, \sqsubseteq_{Abs} \rangle$ be an ordered *abstract* domain.
- Consider mappings $\alpha : Concr \rightarrow Abs$, and $\gamma : Abs \rightarrow Concr$.
- We say that the pair $\langle \alpha, \gamma \rangle$ form a *Galois insertion* if
  - α and γ are monotonic,
  - $\forall a \in Abs : a = \alpha(\gamma(a))$, and
  - $\forall c \in Conc : c \sqsubseteq_{Concr} \gamma(\alpha(c))$.

Abstract Interpretation (cont.)

- The algorithm for deciding reachability can be seen as computing the fixpoint $\mu X.\, min(I) \sqcup minpre(X)$ in the lattice $\langle \mathcal{M}, \sqsubseteq \rangle$, where $M_1 \sqcup M_2 = min(M_1 \cup M_2)$.
- Monotonicity ensures that the above corresponds exactly to the computation $\mu X.\, I \cup pre(X)$ in $\langle 2^S, \subseteq \rangle$ if I is an ideal in S.
- Exactness follows from the identity $pre(\gamma(M)) = \gamma(minpre(M))$ for all $M \in \mathcal{M}$, and ensures that if the fixpoint computation converges to M_k, then $\gamma(M_k)$ is the least fixpoint of $\mu X.\, I \cup pre(X)$ in $\langle 2^S, \subseteq \rangle$.
- Finally, the well quasi-ordering of $\preceq$ implies that all ascending chains in $\langle \mathcal{M}, \sqsubseteq \rangle$ are finite, guaranteeing convergence of any least fixpoint computation.

Eventuality Properties

- The problem: deciding whether each computation starting from an initial state eventually reaches a certain control state satisfying a predicate p over control states.
- In CTL, these properties are of the form AFp.
- We present an algorithm for the dual property EGp; checking AFp is equivalent to checking $\neg$EG$\neg p$.
- The property EGp is true in a state s_0 iff there is a computation from s_0 in which all states have a control part that satisfies p.
- The algorithm will actually solve the more general problem of whether s_0 satisfies a property of the form EGI for an ideal I.
- We write this property as $s_0 \models$ EGI.

Eventuality Properties (cont.)

- The algorithm essentially builds a tree of reachable states.
- We must then consider the possibility that $post(s)$ is infinite for some states s.
- We say that a transition system is *essentially finite branching* if for each state s we can effectively compute a finite subset of $post(s)$, denoted $maxpost(s)$, such that for each state $s' \in post(s)$ there is a state $s'' \in maxpost(s)$ with $s' \preceq s''$.
- If $post(s)$ is finite, then $maxpost(s)$ can be taken as $post(s)$. In the cases where $post(s)$ is infinite (as can be the case, e.g., for real-time automata), the subset $maxpost(s)$ can fully represent the set $post(s)$ for the purposes of this algorithm.

Eventuality Properties (cont.)

- We build a tree labeled by properties of the form $s \models \mathsf{EG}I$. The root node is labeled by $s_0 \models \mathsf{EG}I$. A node labeled by $s \models \mathsf{EG}I$ is a leaf if one of the following holds.
  1. $s \notin I$. In this case, the node is considered *unsuccessful*.
  2. The node has an ancestor labeled $s' \models \mathsf{EG}I$ for some s' with $s' \preceq s$. In this case, the node is considered *successful*.
  3. $s \in I$ and $post(s)$ is empty. In this case, the node is considered *successful*.
- From a non-leaf node labeled $s \models \mathsf{EG}I$ we create a child labeled $s' \models \mathsf{EG}I$ for each state $s' \in maxpost(s)$.
- The algorithm answers "yes" if a successful node is encountered, otherwise it answers "no".

Simulation Relations

- The problem: whether a well-structured system is simulated by a finite transition system.
- Given two transition systems $\mathcal{L}_1 = \langle S_1, \delta_1 \rangle$ and $\mathcal{L}_2 = \langle S_2, \delta_2 \rangle$, we say that a relation $\mathcal{R} \subseteq S_1 \times S_2$ is a *simulation* (of $\mathcal{L}_1$ by $\mathcal{L}_2$) if for each $\langle s_1, s_2 \rangle \in \mathcal{R}$, $s_1' \in S_1$, and $\lambda \in \Lambda$, if $s_1 \xrightarrow{\lambda} s_1'$ then there exists $s_2' \in S_2$ such that $s_2 \xrightarrow{\lambda} s_2'$ and $\langle s_1', s_2' \rangle \in \mathcal{R}$.
- For $s_1 \in S_1$ and $s_2 \in S_2$, we say that s_1 is *simulated* by s_2, denoted $s_1 \sqsubseteq s_2$, if there is a simulation $\mathcal{R}$ of $\mathcal{L}_1$ by $\mathcal{L}_2$ such that $\langle s_1, s_2 \rangle \in \mathcal{R}$.
- A transition system is said to be *intersection effective* if $min(\mathcal{C}(s_1) \cap \mathcal{C}(s_2))$ is computable for any states s_1 and s_2.

Simulation Relations (cont.)

Theorem 6.2. For a state s in an intersection effective well-structured transition system and a state q in a finite transition system, it is decidable whether $s \sqsubseteq q$.

- The idea is to calculate the set of pairs $\langle s, q\rangle$ of states such that $s \not\sqsubseteq q$.
- We observe that for each q, the set $\{s \mid \ s \not\sqsubseteq q\}$ is an ideal.
- This allows us to compute the set by a fixpoint iteration analogous to that used for the reachability problem.

Simulation Relations (cont.)

- For each state q of the finite transition system, we define a sequence $I_0^q, I_1^q, I_2^q, \ldots$, where $I_0^q = \emptyset$, and $s \in I_{j+1}^q$ if and only if either
  - $s \in I_j^q$; or
  - there are λ and s' such that $s \xrightarrow{\lambda} s'$ and for all q' if $q \xrightarrow{\lambda} q'$ then $s' \in I_j^{q'}$.
- Intuitively, with $I_0^q = \emptyset$, I_j^q denotes the set of states (in the infinite transition system), which q can simulate at most $j-1$ steps, for $j > 0$.
- It is clear that I_j^q is an ideal and that $I_0^q \subseteq I_1^q \subseteq I_2^q \subseteq \cdots$.
- By Lemma 3.2, it follows that there is a k such that $I_{k+1}^q = I_k^q$ for all q, and $s \not\sqsubseteq q$ iff $s \in I_k^q$.

Simulation Relations (cont.)

- We represent I_j^q by the canonical set $M_j^q = min(I_j^q)$, where $M_0^q = \emptyset$, and

$$M_{j+1}^q = \bigcup_{\lambda} minpre_{\lambda}\left(\bigcap_{q' \in post_{\lambda}(q)} M_j^{q'}\right)$$

- Note that M_{j+1}^q can be computed from M_j^q for intersection effective well-structured transition systems.
- We iterate until we reach a k such that $M_{k+1}^q \equiv M_k^q$.
- To decide whether $s \not\sqsubseteq q$ we check if $\exists s' \preceq s$ such that $s' \in M_k^q$.