Initiation à la Vérification

Stefan Schwoon

Winter 2012/13
Contact details

Name: Stefan Schwoon

Email: schwoon@lsv.ens-cachan.fr

Personal webpage: http://www.lsv.ens-cachan.fr/~schwoon

Course webpage: see MPRI wiki
Program for the second half

First half of the course:

- CTL/CTL*/LTL, model-checking algorithms
- for finite-state systems
- Algorithms linear in number of states

Second half of the course:

- More refined verification algorithms
- Infinite-state model checking
State-space explosion

CTL*/LTL model checking linear in the size of the (low-level) transition system.

Good, but... the transition system may still be of very large size, in fact exponentially larger than the underlying high-level description.

Examples:

Data: TS with variables $v \in \mathcal{V}$ with domains $D_v$ → low-level states $\prod_{v \in \mathcal{V}} D_v$

Concurrency: Parallel composition of components with states $S_i$ → low-level states $\prod_{i=1}^{n} S_i$

Phenomenon popularly called state-space explosion.
Fighting state-space explosion

Many techniques have been developed to alleviate this problem. We will look at some representative examples:

Compression: represent sets of states concisely, handle many states at once

Reduction: identify “equivalent” computations, examine only one representative

Abstraction: throw away information, obtaining a smaller system, then repair inaccuracies
Infinite-state model checking: vast field, even more techniques.

Reasons for infinite states: data, control, unknown parameters

Problems with (un-)decidability

Will take a glimpse at two computation models:

- **Petri nets**: modelling of concurrency aspects
- **Pushdown systems**: sequential, adequate to model procedure calls
Petri nets
Literature about Petri Nets

For more in-depth coverage, a lot of literature about Petri nets is available, for instance:


Internet resources: Petri Nets World
http://www.informatik.uni-hamburg.de/TGI/PetriNets/
Petri nets

Petri nets are a basic model of parallel and distributed systems, designed by Carl Adam Petri in 1962. The basic idea is to describe state changes in a system with transitions.

Petri nets contain places and transitions that may be connected by directed arcs.

Places symbolise states, conditions, or resources that need to be met/be available before an action can be carried out.

Transitions symbolise actions.
Behaviour of Petri nets

Places may contain tokens that may move to other places by executing ("firing") actions.

A token on a place means that the corresponding condition is fulfilled or that a resource is available:

In the example, transition $t$ may "fire" if there are tokens on places $s_1$ and $s_3$. Firing $t$ will remove those tokens and place new tokens on $s_2$ and $s_4$. 
A Petri net is a tuple $N = \langle P, T, F, W, m_0 \rangle$, where

- $P$ is a finite set of places,
- $T$ is a finite set of transitions,
- the places $P$ and transitions $T$ are disjoint ($P \cap T = \emptyset$),
- $F \subseteq (P \times T) \cup (T \times P)$ is the flow relation,
- $W: ((P \times T) \cup (T \times P)) \rightarrow \mathbb{N}$ is the arc weight mapping (where $W(f) = 0$ for all $f \notin F$, and $W(f) > 0$ for all $f \in F$), and
- $m_0: P \rightarrow \mathbb{N}$ is the initial marking representing the initial distribution of tokens.
Semantics

Let $N = \langle P, T, F, W, m_0 \rangle$ be a Petri net. We associate with it the transition system $\mathcal{M} = \langle S, \Sigma, \Delta, I, AP, \ell \rangle$, where:

\[
S = \{ m \mid m : P \to \mathbb{N} \}, \ I = \{ m_0 \}
\]

\[
\Sigma = T
\]

\[
\Delta = \{ (m, t, m') \mid \forall p \in P : m(p) \geq W(p, t) \land m'(p) = m(p) - W(p, t) + W(t, p) \}
\]

\[
AP = P, \ \ell(m) = \{ p \in P \mid m(p) > 0 \}
\]

When $(m, t, m') \in \Delta$, we say that $t$ is enabled in $m$ and that its firing produces the successor marking $m'$; we also write $m \xrightarrow{t} m'$. 

Petri nets: Remarks

If $\langle p, t \rangle \in F$ for a transition $t$ and a place $p$, then $p$ is an input place of $t$.

If $\langle t, p \rangle \in F$ for a transition $t$ and a place $p$, then $p$ is an output place of $t$.

Let $a \in P \cup T$. The set $\bullet a = \{a' \mid \langle a', a \rangle \in F\}$ is called the pre-set of $a$, and the set $a^\bullet = \{a' \mid \langle a, a' \rangle \in F\}$ is its post-set.

When drawing a Petri net, we usually omit arc weights of 1. Also, we may either denote tokens on a place either by black circles, or by a number.
Example: Dining philosophers

There are philosophers sitting around a round table.

There are forks on the table, one between each pair of philosophers.

The philosophers want to eat spaghetti from a large bowl in the center of the table.
Dining philosophers: Petri net
Synchronization by rendez-vous

Assume that we have a number of components with local actions and actions \( !m \) (send message \( m \)) and \( ?m \) (receive message \( m \)).

Transition into Petri net:

Places = union of local states

Transitions:

– for local actions \( (p, a, p') \) build a Petri transition \( t \) labelled with \( a \) and \( \bullet t = \{p\}, t^\bullet = \{p'\} \);

– for pairs of actions \( (p, !m, p') \) and \( (q, ?m, q') \) build a Petri transition \( t \) labelled with \( m \) and \( \bullet t = \{p, q\}, t^\bullet = \{p', q'\} \).

Similar translations possible for other models discussed in the course (asynchronous product, TS with variables, …)
Notation for markings

Often we will fix an order on the places (e.g., matching the place numbering), and write, e.g., \( m_0 = \langle 2, 5, 0 \rangle \) instead.

When no place contains more than one token, markings are in fact sets, in which case we often use set notation and write instead \( m_0 = \{ p_5, p_7, p_8 \} \).

Alternatively, we could denote a marking as a multiset, e.g.,

\[
m_0 = \{ p_1, p_1, p_2, p_2, p_2, p_2, p_2 \}.
\]
Reachable markings

Let \( m \) be a marking of a Petri net \( N = \langle P, T, F, W, m_0 \rangle \).

The set of markings reachable from \( m \) (the reachability set of \( m \), written \( \text{reach}(m) \)), is the smallest set of markings such that:

1. \( m \in \text{reach}(m) \), and

2. if \( m' \xrightarrow{t} m'' \) for some \( t \in T \), \( m' \in \text{reach}(m) \), then \( m'' \in \text{reach}(m) \).

The set of reachable markings \( \text{reach}(N) \) of a net \( N = \langle P, T, F, W, m_0 \rangle \) is defined to be \( \text{reach}(m_0) \).
Reachability Graph

Let $N = \langle P, T, F, W, m_0 \rangle$ be a Petri net with associated transition system $\mathcal{M} = \langle S, \Sigma, \Delta, I, AP, \ell \rangle$.

The reachability graph of $N$ is the rooted, directed graph $G = \langle S', \Delta', m_0 \rangle$, where $S'$ and $\Delta'$ are the restrictions of $S$ and $\Delta$ to $\text{reach}(N)$.

The reachability graph can be constructed in iterative fashion, starting with the initial marking and then adding, step for step, all reachable markings.
**k-Safeness**

**Definition:** Let $N$ be a net. If no reachable marking of $N$ can contain more than $k$ tokens in any place (where $k \geq 0$ is some constant), then $N$ is said to be $k$-safe.

**Example:** The following net is 1-safe.

Other example: the nets resulting from translating synchronous rendez-vous
\textbf{$k$-safeness and Termination}

A $k$-safe net has at most $(k + 1)^{|P|}$ reachable markings; for 1-safe nets, the limit is $2^{|P|}$.

In this case, there are finitely many reachable markings, and the construction of the reachability graph terminates.

On the other hand, if a net is not $k$-safe for any $k$, then there are infinitely many markings, and the construction will not terminate.

In the following, we will first consider 1-safe nets.
Reachability problem (for 1-safe nets)

Let $N$ be a Petri net and $m$ be a marking. The \textit{reachability problem} for $N$, $m$ is to determine whether $m \in \text{reach}(N)$.

\textbf{Theorem:} The reachability problem for 1-safe Petri nets is PSPACE-complete.

\textbf{Proof:} (sketch)
upper bound: non-deterministically simulate net for at most $2^{|P|}$ steps;
hardness by reduction from QBF.

\textbf{Corollary:} Given a 1-safe net $N$ and a place $p$, it is PSPACE-complete to determine whether $\text{reach}(N)$ contains a marking $m$ such that $m(p) = 1$. 
Algorithms for the reachability problem

The most straightforward way to solve the reachability problem on Petri nets is to construct the reachability graph. However, this can be very inefficient:

If there are $n$ such components, then the reachability graph has size $2^n$. 
In fact, the reachability graph does not take advantage of the concurrent nature of a Petri net.

We shall study a method more adapted to concurrent systems. It constructs a concise representation of the reachable markings. This representation is called unfolding.

Given the unfolding of $N$, it is NP-complete to determine whether a given marking is reachable in $N$. One can thus take advantage of the advances made in SAT-checking.
Petri nets: Unfoldings
Unfoldings are a data structure that represents the behaviour of a Petri net.

We will study it for 1-safe nets.

Unfoldings represent a trade-off in terms of time/space requirements; their size is in between that of a net and its reachability graph, and checking whether a marking is reachable becomes easier than for the net, but more difficult than from the reachability graph.
The unfolding of a Petri net $\mathcal{P}$ (or, a prefix of the same) is an infinite *acyclic* Petri net $\mathcal{U}$. We shall be interested in computing a finite prefix $\mathcal{Q}$ of $\mathcal{U}$.

**Remark:** In the following, we call the places of $\mathcal{Q}$ *conditions*, the transitions of $\mathcal{Q}$ *events*. This merely serves to better distinguish the elements of $\mathcal{P}$ and $\mathcal{Q}$, functionally they are the same!
Every condition of $Q$ is labelled by a place of $P$, every event of $Q$ by a transition of $P$.

Every event $e$ is of the form $(S, t)$, where $S$ is the preset of $e$ and $t$ the label of $e$.

Let $S$ be a set of conditions. $B(S)$ denotes the set of places labelling the elements of $S$.

Every condition has exactly one incoming arc.
Unfolding construction for Petri nets

We first discuss the construction of $U$ (possibly infinite).

1. Let $m_0$ be the initial marking of $P$. Then the initial marking of $U$ contains exactly one condition for each place in $m_0$.

2. Let $S$ the subset of a reachable marking in $U$. Let $B(S) = \bullet t$ for some transition $t$ of $P$ such that $(S, t)$ is not yet contained in $U$.

2a. If no such pair $(S, t)$ exists, we are done.

2b. Add the event $e := (S, t)$ to the prefix (with $S$ as preset and label $t$). Moreover, extend the prefix by one condition for every output place of $t$ and make it an output place of $e$. 
Example 1: Petri net...
... and a possible prefix of the unfolding
We shall now amend the construction so that it prunes certain branches of the unfolding, creating a finite prefix.

More precisely, certain events will be called cutoffs. These events lead to markings that we have already seen.
Prefix construction for Petri nets

1. Let $m_0$ be the initial marking of $\mathcal{P}$. Then the initial marking of $\mathcal{Q}$ contains exactly one condition for each place in $m_0$. We set $\mathcal{E} := \{m_0\}$.

2. Let $S$ the subset of a reachable marking in $\mathcal{Q}$. Suppose that no element of $S$ is the output place of a cutoff event. Let $B(S) = \bullet t$ for some transition $t$ of $\mathcal{P}$ such that $(S, t)$ is not yet contained in $\mathcal{Q}$.

2a. If no such pair $(S, t)$ exists, we are done.

2b. Add the event $e := (S, t)$ to the prefix (with $S$ as preset and label $t$). Moreover, extend the prefix by one condition for every output place of $t$ and make it an output place of $e$.

2c. We associate with $e$ a marking $m_e$ (which is reachable in $\mathcal{P}$) (see below). If $m_e \in \mathcal{E}$, then $e$ is a cutoff. Otherwise $\mathcal{E} := \mathcal{E} \cup \{m_e\}$. 


Determining $m_e$

For the event $e = (S, t)$, we determine $m_e$, a marking of $P$, as follows:

**Idea:** $m_e$ is the label of the marking obtained by making the “minimal” effort to fire $e$.

Let $x, y$ be two nodes (conditions or events) in $Q$. Let $<$ be the smallest partial order where $x < y$ if there is an edge from $x$ to $y$.

Let $x$ be a node of $Q$. We define $\lfloor x \rfloor := \{ y \mid y \leq x \}$.

Let $m_e$ be the labels of the marking obtained by firing the events of $[e]$ (in any order). **Note:** Such a firing sequence exists due to the properties of $S$. 
Let $\mathcal{P}$ be a Petri net and $\mathcal{Q}$ a prefix of its unfolding $\mathcal{U}$ with labelling $B$. We call $\mathcal{Q}$ complete if it satisfies the following property:

A marking $m$ is reachable in $\mathcal{P}$ iff a marking $m'$ with $B(m') = m$ is reachable in $\mathcal{Q}$.

Thus, if $\mathcal{Q}$ is complete, we can decide reachability in $\mathcal{P}$ by examining $\mathcal{Q}$.

Unfortunately, the algorithm given previously does not always produce a complete prefix. Indeed, its shape depends on the order in which events are added. We shall discuss an example that demonstrates this effect.
Example 2

Consider the following Petri net:
In Example 2 the marking \{p\} is reachable, e.g. by firing \(A B T\).

The net can also reach the marking \{e, f\} by firing either \(AC\) or \(BD\), and then return by firing \(EF\) to the initial marking.

We shall see that a prefix generated according to depth-first order will “overlook” the transition \(T\).
Depth-first order generated the prefix shown below (order and cutoffs indicated in red):