Invariant checking for rewriting systems over nested words with data

Ahmed Bouajjani, Cezara Drăgoi, Yan Jurski, and Mihaela Sighireanu

LIAFA, CNRS & University Paris 7, France
175, rue du Chevaleret, 75013 Paris, France
E-mail: cezara.dragoi@liafa.jussieu.fr
Dynamic Networks of Processes

\[ P_1 \parallel P_2 \parallel \ldots \parallel P_n \]

- \( n \) not fixed: dynamic creation and deletion of processes,
- \( P_i \) are infinite state processes that:
  - manipulate data over unbounded complex domains (stacks, lists, multi-dimensional arrays...),
  - recursive procedure calls
- \( P_i \) synchronize: rendez-vous, broadcast, shared variables.

Challenge: two sources of infinity
- number of processes
- data domains
Dynamic Networks of Processes: Safety properties

Verification approach:

- pre/post condition reasoning
- invariants checking

Check that:

- $post(Inv) \subseteq Inv$
- $Inv \subseteq Good$
Verification approach:
- pre/post condition reasoning
- invariants checking
  - Check that:
    - $\text{post}(\text{Inv}) \subseteq \text{Inv}$
    - $\text{Inv} \subseteq \text{Good}$

We introduce a logic
- Nested Data Word Logic to represent sets of states
- Nested Data Word Rewriting System to model the transitions of the network
The domain of Nested Data Words
Nested Data Word Logic
Decidability result
Rewriting Systems over Nested Words
Application to verification
Representing configurations as Nested Data Words

\[ \Sigma = \{ P, M, A, B, C, G \} \quad \mathbb{D} = \mathbb{Z} \quad \text{NDW}_1 \]

\[ \begin{align*}
\sigma[6] &= (P, 7, [0 \mapsto (A, 7), 1 \mapsto (C, 5)]) \\
\sigma[6, 0] &= (A, 7) \\
\sigma[9] &= (G, 1, \epsilon)
\end{align*} \]
Representing configurations as Nested Data Words

We define NDW to be \( \bigcup_{k \geq 0} \text{NDW}_k \) where:

- \( \text{NDW}_1 : \{w : \mathbb{N} \to \Sigma \times \mathbb{D}\} \),
- \( \text{NDW}_k = \{w : \mathbb{N} \to \Sigma \times \mathbb{D} \times \text{NDW}_{k-1}\} \) if \( k > 0 \).

A word in \( \text{NDW}_2 \)

- Assume \( \mathbb{D} \) the data domain for the values of the (local/global) variables

\[0 \ 1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \ 8 \ 9 \ 10 \ 11\]

\[\begin{array}{cccccccccccc}
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \\
\hline
P 1 & P 3 & P 2 & P 7 & P 5 & G 1 & G 3 & G 2 \\
0 & A 3 & 0 & A 2 & A 7 & \\
1 & 1 & & & & C 5 &
\end{array}\]
The domain of Nested Data Words

Nested Data Word Logic

Decidability result

Rewriting Systems over Nested Words

Application to verification
assume $\text{FO}(D, O, P)$ a first order logic on $D$, with operations in $O$ and predicates in $P$

NDWL is a logic on NDW parametrized by $\text{FO}(D, O, P)$
Label predicates

\[ w \in \text{NDW}_2 \]

\[ P(\gamma[x]) \land G(\gamma[t]) \]
Label predicates

\[ w \in \text{NDW}_2 \]

\[
P(\gamma[x]) \land G(\gamma[t])
\]

\[
A(\gamma[x, x']) \land C(\gamma[z, z'])
\]
Label predicates and index constraints

\[ w \in \text{NDW}_2 \]

\[ P(\gamma[x]) \land G(\gamma[t]) \land x < z \]

\[ A(\gamma[x, x']) \land C(\gamma[z, z']) \]
Label predicates and index constraints

\[ w \in \text{NDW}_2 \]

\[ P(\gamma[x]) \land G(\gamma[t]) \land x < z \]

\[ A(\gamma[x, x']) \land C(\gamma[z, z']) \land x' < z' \]
Label predicates and index constraints

$$w \in \text{NDW}_2$$

$$idx(x, \gamma) \land P(\gamma[x]) \land G(\gamma[t]) \land x < z$$

$$idx(x', \gamma[x]) \land A(\gamma[x, x']) \land C(\gamma[z, z']) \land x' < z'$$
**Data constraints**

\[ w \in \text{NDW}_2 \]

\[ \text{idx}(x, \gamma) \land P(\gamma[x]) \land G(\gamma[t]) \land x < z \]

\[ \text{idx}(x', \gamma[x]) \land A(\gamma[x, x']) \land C(\gamma[z, z']) \land x' < z' \]

\[ v(\gamma[x]) = 3 \land v(\gamma[z, z']) = 5 \]

\[ v(\gamma[x]) + d + v(\gamma[z, z']) \geq 17 \]
Data constraints

\[ w \in \text{NDW}_2 \]

\[ \text{idx}(x, \gamma) \land P(\gamma[x]) \land G(\gamma[t]) \land x < z \]

\[ \text{idx}(x', \gamma[x]) \land A(\gamma[x, x']) \land C(\gamma[z, z']) \land x' < z' \]

\[ \nu(\gamma[x]) = 3 \land \nu(\gamma[z, z']) = 5 \]

\[ \nu(\gamma[x]) + d + \nu(\gamma[z, z']) \geq 17 \]

\[ \delta(\gamma[x]) \neq \delta(\gamma[z]) \]
Outline

- The domain of Nested Data Words
- Nested Data Word Logic
- Decidability result
- Rewriting Systems over Nested Words
- Application to verification
The satisfiability problem of NDWL is undecidable

- even if we restrict to NDW₁ for very simple data logics such as \((\mathbb{N}, =)\), the fragment \(\forall^* \exists^*\) is undecidable
Decidability

We define $\Sigma_2$ as the smallest set of formulas closed under disjunction and conjunction, which contains all the closed formulas of the form:

$$
\exists_{\leq k}^* \forall_k^* \exists_{\leq k-1}^* \forall_{k-1}^* \ldots \exists_1^* \forall_1^* \{\exists d, \forall d\}^*. \phi
$$

$\phi$ is a quantifier-free formula in NDWL.
Decidability

We define $\Theta_1$ the smallest set of formulas in $\Sigma_2$ closed under disjunction and conjunction that contains

$$\{\exists^*_k, \forall^*_k\} \ \{\exists^*_{k-1}, \forall^*_{k-1}\} \ \cdots \ \{\exists^*_1, \forall^*_1\} \ \{\exists_d, \forall_d\}^*. \phi$$

$\phi$ is a quantifier-free formula in NDWL.
Theorem

The satisfiability of $\text{NDWL}(\text{FO}(\mathcal{D}, \Omega, \Xi))$ formulas in the fragment $\Sigma_2$ can be reduced to the satisfiability of a formula in $\text{FO}(\mathcal{D}, \Omega, \Xi)$.

Remark

The complexity of the reduction procedure is $\text{NP}$ when the number of universally quantified variables is fixed.
Properties of Dynamic Networks

Stack: “in successive calls of some procedure the values of its parameters decreases”

\[ \forall y \forall y', z'. \ P(\gamma[y]) \land y' < z' \land \text{id}_x(y', \gamma[x]) \land \text{id}_x(z', \gamma[y]) \Rightarrow \nu(\gamma[y, y']) > \nu(\gamma[y, z']) \]

Data: “some local process variable is greater then 2”

\[ \exists x. \text{id}_x(x, \gamma) \land \nu(\gamma[x]) \geq 2 \]

Control structure: “all processes are running”

\[ \forall y \exists x'. \text{id}_x(y, \gamma) \land P(\gamma[y]) \Rightarrow \text{id}_x(x', \gamma[y]) \]
Outline

- Nested Data Words
- Nested Data Word Logic
- Decidability result
- Rewriting Systems over Nested Words
- Application to verification
Transitions of the network:

- create/delete process
- local behavior of processes
- communication and synchronization between processes
  - locks
  - shared variables
  - rendez-vous, e.g. `notify()`
  - broadcast, e.g. `notifyAll()`
R1 : \( P \xrightarrow{m} P P : \quad \nu(\gamma[x_1]) = 2 \quad \land \quad \nu(\gamma'[y_1]) = \nu(\gamma[x_1]) \land \delta(\gamma'[y_1]) = \delta(\gamma[x_1})();
\nu(\gamma'[y_2]) = 2\nu(\gamma[x_1])\)
R2: $C \leftrightarrow_{s} B D : P(\gamma[x]) \land \nu(\gamma[x]) = 7/\nu(\gamma[x, x_1]) = \nu(\gamma'[x, y_1]) \land \nu(\gamma'[x, y_2]) = 1$
\[ R3 : \quad P \rightarrow P : \quad v(\gamma[u]) \geq 2 \quad / \quad v(\gamma'[u]) = v(\gamma[u]) + 1 \]
\( S=(\Sigma, \Delta) \) is a NDW-RS where

- \( \Sigma \) is a finite set of labels,
- \( \Delta \) is a finite set of rewriting rules

\[
(\vec{x}, \vec{A}) \mapsto \# (\vec{y}, \vec{B}) : \varphi_g / \varphi_a \mid (\vec{u}, \vec{C}) \mapsto \vec{D} : \psi_g / \psi_a
\]

For every rule \( R \in \text{NDW-RS} \) its semantics is given by a NDW formula:

\[
reach_R(\gamma, \gamma') = \varphi_g \land \varphi_a \land order(\vec{x}, \vec{y}) \land \psi_g \land \psi_a
\]
Outline

- Nested Data Words
- Nested Data Word Logic
- Decidability result
- Rewriting Systems over Nested Words
- Application to verification
Invariance checking

We denote by NDW-RS[\Sigma_2] the class of NDW-RS s.t. for every rule \( \varphi_g, \varphi_a \in \Sigma_2 \) and \( \psi_g, \psi_a \) in \( \Theta_1 \).

**Proposition**

For every rule of a rewriting system in NDW-RS[\Sigma_2], the associated NDWL formula is in the fragment \( \Sigma_2 \).

**Theorem**

The problem whether a closed formula \( \varphi(\gamma) \in \Theta_1 \) is an inductive invariant of \((S, \varphi_{init})\), where \( S = (\Sigma, \Delta) \) is in NDW-RS[\Sigma_2] and \( \varphi_{init} \) is a closed formula in \( \Sigma_2 \) is decidable.
Using NDW−RS we verified:

- mutual exclusion algorithms: Burns, Richard - Agrawala
- Java-like code with recursive procedure calls and synchronization by monitors
Conclusions

Using NDW−RS we verified:

- mutual exclusion algorithms : Burns, Richard - Agrawala
- Java-like code with recursive procedure calls and synchronization by monitors

Thank You!