GRAPH-REWITING PETRI NETS

ICGT@STAF 2018, Toulouse

Géza Kulcsár, Malte Lochau, Andy Schürr
EXAMPLE: WSN TOPOLOGY CONTROL

- Mobile sensor nodes
- Wireless communication channels
- Short active edge
- Long active edge
- Unclassified edge
GRAPH-REWRITING PROCESSES

Environmental event: short edge becomes long edge

Match unclassified long edge

Match two adjacent short edges

“Inactivate long edge”

“Activate long edge”

“Loop forever”
OPEN CHALLENGES IN CONTROLLED GRAPH REWRITING

- Control-flow specification for graph-rewriting processes to restrict the orderings of graph-rewriting rule applications
- Data-flow specification for graph-rewriting processes to restrict the matches of graph-rewriting rule applications
- Control-flow and data-flow specification for graph-rewriting processes with concurrent rule composition and synchronization of rule applications
- Techniques for automated analysis of correctness properties for controlled graph-rewriting process specifications
  - „TC must preserve connectedness of input topology graphs“
  - „TC should eventually inactivate redundant long edges“
  - „TC must not deadlock due to concurrent interactions with environmental events“

Graph-Rewriting Petri Nets (GPN) = Coloured Petri Nets + DPO Graph Rewriting
PETRI NETS, VISUALLY

Token → Pre-Place → Arc → Transition → Arc → Post-Place → Loops → Sequence → Choice → Concurrency
PETRI NETS, FORMALLY

- A Petri Net $N$ consists of a set of places $P$ and a set of transitions $T$.
- A marking $M \in \mathbb{N}^P$ of $N$ is a multiset, where $M(p) \geq 0$ denotes the number of tokens on place $p$.
- A step $M \rightarrow M'$ consists of a multiset of transitions $X \in \mathbb{N}^T$.
- A marking $M$ is reachable from initial marking $M_0$ iff there exists a sequence of steps $M_0 \xrightarrow{X_0} \cdots \xrightarrow{X} M$.
- Two transitions $t, u \in T$ are concurrent if there exists a reachable marking $M \xrightarrow{\{t, u\}} M'$.
- A Petri Net $N$ is $k$-bounded iff $M(p) \leq k$ for all reachable markings $M$ and all places $p$.
- A Petri Net $N$ is live if for all reachable markings $M \rightarrow M'$. 
COLOURED PETRI NETS [JENSEN & CHRISTENSEN 2008]

- Places are **typed** over sets $\Sigma$ of **colours**
- Tokens carry **data**, typed over sets of **colours**
- Transitions and arcs are augmented with **inscriptions** over typed **variables** $v \in V$

\[ \begin{align*}
C(p) \in \Sigma & \quad \Gamma(t) \quad C(p') \in \Sigma \\
\sigma \in C(p) & \quad E(a) \quad E(a') \\
\text{pre-place colour set} & \quad \text{transition guard over variables} \quad \text{post-place colour set} \\
\text{token colour} & \quad \text{arc expression binds token data to variables} \quad \text{arc expression binds variable values to token data}
\end{align*} \]
DPO RULE APPLICATION AS GPN TRANSITION

- **GPN colour set** \( \Sigma_{TG} = \{ \text{Obj}(\text{Graph}_{TG}), \text{Mor}(\text{Graph}_{TG}) \} \)
- Variable G is bound to graph \( G \in \text{Obj}(\text{Graph}_{TG}) \) carried by input token \( 1'G \)
- Transition guard \( \rho \) corresponds to the DPO diagram for the application of rule \( \rho : (L \xleftarrow{l} K \xrightarrow{r} R) \) on match \( m \) in \( G \)
- Variable H is bound to output graph \( H \in \text{Obj}(\text{Graph}_{TG}) \) of the rule application and assigned to output token \( 1'H \)
NON-DETERMINISTIC CHOICE
NEGATIVE RULE APPLICATIONS & DETERMINISTIC CHOICE

- Non-applicability of rule
  \[ \rho : (L \xleftarrow{l} K \xrightarrow{r} R) \]

- If-then-else fragment
CONCURRENCY

- Rule application produces **multiple concurrent copies of the output** graph
- Rule application requires **multiple concurrent input graphs**
SUB-GRAPH BINDING AND MATCHING

- Output token $b \in \text{Mor}(\text{Graph}_T_G)$ denote sub-graph bound in output graph $H$
- Input token $b \in \text{Mor}(\text{Graph}_T_G)$ denotes sub-graph to be matched in input graph $G$
GENERIC GPN TRANSITION TEMPLATE

Multiple input sub-graph bindings

Multiple output sub-graph matchings

Multiple concurrent input graphs

Multiple concurrent copies of output graph
GPN SEMANTICS: TWO PERSPECTIVES

GPN language

- A GPN marking $M$ is completed if all tokens are from $\text{Obj}(\text{Graph}_{TG})$
- The language of a GPN is the set of all completed markings reachable from initial completed marking $M_0$

GPN processes

- Let $\mathcal{P} = \{\text{pred} \mid \text{pred} : \text{Obj}(\text{Graph}_{TG}) \to \text{Bool}\}$ be a set of graph predicates
- The processes of a GPN are defined by the LTS $(S, s_0, \rightarrow, \mathcal{P})$ with set of states $S = \mathbb{N}^{\Sigma_{TG}}$, initial state $s_0 = M_0$, step relation $M \rightarrow M'$, and a set of state properties $\mathcal{P}$
WSN CASE STUDY REVISITED

- Centralized model

- Concurrent model

- „TC must preserve connectedness of input topology graphs“ → Invariant property
- „TC should eventually inactivate redundant long edges“ → Fairness property
- „TC must not deadlock …“ → Liveness property
SUMMARY & FUTURE WORK

- GPN provide a visual, very expressive and formally founded modeling language for controlled graph-rewriting processes
- Novel features: explicit notion of concurrency and sub-graph binding/matching

Future Work
- Tool support based on CPN tools
- Further (application-specific) merge-operators
- Notions of expressiveness for GPN languages
- Notions of parallel independence for GPN processes
- Notions of equivalence for GPN processes
RELATED WORK

- First proposal of programmed (a.k.a. controlled) graph grammers [Bunke 1978]
- Formal notion of concurrent graph processes [Corradini et al. 1996, Baldan et al. 1999]
- Denotational characterization of input/output semantics of graph-rewriting processes [Schürr 1996]
- Composable (graph) transformation units [Kreowski et al. 2008]
- Operational semantics of GP language [Plump & Steinert 2009]
- Tool support: PROGRES, GReAT, Fujaba, eMoflon, Henshin…
REFERENCES


REFERENCES

GRAPH-REWRITING INSCRIPTIONS

- Colour set of GPN \( \Sigma_{TG} = \{\text{Obj}(\text{Graph}_{TG}), \text{Mor}(\text{Graph}_{TG})\} \)

- Graph-rewriting inscriptions

  - Finite category of bound variables \( \text{BV}_I \)
  
  - Finite category of free variables \( \text{FV}_I \) s.t. \( \text{BV}_I \subseteq \text{FV}_I \)
  
  - Binding functor \( B_I : \text{BV}_I \rightarrow \text{Graph}_{TG} \) s.t.
    \[ \forall v \in \text{BV}_I : \text{Type}(v) = \text{Type}(B_I(v)) \]

  - A set of categorial properties \( \Phi_I \) for the images of \( \text{FV}_I \)

- An inscription binding is a functor \( B : \text{FV}_I \rightarrow \text{Graph}_{TG} \) s.t.

  - \( B \mid_{\text{BV}_I} = B_I \) and \( \forall v \in \text{FV}_I : \text{Type}(v) = \text{Type}(B(v)) \)

  - \( I(B) \) is satisfied if \( B(\text{FV}_I) \) satisfies \( \Phi_I \)