Reactive graphs in Action

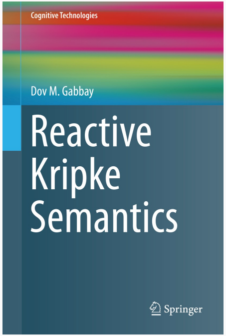
David Tinoco, Alexandre Madeira, José Proença and Manuel A. Martins

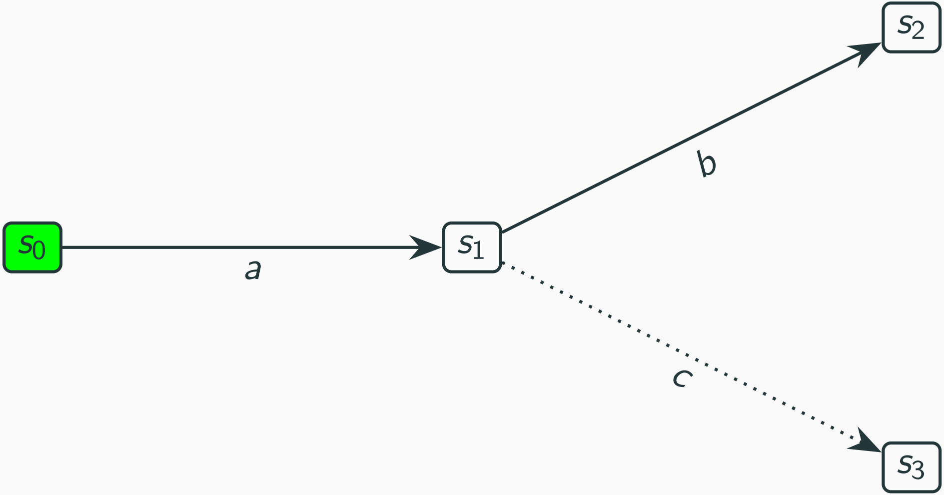
FACS'24 - 20th International Conference in Formal Aspects of Components
Milan, 9-10 September 2024

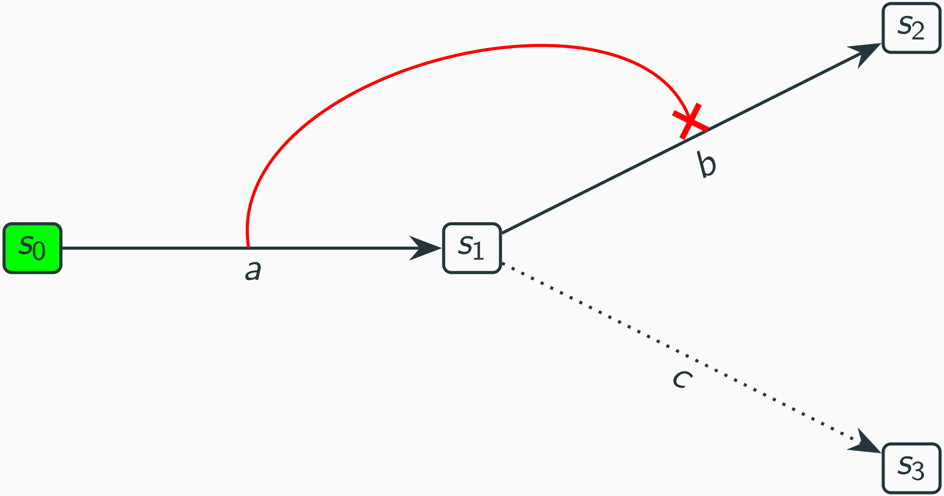
Reactive graphs?

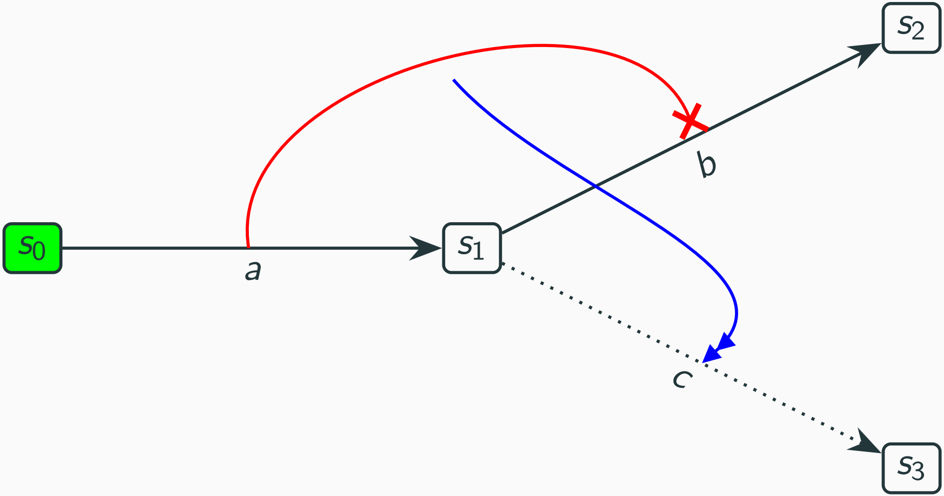
A reconfigurability dimension on LTS?

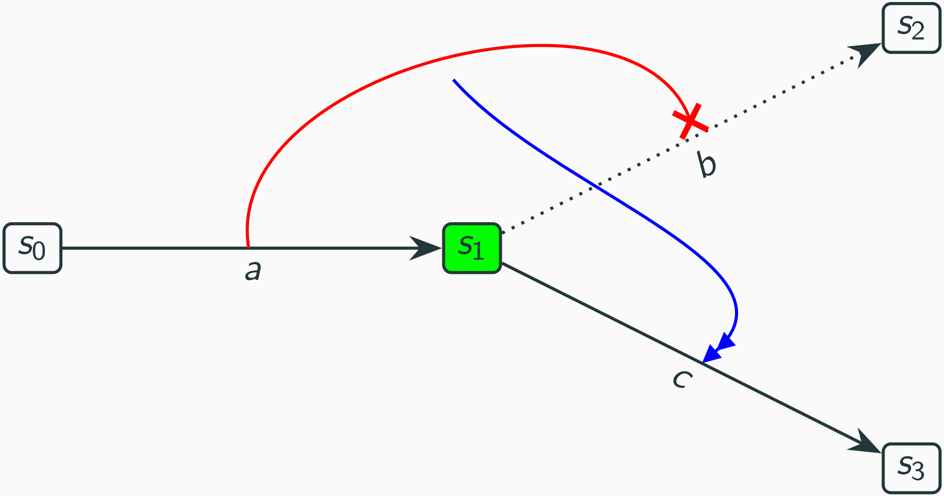
"In computer science the word reactivity has been used to denote systems that react to their environment and are not meant to terminate, as coined by Pnueli and Harel in [104]. In this work the word has a different meaning, reactive systems are history-dependent relational structures, where the accessibility relation is determined not only by the point where one is, but also by the previous transitions"




s_2
b
s_0
a
s_1
c
s_3

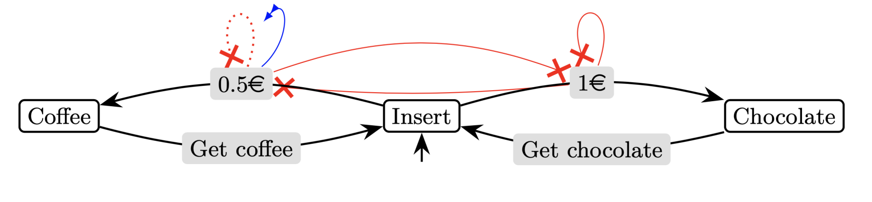

s_0
a
s_1
b
s_2
c
s_3


s_0
a
s_1
b
s_2
c
s_3


s_0
a
s_1
b
s_2
c
s_3

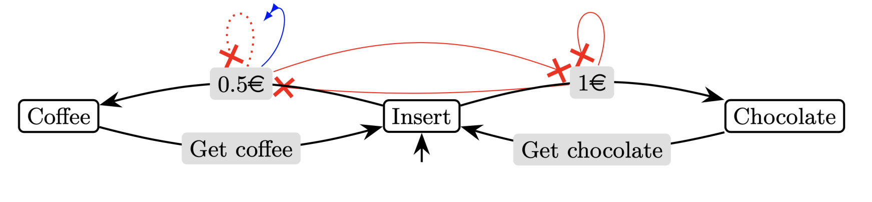
An example

The reactive graph

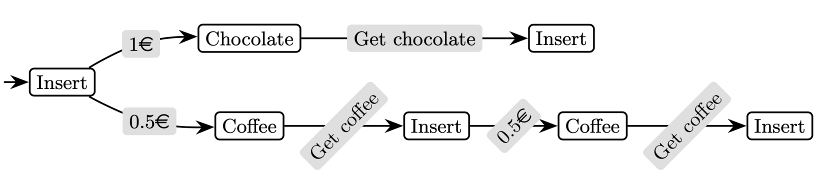


An example

The reactive graph



can be encoded as



Contributions

In this talk we

- formalize a labelled versions of reactive graphs (LRG)

Contributions

In this talk we

- formalize a labelled versions of reactive graphs (LRG)
- introduce a set of constructors to build LRG, including the intrusive product - a composition operator that captures interference of actions within models

Contributions

In this talk we

- formalize a labelled versions of reactive graphs (*LRG*)
- introduce a set of constructors to build *LRG*, including the intrusive product - a composition operator that captures interference of actions within models
- introduce Marge, an animator and analyser for *LRG*

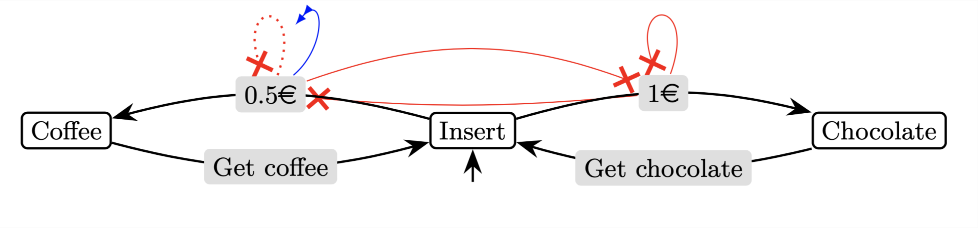
Labelled Reactive Graphs

A labelled version of a reactive graph

A Multi-Actions Reactive Graph is a tuple $M = (W, Act, E, \rightarrow, \twoheadrightarrow, \rightarrow\!\!\!\times, \bar{\cdot}, w_0, \alpha_0)$ where:

- W – states
- Act – actions
- E – edges
- $w_0 \in W$ – initial state;
- $\alpha_0 \subseteq E$ – initially active edges

- $\rightarrow \; \subseteq \; W \times Act \times W$ – ground edges
- $\twoheadrightarrow \; \subseteq \; E \times E$ – activating edges
- $\rightarrow\!\!\!\times \; \subseteq \; E \times E$ – deactivating edges
- $\bar{\cdot} \; : \; E \longrightarrow (\rightarrow \cup \twoheadrightarrow \cup \rightarrow\!\!\!\times)$ – internal details of edges



Semantics

Semantics

Some auxiliary notions

For $e \in E_M$ and a set of active edges $\alpha \subseteq E_M$:

$$\begin{aligned}
\mathsf{from}(e_s) &= \{e \mid \exists e_t \cdot \overline{e} = (e_s, e_t)\} \\
\mathsf{from}^*(e, \alpha) &= \bigcup_{r \in (\mathsf{from}(e) \cap \alpha)} \mathsf{from}^*(r, \alpha \backslash \{e\}) \cup \{r\} \\
\mathsf{on}(e, \alpha) &= \{e_t \mid e_{trg} \in \mathsf{from}^*(e, \alpha) \wedge \exists e_s \cdot \overline{e_{trg}} = (e_s, e_t) \in \twoheadrightarrow\} \\
\mathsf{off}(e, \alpha) &= \{e_t \mid e_{trg} \in \mathsf{from}^*(e, \alpha) \wedge \exists e_s \cdot \overline{e_{trg}} = (e_s, e_t) \in \rightarrow\!\!\times\}
\end{aligned}$$

Semantics

Some auxiliary notions

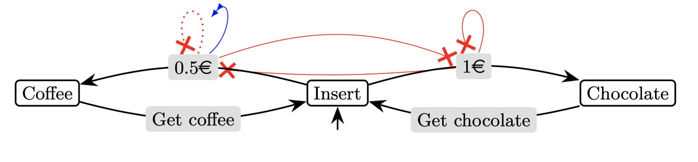
For $e \in E_M$ and a set of active edges $\alpha \subseteq E_M$:

$$\mathsf{from}(e_s) = \{e \mid \exists e_t \cdot \overline{e} = (e_s, e_t)\}$$

$$\mathsf{from}^*(e,\alpha) = \bigcup_{r \in (\mathsf{from}(e) \cap \alpha)} \mathsf{from}^*(r, \alpha \backslash \{e\}) \cup \{r\}$$

$$\mathsf{on}(e,\alpha) = \{e_t \mid e_{trg} \in \mathsf{from}^*(e,\alpha) \wedge \exists e_s \cdot \overline{e_{trg}} = (e_s, e_t) \in \twoheadrightarrow\}$$

$$\mathsf{off}(e,\alpha) = \{e_t \mid e_{trg} \in \mathsf{from}^*(e,\alpha) \wedge \exists e_s \cdot \overline{e_{trg}} = (e_s, e_t) \in \rightarrow\!\!\!\!\times\}$$



- $\mathsf{off}(e_1, \alpha_0) = \{e_1, e_2\}$, where $\overline{e_1} = \langle \mathit{Insert}, 1€, \mathit{Chocolate}\rangle$ and $\overline{e_2} = \langle \mathit{Insert}, 0.5€, \mathit{Coffee}\rangle$.

Semantics

Semantics

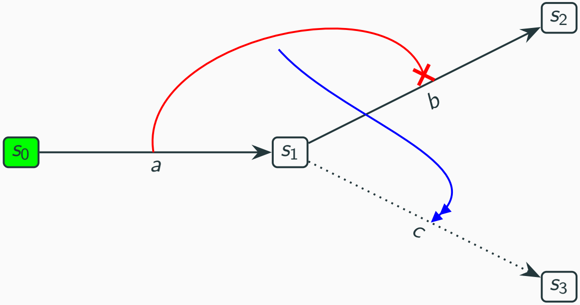
The semantics of a reactive graph M is given by the evolution of the configuration $\langle w_0, \alpha_0 \rangle$ by the rule

$$\frac{\exists e \in \alpha \quad \cdot \quad \overline{e} = w \xrightarrow{a} w' \quad \wedge \quad \alpha' = \big(\alpha \cup \mathsf{on}(e, \alpha)\big) \backslash \mathsf{off}(e, \alpha)}{\langle w, \alpha \rangle \xrightarrow{a}_M \langle w', \alpha' \rangle}$$

Semantics

The semantics of a reactive graph M is given by the evolution of the configuration $\langle w_0, \alpha_0 \rangle$ by the rule

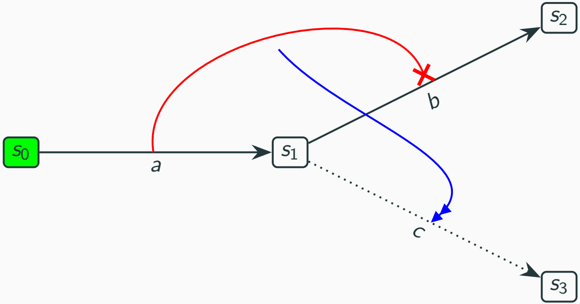
$$\frac{\exists e \in \alpha \quad \cdot \quad \overline{e} = w \xrightarrow{a} w' \quad \wedge \quad \alpha' = \left(\alpha \cup \mathsf{on}(e,\alpha)\right) \backslash \mathsf{off}(e,\alpha)}{\langle w, \alpha \rangle \xrightarrow{a}_M \langle w', \alpha' \rangle}$$



Semantics

The semantics of a reactive graph M is given by the evolution of the configuration $\langle w_0, \alpha_0 \rangle$ by the rule

$$\frac{\exists e \in \alpha \quad \cdot \quad \overline{e} = w \xrightarrow{a} w' \quad \wedge \quad \alpha' = \big(\alpha \cup \mathrm{on}(e,\alpha)\big) \backslash \mathrm{off}(e,\alpha)}{\langle w, \alpha \rangle \xrightarrow{a}_M \langle w', \alpha' \rangle}$$

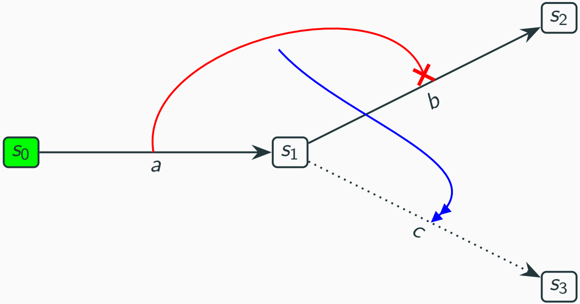


$\left\langle s_0, \left\{ s_0 \xrightarrow{a} s_1, s_1 \xrightarrow{b} s_2, \ldots \right\} \right\rangle \quad \xrightarrow{a}$

Semantics

The semantics of a reactive graph M is given by the evolution of the configuration $\langle w_0, \alpha_0 \rangle$ by the rule

$$\frac{\exists e \in \alpha \quad \cdot \quad \overline{e} = w \xrightarrow{a} w' \quad \wedge \quad \alpha' = \big(\alpha \cup \mathsf{on}(e, \alpha)\big) \backslash \mathsf{off}(e, \alpha)}{\langle w, \alpha \rangle \xrightarrow{a}_M \langle w', \alpha' \rangle}$$

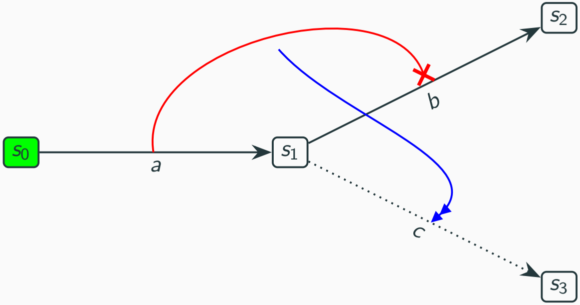


$$\langle s_0, \{s_0 \xrightarrow{a} s_1, s_1 \xrightarrow{b} s_2, \dots\} \rangle \quad \xrightarrow{a} \quad \langle s_1, (\{s_0 \xrightarrow{a} s_1, s_1 \xrightarrow{b} s_2, \dots\} \cup \{s_1 \xrightarrow{c} s_3\}) \setminus \{s_1 \xrightarrow{b} s_2\} \rangle$$

Semantics

The semantics of a $\mathrm{reactive\ graph}$ M is given by the evolution of the configuration $\langle w_0, \alpha_0 \rangle$ by the rule

$$\frac{\exists e \in \alpha \quad \cdot \quad \overline{e} = w \xrightarrow{a} w' \quad \wedge \quad \alpha' = \big(\alpha \cup \mathrm{on}(e, \alpha)\big) \backslash \mathrm{off}(e, \alpha)}{\langle w, \alpha \rangle \xrightarrow{a}_M \langle w', \alpha' \rangle}$$



$$\langle s_0, \{s_0 \xrightarrow{a} s_1, s_1 \xrightarrow{b} s_2, \ldots\} \rangle \quad \xrightarrow{a} \quad \langle s_1, (\{s_0 \xrightarrow{a} s_1, s_1 \xrightarrow{c} s_3, \ldots\} \rangle$$

Relevant Properties on modelling with reactive graphs

The specifier shall be aware of the presence of

Contradictory effects: a transition triggers both the enabling and disabling of the same edge.

Our semantics solve this conflict by giving priority to disabling

Relevant Properties on modelling with reactive graphs

The specifier shall be aware of the presence of

Contradictory effects: a transition triggers both the enabling and disabling of the same edge. Our semantics solve this conflict by giving priority to disabling

Unreachable transitions: even from reachable states, there are transitions that can not be never triggered

Relevant Properties on modelling with reactive graphs

The specifier shall be aware of the presence of

Contradictory effects: a transition triggers both the enabling and disabling of the same edge.
Our semantics solve this conflict by giving priority to disabling

Unreachable transitions: even from reachable states, there are transitions that can not be never triggered

Other properties of reactive graphs can be defined over its behaviour:

$$\rightarrow_M = \bigcup \left\{ \xrightarrow{a} \mid a \in Act \right\}$$

- Deadlocks
- Unreachable states
- Observational equivalence
- ...

Composition of models

Composition of reactive graphs

Goal

Help building complex systems by composing simpler modules.

Composition of reactive graphs

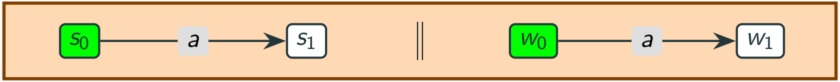
Goal

Help building complex systems by composing simpler modules.

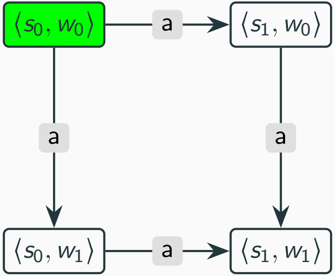
Four products of reactive graphs

- asynchronous and synchronous
- with and without intrusive transitions

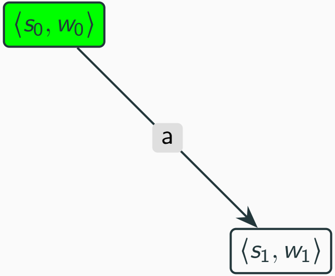
Traditional composition



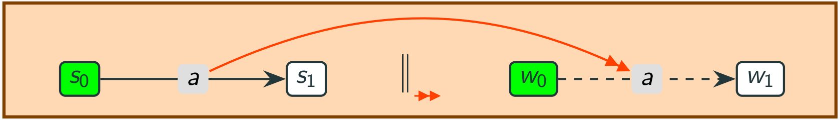
Asynchronous (interleaving)



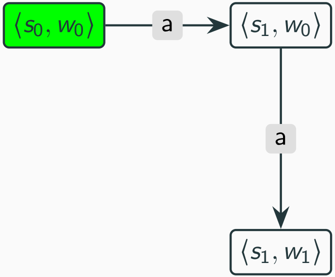
Synchronous (communicating)



Composition with intrusive transitions (example 1)

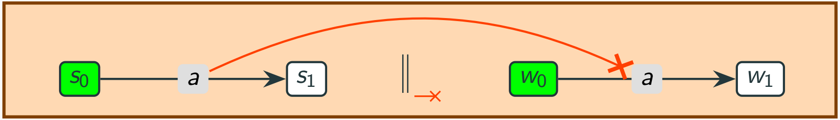


Asynchronous (interleaving)

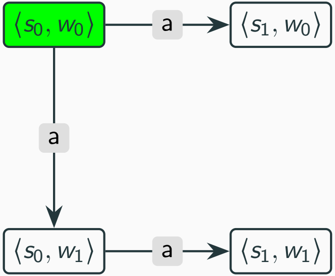


Synchronous (communicating)

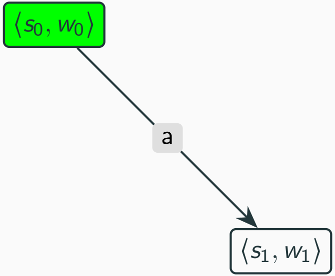
Composition with intrusive transitions (example 2)



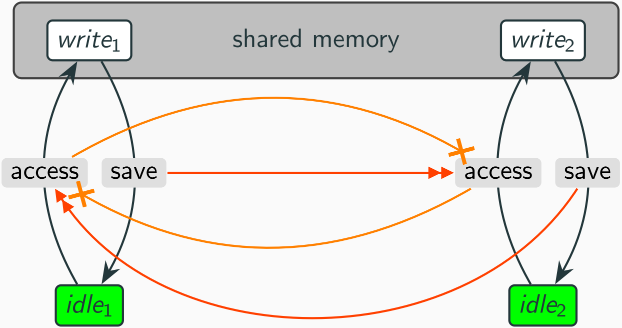
Asynchronous



Synchronous



Intrusive transitions – a different way to communicate



Products – formally

Asynchronous product with intrusive transitions
Given two multi-action reactive graphs M_1, M_2, and $\Gamma^{\oplus}, \Gamma^{\ominus} \subseteq E_1 \times E_2 \cup E_2 \times E_1$ is the set of intrusive edges between M_1 and M_2. The effects produced by edge $e \in E_{M_i}$ in M_i is given for the set follow:

$$\alpha_i(\Gamma^{\oplus}, \Gamma^{\ominus}, e) \quad = \quad \big(\alpha_i \cup \mathrm{on}(e, \alpha_i) \cup \Gamma^{\oplus}(e)\big) \setminus \big(\mathrm{off}(e, \alpha_i) \cup \Gamma^{\ominus}(e)\big)$$

with the rules

$$\frac{\exists\ e \in \alpha_1 \quad \cdot \quad \overline{e} = s_1 \xrightarrow{a} s_1' \quad \wedge \quad \alpha_1' = \big(\alpha_1 \cup \mathrm{on}(e, \alpha_1)\big)\backslash \mathrm{off}(e, \alpha_1) \quad \wedge \quad \alpha_2' = \alpha_2(\Gamma^{\oplus}, \Gamma^{\ominus}, e)}{\langle s_1, \alpha_1 \rangle \mathbin{/\!/}_{\Gamma^{\oplus}, \Gamma^{\ominus}} \langle s_2, \alpha_2 \rangle \quad \xrightarrow{a} \quad \langle s_1', \alpha_1' \rangle \mathbin{/\!/}_{\Gamma^{\oplus}, \Gamma^{\ominus}} \langle s_2, \alpha_2' \rangle}$$

$$\frac{\exists\ e \in \alpha_2 \quad \cdot \quad \overline{e} = s_2 \xrightarrow{a} s_2' \quad \wedge \quad \alpha_2' = \big(\alpha_2 \cup \mathrm{on}(e, \alpha_2)\big)\backslash \mathrm{off}(e, \alpha_2) \quad \wedge \quad \alpha_1' = \alpha_1(\Gamma^{\oplus}, \Gamma^{\ominus}, e)}{\langle s_1, \alpha_1 \rangle \mathbin{/\!/}_{\Gamma^{\oplus}, \Gamma^{\ominus}} \langle s_2, \alpha_2 \rangle \quad \xrightarrow{a} \quad \langle s_1, \alpha_1' \rangle \mathbin{/\!/}_{\Gamma^{\oplus}, \Gamma^{\ominus}} \langle s_2', \alpha_2' \rangle}$$

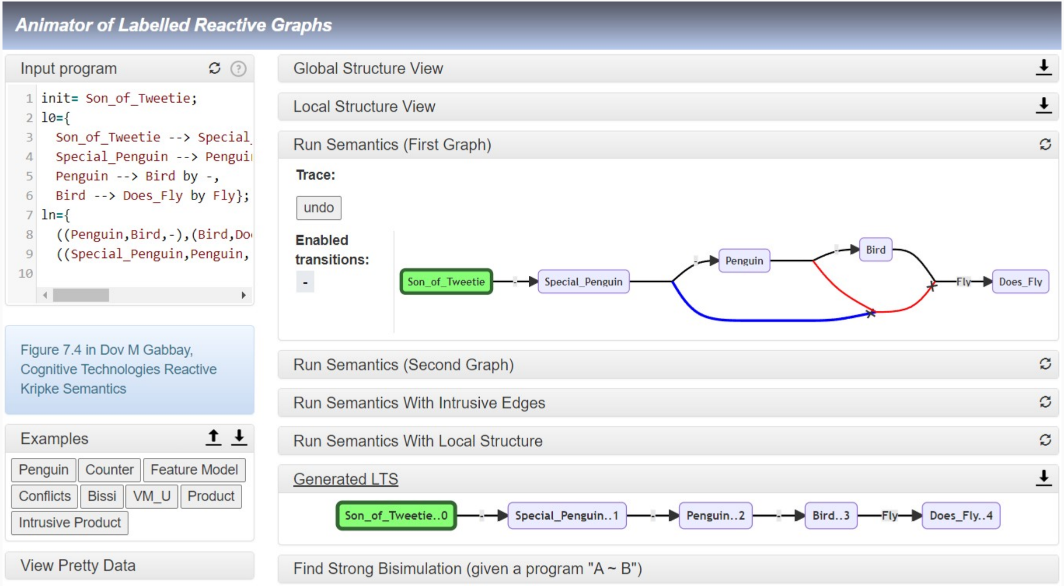
Products – formally

SOS rule for synchronous product with intrusive transitions

$$\frac{\begin{array}{lcl} \exists\ e_1 \in \alpha_1 \cdot \overline{e_1} = s_1 \xrightarrow{a} s_1' & & \alpha_1' = \big(\alpha_1 \cup \mathsf{on}(e_1, \alpha_1) \cup \Gamma^{\oplus}(e_1)\big) \setminus \big(\mathsf{off}(e_1, \alpha_1) \cup \Gamma^{\ominus}(e_1)\big) \\ \exists\ e_2 \in \alpha_2 \cdot \overline{e_2} = s_2 \xrightarrow{a} s_2' & \wedge & \alpha_2' = \big(\alpha_2 \cup \mathsf{on}(e_2, \alpha_2) \cup \Gamma^{\oplus}(e_2)\big) \setminus \big(\mathsf{off}(e_2, \alpha_2) \cup \Gamma^{\ominus}(e_2)\big) \end{array}}{\langle s_1, \alpha_1 \rangle \mathbin{\wr\wr}_{\Gamma^{\oplus}, \Gamma^{\ominus}} \langle s_2, \alpha_2 \rangle \xrightarrow{a} \langle s_1', \alpha_1' \rangle \mathbin{\wr\wr}_{\Gamma^{\oplus}, \Gamma^{\ominus}} \langle s_2', \alpha_2' \rangle}$$

Marge – animator of Multi Action Reactive Graphs

Marge – https://fm-dcc.github.io/MARGe



Marge - overview

- Developed in Scala, using CAOS (generating JavaScript with ScalaJS)
- Static website that loads a compiled JavaScript (fully offline, no server)

Available widgets

- Input
- View graph
- Run step-by-step
- Produce LTS
- Count edges/states
- Compare graphs (bisim)
- Find conflicts
- Find deadlocks
- Run a product

Marge - overview

- Developed in Scala, using CAOS (generating JavaScript with ScalaJS)
- Static website that loads a compiled JavaScript (fully offline, no server)

Available widgets

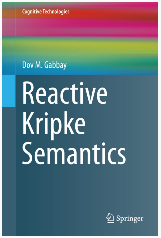
- Input
- View graph
- Run step-by-step
- Produce LTS
- Count edges/states
- Compare graphs (bisim)
- Find conflicts
- Find deadlocks
- Run a product

Demo: `https://fm-dcc.github.io/MARGe`

Wrap up

In this talk:

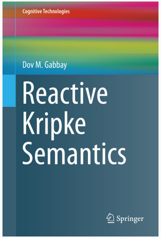
- Analysed **reactive graphs**
- Revisited (operational) semantics
- Animated and increased insights
  - Marge
- Proposed compositions
  (to improve specifications of
  more complex systems)



Wrap up

In this talk:

- Analysed **reactive graphs**
- Revisited (operational) semantics
- Animated and increased insights
  – Marge
- Proposed compositions
  (to improve specifications of
  more complex systems)



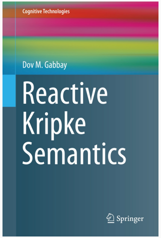
Next steps

- Further improve specifications
  - exploit labels for intrusion
  - support dynamic SPL
  - propose betted DSL in Marge
- support weights
- investigate logics
  - capturing reconfigurations
  - capturing weights
  - support model checking
    (e.g., via mCRL2)

Wrap up

In this talk:

- Analysed **reactive graphs**
- Revisited (operational) semantics
- Animated and increased insights
  – Marge
- Proposed compositions
  (to improve specifications of
  more complex systems)



Next steps

- Further improve specifications
  - exploit labels for intrusion
  - support dynamic SPL
  - propose betted DSL in Marge
- support weights
- investigate logics
  - capturing reconfigurations
  - capturing weights
  - support model checking
    (e.g., via mCRL2)

Thank you!

Reactive graphs in Action

David Tinoco, Alexandre Madeira, José Proença and Manuel A. Martins

FACS'24 - 20th International Conference in Formal Aspects of Components
Milan, 9-10 September 2024