
# Sometimes Less is More:

## Why too many input languages can harm a system

Peter Höfner

June 2019

www.csiro.au

CSIRO

# A Standard Conference Talk

**Theorem:** We developed an efficient algorithm to solve a complex problem using problem X, using the *Brisbane-methodology* and the *FMOz-formalism*.

**Remark:** The system *COOL* is implemented in the programming language PL, available at

**http://fmoz_problemX.com.au/2019/**

# A Standard Conference Talk

**Proof:** By simple induction

**Deliver (T1)**: AWN defines the inference rule

$$\frac{}{\xi, \mathbf{deliver}(data).\mathrm{p} \xrightarrow{\mathbf{deliver}(\xi(data))} \xi, \mathrm{p}}\ \textsc{Deliver (T1)}$$

There exists an $(A_1, A_2)$ in $\mathscr{A}$ such that $\mathbf{deliver}(\xi(data)) \in A_1 \wedge \xi, \mathbf{deliver}(data).\mathrm{p} \xrightarrow{A_1} \xi, \mathrm{p}$, namely Pair 2.9 in Table 2.10. This base case can therefore be proven by finding a set of derivations in mCRL2 such that $\mathrm{T}(\xi, \mathbf{deliver}(data).\mathrm{p}) \xrightarrow{\ } \equiv \mathrm{T}(\xi, \mathrm{p})$ for all $a \in A_2 = \left\{ \mathbf{del}(\hat{\mathtt{i}}, [\![\mathrm{T}_\xi(data)]\!]) \;\middle|\; \hat{\mathtt{i}} \in \mathrm{T}(\mathrm{IP}) \right\}$.
In mCRL2, the following derivation can be made for all $\hat{\mathtt{i}} \in \mathrm{T}(\mathrm{IP})$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\mathbf{del}(t_{\hat{\mathtt{i}}}, \mathrm{T}_\xi(data)) \xrightarrow{[\![\mathbf{del}(t_{\hat{\mathtt{i}}}, \mathrm{T}_\xi(data))]\!]} \checkmark}\ \textsc{Axiom}}{\mathbf{del}(t_{\hat{\mathtt{i}}}, \mathrm{T}_\xi(data)) \xrightarrow{\mathbf{del}(\hat{\mathtt{i}}, [\![\mathrm{T}_\xi(data)]\!])} \checkmark}\ \text{Definition 2.1}}{\mathbf{del}(t_{\hat{\mathtt{i}}}, \mathrm{T}_\xi(data)).\mathrm{T}_{\mathrm{DOM}(\xi)}(\xi, \mathrm{q}) \xrightarrow{\mathbf{del}(\hat{\mathtt{i}}, [\![\mathrm{T}_\xi(data)]\!])} \mathrm{T}_{\mathrm{DOM}(\xi)}(\xi, \mathrm{p})}\ \textsc{Seq 1}}{\mathbf{del}(\mathtt{ip}, \mathrm{T}_\xi(data)).\mathrm{T}_{\mathrm{DOM}(\xi)}(\xi, \mathrm{q})[\mathtt{ip} := t_{\hat{\mathtt{i}}}] \xrightarrow{\mathbf{del}(\hat{\mathtt{i}}, [\![\mathrm{T}_\xi(data)]\!])} \mathrm{T}_{\mathrm{DOM}(\xi)}(\xi, \mathrm{p})}\ \text{Substitution}}{\sum_{\mathtt{ip}:\mathrm{T}(\mathrm{IP})} \mathbf{del}(\mathtt{ip}, \mathrm{T}_\xi(data)).\mathrm{T}_{\mathrm{DOM}(\xi)}(\xi, \mathrm{q}) \xrightarrow{\mathbf{del}(\hat{\mathtt{i}}, [\![\mathrm{T}_\xi(data)]\!])} \mathrm{T}_{\mathrm{DOM}(\xi)}(\xi, \mathrm{p})}\ \textsc{Sum 2}}{\mathrm{T}_{\mathrm{DOM}(\xi)}(\xi, \mathbf{deliver}(ms).\mathrm{p}) \xrightarrow{\mathbf{del}(\hat{\mathtt{i}}, [\![\mathrm{T}_\xi(data)]\!])} \mathrm{T}_{\mathrm{DOM}(\xi)}(\xi, \mathrm{p})}\ \text{T5}}{\mathrm{T}(\xi, \mathbf{deliver}(ms).\mathrm{p}) \xrightarrow{\mathbf{del}(\hat{\mathtt{i}}, [\![\mathrm{T}_\xi(data)]\!])} \mathrm{T}(\xi, \mathrm{p})}\ \text{T12}$$

for $\mathtt{ip} \notin \mathrm{T}(V)$. In conclusion, the induction hypothesis holds for this base case.

# A Standard Conference Talk

**Remark:** Our tool also supports the following input languages

- process algebras CCS, CSP, ABC
- Petri Nets
- Timed Automata
- Markov Decision Diagrams and Markow Chains
- …

Therefore it is really user-friendly, isn't it?

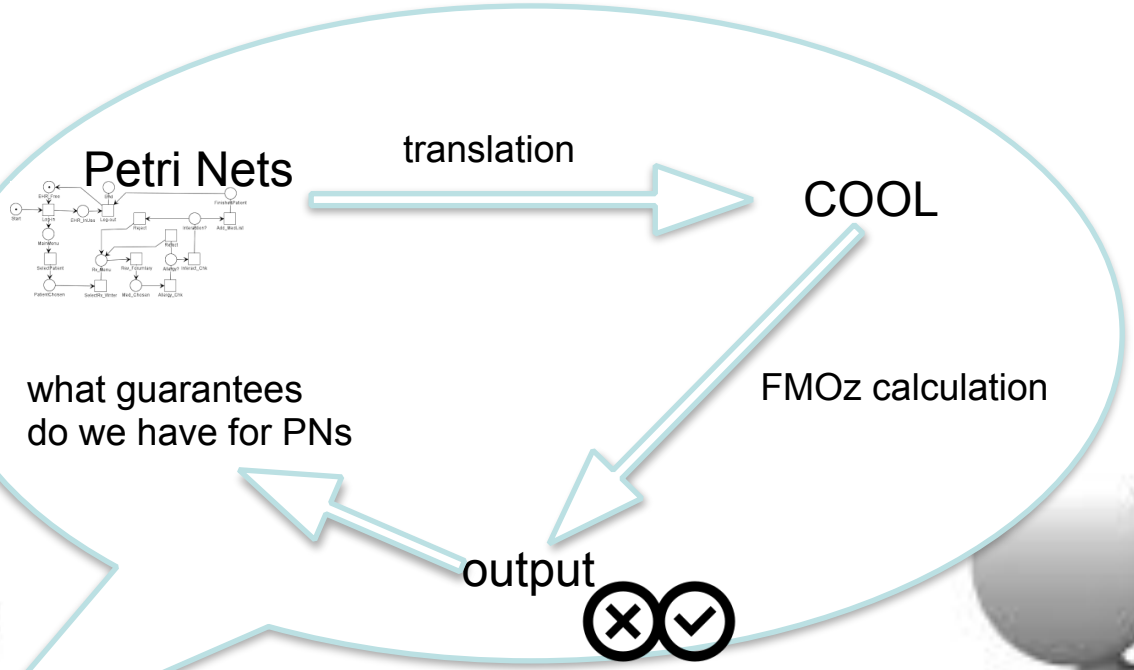
# A Standard Conference Talk

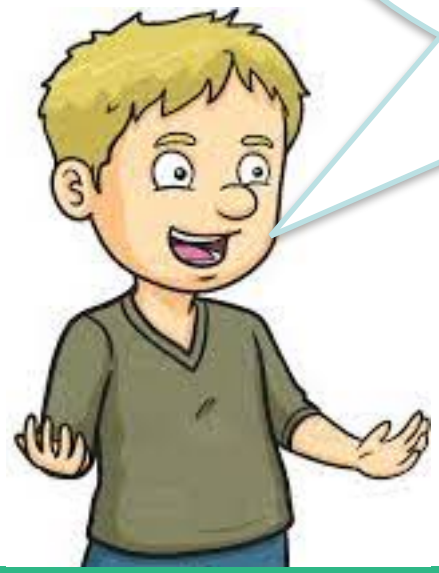
## Questions

# Question Time

You accept all these input languages and transform them (internally) into the formalism FMOz, what guarantees can you give?

# Question Time



# Further Discussion

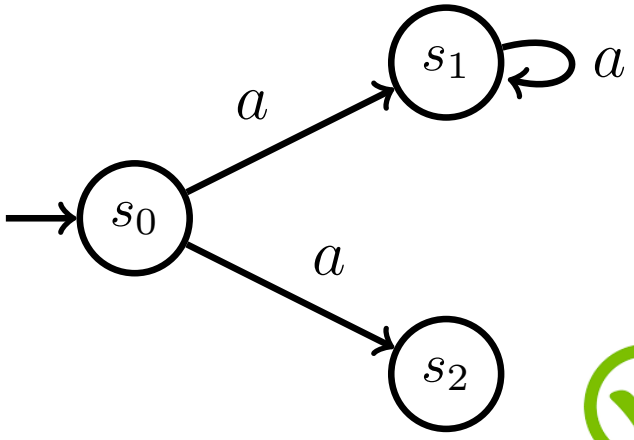
- What structure does your translation have
  - branching bisimulation
  - strong/weak bisimulation
  - justness-preserving
  - …

- What properties can be checked
  - deadlock
  - livelock
  - timelock
  - safety properties
  - liveness properties
  - security/infoflow properties

# Standard Reply

# A Trivial Example

- From automata to some formal language



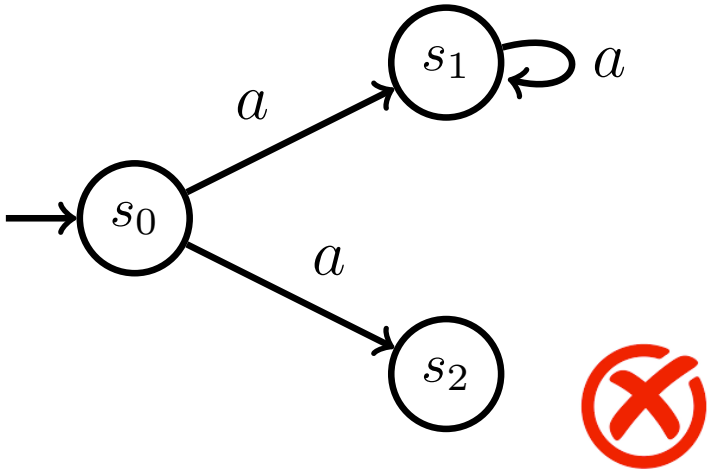

$$\textbf{while}\ (\texttt{true})\ \{\ a\ \}$$

- Is this a good translation?
  (CTL) Property:

$$\mathbf{E}\,\mathbf{G}\,a$$

# A Trivial Example

- From automata to some formal language

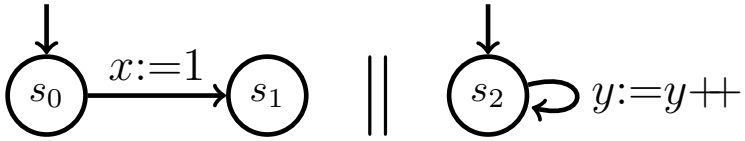



$$\textbf{while}\ (\texttt{true})\ \{\ a\ \}$$

- Is this a good translation?
  (CTL) Property:
  $\mathbf{A}\,\mathbf{G}\,a$
- It is a (strong) simulation

# A (Non-)Trivial Example

- From automata to some formal language



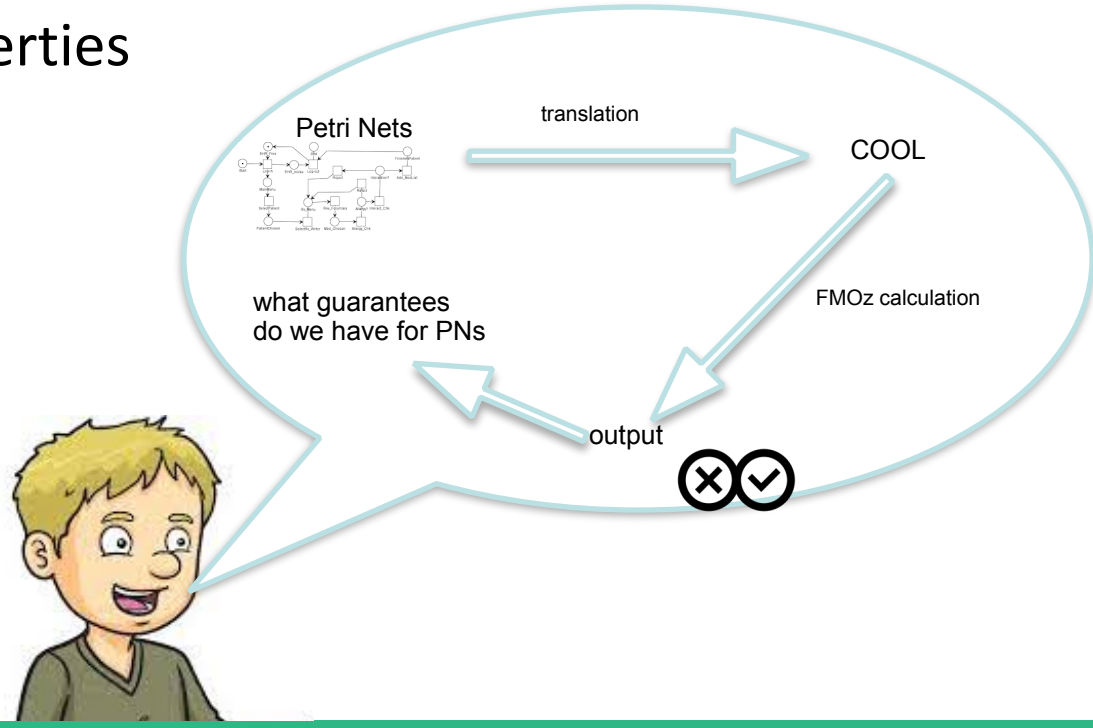

$$
\begin{array}{l}
\textbf{while}\ (\texttt{true})\ \{ \\
\qquad x == 0 \rightarrow x := 1 \\
\quad [] \ y := y\,{+}{+} \\
\}
\end{array}
$$

- It is a strong bisimulation (roughly)

- Is this a good translation?
  good for safety, not necessary for liveness (see Rob's talk@FMOz'18)

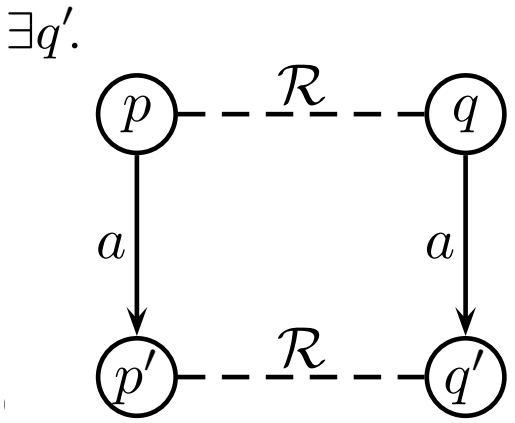
# Intermediate Results and Observations

- Sometimes Less is More
- **“Do not just add further input languages to your tool”**
- If you do, state which properties can be shown (and prove it)
- ask others!



# Bisimulation: A Formal Definition

- standard technique to compare labelled transition systems (LTSs)




A binary relation $\mathcal{R} \subseteq S_1 \times S_2$ is a *strong simulation* between two transition systems if it satisfies, for all $a$

$$\text{if } p \,\mathcal{R}\, q \text{ and } p \xrightarrow{a}_1 p' \text{ then } \exists q'.\ q \xrightarrow{a}_2 q' \text{ and } p' \,\mathcal{R}\, q' \ .$$

A *bisimulation* is a relation $\mathcal{R}$ with both $\mathcal{R}$ and $\mathcal{R}^{\smile}$ being strong simulations.

# It’s Obvious: A Case Study

- From AWN (Algebra for Wireless Networks)
  to mCRL2 (milli Common Representation Language)

- **why:** AWN is ideal for reasoning about routing protocols
  (see FMOz’18)
  mCRL2 provides more than 50 tools
  (model checker, simulator, analyser, …)
  **http://mcrl2.org**

# AWN vs mCRL2

| **AWN** (process algebra) | **mCRL2** (process algebra) |
|---|---|
| choice (+), composition (.), … | choice (+), composition (.), … |
| data structure | data structure |
| process call | process call |

creating a bisimulation between both is straightforward

# AWN vs mCRL2

| **AWN** (process algebra) | **mCRL2** (process algebra) |
|---|---|
| choice (+), composition (.), … | choice (+), composition (.), … |
| data structure | data structure |
| process call | process call |
| | |
| layered (nodes, networks, …) | single layer |
| 4 different synchronisations | 1 operator |
| dynamic topologies | ??? |
| | |

creating a bisimulation between both is straightforward

# AWN to mCRL2: A Summary

- 1 Master's Thesis (of a very smart student)
- 12 months of work
- 1 conference publication
- 60+ pages proof
- required the introduction of 2 new variants of bisimulation
- we did several mistakes in our translation that were only discovered through formal proof

- **But yes, a translation from any input language is obvious and straightforward**

# AWN to mCRL2

- The formal translation (most parts)

$$
\begin{aligned}
\mathrm{T}_V(\zeta, \mathbf{broadcast}(ms).P) &= \textstyle\sum_{\mathtt{D}:\mathrm{Set(IP)}} \mathbf{cast}(\mathrm{IP}, \mathtt{D}, ms^\zeta) \cdot \mathrm{T}_V(\zeta, P) \\
\mathrm{T}_V(\zeta, \mathbf{groupcast}(dests, ms).P) &= \textstyle\sum_{\mathtt{D}:\mathrm{Set(IP)}} \mathbf{cast}(dests^\zeta, \mathtt{D}, ms^\zeta) \cdot \mathrm{T}_V(\zeta, P) \\
\mathrm{T}_V(\zeta, \mathbf{unicast}(dest, ms).P \blacktriangleright Q) &= \mathbf{cast}(\{dest^\zeta\}, \{dest^\zeta\}, ms^\zeta) \cdot \mathrm{T}_V(\zeta, P) \\
&\quad + \neg\mathbf{uni}(\{dest^\zeta\}, \emptyset, ms^\zeta) \cdot \mathrm{T}_V(\zeta, Q) \\
\mathrm{T}_V(\zeta, \mathbf{send}(ms).P) &= \mathbf{send}(\emptyset, \emptyset, ms^\zeta) \cdot \mathrm{T}_V(\zeta, P) \\
\mathrm{T}_V(\zeta, \mathbf{deliver}(data).P) &= \textstyle\sum_{\mathtt{ip}:\mathrm{IP}} \mathbf{del}(\mathtt{ip}, data^\zeta) \cdot \mathrm{T}_V(\zeta, P) \\
\mathrm{T}_V(\zeta, \mathbf{receive}(\mathtt{m}).P) &= \textstyle\sum_{\substack{\mathtt{D},\mathtt{D}':\mathrm{Set(IP)} \\ \mathtt{m}:\mathrm{MSG}}} \mathbf{receive}(\mathtt{D}, \mathtt{D}', \mathtt{m}) \cdot \mathrm{T}_{V \cup \{\mathtt{m}\}}(\zeta^{\setminus \mathtt{m}}, P) \\
\mathrm{T}_V(\zeta, [\![\mathtt{v} := exp]\!]\ P) &= \textstyle\sum_{\mathtt{y}:\mathrm{sort(v)}} (\mathtt{y} = exp^\zeta) \rightarrow \\
&\quad (\textstyle\sum_{\mathtt{v}:\mathrm{sort(y)}} (\mathtt{v} = \mathtt{y}) \rightarrow \mathbf{t} \cdot \mathrm{T}_{V \cup \{\mathtt{v}\}}(\zeta^{\setminus \mathtt{v}}, P)) \\
\mathrm{T}_V(\zeta, X(exp_1, \cdots, exp_n)) &= X(exp_1^\zeta, \cdots, exp_n^\zeta) \\
\mathrm{T}_V(\zeta, P + Q) &= \mathrm{T}_V(\zeta, P) + \mathrm{T}_V(\zeta, Q) \\
\mathrm{T}_V(\zeta, [\varphi]P) &= \textstyle\sum_{\mathrm{Fv}(\varphi) \setminus V} \varphi^\zeta \rightarrow \mathbf{t} \cdot \mathrm{T}_{V \cup \mathrm{Fv}(\varphi)}(\zeta, P)
\end{aligned}
$$

# AWN to mCRL2

- the translation (the ugly parts)

$$\mathrm{T}(ip : P : R) = \nabla_V \Gamma_C(\mathrm{T}(P) \| G(ip, R))$$

where $V = \{\mathbf{t}, \mathbf{starcast}, \mathbf{arrive}, \mathbf{deliver}, \mathbf{connect}, \mathbf{disconnect}\}$

where $C = \{\mathbf{cast}|\overline{\mathbf{cast}} \to \mathbf{starcast}, \neg\mathbf{uni}|\overline{\neg\mathbf{uni}} \to \mathbf{t},$
$\mathbf{del}|\overline{\mathbf{del}} \to \mathbf{deliver}, \mathbf{receive}|\overline{\mathbf{receive}} \to \mathbf{arrive}\}$

where $G(\mathtt{ip}, \mathtt{R}) \stackrel{\text{def}}{=} \sum_{\substack{\mathtt{D},\mathtt{D}':\mathrm{Set(IP)} \\ \mathtt{m}:\mathrm{MSG}}} (\mathtt{R} \cap \mathtt{D} = \mathtt{D}') \to \overline{\mathbf{cast}}(\mathtt{D}, \mathtt{D}', \mathtt{m}) \cdot G(\mathtt{ip}, \mathtt{R})$
$+ \sum_{\substack{\mathtt{d}:\mathrm{IP} \\ \mathtt{m}:\mathrm{MSG}}} (\mathtt{d} \notin \mathtt{R}) \to \overline{\neg\mathbf{uni}}(\{\mathtt{d}\}, \emptyset, \mathtt{m}) \cdot G(\mathtt{ip}, \mathtt{R})$
$+ \sum_{\mathtt{data}:\mathrm{DATA}} \overline{\mathbf{del}}(\mathtt{ip}, \mathtt{data}) \cdot G(\mathtt{ip}, \mathtt{R})$
$+ \sum_{\mathtt{ip}':\mathrm{IP}} \mathbf{connect}(\mathtt{ip}, \mathtt{ip}') \cdot G(\mathtt{ip}, \mathtt{R} \cup \{\mathtt{ip}'\})$
$+ \sum_{\mathtt{ip}':\mathrm{IP}} \mathbf{connect}(\mathtt{ip}', \mathtt{ip}) \cdot G(\mathtt{ip}, \mathtt{R} \cup \{\mathtt{ip}'\})$
$+ \sum_{\mathtt{ip}',\mathtt{ip}'':\mathrm{IP}} (\mathtt{ip} \notin \{\mathtt{ip}', \mathtt{ip}''\}) \to \mathbf{connect}(\mathtt{ip}', \mathtt{ip}'') \cdot G(\mathtt{ip}, \mathtt{R})$
$+ \sum_{\mathtt{ip}':\mathrm{IP}} \mathbf{disconnect}(\mathtt{ip}, \mathtt{ip}') \cdot G(\mathtt{ip}, \mathtt{R} \backslash \{\mathtt{ip}'\})$
$+ \sum_{\mathtt{ip}':\mathrm{IP}} \mathbf{disconnect}(\mathtt{ip}', \mathtt{ip}) \cdot G(\mathtt{ip}, \mathtt{R} \backslash \{\mathtt{ip}'\})$
$+ \sum_{\mathtt{ip}',\mathtt{ip}'':\mathrm{IP}} (\mathtt{ip} \notin \{\mathtt{ip}', \mathtt{ip}''\}) \to \mathbf{disconnect}(\mathtt{ip}', \mathtt{ip}'') \cdot G(\mathtt{ip}, \mathtt{R})$
$+ \sum_{\substack{\mathtt{D},\mathtt{D}':\mathrm{Set(IP)} \\ \mathtt{m}:\mathrm{MSG}}} (\mathtt{ip} \in \mathtt{D}') \to \overline{\mathbf{receive}}(\mathtt{D}, \mathtt{D}', \mathtt{m}) \cdot G(\mathtt{ip}, \mathtt{R})$
$+ \sum_{\substack{\mathtt{D},\mathtt{D}':\mathrm{Set(IP)} \\ \mathtt{m}:\mathrm{MSG}}} (\mathtt{ip} \notin \mathtt{D}') \to \mathbf{arrive}(\mathtt{D}, \mathtt{D}', \mathtt{m}) \cdot G(\mathtt{ip}, \mathtt{R})$

# AWN to mCRL2

- The formal translation (most parts)

$$\mathrm{T}_V(\zeta, \mathbf{broadcast}(ms).P) = \sum_{\mathtt{D}:\mathrm{Set}(\mathrm{IP})} \mathbf{cast}(\mathrm{IP}, \mathtt{D}, ms^\zeta) \cdot \mathrm{T}_V(\zeta, P)$$

$$\mathrm{T}_V(\zeta, \mathbf{groupcast}(dests, ms).P) = \sum_{\mathtt{D}:\mathrm{Set}(\mathrm{IP})} \mathbf{cast}(dests^\zeta, \mathtt{D}, ms^\zeta) \cdot \mathrm{T}_V(\zeta, P)$$

$$\begin{aligned}\mathrm{T}_V(\zeta, \mathbf{unicast}(dest, ms).P \blacktriangleright Q) = {} & \mathbf{cast}(\{dest^\zeta\}, \{dest^\zeta\}, ms^\zeta) \cdot \mathrm{T}_V(\zeta, P) \\ & + \neg\mathbf{uni}(\{dest^\zeta\}, \emptyset, ms^\zeta) \cdot \mathrm{T}_V(\zeta, Q)\end{aligned}$$

$$\mathrm{T}_V(\zeta, \mathbf{send}(ms).P) = \mathbf{send}(\emptyset, \emptyset, ms^\zeta) \cdot \mathrm{T}_V(\zeta, P)$$

$$\mathrm{T}_V(\zeta, \mathbf{deliver}(data).P) = \sum_{\mathtt{ip}:\mathrm{IP}} \mathbf{del}(\mathtt{ip}, data^\zeta) \cdot \mathrm{T}_V(\zeta, P)$$

$$\mathrm{T}_V(\zeta, \mathbf{receive}(\mathtt{m}).P) = \sum_{\substack{\mathtt{D},\mathtt{D}':\mathrm{Set}(\mathrm{IP}) \\ \mathtt{m}:\mathrm{MSG}}} \mathbf{receive}(\mathtt{D}, \mathtt{D}', \mathtt{m}) \cdot \mathrm{T}_{V \cup \{\mathtt{m}\}}(\zeta^{\setminus \mathtt{m}}, P)$$

$$\begin{aligned}\mathrm{T}_V(\zeta, [\![\mathtt{v} := exp]\!]\ P) = {} & \sum_{\mathtt{y}:\mathrm{sort}(\mathtt{v})} (\mathtt{y} = exp^\zeta) \rightarrow \\ & (\sum_{\mathtt{v}:\mathrm{sort}(\mathtt{y})} (\mathtt{v} = \mathtt{y}) \rightarrow \mathbf{t} \cdot \mathrm{T}_{V \cup \{\mathtt{v}\}}(\zeta^{\setminus \mathtt{v}}, P))\end{aligned}$$

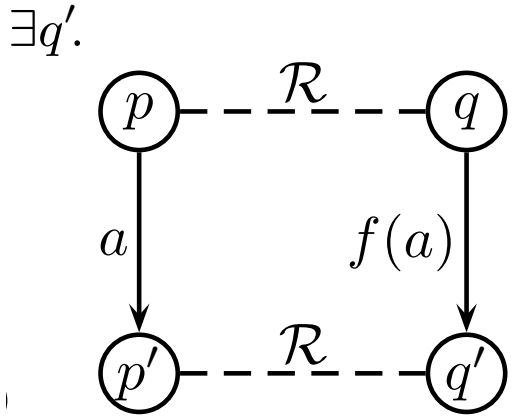
$$\mathrm{T}_V(\zeta, X(exp_1, \cdots, exp_n)) = X(exp_1^\zeta, \cdots, exp_n^\zeta)$$

$$\mathrm{T}_V(\zeta, P + Q) = \mathrm{T}_V(\zeta, P) + \mathrm{T}_V(\zeta, Q)$$

$$\mathrm{T}_V(\zeta, [\varphi]P) = \sum_{\mathrm{Fv}(\varphi) \setminus V} \varphi^\zeta \rightarrow \mathbf{t} \cdot \mathrm{T}_{V \cup \mathrm{Fv}(\varphi)}(\zeta, P)$$

# Bisimulation modulo Renaming

- Source and target language often have different primitives (different labels on transitions)




A binary relation $\mathcal{R} \subseteq S_1 \times S_2$ is a *strong simulation modulo renaming* between two transition systems for a bijective *renaming function* $f$, if, for all $a$

$$\text{if } p \,\mathcal{R}\, q \text{ and } p \xrightarrow{a}_1 p' \text{ then } \exists q'.\ q \xrightarrow{f(a)}_2 q' \text{ and } p' \,\mathcal{R}\, q' \ .$$

A *bisimulation modulo renaming* is a relation $\mathcal{R}$ with both $\mathcal{R}$ and $\mathcal{R}^\smile$ being strong simulations modulo renaming.

# Bisimulation and Data Congruence

- we also had to take data congruence into account (actions $a(3)$, $a(1+2)$ and $a(6-3)$ should be treated the way)
- luckily, data congruences are “standard”
- in our setting data concurrence ($\equiv$) is a strong bisimulation
- strong bisimulations are compositional

# AWN to mCRL2

- The formal translation (most parts)

$$\mathrm{T}_V(\zeta, \mathbf{broadcast}(ms).P) = \sum_{\mathtt{D}:\mathrm{Set}(\mathrm{IP})} \mathbf{cast}(\mathrm{IP}, \mathtt{D}, ms^\zeta) \cdot \mathrm{T}_V(\zeta, P)$$

$$\mathrm{T}_V(\zeta, \mathbf{groupcast}(dests, ms).P) = \sum_{\mathtt{D}:\mathrm{Set}(\mathrm{IP})} \mathbf{cast}(dests^\zeta, \mathtt{D}, ms^\zeta) \cdot \mathrm{T}_V(\zeta, P)$$

$$\mathrm{T}_V(\zeta, \mathbf{unicast}(dest, ms).P \blacktriangleright Q) = \mathbf{cast}(\{dest^\zeta\}, \{dest^\zeta\}, ms^\zeta) \cdot \mathrm{T}_V(\zeta, P) + \neg\mathbf{uni}(\{dest^\zeta\}, \emptyset, ms^\zeta) \cdot \mathrm{T}_V(\zeta, Q)$$

$$\mathrm{T}_V(\zeta, \mathbf{send}(ms).P) = \mathbf{send}(\emptyset, \emptyset, ms^\zeta) \cdot \mathrm{T}_V(\zeta, P)$$

$$\mathrm{T}_V(\zeta, \mathbf{deliver}(data).P) = \sum_{\mathtt{ip}:\mathrm{IP}} \mathbf{del}(\mathtt{ip}, data^\zeta) \cdot \mathrm{T}_V(\zeta, P)$$

$$\mathrm{T}_V(\zeta, \mathbf{receive}(\mathtt{m}).P) = \sum_{\substack{\mathtt{D},\mathtt{D}':\mathrm{Set}(\mathrm{IP}) \\ \mathtt{m}:\mathrm{MSG}}} \mathbf{receive}(\mathtt{D}, \mathtt{D}', \mathtt{m}) \cdot \mathrm{T}_{V \cup \{\mathtt{m}\}}(\zeta^{\setminus \mathtt{m}}, P)$$

$$\mathrm{T}_V(\zeta, [\![ \mathtt{v} := exp ]\!] P) = \sum_{\mathtt{y}:\mathrm{sort}(\mathtt{v})} (\mathtt{y} = exp^\zeta) \to (\sum_{\mathtt{v}:\mathrm{sort}(\mathtt{y})} (\mathtt{v} = \mathtt{y}) \to \mathbf{t} \cdot \mathrm{T}_{V \cup \{\mathtt{v}\}}(\zeta^{\setminus \mathtt{v}}, P))$$

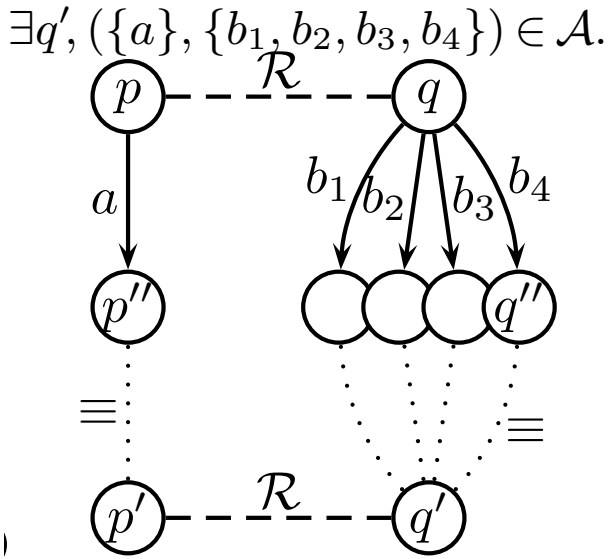
$$\mathrm{T}_V(\zeta, X(exp_1, \cdots, exp_n)) = X(exp_1^\zeta, \cdots, exp_n^\zeta)$$

$$\mathrm{T}_V(\zeta, P + Q) = \mathrm{T}_V(\zeta, P) + \mathrm{T}_V(\zeta, Q)$$

$$\mathrm{T}_V(\zeta, [\varphi]P) = \sum_{\mathrm{Fv}(\varphi) \setminus V} \varphi^\zeta \to \mathbf{t} \cdot \mathrm{T}_{V \cup \mathrm{Fv}(\varphi)}(\zeta, P)$$

# Warped Bisimulation

- the layered structure of AWN required another generalisation




A binary relation $\mathcal{R} \subseteq S_1 \times S_2$ is a $\mathcal{A}$-*warped simulation up to* $\equiv$ between two transition systems for a relation $\mathcal{A}$ (between sets of action labels) if it satisfies

$$\text{if } p\,\mathcal{R}\,q \text{ and } p \xrightarrow{a}_1 p'' \text{ then } \exists \mathsf{A}_1, \mathsf{A}_2, p', q'.$$
$$(a \in \mathsf{A}_1,\ p'' \equiv p',\ \mathsf{A}_1\,\mathcal{A}\,\mathsf{A}_2,\ p \xrightarrow{\mathsf{A}_1}_1 \equiv p',\ q \xrightarrow{\mathsf{A}_2}_2 \equiv q' \text{ and } p'\ R\ q')\,,$$

where $p \xrightarrow{\mathsf{A}}_1 \equiv p' :\Leftrightarrow \forall a \in \mathsf{A}. \exists p''.\, p \xrightarrow{a}_1 p'' \wedge p'' \equiv p'$.

An $\mathcal{A}$-*warped bisimulation up to* $\equiv$ is a relation $\mathcal{R}$ with $\mathcal{R}$ is an $\mathcal{A}$-warped bisimulation up to $\equiv$ and $\mathcal{R}^\smile$ $\mathcal{A}^\smile$-warped bisimulation up to $\equiv$.

# Conclusion

- we have brilliant tools that give **formal guarantees**
- adding another input language carelessly may lead
  - **false results**
  - **reduces confidence** in formal methods and formal tools
  - reduces our credibility

# Future Work

- keep asking (raise awareness for the problem)

- how to scale the proof effort
  (our case study were 2 process algebras, no time, no probabilities)

- what structure is needed to maintain properties
  - safety = (strong) bisimulation
  - liveness = justness (or fairness) ?
  - security = ???

- is there a generic way to prove these translations
  (I doubt)


# Thank you

**Data61**
Peter Höfner

**t** +61 2 9490 5861
**e** peter.hoefner@data61.csiro.au
**w** www.data61.csiro.au