Modelling an implementation & implementing a model

Thanks to the organisers

Apologies to the audience

Thank Michele Pinna
Modelling an implementation

Joint work with

Roland Kuhn @ Actyx
Hernán Melgratti @ UBA
- Systems don't stop
- Partitioning is intrinsic
- Availability is not negotiable

Keep going and don't be afraid to make mistakes

Local First

Consistency

Availability

Partitioning
Actyx

- Platform to develop applications for controlling / managing industrial production processes
- Command execution generates events
- Pub-Sub middleware (asynchronously) replicates logs of events by merging newly generated events
- Temporary partitioning
- Need for eventual consistency of replicated logs and other application domains too
Top down

- Global type (and well-formedness conditions) to handle protocols where
  \[ c_1, b_1 @ P_1 \rightarrow O \rightarrow \quad \text{with } P_1 \neq P_2 \text{ and arbitrary num. of inst.} \]

- Local type wee FSAs too

proj ection op. \text{parametrised by subscrip tives}
Some comments/results

- Session infidelity
- Projections yield correct realisations
- Deadlock freedom by construction!
- Dropping events is harmless in W.F. protocols

① Details in Kuhn, Melgratti, T. @ ECOOP 23)
Open Problems

- Generalise WF
- Optimise
- Support to “design” subscriptions

- Collaborative vs. Adversarial
- Failures
- Unreliable pub-sub
- Impure / non-compensable events
Implementing a model

Joint work in progress with

Philipp Heller @ ZTH
Hernán Helgastti @ UBA
Aleste Scalos @ DTU
Dogstuhl, September 2021

The join calculus is a powerful abstraction for concurrent programming:

\[
\text{banana (b) & apple (a) if } a = b \Rightarrow \text{fruit salad (b, a)} \\
\text{lettuce (l) & apple (a) & olives (o) \Rightarrow \text{salad (l, a, o)}}
\]

Q: Why hasn't the join calculus become popular?

A: Its implementations are not very efficient.
Idea: reuse partial matches

Fix a signature $\Sigma$ and a set of variable symbols $X$

$\mu$ range on $\text{Term}_\Sigma, x$

$\gamma$ range on propositions on $X$

$Q \in \text{Term}_\Sigma$  

Pattern matching

$\pi = \mu_1 \land \ldots \land \mu_n \iff \gamma \quad Q \vdash_{\cdot} \pi$ where

$Q \in \text{Term}_\Sigma$

$\forall i \leq n: Q[i] = \pi[i]$

\[ \sigma \]
Chasing matches in

Given \( Q \subseteq T \subseteq \mathbb{Z} \) and \( \pi = \mu_1 \& \cdots \& \mu_m \)

\[ \text{cidx} (Q, \mu) = \{ i \mid \exists j, \sigma: Q[i] = \mu[j] \sigma \} \]

\[ A = \{ a \in \{1, \ldots, m\}^+ | \exists i \in \text{cidx}(Q, \mu) \text{ and } a : i \rightarrow j \Rightarrow \exists \sigma: Q[i] = \mu[a(i)] \sigma \} \]

\[ a \leq b \Leftrightarrow \min(\text{dom} a \setminus \text{dom} b) \leq \min(\text{dom} b \setminus \text{dom} a) \quad \text{or} \quad \text{dom} a = \text{dom} b \]

Find \( a \in A \) "minimal" s.t.

\[ \exists \sigma: \forall i : \text{dom} a \ni Q[i] = \mu[a(i)] \sigma \quad \text{and} \quad \sigma \sigma \] holds
\[ r \in U \quad \text{for } r \in U \]

\[ X_1, \ldots, X_n \]

level 0

level 1

level h

\[ X_{h+1} \]

\[ X_{x_{i+1}} \]

\[ \min X \leq \min Y \]

\[ \max X \neq \max Y \]
\[
x \text{ father } Y \implies \min X \leq \min Y \\text{ and } \max X \neq \max Y
\]
Open Problems

- Does this work better?
  - Experiments
  - Benchmarks (e.g., Savina)

- The algorithms is exponential in the worst case, but how realistic pathological cases are?

- Are there other interesting kinds of join patterns?

- Other notions of "best" match?