From Communicating Machines to Graphical Choreographies

Julien Lange

based on joint works with L. Bocchi, E. Tuosto, and N. Yoshida

May 2015
Single bird \(\simeq\) local behaviour \(\simeq\) CFSM

- alignment
- separation
- cohesion
Flock \sim global\ behaviour \sim choreography
Introduction

- Parts of distributed systems change/evolve, not always in a coordinated way,
- these changes are often *not* documented.
- Service oriented systems are sometimes composed dynamically,
- it is often unclear how complex the overall system has become.
- Cognizant’s Zero Deviation Lifecycle Business Unit.

*A global* point of view of a distributed system is *essential* for top-level management.
- Choreography-driven development, cf. Multiparty Session Types top-down approach (POPL’08 & ESOP’12)
- Not applicable without *a priori knowledge* of a choreography
Choreography-driven development, cf. Multiparty Session Types top-down approach (POPL’08 & ESOP’12)
- Not applicable without *a priori knowledge* of a choreography
- Our goal: from *Communicating Finite-State Machines* to *Global Graphs*
Background: CFSMs

“On Communicating Finite-State Machines”, Brand & Zafiropulo (’83)
Background: CFSMs

“On Communicating Finite-State Machines”, Brand & Zafiropulo (’83)
Background: CFSMs

"On Communicating Finite-State Machines", Brand & Zafiropulo ('83)
Background: CFSMs

“On Communicating Finite-State Machines”, Brand & Zafiropulo (’83)
Background: CFSMs

“On Communicating Finite-State Machines”, Brand & Zafiropulo ('83)
“On Communicating Finite-State Machines”, Brand & Zafiropulo (’83)
Background: CFSMs

“On Communicating Finite-State Machines”, Brand & Zafiropulo ('83)
Background: CFSMs

“On Communicating Finite-State Machines”, Brand & Zafiropulo (’83)
Background: CFSMs

"On Communicating Finite-State Machines", Brand & Zafiropulo ('83)
Background: CFSMs

“On Communicating Finite-State Machines”, Brand & Zafiropulo (’83)
Background: CFSMs

“On Communicating Finite-State Machines”, Brand & Zafiropulo ('83)
Global Graphs

Four Player Game
Global Graphs

Four Player Game

Alice

Bob

Dave

Carol
Objectives

Two main objectives:

- **Sound Condition for Safety**: generalised multiparty compatibility
  
  If \( S = (M_1, \ldots, M_k) \) is *compatible* then \( S \) is “safe”, i.e., every sent message is eventually received and no deadlock.

- **Construction of a Global Graph**:
  
  If \( G \) is the global graph constructed from \( S \), then

  \[
  S = (M_1, \ldots, M_k) \equiv (G|_1, \ldots, G|_k)
  \]
The Plan

1. Build TS($S$), the transition system of all *synchronous* executions
2. Check for safety on TS($S$) to
   - ensure equivalence between original system and the projections of the choreography,
   - guarantee safety (no deadlock, no orphan message)
3. Build a choreography (global graph) from TS($S$), relying on
   - the theory of regions, and
   - safe Petri nets.
1. Synchronous Transition System of CFSMs
CFSMs

L. Bocchi, J. Lange, E. Tuosto, N. Yoshida
From Communicating Machines to Graphical Choreographies
CFSMs

\[ (A_0, B_0, A \rightarrow B : \text{apple}) \]

\[ (A_0, B_0, A \rightarrow B : \text{orange}) \]

\[ (A_1, B_1, A \rightarrow B : \text{apple}) \]

\[ (A_1, B_1, A \rightarrow B : \text{orange}) \]

\[ (A_2, B_2, A \rightarrow B : \text{drink}) \]

\[ (A_3, B_3) \]
CFSMs

(A_0, B_0, \rightarrow B: apple) (A_0, B_0, \rightarrow B: orange)

(A_1, B_1, \rightarrow B: apple) (A_1, B_1, \rightarrow B: orange)

(A_2, B_2, \rightarrow B: apple) (A_2, B_2, \rightarrow B: drink)

(A_3, B_3)

TS(S)

L. Bocchi, J. Lange, E. Tuosto, N. Yoshida
From Communicating Machines to Graphical Choreographies
Synchronous Transition System (TS(S))
2. Check for Safety: *Generalised Multiparty Compatibility* (GMC)

1. Representability
2. Branching Property
Checking Compatibility (i) – Representability

Representability

- The projected TS\( (S) \equiv \) original machine
- Each branching in each machine must be represented in TS
Checking Compatibility (i) – Representability

Representability

- The projected TS(\(S\)) \(\equiv\) original machine
- Each branching in each machine must be represented in TS
Checking Compatibility (i) – Representability

Representability

- The projected $TS(S) \equiv$ original machine
- Each branching in each machine must be represented in $TS$
Checking Compatibility (i) – Representability

Representability

- The projected TS$(S)$ $\equiv$ original machine
- Each branching in each machine must be represented in TS

\[(A \rightarrow B : bwin) \downarrow_B = AB?bwin\]
\[(B \rightarrow A : sig) \downarrow_B = BA!sig\]
\[(C \rightarrow D : busy) \downarrow_B = \varepsilon\]
Checking Compatibility (i) – Representability

Representability

- The projected $TS(S) \equiv$ original machine
- Each branching in each machine must be represented in $TS$

\[(A \rightarrow B : \text{bwin}) |_B = AB?\text{bwin}\]
\[(B \rightarrow A : \text{sig}) |_B = BA!\text{sig}\]
\[(C \rightarrow D : \text{busy}) |_B = \varepsilon\]
Checking Compatibility (ii) – Branching Property

Each branching in TS must be either

\[ e \xleftarrow{n_1} e' \]
Checking Compatibility (ii) – Branching Property

Each branching in TS must be either commuting:

- commuting:
Each branching in TS must be either

- or, each *last node* $n_k$

  $n_1 \xleftarrow{e} n_2 \xrightarrow{e'} n_k \xleftarrow{e}$

must be a “well-formed” choice, i.e.,
- for each participant
  - it receives a different message in each branch, or
  - it is not involved in the choice
- there is a unique sender
- there is no “race” between branches

Last node, reachable from $n_1$, from which $e$ and $e'$ can be fired.
C → D: busy

(A₀, B₀, C₀, D₀)

A → C: cwin

(A₁, B₀, C₁, D₀)

A → B: bwin

C → B: close

B → C: close

C → D: busy

(A₁, B₁, C₀, D₀)

B → A: sig

A → C: cwin

(A₁, B₂, C₂, D₁)

B → C: close

C → D: busy

(A₁, B₂, C₃, D₀)

B → A: sig

C → B: close

(A₁, B₂, C₅, D₁)

B → A: sig

(A₂, B₂, C₂, D₀)

No race, e.g., between AC? cwin and BC? close

D: not involved in the choice
L. Bocchi, J. Lange, E. Tuosto, N. Yoshida
From Communicating Machines to Graphical Choreographies
A : $\text{AB}!\text{bwin} \neq \text{AC}!\text{cwin}$

B : $\text{AB}?\text{bwin} \neq \text{CB}?\text{blose}$

D : not involved in the choice

C : $\text{AC}?\text{cwin} \neq \text{BC}?\text{close}$

No race, e.g., between $\text{AC}?\text{cwin}$ and $\text{BC}?\text{close}$
Violating the no-race condition

L. Bocchi, J. Lange, E. Tuosto, N. Yoshida
From Communicating Machines to Graphical Choreographies
Violating the no-race condition
Violating the no-race condition
Violating the no-race condition

A → B: apple
A → C: carrot
A → C: cabbage
C → B: date
C → B: date
B → A: banana
A → B: apple
A → B: cake
B → A: biscuit
AB : apple
CB : ε
AC : cabbage
BA : ε
Violating the no-race condition
Violating the no-race condition

A → B : apple
A → C : carrot
A → C : cabbage
C → B : date
C → B : date
B → A : banana
B → A : banana
A → B : cake
A → B : cake
A → B : cake

AB : apple
CB : date
AC : ε
BA : ε
Violating the no-race condition

A → B: apple
A → C: carrot
A → C: cabbage
C → B: date
C → B: date
B → A: banana
B → A: banana
A → B: cake
A → B: cake
A → B: apple
B → A: biscuit
B → A: biscuit

AB : apple
CB : date
AC : ε
BA : ε
Violating the no-race condition

A → B: apple
A → C: carrot
A → C: cabbage
C → B: date
C → B: date
B → A: banana
A → B: apple
A → B: cake
B → A: biscuit
B → A: biscuit

AB: apple
CB: ε
AC: ε
BA: ε
Violating the no-race condition

A → B : apple
A → C : carrot
A → C : cabbage
C → B : date
C → B : date
B → A : banana
B → A : banana
A → B : cake
A → B : cake
B → A : biscuit
B → A : biscuit

AB : apple
CB : ε
AC : ε
BA : banana
Violating the no-race condition

L. Bocchi, J. Lange, E. Tuosto, N. Yoshida
From Communicating Machines to Graphical Choreographies
Violating the no-race condition

A → B: apple
A → C: carrot
A → C: cabbage
C → B: date
B → A: banana
A → B: apple
A → B: cake
B → A: biscuit

AB : apple · cake
CB : ε
AC : ε
BA : ε
Violating the no-race condition
Violating the no-race condition

A → B: apple
A → C: carrot
A → C: cabbage
C → B: date
C → B: date
B → A: banana
B → A: banana
A → B: cake
A → B: cake

L. Bocchi, J. Lange, E. Tuosto, N. Yoshida
From Communicating Machines to Graphical Choreographies
Violating the no-race condition

L. Bocchi, J. Lange, E. Tuosto, N. Yoshida
From Communicating Machines to Graphical Choreographies
Violating the no-race condition

A → B : apple
A → C : cabbage
C → B : date
B → A : banana
A → B : cake

L. Bocchi, J. Lange, E. Tuosto, N. Yoshida
From Communicating Machines to Graphical Choreographies
Violating the no-race condition

L. Bocchi, J. Lange, E. Tuosto, N. Yoshida
From Communicating Machines to Graphical Choreographies
3. Build a global graph
We use the work of Cortadella et al. (1998), based on the theory of regions, to derive a safe and extended free-choice Petri net from the Synchronous Transition System.
2: From Petri Net to One-source Petri Net

L. Bocchi, J. Lange, E. Tuosto, N. Yoshida
From Communicating Machines to Graphical Choreographies
3: From One-source Petri Net to Joined Petri Net

From Communicating Machines to Graphical Choreographies
4 & 5 : From Joined Petri Net to Global Graph

[Diagram of a Petri net transformation process]

L. Bocchi, J. Lange, E. Tuosto, N. Yoshida
From Communicating Machines to Graphical Choreographies
Prototype Implementation

https://bitbucket.org/julien-lange/gmc-synthesis
## Experimental Evaluation

| $S$                        | $|P|$ | $|N|$  | $|\rightarrow|$ | GMC | $|G|$ | Time (s) |
|---------------------------|------|-------|-----------------|-----|------|----------|
| Running Example           | 4    | 12    | 19              | ✓   | 16   | 0.184    |
| Running Example $\times 2$| 8    | 144   | 456             | ✓   | 32   | 22.307   |
| Bargain                   | 3    | 4     | 4               | ✓   | 8    | 0.103    |
| Bargain $\times 2$        | 6    | 16    | 32              | ✓   | 16   | 0.161    |
| Alternating 3-bit         | 2    | 24    | 48              | ✓   | 18   | 3.164    |
| Alternating 3-bit $\times 2$| 4 | 576   | 2304            | ✓   | 34   | 12.069   |
| TPMContract v2 $\times 2$ | 4    | 25    | 80              | ✓   | 30   | 0.362    |
| Sanitary Agency           | 4    | 17    | 21              | ✓   | 22   | 0.241    |
| Sanitary Agency $\times 2$| 8    | 196   | 476             | ✓   | 44   | 3.165    |
| Health System             | 6    | 10    | 11              | ✓   | 14   | 0.17     |
| Health System $\times 2$  | 12   | 100   | 220             | ✓   | 28   | 1.702    |
| Logistic                   | 4    | 13    | 17              | ✓   | 27   | 0.276    |
| Logistic $\times 2$        | 8    | 169   | 442             | ✓   | 54   | 2.155    |
| Cloud System v4           | 4    | 7     | 8               | ✓   | 12   | 0.14     |
| Cloud System v4 $\times 2$| 8    | 49    | 112             | ✓   | 24   | 0.432    |
## Experimental Evaluation

| S                  | $|P|$ | $|N|$ | $|\rightarrow|$ | GMC | $|G|$ | Time (s) |
|--------------------|-----|-----|--------------|-----|-----|---------|
| Running Example    | 4   | 12  | 19           | √   | 16  | 0.184   |
| Running Example × 2 | 8   | 144 | 456          | √   | 32  | 22.307  |
| Bargain            | 3   | 4   | 4            | √   | 8   | 0.103   |
| Bargain × 2        | 6   | 16  | 32           | √   | 16  | 0.161   |
| Alternating 3-bit  | 2   | 24  | 48           | √   | 18  | 3.164   |
| Alternating 3-bit × 2 | 4 | 576 | 2304       | √   | 34  | 12.069  |
| TPMContract v2 × 2 | 4   | 25  | 80           | √   | 30  | 0.362   |
| Sanitary Agency    | 4   | 17  | 21           | √   | 22  | 0.241   |
| Sanitary Agency × 2 | 8 | 196 | 476         | √   | 44  | 3.165   |
| Health System      | 6   | 10  | 11           | √   | 14  | 0.17    |
| Health System × 2  | 12  | 100 | 220          | √   | 28  | 1.702   |
| Logistic           | 4   | 13  | 17           | √   | 27  | 0.276   |
| Logistic × 2       | 8   | 169 | 442          | √   | 54  | 2.155   |
| Cloud System v4    | 4   | 7   | 8            | √   | 12  | 0.14    |
| Cloud System v4 × 2 | 8 | 49  | 112         | √   | 24  | 0.432   |
Each machine owns several clocks (not shared)

Time elapses at the same pace for each clock
Construction as in the untimed setting

Additional safety checks for time constraints
Safety checks for CTAs

- **MC**: (Generalised) Multiparty Compatibility
- **IE**: Interaction Enabling
- **CE**: Cycle Enabling

<table>
<thead>
<tr>
<th>Property</th>
<th>$S \sim (TS(S) \downarrow_p)_{p \in P}$</th>
<th>Safety</th>
<th>Progress</th>
<th>Non-Zeno</th>
<th>Eventual Reception</th>
</tr>
</thead>
<tbody>
<tr>
<td>MC</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>MC+IE</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>MC+CE</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>MC+IE+CE</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>


Draft: [www.doc.ic.ac.uk/~jlange/cta/](http://www.doc.ic.ac.uk/~jlange/cta/)
Summing up

- An effective way of checking properties of CFSMs, and whether one can construct a global graph from them.
- An algorithm based on the theory of regions.
- A CFSMs characterisation of well-formed *generalised global types*.
- [https://bitbucket.org/julien-lange/gmc-synthesis](https://bitbucket.org/julien-lange/gmc-synthesis)
- An extension for communicating timed automata
References

- Julien Lange, Emilio Tuosto, and Nobuko Yoshida. “From Communicating Machines to Graphical Choreographies”. In: *POPL 2015*. 2015

Related work (i)

- Julien Lange and Emilio Tuosto. “Synthesising Choreographies from Local Session Types”. In: CONCUR. LNCS. Springer, 2012
- Pierre-Malo Deniélou and Nobuko Yoshida. “Multiparty Compatibility in Communicating Automata: Characterisation and Synthesis of Global Session Types”. In: ICALP. LNCS. 2013
- Pavel Krcal and Wang Yi. “Communicating Timed Automata: The More Synchronous, the More Difficult to Verify”. In: CAV. LNCS. 2006
Related work (ii)

Thanks!

Any questions?

https://bitbucket.org/julien-lange/gmc-synthesis

L. Bocchi, J. Lange, E. Tuosto, N. Yoshida
From Communicating Machines to Graphical Choreographies