Team Automata: Communication and Compatibility

Maurice H. ter Beek

ISTI–CNR, Pisa, Italy

LIACS, Leiden University
15 July 2015

Team Automata: Communication and Compatibility

Maurice H. ter Beek

ISTI–CNR, Pisa, Italy

LIACS, Leiden University
15 July 2015

Outline

Some background. . .

Computer Supported Cooperative Work

- ? how do groups of people work together
- ? how can technology (groupware) assist

Ellis @ GROUP'97: Team Automata for Groupware Systems

Conceptual level: define concepts and terminology consistently

Architectural level: describe and compare groupware systems

Theoretical source of inspiration and foundations

- Lynch & Tuttle @ PODC'87: Input/Output Automata (IOA)
- ter Beek, Ellis, Kleijn & Rozenberg in CSCW journal (2003): Synchronizations in Team Automata for Groupware Systems

Component Automaton

$$\mathcal{C} = (Q, (\Sigma_{inp}, \Sigma_{out}, \Sigma_{int}), \delta, I)$$

- Q set of *states* $\qquad Q \cap \Sigma = \varnothing$
- $\Sigma = \Sigma_{inp} \cup \Sigma_{out} \cup \Sigma_{int}$ *alphabet* (partition)
- $\delta \subseteq Q \times \Sigma \times Q$ *transition relation* $\qquad q \xrightarrow{a} q'$
- $I \subseteq Q$ set of *initial states*

$$\left.\begin{array}{l} \Sigma_{inp} \ \textit{input actions} \\ \Sigma_{out} \ \textit{output actions} \end{array}\right\} \Sigma_{ext} \text{ externally observable}$$

Σ_{int} *internal actions* — cannot be observed

$\delta_a = \{(q, q') \mid (q, a, q') \in \delta\}$ set of *a-transitions*

$a \ \mathrm{en}_{\mathcal{C}} \ q$ if $\exists q' \in Q : (q, q') \in \delta_a$ $\qquad$ *a enabled* at q

Composable System

$\mathcal{S} = \{\mathcal{C}_1, \ldots, \mathcal{C}_n\}$ of component automata is a *composable system* if $\forall\, i \in \{1, \ldots, n\}$:

$$\Sigma_{i,int} \cap \bigcup_{j \in \{1,\ldots,n\} \setminus \{i\}} \Sigma_j = \varnothing$$

$\Rightarrow$ internal actions are private, uniquely associated to one component

Fix: $\mathcal{S} = \{\mathcal{C}_1, \ldots, \mathcal{C}_n\}$ is a composable system with $\Sigma = \bigcup_{i \in \{1,\ldots,n\}} (\Sigma_{i,inp} \cup \Sigma_{i,out} \cup \Sigma_{i,int})$

? how to form a team automaton over a composable system

Complete Transition Space

The *complete transition space* of $a \in \Sigma$ in $\mathcal{S}$ is

$$\Delta_a(\mathcal{S}) = \{(q, q') \in \prod_{i \in \{1,\ldots,n\}} Q_i \times \prod_{i \in \{1,\ldots,n\}} Q_i \mid$$

$$\exists j \in \{1, \ldots, n\} : \ (\mathrm{proj}_j(q), a, \mathrm{proj}_j(q')) \in \delta_j \ \wedge$$
$$\forall i \in \{1, \ldots, n\} : \ (\mathrm{proj}_i(q), a, \mathrm{proj}_i(q')) \in \delta_i \ \vee \ \mathrm{proj}_i(q) = \mathrm{proj}_i(q')\}$$

Hence, in every team transition:

1. at least one component acts according to its transition relation
2. all other components either join or are idle

$\Rightarrow$ Moreover, each choice of team transition relations $\delta_a \subseteq \Delta_a(\mathcal{S})$, $\forall a \in \Sigma$, defines a specific team automaton

Team Automata

$$\mathcal{T} = (\prod_{i\in\{1,\ldots,n\}} Q_i, (\Sigma_{inp}, \Sigma_{out}, \Sigma_{int}), \delta, \prod_{i\in\{1,\ldots,n\}} I_i)$$

is **a** *team automaton composed over* $\mathcal{S}$ if

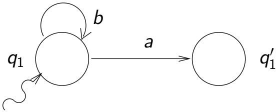
$$\left.\begin{array}{l} \bullet\ \Sigma_{int} = \bigcup_{i\in\{1,\ldots,n\}} \Sigma_{i,int} \\ \bullet\ \Sigma_{out} = \bigcup_{i\in\{1,\ldots,n\}} \Sigma_{i,out} \\ \bullet\ \Sigma_{inp} = (\bigcup_{i\in\{1,\ldots,n\}} \Sigma_{i,inp}) \setminus \Sigma_{out} \end{array}\right\} = \Sigma$$

- $\delta \subseteq \prod_{i\in\{1,\ldots,n\}} Q_i \times \Sigma \times \prod_{i\in\{1,\ldots,n\}} Q_i$ such that $\forall a \in \Sigma$: $\delta_a \subseteq \Delta_a(\mathcal{S})$ and $\delta_a = \Delta_a(\mathcal{S})$ if $a \in \Sigma_{int}$

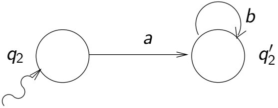
$\Rightarrow$ every team is a component and can thus be used to iteratively compose team automata

Examples

$\mathcal{C}_1$:

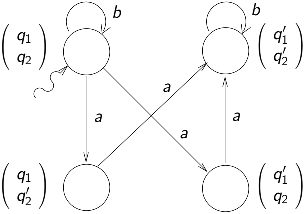


$\mathcal{C}_2$:

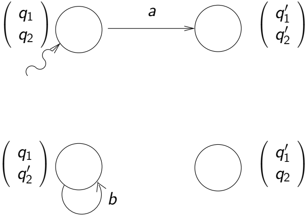


Team automata over $\{\mathcal{C}_1, \mathcal{C}_2\}$ defined by choosing their transitions:

$\mathcal{T}^{free}$:



$\mathcal{T}^{ai}$:



? who participates in $((q_1, q'_2), b, (q_1, q'_2))$ of $\mathcal{T}^{ai}$: only $\mathcal{C}_1$ or $\mathcal{C}_2$ or both

! adopt maximal interpretation: both execute b

Computations and behaviour (languages)

$$\alpha = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} a \begin{pmatrix} q_1 \\ q_2' \end{pmatrix} a \begin{pmatrix} q_1' \\ q_2' \end{pmatrix} \in \mathbf{C}_{\mathcal{T}^{free}}$$

$\pi_{\mathcal{C}_1}(\alpha) = q_1 a q_1' \in \mathbf{C}_{\mathcal{C}_1}$ $\qquad$ $\pi_{\mathcal{C}_2}(\alpha) = q_2 a q_2' \in \mathbf{C}_{\mathcal{C}_2}$

$\Rightarrow$ if $\mathcal{T}$ is a team automaton over $\mathcal{S}$, then $\forall i \in \{1, \ldots, n\}$:

$$\pi_{\mathcal{C}_i}(\mathbf{C}_{\mathcal{T}}) \subseteq \mathbf{C}_{\mathcal{C}_i}$$

$\mathrm{pres}_{\{a,b\}}(\alpha) = aa \in \mathbf{B}_{\mathcal{T}^{free}}$ $\qquad$ $aa \notin \mathbf{B}_{\mathcal{C}_1} \cup \mathbf{B}_{\mathcal{C}_2}$

$ab \notin \mathbf{B}_{\mathcal{T}^{free}} \cup \mathbf{B}_{\mathcal{T}^{ai}}$ $\qquad$ $ab \in \mathbf{B}_{\mathcal{C}_2}$

Subteams

The *subteam of* $\mathcal{T}$ *determined by* $J \subseteq \{1, \ldots, n\}$ is

$$SUB_J(\mathcal{T}) = (\prod_{j \in J} Q_j, (\Sigma_{J,inp}, \Sigma_{J,out}, \Sigma_{J,int}), \delta_J, \prod_{j \in J} I_j)$$

$$\left.\begin{array}{l} \bullet\ \Sigma_{J,int} = \bigcup_{j \in J} \Sigma_{j,int} \\[1em] \bullet\ \Sigma_{J,out} = \bigcup_{j \in J} \Sigma_{j,out} \\[1em] \bullet\ \Sigma_{J,inp} = (\bigcup_{j \in J} \Sigma_{j,inp}) \setminus \Sigma_{J,out} \end{array}\right\} = \Sigma_J$$

- δ_J is defined by, $\forall\, a \in \Sigma_J$,

$$(\delta_J)_a = \{(\mathrm{proj}_J(q), \mathrm{proj}_J(q')) \mid (q, q') \in \delta_a)\} \cap \Delta_a(\{\mathcal{C}_j \mid j \in J\})$$

$\Rightarrow$ $SUB_J(\mathcal{T})$ is a team automaton over $\{\mathcal{C}_j \mid j \in J\}$

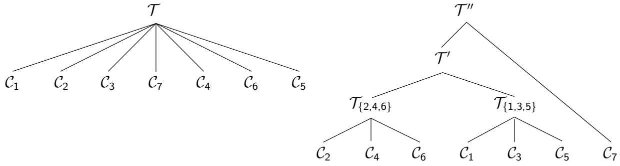
Teams over teams

Key notions so far

- Component automaton
- Composable system (of component automata)
- Team automaton (over a composable system)
- Subteam (of a team automaton)

Next: compositionality

- Iterated team automata
- Synchronizations in team automata



$\Rightarrow$ after *reordering* its elements, $\mathcal{T}''$ is a team automaton over $\{\mathcal{C}_1, \ldots, \mathcal{C}_7\}$

Synchronizations: *free*, *action*- and *state-indispensable*

An action a of a team automaton $\mathcal{T}$ over $\mathcal{S}$ is:

- free: if no a-transition of $\mathcal{T}$ contains the simultaneous execution of a by two or more components
- ai: if all components with a as an action participate in every a-transition of $\mathcal{T}$ (blocking can occur)
- si: if all a-transitions of $\mathcal{T}$ involve all components in which a is enabled in the current state (no blocking can occur)
- no: no restrictions whatsoever

Synchronizations: *free*, *action-* and *state-indispensable*

An action a of a team automaton $\mathcal{T}$ over $\mathcal{S}$ is:

$$\begin{array}{rl}
\text{free} & \delta_a = \{(q,q') \in \Delta_a(\mathcal{S}) \mid \\
& \quad \#\{i \in \{1,\ldots,n\} \mid a \in \Sigma_i \wedge (\mathrm{proj}_i(q), \mathrm{proj}_i(q')) \in \delta_{i,a}\} = 1\} \\
\text{ai} & \delta_a = \{(q,q') \in \Delta_a(\mathcal{S}) \mid \forall i \in \{1,\ldots,n\} : \\
& \quad a \in \Sigma_i \Rightarrow (\mathrm{proj}_i(q), \mathrm{proj}_i(q')) \in \delta_{i,a}\} \\
\text{si} & \delta_a = \{(q,q') \in \Delta_a(\mathcal{S}) \mid \forall i \in \{1,\ldots,n\} : \\
& \quad (a \in \Sigma_i \wedge a \ \mathrm{en}_{\mathcal{C}_i} \ \mathrm{proj}_i(q)) \Rightarrow (\mathrm{proj}_i(q), \mathrm{proj}_i(q')) \in \delta_{i,a}\} \\
\text{no} & \delta_a = \Delta_a(\mathcal{S})
\end{array}$$

Fix: team automaton $\mathcal{T} = (Q, (\Sigma_{inp}, \Sigma_{out}, \Sigma_{int}), \delta, I)$ over $\mathcal{S}$

recall $\Sigma = \Sigma_{inp} \cup \Sigma_{out} \cup \Sigma_{int}$ and let $a \in \Sigma$

Synchronizations: *free*, *action-* and *state-indispensable*

An action a of a team automaton $\mathcal{T}$ over $\mathcal{S}$ is:

$$\begin{array}{rl}
\text{free} & \delta_a = \{(q,q') \in \Delta_a(\mathcal{S}) \mid \\
& \quad \#\{i \in \{1,\ldots,n\} \mid a \in \Sigma_i \wedge (\text{proj}_i(q), \text{proj}_i(q')) \in \delta_{i,a}\} = 1\} \\
\text{ai} & \delta_a = \{(q,q') \in \Delta_a(\mathcal{S}) \mid \forall i \in \{1,\ldots,n\} : \\
& \quad a \in \Sigma_i \Rightarrow (\text{proj}_i(q), \text{proj}_i(q')) \in \delta_{i,a}\} \\
\text{si} & \delta_a = \{(q,q') \in \Delta_a(\mathcal{S}) \mid \forall i \in \{1,\ldots,n\} : \\
& \quad (a \in \Sigma_i \wedge a \ \text{en}_{\mathcal{C}_i} \ \text{proj}_i(q)) \Rightarrow (\text{proj}_i(q), \text{proj}_i(q')) \in \delta_{i,a}\} \\
\text{no} & \delta_a = \Delta_a(\mathcal{S})
\end{array}$$

Fix: team automaton $\mathcal{T} = (Q, (\Sigma_{inp}, \Sigma_{out}, \Sigma_{int}), \delta, I)$ over $\mathcal{S}$

recall $\Sigma = \Sigma_{inp} \cup \Sigma_{out} \cup \Sigma_{int}$ and let $a \in \Sigma$

Fixed team automata

Let $\mathcal{R}_a(\mathcal{S}) \subseteq \Delta_a(\mathcal{S})$, for all $a \in \Sigma$, and let $\mathcal{R}_\Sigma = \{\mathcal{R}_a(\mathcal{S}) \mid a \in \Sigma\}$
Then $\mathcal{T}$ is the $\mathcal{R}_\Sigma$-*team automaton* over $\mathcal{S}$ if $\forall a \in \Sigma : \delta_a = \mathcal{R}_a(\mathcal{S})$

$\Rightarrow$ maximal interpretation: $((q_1, q_2'), b, (q_1, q_2'))$ is part of the $\mathcal{R}^{ai}$-team automaton $\mathcal{T}^{ai}$ but not of the $\mathcal{R}^{free}$-team automaton $\mathcal{T}^{free}$ (omit Σ)

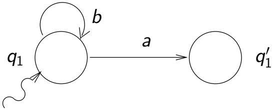
Synchronization strategies per action, but composition can be based on combinations, e.g. $(\mathcal{R}^{ai}_\Gamma \cup \mathcal{R}^{free}_{\Sigma\setminus\Gamma})$-team automaton

Most commonly used synchronization strategy in the literature is ai-based (e.g. IOA)

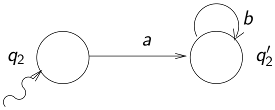
$\Rightarrow$ peer-to-peer and master-slave types of synchronization cannot be distinguished in IOA

Recall examples

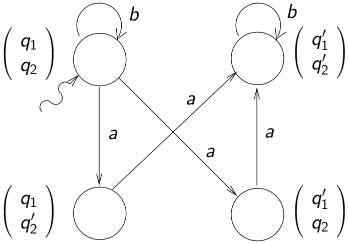
$\mathcal{C}_1$:



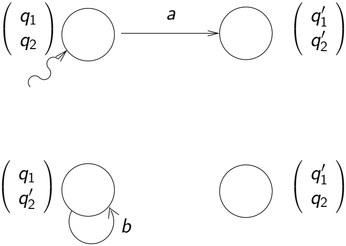
$\mathcal{C}_2$:



$\mathcal{T}^{free}$:

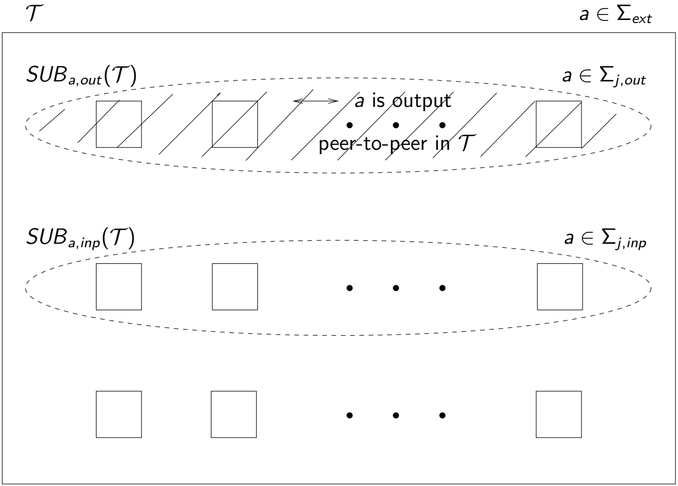


$\mathcal{T}^{ai}$:



Synchronizations: *peer-to-peer* collaboration

Action a is *strong* (*weak*) *output peer-to-peer* in $\mathcal{T}$ if a is *ai* (*si*) in $SUB_{a,out}(\mathcal{T})$

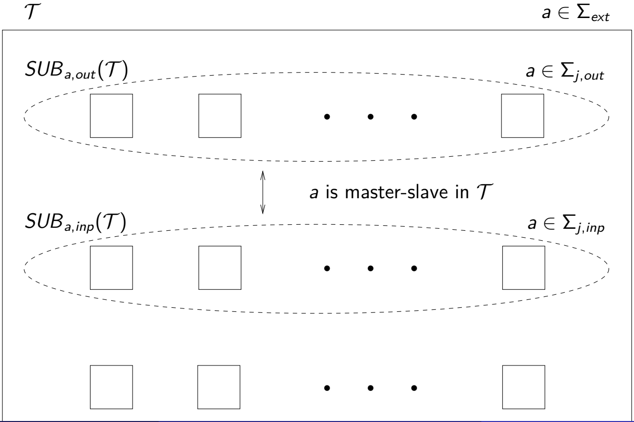


Action a is *strong* (*weak*) *input peer-to-peer* in $\mathcal{T}$ if a is *ai* (*si*) in $SUB_{a,in}(\mathcal{T})$

Synchronizations: *master-slave* cooperation

Action a is *strong (weak) master-slave* in $\mathcal{T}$ if

1. $SUB_{a,out}(\mathcal{T})$ participates in every a-transition of $\mathcal{T}$, i.e. slaves may never proceed on their own
2. $SUB_{a,inp}(\mathcal{T})$ participates in every a-transition of $\mathcal{T}$, i.e. slaves must obey the master (if enabled)



Compositionality of team automata

A team automaton over $\mathcal{S}$ is said to *satisfy compositionality* if its behaviour can be described in terms of that of its component automata:

There exists a set-theoretic operation that, if applied to the languages of the component automata in $\mathcal{S}$, yields the language of the team automaton

Compositionality of fixed team automata

? which (combinations of) synchronization strategies result in team automata satisfying compositionality

Compositionality of team automata

A team automaton over $\mathcal{S}$ is said to *satisfy compositionality* if its behaviour can be described in terms of that of its component automata:

There exists a set-theoretic operation that, if applied to the languages of the component automata in $\mathcal{S}$, yields the language of the team automaton

Compositionality of fixed team automata

? which (combinations of) synchronization strategies result in team automata satisfying compositionality

Motivation for (synchronized) shuffles

- Component automaton: ordinary FSA without final states and its alphabet partitioned into input, output and internal actions
  - $\Rightarrow$ prefix-closed languages that may contain *infinite* words
- Team automaton: (iterative) composition of component automata that may collaborate by synchronizing on shared actions
  - $\Rightarrow$ languages may be *unfair* in the sense that a component may execute ad infinitum, never giving others a turn
- Compositionality: language of a team automaton can be defined in terms of the languages of its constituting component automata
  - $\Rightarrow$ (synchronized) shuffling of languages must be *associative*

Shuffling: basic definitions

Let $u, v \in \Delta^\infty$. Then

1. $w \in \Delta^\infty$ is a *fair shuffle* of u and v if $w = u_1 v_1 u_2 v_2 \cdots$, where $u_i, v_i \in \Delta^*$, for all $i \geq 1$, are such that $u = u_1 u_2 \cdots$ and $v = v_1 v_2 \cdots$
2. $w \in \Delta^\infty$ is a *shuffle* of u and v if either
   1. w is a fair shuffle of u and v, or
   2. $w = u_1 v_1 u_2 v_2 \cdots$, where $u_i, v_i \in \Delta^*$, for all $i \geq 1$, and either $u_1 u_2 \cdots \in \mathrm{pref}(u)$ and $v = v_1 v_2 \cdots \in \Delta^\omega$ or $u = u_1 u_2 \cdots \in \Delta^\omega$ and $v_1 v_2 \cdots \in \mathrm{pref}(v)$

Notations:

$$\begin{aligned}
u \,|||\, v &= \{ w \in \Delta^\infty : w \text{ is a fair shuffle of } u \text{ and } v \} \\
u \,||\, v &= \{ w \in \Delta^\infty : w \text{ is a shuffle of } u \text{ and } v \} \\
L_1 \,|||\, L_2 &= \textstyle\bigcup_{u \in L_1, v \in L_2} \; u \,|||\, v \\
L_1 \,||\, L_2 &= \textstyle\bigcup_{u \in L_1, v \in L_2} \; u \,||\, v
\end{aligned}$$

Examples: fair and unfair shuffling

$a \,||\, b = \{ab, ba\}$ $\qquad$ $a^2 \,||\, b = \{a^2 b, aba, ba^2\}$

$a^n \,||\, b = \{\, a^i b a^j : i, j \geq 0,\ i + j = n \,\}$

$\Rightarrow$ every shuffle of a^n and b is fair

$a^\omega \,|||\, b = \{\, a^i b a^\omega : i \geq 0 \,\}$ $\qquad$ $a^\omega \,||\, b = (a^\omega \,|||\, b) \cup a^\omega$

$\Rightarrow$ a^ω is an unfair shuffle of a^ω and b

$a^\omega \,|||\, a = a^\omega \,||\, a = a^\omega$

$\Rightarrow$ infinite words need not result in unfair shuffles

Shuffling: basic observations and our result

Let $u, v, w \in \Delta^\infty$. Then

1. (fair) shuffling is *commutative*: $u \,|||\, v = v \,|||\, u$ and $u \,||\, v = v \,||\, u$
2. fair shuffling is *associative*: $u \,|||\, (v \,|||\, w) = (u \,|||\, v) \,|||\, w$
3. $\mathrm{pref}(u) \,||\, \mathrm{pref}(v) = \mathrm{pref}(u \,|||\, v) = \mathrm{pref}(u \,||\, v) = \mathrm{pref}(u) \,|||\, \mathrm{pref}(v)$

$\Rightarrow$ $\mathrm{pref}(a^\omega \,|||\, b) = \mathrm{pref}(a^\omega \,||\, b) = \{\, a^i b a^\omega : i \geq 0 \,\} \cup a^*$, but recall that $a^\omega \,|||\, b \neq a^\omega \,||\, b$

ter Beek & Kleijn in TCS (2007): Infinite Unfair Shuffles and Associativity

4. shuffling is *associative*: $u \,||\, (v \,||\, w) = (u \,||\, v) \,||\, w$

Shuffling: basic observations and our result

Let $u, v, w \in \Delta^\infty$. Then

1. (fair) shuffling is *commutative*: $u \mathbin{|||} v = v \mathbin{|||} u$ and $u \mathbin{||} v = v \mathbin{||} u$
2. fair shuffling is *associative*: $u \mathbin{|||} (v \mathbin{|||} w) = (u \mathbin{|||} v) \mathbin{|||} w$
3. $\mathrm{pref}(u) \mathbin{||} \mathrm{pref}(v) = \mathrm{pref}(u \mathbin{|||} v) = \mathrm{pref}(u \mathbin{||} v) = \mathrm{pref}(u) \mathbin{|||} \mathrm{pref}(v)$

$\Rightarrow$ $\mathrm{pref}(a^\omega \mathbin{|||} b) = \mathrm{pref}(a^\omega \mathbin{||} b) = \{\, a^i b a^\omega : i \geq 0 \,\} \cup a^*$, but recall that $a^\omega \mathbin{|||} b \neq a^\omega \mathbin{||} b$

ter Beek & Kleijn in TCS (2007): Infinite Unfair Shuffles and Associativity

4. shuffling is *associative*: $u \mathbin{||} (v \mathbin{||} w) = (u \mathbin{||} v) \mathbin{||} w$

Synchronized shuffling: definitions and examples

Let $u, v \in \Delta^\infty$. Then a word $w \in \Delta^\infty$ is a *synchronized shuffle* (*S-shuffle*) *on* Γ of u and v if

1. either $u = u_1 x_1 \cdots x_{n-1} u_n$ and $v = v_1 x_1 \cdots x_{n-1} v_n$, with $u_i, v_i \in (\Delta \setminus \Gamma)^*$ and $x_i \in \Gamma$ for $1 \leq i \leq n-1$, and $u_n, v_n \in (\Delta \setminus \Gamma)^\infty$, in which case $w = w_1 x_1 \cdots x_{n-1} w_n$ with $w_i \in u_i \,||\, v_i$ for all $1 \leq i \leq n$ (and w is *fair* whenever $w_n \in (u_n \,|||\, v_n)$)
2. or $u = u_1 x_1 u_2 x_2 \cdots$ and $v = v_1 x_1 v_2 x_2 \cdots$, with $u_i, v_i \in (\Delta \setminus \Gamma)^*$ and $x_i \in \Gamma$ for all $i \geq 1$, in which case $w = w_1 x_1 w_2 x_2 \cdots$ with $w_i \in u_i \,||\, v_i$ for all $i \geq 1$ (and w is *fair*)

$$
\begin{aligned}
u \,||^\Gamma\, v &= \{\, w \in \Delta^\infty : w \text{ is an S-shuffle on } \Gamma \text{ of } u \text{ and } v \,\} \\
u \,|||^\Gamma\, v &= \{\, w \in \Delta^\infty : w \text{ is a fair S-shuffle on } \Gamma \text{ of } u \text{ and } v \,\}
\end{aligned}
$$

$abc \,||^{\{c\}}\, cd = \{abcd\}$, but $abcd \notin abc \,||\, cd$ and $abc \,||^{\{b,c\}}\, cd = \varnothing$

Full Synchronized Shuffles

A *full S-shuffle* of two words is an S-shuffle in which all and only the symbols shared by their alphabets act as synchronizing symbols

Let $u \in \Delta_1^\infty$, $v \in \Delta_2^\infty$, and $w \in (\Delta_1 \cup \Delta_2)^\infty$. Then w is a *full S-shuffle* (*fS-shuffle*) of u and v w.r.t. Δ_1 and Δ_2 if w is an S-shuffle on $\Delta_1 \cap \Delta_2$ of u and v (*fair* if w is a fair S-shuffle on $\Delta_1 \cap \Delta_2$ of u and v)

$$u \,{}_{\Delta_1}\underline{|||}_{\Delta_2}\, v = \{w \in (\Delta_1 \cup \Delta_2)^\infty : w \text{ is a fair fS-shuffle of } u \text{ and } v \text{ w.r.t. } \Delta_1 \text{ and } \Delta_2\}$$
$$u \,{}_{\Delta_1}\underline{||}_{\Delta_2}\, v = \{w \in (\Delta_1 \cup \Delta_2)^\infty : w \text{ is an fS-shuffle of } u \text{ and } v \text{ w.r.t. } \Delta_1 \text{ and } \Delta_2\}$$

If $\Delta = \{a, b, c, d\}$, then $abc \,{}_{\Delta}\underline{||}_{\Delta}\, cd = \varnothing$, but
with $\Delta' = \{a, b, c\}$ and $\Delta'' = \{c, d\}$, $abc \,{}_{\Delta'}\underline{||}_{\Delta''}\, cd = abc \,||^{\{c\}}\, cd = \{abcd\}$
Also, $a^\omega \,{}_{\Delta}\underline{||}_{\Delta}\, b = a^\omega \,{}_{\Delta}\underline{|||}_{\Delta}\, b = \varnothing$, while
$a^\omega \,{}_{\{a\}}\underline{|||}_{\{b\}}\, b = a^\omega \,|||\, b = \{\, w \in \{a, b\}^\omega : w \text{ has one occurrence of } b \,\}$ and
$a^\omega \,{}_{\{a\}}\underline{||}_{\{b\}}\, b = a^\omega \,||\, b = (a^\omega \,{}_{\{a\}}\underline{|||}_{\{b\}}\, b) \cup \{a^\omega\}$

Relaxed Synchronized Shuffles

A *relaxed S-shuffle* of two words is an S-shuffle in which only some of the symbols shared by their alphabets are synchronizing symbols

Let $u \in \Delta_1^\infty$, $v \in \Delta_2^\infty$ and $w \in (\Delta_1 \cup \Delta_2)^\infty$. Then w is a *relaxed S-shuffle* (*rS-shuffle*) *on* Γ of u and v w.r.t. Δ_1 and Δ_2 if w is an S-shuffle on $\Gamma \cap \Delta_1 \cap \Delta_2$ of u and v (*fair* if w is a fair S-shuffle on $\Gamma \cap \Delta_1 \cap \Delta_2$ of u and v)

$$u \,{}_{\Delta_1}\underline{|||}^{\Gamma}_{\Delta_2}\, v = \{ w \in (\Delta_1 \cup \Delta_2)^\infty : w \text{ is a fair rS-shuffle on } \Gamma \text{ of } u \text{ and } v \text{ w.r.t. } \Delta_1 \text{ and } \Delta_2 \}$$

$$u \,{}_{\Delta_1}\underline{||}^{\Gamma}_{\Delta_2}\, v = \{ w \in (\Delta_1 \cup \Delta_2)^\infty : w \text{ is an rS-shuffle on } \Gamma \text{ of } u \text{ and } v \text{ w.r.t. } \Delta_1 \text{ and } \Delta_2 \}$$

$(a \,{}_{\{a\}}\underline{|||}^{\{a\}}_{\{b\}}\, b) \,{}_{\{a,b\}}\underline{|||}^{\{b\}}_{\{a,b\}}\, ab = \{ab, ba\} \,{}_{\{a,b\}}\underline{||}^{\{b\}}_{\{a,b\}}\, ab = \{aab, aba\}$,
which differs from $a \,{}_{\{a\}}\underline{|||}^{\{a\}}_{\{a,b\}}\, (b \,{}_{\{b\}}\underline{|||}^{\{b\}}_{\{a,b\}}\, ab) = a \,{}_{\{a\}}\underline{|||}^{\{a\}}_{\{a,b\}}\, ab = \{ab\}$

Arbitrary Synchronized Shuffles

An *arbitrary S-shuffle* of two words is an S-shuffle in which synchronization on occurrences of the specified synchronizing symbols is only an option

Let $u, v \in \Delta^\infty$. Then a word $w \in \Delta^\infty$ is an *arbitrary S-shuffle* (*aS-shuffle*) *on* Γ of u and v if

1. either $u = u_1 x_1 u_2 x_2 \cdots x_{n-1} u_n$ and $v = v_1 x_1 v_2 x_2 \cdots x_{n-1} v_n$, with $u_i, v_i \in \Delta^*$ and $x_i \in \Gamma$, for $1 \leq i \leq n-1$, and $u_n, v_n \in \Delta^\infty$, in which case $w = w_1 x_1 w_2 x_2 \cdots x_{n-1} w_n$ with $w_i \in u_i \,||\, v_i$ for all $1 \leq i \leq n$ (and w is *fair* whenever $w_n \in (u_n \,|||\, v_n)$)
2. or $u = u_1 x_1 u_2 x_2 \cdots$ and $v = v_1 x_1 v_2 x_2 \cdots$, with $u_i, v_i \in \Delta^*$ and $x_i \in \Gamma$, for all $i \geq 1$, in which case $w = w_1 x_1 w_2 x_2 \cdots$ with $w_i \in u_i \,||\, v_i$ for all $i \geq 1$ (and w is *fair*)

$$\begin{aligned} u \wr\wr\wr^{\Gamma} v &= \{ w \in \Delta^\infty : w \text{ is a fair aS-shuffle on } \Gamma \text{ of } u \text{ and } v \} \\ u \wr\wr^{\Gamma} v &= \{ w \in \Delta^\infty : w \text{ is an aS-shuffle on } \Gamma \text{ of } u \text{ and } v \} \end{aligned}$$

$cab \in ((ca \wr\wr^{\{a\}} ab) \wr\wr^{\{c\}} c)$, whereas all the words in $(ca \wr\wr^{\{a\}} (ab \wr\wr^{\{c\}} c))$ contain two occurrences of c

Weak Synchronized Shuffles

A *weak S-shuffle* of two words is an S-shuffle in which synchronization on occurrences of the specified synchronizing symbols is required, if possible

Let $u, v \in \Delta^\infty$. Then a word $w \in \Delta^\infty$ is a *weak S-shuffle* (*wS-shuffle*) *on* Γ of u and v if

1. either $u = u_1 x_1 u_2 x_2 \cdots x_{n-1} u_n$ and $v = v_1 x_1 v_2 x_2 \cdots x_{n-1} v_n$, with $u_i, v_i \in \Delta^*$, $alph(u_i) \cap alph(v_i) \cap \Gamma = \varnothing$ and $x_i \in \Gamma$ for $1 \leq i \leq n-1$, and $u_n, v_n \in \Delta^\infty$, in which case $w = w_1 x_1 w_2 x_2 \cdots x_{n-1} w_n$ with $w_i \in u_i \,\|\, v_i$ for all $1 \leq i \leq n$ (and w is *fair* whenever $w_n \in (u_n \,|||\, v_n)$)
2. or $u = u_1 x_1 u_2 x_2 \cdots$ and $v = v_1 x_1 v_2 x_2 \cdots$, with $u_i, v_i \in \Delta^*$, $alph(u_i) \cap alph(v_i) \cap \Gamma = \varnothing$ and $x_i \in \Gamma$, for all $i \geq 1$, in which case $w = w_1 x_1 w_2 x_2 \cdots$ with $w_i \in u_i \,\|\, v_i$ for all $i \geq 1$ (and w is *fair*)

$$
\begin{aligned}
u \,\underset{\sim}{\|}^{\Gamma}\, v &= \{\, w \in \Delta^\infty : w \text{ is a wS-shuffle on } \Gamma \text{ of } u \text{ and } v \,\} \\
u \,\underset{\sim}{|||}^{\Gamma}\, v &= \{\, w \in \Delta^\infty : w \text{ is a fair wS-shuffle on } \Gamma \text{ of } u \text{ and } v \,\}
\end{aligned}
$$

$abc \,\underset{\sim}{\|}^{\{c\}}\, cd = \{abcd\}$

$a^\omega \,\underset{\sim}{\|}^{\{a\}}\, a = a^\omega \,\underset{\sim}{|||}^{\{a\}}\, a = a^\omega \,\|\, a = a^\omega \,|||\, a = a^\omega$, but $a^\omega \,\|^{\{a\}}\, a = a^\omega \,|||^{\{a\}}\, a = \varnothing$

Our observations on commutativity

ter Beek & Kleijn in FI (2009): Associativity of Infinite Synchronized Shuffles and Team Automata

1. (fair) S-shuffling is
2. (fair) fS-shuffling is: $u \,_{\Delta_1}\underline{|||}_{\Delta_2}\, v = v \,_{\Delta_2}\underline{|||}_{\Delta_1}\, u$ and $u \,_{\Delta_1}\underline{||}_{\Delta_2}\, v = v \,_{\Delta_2}\underline{||}_{\Delta_1}\, u$
3. (fair) rS-shuffling is: $u \,_{\Delta_1}\underline{|||}^{\Gamma}_{\Delta_2}\, v = v \,_{\Delta_2}\underline{|||}^{\Gamma}_{\Delta_1}\, u$ and $u \,_{\Delta_1}\underline{||}^{\Gamma}_{\Delta_2}\, v = v \,_{\Delta_2}\underline{||}^{\Gamma}_{\Delta_1}\, u$
4. (fair) aS-shuffling is

ter Beek, Martín-Vide & Mitrana in TCS (2005): Synchronized Shuffles

5. (fair) wS-shuffling is

Our observations on commutativity

ter Beek & Kleijn in FI (2009): Associativity of Infinite Synchronized Shuffles and Team Automata

1. (fair) S-shuffling is
2. (fair) fS-shuffling is: $u \;{}_{\Delta_1}\underline{|||}_{\Delta_2}\; v = v \;{}_{\Delta_2}\underline{|||}_{\Delta_1}\; u$ and $u \;{}_{\Delta_1}\underline{||}_{\Delta_2}\; v = v \;{}_{\Delta_2}\underline{||}_{\Delta_1}\; u$
3. (fair) rS-shuffling is: $u \;{}_{\Delta_1}\underline{|||}^{\Gamma}_{\Delta_2}\; v = v \;{}_{\Delta_2}\underline{|||}^{\Gamma}_{\Delta_1}\; u$ and $u \;{}_{\Delta_1}\underline{||}^{\Gamma}_{\Delta_2}\; v = v \;{}_{\Delta_2}\underline{||}^{\Gamma}_{\Delta_1}\; u$
4. (fair) aS-shuffling is

ter Beek, Martín-Vide & Mitrana in TCS (2005): Synchronized Shuffles

5. (fair) wS-shuffling is

Our results on associativity

ter Beek & Kleijn in FI (2009)

1. (fair) S-shuffling is: $u \;|||^{\Gamma}\; (v \;|||^{\Gamma}\; w) = (u \;|||^{\Gamma}\; v) \;|||^{\Gamma}\; w$ and $u \,||^{\Gamma}\, (v \,||^{\Gamma}\, w) = (u \,||^{\Gamma}\, v) \,||^{\Gamma}\, w$
2. fair fS-shuffling is: $u \,{}_{\Delta_1}\underline{|||}_{\Delta_2 \cup \Delta_3} (v \,{}_{\Delta_2}\underline{|||}_{\Delta_3}\, w) = (u \,{}_{\Delta_1}\underline{|||}_{\Delta_2}\, v) \,{}_{\Delta_1 \cup \Delta_2}\underline{|||}_{\Delta_3}\, w$
3. unfair fS-shuffling is *not*, but it is in case of prefix-closed languages: $(\{u\} \cup \mathrm{pref}(u)) \,{}_{\Delta_1}\underline{||}_{\Delta_2 \cup \Delta_3} ((\{v\} \cup \mathrm{pref}(v)) \,{}_{\Delta_2}\underline{||}_{\Delta_3} (\{w\} \cup \mathrm{pref}(w))) = ((\{u\} \cup \mathrm{pref}(u)) \,{}_{\Delta_1}\underline{||}_{\Delta_2} (\{v\} \cup \mathrm{pref}(v))) \,{}_{\Delta_1 \cup \Delta_2}\underline{||}_{\Delta_3} (\{w\} \cup \mathrm{pref}(w))$
4. fair rS-shuffling is: $u \,{}_{\Delta_1}\underline{|||}^{\Gamma}_{\Delta_2 \cup \Delta_3} (v \,{}_{\Delta_2}\underline{|||}^{\Gamma}_{\Delta_3}\, w) = (u \,{}_{\Delta_1}\underline{|||}^{\Gamma}_{\Delta_2}\, v) \,{}_{\Delta_1 \cup \Delta_2}\underline{|||}^{\Gamma}_{\Delta_3}\, w$
5. unfair rS-shuffling is *not*, but it is in case of prefix-closed languages: $(\{u\} \cup \mathrm{pref}(u)) \,{}_{\Delta_1}\underline{||}^{\Gamma}_{\Delta_2 \cup \Delta_3} ((\{v\} \cup \mathrm{pref}(v)) \,{}_{\Delta_2}\underline{||}^{\Gamma}_{\Delta_3} (\{w\} \cup \mathrm{pref}(w))) = ((\{u\} \cup \mathrm{pref}(u)) \,{}_{\Delta_1}\underline{||}^{\Gamma}_{\Delta_2} (\{v\} \cup \mathrm{pref}(v))) \,{}_{\Delta_1 \cup \Delta_2}\underline{||}^{\Gamma}_{\Delta_3} (\{w\} \cup \mathrm{pref}(w))$
6. (fair) aS-shuffling is: $u \wr\wr\wr^{\Gamma} (v \wr\wr\wr^{\Gamma} w) = (u \wr\wr\wr^{\Gamma} v) \wr\wr\wr^{\Gamma} w$ and $u \wr\wr^{\Gamma} (v \wr\wr^{\Gamma} w) = (u \wr\wr^{\Gamma} v) \wr\wr^{\Gamma} w$

ter Beek, Martín-Vide & Mitrana in TCS (2005)

7. (fair) wS-shuffling is *not*

Our results on associativity

ter Beek & Kleijn in FI (2009)

1. (fair) S-shuffling is: $u \mathbin{|||}^{\Gamma} (v \mathbin{|||}^{\Gamma} w) = (u \mathbin{|||}^{\Gamma} v) \mathbin{|||}^{\Gamma} w$ and $u \mathbin{||}^{\Gamma} (v \mathbin{||}^{\Gamma} w) = (u \mathbin{||}^{\Gamma} v) \mathbin{||}^{\Gamma} w$
2. fair fS-shuffling is: $u \;{}_{\Delta_1}\underline{|||}_{\Delta_2 \cup \Delta_3} (v \;{}_{\Delta_2}\underline{|||}_{\Delta_3} w) = (u \;{}_{\Delta_1}\underline{|||}_{\Delta_2} v) \;{}_{\Delta_1 \cup \Delta_2}\underline{|||}_{\Delta_3} w$
3. unfair fS-shuffling is *not*, but it is in case of prefix-closed languages: $(\{u\} \cup \mathrm{pref}(u)) \;{}_{\Delta_1}\underline{||}_{\Delta_2 \cup \Delta_3} ((\{v\} \cup \mathrm{pref}(v)) \;{}_{\Delta_2}\underline{||}_{\Delta_3} (\{w\} \cup \mathrm{pref}(w))) = ((\{u\} \cup \mathrm{pref}(u)) \;{}_{\Delta_1}\underline{||}_{\Delta_2} (\{v\} \cup \mathrm{pref}(v))) \;{}_{\Delta_1 \cup \Delta_2}\underline{||}_{\Delta_3} (\{w\} \cup \mathrm{pref}(w))$
4. fair rS-shuffling is: $u \;{}_{\Delta_1}\underline{|||}^{\Gamma}_{\Delta_2 \cup \Delta_3} (v \;{}_{\Delta_2}\underline{|||}^{\Gamma}_{\Delta_3} w) = (u \;{}_{\Delta_1}\underline{|||}^{\Gamma}_{\Delta_2} v) \;{}_{\Delta_1 \cup \Delta_2}\underline{|||}^{\Gamma}_{\Delta_3} w$
5. unfair rS-shuffling is *not*, but it is in case of prefix-closed languages: $(\{u\} \cup \mathrm{pref}(u)) \;{}_{\Delta_1}\underline{||}^{\Gamma}_{\Delta_2 \cup \Delta_3} ((\{v\} \cup \mathrm{pref}(v)) \;{}_{\Delta_2}\underline{||}^{\Gamma}_{\Delta_3} (\{w\} \cup \mathrm{pref}(w))) = ((\{u\} \cup \mathrm{pref}(u)) \;{}_{\Delta_1}\underline{||}^{\Gamma}_{\Delta_2} (\{v\} \cup \mathrm{pref}(v))) \;{}_{\Delta_1 \cup \Delta_2}\underline{||}^{\Gamma}_{\Delta_3} (\{w\} \cup \mathrm{pref}(w))$
6. (fair) aS-shuffling is: $u \wr\wr\wr^{\Gamma} (v \wr\wr\wr^{\Gamma} w) = (u \wr\wr\wr^{\Gamma} v) \wr\wr\wr^{\Gamma} w$ and $u \wr\wr^{\Gamma} (v \wr\wr^{\Gamma} w) = (u \wr\wr^{\Gamma} v) \wr\wr^{\Gamma} w$

ter Beek, Martín-Vide & Mitrana in TCS (2005)

7. (fair) wS-shuffling is *not*

Compositionality: a negative result

Recall: a team automaton over $\mathcal{S}$ is said to *satisfy compositionality* if its behaviour can be described in terms of that of its component automata:

There exists a set-theoretic operation that, if applied to the languages of the component automata in $\mathcal{S}$, yields the language of the team automaton

ter Beek, Gadducci & Janssens in ENTCS (2007):
A calculus for team automata

Let $n = 2$ and let $\mathcal{T}$ be the $\mathcal{R}^{si}$-team automaton over $\mathcal{S}$.
Then there exists no set-theoretic operation $|$ on languages such that

$$\mathbf{B}^{\Sigma}_{\mathcal{T}} = \mathbf{B}^{\Sigma_1}_{\mathcal{C}_1} \mid \mathbf{B}^{\Sigma_2}_{\mathcal{C}_2}$$

Compositionality: a negative result

Recall: a team automaton over $\mathcal{S}$ is said to *satisfy compositionality* if its behaviour can be described in terms of that of its component automata:

There exists a set-theoretic operation that, if applied to the languages of the component automata in $\mathcal{S}$, yields the language of the team automaton

ter Beek, Gadducci & Janssens in ENTCS (2007): A calculus for team automata

Let $n = 2$ and let $\mathcal{T}$ be the $\mathcal{R}^{si}$-team automaton over $\mathcal{S}$.
Then there exists no set-theoretic operation $|$ on languages such that

$$\mathbf{B}^{\Sigma}_{\mathcal{T}} = \mathbf{B}^{\Sigma_1}_{\mathcal{C}_1} \mid \mathbf{B}^{\Sigma_2}_{\mathcal{C}_2}$$

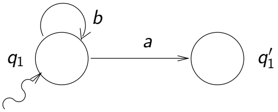
Our results on compositionality

ter Beek & Kleijn @ FM'03: Team Automata Satisfying Compositionality and in FI (2009): Associativity of Infinite Synchronized Shuffles and Team Automata

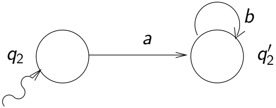
1. Let $\mathcal{T}$ be the $\mathcal{R}^{free}$-team automaton over $\mathcal{S}$. If $\mathcal{S}$ is loop-limited, then
$$\mathbf{B}^{\Sigma}_{\mathcal{T}} = ||_{i \in \{1,\ldots,n\}} \mathbf{B}^{\Sigma_i}_{\mathcal{C}_i}$$
2. Let $\mathcal{T}$ be the $\mathcal{R}^{ai}$-team automaton over $\mathcal{S}$. Then
$$\mathbf{B}^{\Sigma}_{\mathcal{T}} = \underline{||}_{\{\Sigma_i \mid i \in \{1,\ldots,n\}\}} \mathbf{B}^{\Sigma_i}_{\mathcal{C}_i}$$
3. Let $\mathcal{T}$ be the $(\mathcal{R}^{ai}_{\Gamma} \cup \mathcal{R}^{free}_{\Sigma\setminus\Gamma})$-team automaton over $\mathcal{S}$. If $\mathcal{S}$ is $(\Sigma \setminus \Gamma)$-loop-limited, then
$$\mathbf{B}^{\Sigma}_{\mathcal{T}} = \underline{||}^{\Gamma}_{\{\Sigma_i \mid i \in \{1,\ldots,n\}\}} \mathbf{B}^{\Sigma_i}_{\mathcal{C}_i}$$
4. Let $\mathcal{T}$ be the $\mathcal{R}^{no}$-team automaton over $\mathcal{S}$. Then
$$\mathbf{B}^{\Sigma}_{\mathcal{T}} = \wr\wr^{\Sigma}_{\{i \in \{1,\ldots,n\}\}} \mathbf{B}^{\Sigma_i}_{\mathcal{C}_i}$$

Recall examples

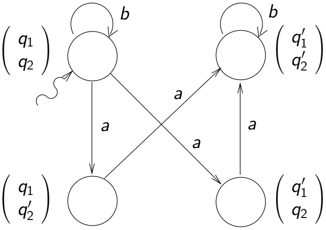
$\mathcal{C}_1$:



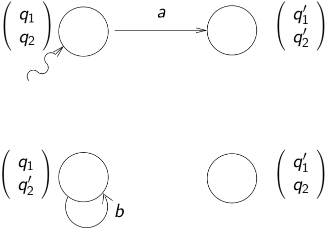
$\mathcal{C}_2$:



$\mathcal{T}^{free}$:



$\mathcal{T}^{ai}$:



$$\Rightarrow \mathbf{B}_{\mathcal{T}^{ai}} = \{\lambda, a\} = \{b^n, b^n a \mid n \geq 0\} \; {}_{\Sigma_1}\underline{\|}_{\Sigma_2} \{\lambda, ab^n \mid n \geq 0\} = \mathbf{B}_{\mathcal{C}_1} {}_{\Sigma_1}\underline{\|}_{\Sigma_2} \mathbf{B}_{\mathcal{C}_2}$$

Communication and compatibility

Correct-by-construction systems of systems

A concept of compatibility is needed to establish that components within a system (or a system and its environment) always communicate correctly

Compatibility

- represents an aspect of successful communication behaviour, a necessary ingredient for the correctness of a distributed system
- compatibility failures in a system model may reveal problems in the design of some component(s) to be repaired before implementation

Carmona & Cortadella @ FMCAD'02: Input/Output Compatibility of Reactive Systems

- define compatibility of two components (in a closed environment) that should engage in a dialogue free from *message loss* and *deadlocks* (as these may have severe repercussions on reliability, safety and security)

Communication and compatibility

Correct-by-construction systems of systems

A concept of compatibility is needed to establish that components within a system (or a system and its environment) always communicate correctly

Compatibility

- represents an aspect of successful communication behaviour, a necessary ingredient for the correctness of a distributed system
- compatibility failures in a system model may reveal problems in the design of some component(s) to be repaired before implementation

Carmona & Cortadella @ FMCAD'02: Input/Output Compatibility of Reactive Systems

- define compatibility of two components (in a closed environment) that should engage in a dialogue free from *message loss* and *deadlocks* (as these may have severe repercussions on reliability, safety and security)

Compatibility

Message loss and deadlock

- message loss occurs when one component sends a message that cannot be received as input by another component
- deadlock occurs when a component is indefinitely waiting for a message that never arrives

Carmona & Kleijn in TCS (2013): Compatibility in a multi-component environment

- compatibility generalized to multi-component systems modelled as team automata, within which communication may take place between more than two components at the same time (e.g. broadcasting)
- emphasis on ai-based synchronizations (i.e. $\mathcal{R}^{ai}$-team automata)

ter Beek, Carmona & Kleijn in ERCIM News (2015): Communication and Compatibility in Systems of Systems: Correctness-by-Construction

A research proposal for investigating other composition strategies

Compatibility

Message loss and deadlock

- message loss occurs when one component sends a message that cannot be received as input by another component
- deadlock occurs when a component is indefinitely waiting for a message that never arrives

Carmona & Kleijn in TCS (2013): Compatibility in a multi-component environment

- compatibility generalized to multi-component systems modelled as team automata, within which communication may take place between more than two components at the same time (e.g. broadcasting)
- emphasis on ai-based synchronizations (i.e. $\mathcal{R}^{ai}$-team automata)

ter Beek, Carmona & Kleijn in ERCIM News (2015): Communication and Compatibility in Systems of Systems: Correctness-by-Construction

A research proposal for investigating other composition strategies

Future work

Generalize definition of compatibility in $\mathcal{R}^{ai}$-team automata

1. non-communicating progress: internal actions do not alter enabledness
2. receptiveness: every output action that is sent, is received as input (i.e. communication never fails, *input-enabledness* in IOA)
3. deadlock-freeness: whenever a component is waiting for input, then the team automaton cannot 'terminate'

Study influence of team automata being *state-sharing*

! the occurrence of a synchronization (in general) depends also on the current states of the components *not* involved in the synchronization

Compatibility in team automata with master-slave synchronizations

? how is compatibility affected when slaves are added

? in what way does compatibility depend on the collaboration among slaves