Coinduction in concurrent timed systems

Jan Komenda

Institute of Mathematics, Czech Academy of Sciences, Brno, Czech Republic

10th Workskop on Coalgebraic Methods in Computer Science (CMCS'10)

Paphos, Cyprus, March 28, 2010

Outline

Outline

Coalgebra and automata theory

- Labelled transition systems (incl. timed) are coalgebras
- Various automata are coalgebras of suitable set functors
- Weighted automata (automata with multiplicities) are coalgebras
- Deterministic automata have simple final coalgebras: e.g. languages, formal power series (Moore automata)
- 2 ways of coding concurrency using weighted automata : nondeterminism (heap automata) and synchronous composition (like timed automata)
- Classes of timed automata (product interval automata) and corresponding classes of Petri nets
- Behaviors of synchronous compositions

Deterministic K-weighted automata as coalgebras

- Mealy automata (inputs in A, outputs in K) are coalgebras (S, t), S set of states, $t : S \to (K \times S)^A$ transition function.
- A partial MA is (S, t), where $t : S \to (1 + (K \times S))^A$ with $1 = \{\emptyset\}$.
- Partial Mealy automata are deterministic K-weighted automata with all states final
- A deterministic K-weighted automaton is viewed as partial Mealy automaton (S, t) above.
- Examples of multiplicity semirings :
    - $K = \mathbb{R}_{min} = (\mathbb{R} \cup \{\infty\}, min, +, \infty, 0)$ ... (min,+)-automata (price)
    - $\mathbb{R}_{\max} = (\mathbb{R} \cup \{-\infty\}, \max, +, -\infty, 0)$ ... (max,+)-automata (time)
    - $K = \mathcal{I}_{max}^{max} = (\mathbb{R}_{\max} \times \mathbb{R}_{\max} \cup (-\infty, -\infty), \oplus, \otimes, (-\infty, -\infty), (0, 0))$ ... interval automaton
    - $(R^+, +, \times, 0, 1)$... stochastic automata (probability semiring)

Outline

Stream coalgebra

Streams are infinite sequences over a set, e.g. a semiring $K = (K, \oplus, \otimes, 0, 1)$.

$(K^\omega, \langle head, tail \rangle)$ is the *final coalgebra* of $F(S) = K \times S$.
Definition. For $s = (s(0), s(1), s(2), s(3), \dots) \in K^\omega$:
$head(s) = s(0)$ and $tail(s) = s' = (s(1), s(2), s(3), \dots)$.

Other notation:

$[r] = (r, 0, 0, \dots) \dots$ constant stream for $r \in K$.

$X = (0, 1, 0, \dots) \dots$ important to describe any stream

Final Mealy automaton

- Behaviors of Mealy automata are causal stream functions $f : A^\omega \to K^\omega$. $f :\ A^\omega \to K^\omega$ is *causal* if $\forall n \in \mathbb{N}$, $\sigma, \tau \in A^\infty$: $\forall i :\ i \leq n$: $\sigma(i) = \tau(i)$ then $f(\sigma)(n) = f(\tau)(n)$.
- Stream derivatives: $\omega = (\omega_0, \omega_1, \dots) \in K^\omega,\ \omega \to \omega' = (\omega_1, \dots)$.
- Stream functions form final coalgebra of Mealy automata with $t(f) = \langle f[a], f_a\rangle$ $f[a] = f(a : \sigma)(0)$ and $f_a(\sigma) = f(a : \sigma)'$
- For partial Mealy automata consider $f :\ A^\omega \to (1+K)^\omega$ f is *consistent* if $\sigma \in A^\omega$: $f(\sigma)(k) = \emptyset$ then $f(\sigma)(n) = \emptyset$ for any $n > k$.
- $\mathcal{F} = (\mathcal{F}, t_{\mathcal{F}})$ is the final coalgebra of partial Mealy automata: $\mathcal{F} = \{f :\ A^\omega \to (1+K)^\omega \ | f \text{ is causal and consistent}\}$.

$$t_{\mathcal{F}}(f)(a) = \begin{cases} \langle f[a], f_a\rangle & \text{if } f[a] \neq \emptyset \in 1, \\ \emptyset & \text{otherwise,} \end{cases}$$

Equivalent presentation of behaviors

- $s_0 \stackrel{\sigma(0)|k_0}{\rightarrow} s_1 \stackrel{\sigma(1)|k_1}{\rightarrow} s_2 \cdots \stackrel{\sigma(n)|k_n}{\rightarrow} s_{n+1}$. We define

$$l(s_0)(\sigma)(n) = k_n.$$

- $A^\infty = A^\omega \cup A^+$, where $A^+ = A^* \setminus \{\lambda\}$
- $\mathcal{F}$ is isomorphic to functions between finite and infinite sequences!

$$\mathcal{F}_\infty = \{f : A^\infty \to K^\infty | f \text{ length preserving, causal, } dom(f) \text{ prefix-closed}\}$$

- $f[a] = f(a)(0)$ whenever f is defined for $a \in A$.
- $f_a : A^\infty \to (1 + K)^\infty$ given by $f_a(s) = f(a : s)'$
-

$$t_{\mathcal{F}_\infty}(f)(a) = \begin{cases} \langle f[a], f_a \rangle & \text{if } f[a] \text{ is defined} \\ \text{undefined} & \text{otherwise,} \end{cases}$$

Fundamental theorem of stream functionals

Fundamental theorem of stream calculus:

$$\sigma = \sigma(0) \oplus X\sigma'(0) \oplus X^2\sigma''(0) \oplus \dots$$

has its stream functional counterpart:

Theorem. For any $f \in \mathcal{F}$ and $\sigma = (\sigma(0), \sigma(1), \dots, \sigma(k), \dots) \in A^\omega$ we have:

$$f(\sigma) = f(\sigma)(0) \oplus X f_{\sigma(0)}(\sigma')(0) \oplus \dots X^k f_{\sigma(0)\dots,\sigma(k-1)}(\omega^{(k)})(0) \oplus \dots$$

or equivalently,

$$f(\sigma) = f[\sigma(0)] \oplus X f_{\sigma(0)}[\sigma(1)] \oplus \dots X^k f_{\sigma(0)\dots,\sigma(k-1)}[\sigma(k)] \oplus \dots$$

Proposition.

1. For any $f \in \mathcal{F}_\infty$, $\omega \in A^\infty$, and $a \in A$: $f(a) : f_a(\omega) = f(a\omega)$.
2. More generally, for any $u \in A^+$ and $\omega \in A^\infty$: $f(u) : f_u(\omega) = f(u\omega)$.

Properties of stream functionals

Initial output is a particular partial stream functional defined by

$$f^{\infty}[a](\sigma) = \begin{cases} f[a] & \text{if } \sigma = a, \\ \text{undefined} & \text{otherwise: } \sigma \neq a, \end{cases}$$

Definition. For $f, g \in \mathcal{F}_{\infty}$, $\sigma = (\sigma(0) : \sigma') \in A^{\infty}$, and $a \in A$ we define

$$(f^{\infty}[a] \odot g)(\sigma(0) : \sigma') = \begin{cases} f(\sigma(0)) : g(\sigma') & \text{if } a = \sigma(0) \in dom(f), \\ \text{undefined} & \text{otherwise,} \end{cases}$$

Theorem 1. For any $f \in \mathcal{F}_{\infty}$ we have: $f = \bigoplus_{a \in A} f^{\infty}[a] \odot f_a$.

Theorem 2. For any $f \in \mathcal{F}_{\infty}$ and $a \in A$: $(f^{\infty}[a] \odot f)_a = f$

Outline

(max,+)- automata

- (max,+) automata are $G = (Q, \alpha, t, \beta)$, where Q is a finite set of states, $\alpha : Q \to \mathbb{R}_{max}$, $t : Q \times A \times Q \to \mathbb{R}_{max}$, and $\beta : Q \to \mathbb{R}_{max}$, called initial, transition, and final delays.
- Also: $G = (Q, A, q_0, Q_m, t)$, where
  - A set of discrete events,
  - q_0 initial state, Q_m subset of final or marked states,
  - $t : Q \times A \times Q \to \mathbb{R}_{max}$ transition function

Meaning: output value $t(q, a, q') \in \mathbb{R}_{max}$ corresponds to the duration of $a-$transition from q to q' and

$t(q, a, q') = \varepsilon$ if there is no transition from q to q' labeled by a.

Algebraic behaviors of (max,+)- automata

Formal power series with variables in A and coefficients in $\mathbb{R}_{max}$.

$\mathbb{R}_{max}(A)$ isomorphic to $\{\omega : A^* \to \mathbb{R}_{max}\}$.

Behavior of $G = \langle Q, A, q_0, Q_m, t\rangle$ for $w = a_1 \dots a_n \in A^*$:

$$l(G)(w) = \max_{q_1,\dots,q_n \in Q:\ q_n \in Q_m} (t(q_o, a_1, q_1) + t(q_1, a_2, q_2) + \cdots + t(q_{n-1}, a_n, q_n)).$$

$l(G)(w)$ is the longest path corresponding to label w from the initial state to a final state.

Using the matrix formalism:

$$l(G)(w) = \alpha \otimes t(w) \otimes \beta,$$

typically $\alpha = (e, \varepsilon, \dots, \varepsilon)$ and similarly for β

Unambiguous and deterministic (max,+)- automata

- (max,+) automata are seemingly simple, still powerful model, cf. 1- safe (timed) Petri nets!
- A K-automaton is unambiguous if, for every word w, there is at most one successful path labeled by w.
- Unambiguous series: ∃unambiguous automaton recognizing it.
- Lombardy and Mairesse: unambiguous series are intersection of (max,+) and (min,+)-rational series
- Beyond unambiguous series equality and inequality is undecidable and no rational controllers exist!
- Decidable classes of timed automata : one clock timed automata (e.g. interval automata)
  and their synchronous products called product interval automata compositions

(max,+)- automata coalgebraically

det. (max,+)- automata $\quad S = (S, t),\ t : S \to (1 + (\mathbb{R}_{\max} \times S))^A$

A *homomorphism* between $S = (S, t)$ and $S' = (S', t')$ is $f : S \to S'$ s.t. $\forall s \in S$ and $\forall a \in A$: if $s \stackrel{a|b}{\to} s'$ then $f(s) \stackrel{a|b}{\to} f(s')$, i.e.:

$$\begin{array}{ccc}
(1 + (\mathbb{R}_{\max} \times S))^A & \stackrel{t}{\longleftarrow} & S \\
\Big\downarrow F(f) & & \Big\downarrow f \\
(1 + (\mathbb{R}_{\max} \times S'))^A & \stackrel{t'}{\longleftarrow} & S'
\end{array}$$

A *bisimulation* between $S = (S, t)$ and $S' = (S', t')$ is $R \subseteq S \times S'$ s.t. $\forall s \in S$ and $\forall s' \in S'$: if $\langle s, s' \rangle \in R$ then

(i) $\forall a \in A$: $s \stackrel{a}{\not\to}$ iff $s' \stackrel{a}{\not\to}$

(ii) $\forall a \in A: \ s \stackrel{a|b}{\to} q \ \Rightarrow \ s' \stackrel{a|b'}{\to} q'$ s. t. $\langle q, q' \rangle \in R, b = b'$, and

(iii) $\forall a \in A: \ s' \stackrel{a|b'}{\to} q' \ \Rightarrow \ s \stackrel{a|b}{\to} q$ such that $\langle q, q' \rangle \in R$, and $b = b'$.

Behaviors of (max,+)-automata

Algebraic behaviors: formal power series

Coalgebraic behaviors: stream functions from $\mathcal{F}$.

$\mathcal{F} = \{f : A^\infty \to \mathbb{R}_{\max}^\infty | f$ length preserving, causal, $dom(f)$ prefix-closed$\}$.

Similar to timed languages: $L_t \subseteq (A \times \mathbb{R})^\infty$, but tailored to one clock timed automata!

Timed languages give the cumulated execution time of a sequence!

Stream functions give the duration of events in the sequence!

Outline

Synchronous extensions of (max,+)-automata

Major problem : (max,+)-automata as a class of timed automata are **not closed** under synchronous composition!

Our Solution: using extended multi-event alphabets
Let G_1 and G_2 be (max,+) automata over local alphabets A_1 and A_2. Associated natural projections are denoted by: $P_1 : (A_1 \cup A_2)^* \to A_1^*$ et $P_2 : (A_1 \cup A_2)^* \to A_2^*$. Boolean morphism matrices are needed:

$$[B\mu(a)]_{ij} = \begin{cases} e, & \text{if } [\mu(a)]_{ij} \neq \varepsilon \\ \varepsilon, & \text{else} \end{cases}$$

To alleviate notation $B(a)$ instead of $B\mu(a)$.
This can be extended to words of A^* by:

$$B(a_1 \dots a_n) = B(a_1) \dots B(a_n).$$

Tensor linear algebra

If A is a matrix of dimension $m \times n$ and B a matrix of dimension $p \times q$ over a dioid, their tensor (Kronecker) product $A \otimes^t B$ is the matrix of dimension $mp \times nq$:

$$A \otimes^t B = \begin{bmatrix} a_{11}B & \cdots & a_{11}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix}$$

In particular, for square matrices $A = (a_{ij})_{i,j=1}^{n}$ and $B = (b_{kl})_{k,l=1}^{m}$, $C = A \otimes^t B$ is a matrix of dimension $n.m \times n.m$ with

$$C_{ik,jl} = a_{ij} \otimes^t b_{kl}.$$

Synchronous composition using extended alphabets

Definition. (Synchronous product)
Synchronous product of (max,+) automata
$G_1 = (Q_1, A_1, \alpha_1, \mu_1, \beta_1)$ and $G_2 = (Q_2, A_2, \alpha_2, \mu_2, \beta_2)$, is (max,+) automaton defined over alphabet

$$\mathcal{A} = (A_1 \cap A_2) \cup (A_1 \setminus A_2)^* \times (A_2 \setminus A_1)^*$$

by

$$G_1 \| G_2 = \mathcal{G} = (Q_1 \times Q_2, \mathcal{A}, \alpha, \mu, \beta)$$

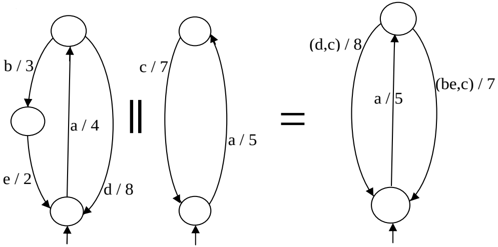
with $Q_1 \times Q_2$ state set, $\mathcal{A}$ event set, $\alpha = \alpha_1 \otimes^t \alpha_2$ initial delays, $\mu : \mathcal{A}^* \to \mathbb{R}_{max}^{|Q| \times |Q|}$ morphism matrix and $\beta = \beta_1 \otimes^t \beta_2$ final delays.
Morphism matrix :

$$\mu(v) = \begin{cases} \mu_1(v) \otimes^t B_2(v) \oplus B_1(v) \otimes^t \mu_2(v), & \text{if } v = a \in A_1 \cap A_2 \\ \mu_1(P_1(v)) \otimes^t B_2(P_2(v)) \oplus B_1(P_1(v)) \otimes^t \mu_2(P_2(v)), & \text{if } v = (P_1(v), P_2(v)) \end{cases}$$

Illustration of the synchronous product

$$\mathbf{G_1} \| \mathbf{G_2} = \mathcal{G} = (\mathbf{Q_1} \times \mathbf{Q_2}, \mathcal{A}, \alpha, \mu, \beta), \text{where}$$

$$\mathcal{A} = \{a, (be, c), (d, c)\} \subseteq (A_1 \cap A_2) \cup (A \setminus (A_1 \cap A_2))^*,$$



$\alpha = \alpha_1 \otimes^t \alpha_2$, $\beta = \beta_1 \otimes^t \beta_2$, and

$$\nu(v) = \begin{cases} \mu_1(a) \otimes^t B_2(a) \oplus B_1(a) \otimes^t \mu_2(a), & \text{if } v = a \in A_1 \cap A_2 \\ \mu_1(be) \otimes^t B_2(c) \oplus B_1(be) \otimes^t \mu_2(c), & \text{if } v = (be, c) \\ \mu_1(d) \otimes^t B_2(c) \oplus B_1(d) \otimes^t \mu_2(c), & \text{if } v = (d, c) \end{cases}$$

Induced behavior

- Behaviors of $G_1 \| G_2$ are formal power series of $\mathbb{R}_{\max}(\mathcal{A})$. From practical viewpoint (performance analysis, control) $I(G_1 \| G_2)(w)$ for $w \in A^*$ are more interesting (durations of tasks)
- Any $w \in A^*$ admits decomposition $w = v_0 a_1 v_1 \dots a_n v_n$, with $a_i \in A_1 \cap A_2,\ i = 1, \dots, n$ shared events and $v_i \in (A \setminus (A_1 \cap A_2))^*,\ i = 0, \dots, n$ private sequences.
- The local tasks of G_1 and G_2 corresponding to v_i are given by $P_1(v_i)$ et $P_2(v_i)$, resp.
- Any word from A^* can be seen as an element of $\mathcal{A}^*$, namely

$$w = P_1(v_0) \times P_2(v_0).a_1 P_1(v_1) \times P_2(v_1) \dots a_n P_1(v_n) \times P_2(v_n)$$

- Morphism μ induces the matrix mapping $\nu : A^* \to \mathbb{R}_{\max}$:

$$\nu(w) = \mu(P_1(v_0) \times P_2(v_0))\mu(a_1)\mu(P_1(v_1) \times P_2(v_1)) \dots \mu(a_n)\mu(P_1(v_n) \times P_2(v_n)).$$

Induced behavior continued

Definition. Induced behavior of $G_1 \| G_2$ is given by:

$$I(G_1 \| G_2)(w) = \alpha \nu(w) \beta.$$

Notation: $Z = \{\nu, B\}$ with complement $\bar{\nu} = B$ and $\bar{B} = \nu$.
Extension to words: $m = m^1 \dots m^k$, $\bar{m} = \bar{m}^1 \dots \bar{m}^k$.
Theorem. Induced behavior of $G_1 \| G_2$ for $w = v_0 a_1 v_1 \dots a_n v_n \in A^*$ is :

$$I(G_1 \| G_2)(w) = \bigoplus_{m \in Z^{2n+1}} \alpha_1 m_1(P_1(w)) \beta_1 \otimes \alpha_2 \bar{m}_2(P_2(w)) \otimes \beta_2.$$

Special case n=0.

$$l_1 \| l_2 = l_1(P_1(w)) \otimes supp(l_2)(P_2(w)) \oplus l_2(P_2(w)) \otimes supp(l_1)(P_1(w)),$$

because $supp(l_i)(P_i(w)) = \alpha_i B_i(P_i(w)) \beta_i$ for $i = 1, 2$.
Hint for better understanding.
$L_1 \| L_2 = P_1^{-1}(L_1) \cap P_2^{-1}(L_2)$, in terms of Boolean series:

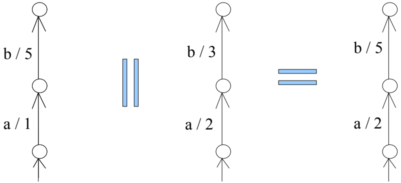
$$L_1 \| L_2(w) = L_1(P_1 w) \otimes L_2(P_2 w).$$

Synchronous product of series: example

Example shows that there is no simple formula for $l_1 \| l_2(w)$.

Here, $l_1 = 1a \oplus 6ab$, $l_2 = 2a \oplus 5ab$, and $l_1 \| l_2 = 2a \oplus 7ab$.

Linear representations of l_i are needed!



Synchronous product defined by coinduction

For $l_i \in \mathcal{F}$ over A_i and $v_i = a_1 \dots a_k \in A_i^+$ we define for $i = 1, 2$:

$$(l_i)[v_i] = (l_i)[a_1] \otimes (l_i)_{a_1}[a_2] \otimes \cdots \otimes (l_i)_{a_1 \dots a_{k-1}}[a_k].$$

Definition. for $l_1, l_2 \in \mathcal{F}$ and $\forall v \in \mathcal{A}$:

$$\begin{aligned} (l_1 \| l_2)_v &= (l_1)_{P_1(v)} \| (l_2)_{P_2(v)} \text{ and} \\ (l_1 \| l_2)[v] &= l_1[P_1(v)] \otimes Bl_2[P_2(v)] \oplus Bl_1[P_1(v)] \otimes l_2[P_2(v)]. \end{aligned}$$

Special case with full synchronization ($A_1 = A_2$): no need for using extended alphabet, in fact $\mathcal{A} = A_1 = A_2$.
For $v = a \in \mathcal{A}$ we have in fact $P_1(v) = P_2(v) = a$. Hence,

$$(l_1 \| l_2)_a = (l_1)_a \| (l_2)_a$$

and $(l_1 \| l_2)[a] = l_1[a] \otimes Bl_2[a] \oplus Bl_1[a] \otimes l_2[a]$.

Synchronous product continued

Equivalent expression for first input:

$$(l_1 \| l_2)[v] = \begin{cases} max(l_1[P_1(v)], l_2[P_2(v)]) & \text{if } l_i[P_i(v)] \neq \varepsilon \text{ for } i = 1,2 \\ \varepsilon & \text{else, i.e. } \exists i = 1,2 : l_i[P_i(v)] = \varepsilon \end{cases}$$

Hint for understanding:

for partial languages $L_1 = (L_1^1, L_1^2)$, $L_2 = (L_2^1, L_2^2)$, and $w \in A^*$ we have in fact

$$(L_1 \| L_2)_w = (L_1)_{P_1(w)} \| (L_2)_{P_2(w)}.$$

Behavior of synchronous product: example

$$(l_1 \| l_2)(a) = (l_1 \| l_2)(a)(0) = 5 = (l_1 \| l_2)[a]$$

$$(l_1 \| l_2)(a(d,c)) = (l_1 \| l_2)(a(d,c))(0) \oplus X(l_1 \| l_2)_a(d,c).$$

Formulas for derivative and first output function yield:

$$(l_1 \| l_2)_a(dc) = ((l_1)_a \| (l_2)_a)(dc) = ((l_1)_a \| (l_2)_a)(dc)(0) = ((l_1)_a \| (l_2)_a)[dc]$$

$$= (l_1)_a[d] \otimes B(l_2)_a[c] \oplus B(l_1)_a[d] \otimes (l_2)_a[c] = (l_1)(ad)(1) \otimes B(l_2)(ac)(1) \oplus$$

$$B(l_1)(ad)(1) \otimes (l_2)(ac)(1) = 8 \otimes 0 \oplus 0 \times 7 = 8.$$

Note that $(l_1)_a[d] = (l_1)_a(d)(0) = (l_1)(a:d)'(0) = (l_1)(ad)(1)$.
Similarly,
$(l_1 \| l_2)(a(be,c)) = (l_1 \| l_2)(a(be,c))(0) \oplus (l_1 \| l_2)_a((be,c))$, where

$$(l_1 \| l_2)_a(be,c) \quad = \quad \cdots = (l_1)(a(be))(1) \otimes B(l_2)(ac)(1) \oplus$$

$$B(l_1)(a(be))(1) \otimes (l_2)(ac)(1) = 5 \otimes 0 \oplus 0 \times 7 = 7.$$

Induced behavior example algebraically

- Induced behaviors translate series from $\mathbb{R}_{\max}(\mathcal{A})$, i.e. over $\mathcal{A}$ into standard series from $\mathbb{R}_{\max}(A)$ and correspond to duration of distributed tasks
- In accordance with Proposition we obtain for $w = abec$ the induced behavior

$$I(abec) = \alpha\nu(abec)\beta = \alpha\mu(a)\mu(be \times c)\beta = 5 + 7 = 12.$$

- Similarly, for $w = a(bec)a(cd)$ we obtain :

$$I(abecacd) = \alpha\nu(abecacd)\beta = \alpha\mu(a)\mu(be{\times}c)\mu(a)\mu(d{\times}c)\beta = 23.$$

- Coalgebraic definition is for series and saves on complexity (no matrices)!

Extension to more local subsystems

For n=3 there are 4types of synchronizations: all the three and all couples of subsystems
Hence, 5 types of multi-events (including no synchronization)

$$\mathcal{A} = (A_1 \cap A_2 \cap A_3) \cup (A_1 \cap A_2) \times (A_3 \setminus (A_1 \cup A_2))^* \cup (A_1 \cap A_3) \times (A_2 \setminus (A_1 \cup A_3))^* \cup (A_2 \cap A_3) \times (A_1 \setminus (A_2 \cup A_3))^*$$

$$(A_1 \setminus (A_2 \cup A_3))^* \times (A_2 \setminus (A_1 \cup A_3))^* \times (A_3 \setminus (A_1 \cup A_2))^*$$

Equivalently,

$$\mathcal{A} = (A_1 \cap A_2 \cap A_3) \times (A_1 \cap A_2 \cap A_3) \times (A_1 \cap A_2 \cap A_3) \cup (A_1 \cap A_2) \times (A_1 \cap A_2) \times (A_3 \setminus (A_1 \cup A_2))^*$$

$$\cup (A_1 \cap A_3) \times (A_2 \setminus (A_1 \cup A_3))^* \times (A_1 \cap A_3) \cup (A_1 \setminus (A_2 \cup A_3))^* \times (A_2 \cap A_3) \times (A_2 \cap A_3)$$

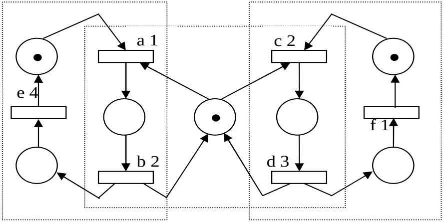
$$\cup (A_1 \setminus (A_2 \cup A_3))^* \times (A_2 \setminus (A_1 \cup A_3))^* \times (A_3 \setminus (A_1 \cup A_2))^*$$

$v = a \in A_1 \cap A_2 \cap A_3$ is here $a \times a \times a$
$v = a \times v_3$ is here $a \times a \times v_3$.

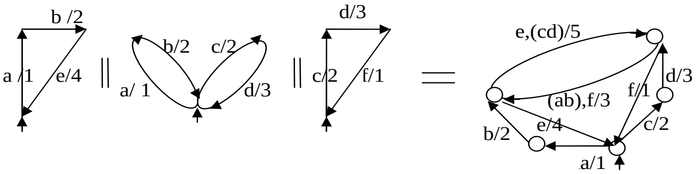
Interpretation in terms of timed Petri nets

- Synchronous products of (max,+) automata correspond to safe Timed Petri nets formed in a compositional way from safe timed state graphs (timed machines)
- Local automata correspond to marking graphs of timed state graphs (no synchronization: each transition has 1 upstream and 1 downstream place)
- Synchronous composition models synchronization of timed state graphs via synchronizing transitions

Example of timed Petri nets



Corresponding automaton model is below:



Outline

Timed automata and distributed interval automata

Timed automaton $\mathcal{A} = (S, A, C, t, s_0)$ with

- S . . . state set
- A . . . event set
- C . . . set of clocks
- $t \subseteq S \times A \times S \times EC \times 2^C$ . . . transition function

Transition labels: $Tr = \langle s, a, s', Cond, Z \rangle$, where
s origin, s' destination, a event label,
t can occur only if $Cond = TRUE$ and the clocks in Z are reset.
Syntax for enabling conditions (EC):
$c \equiv k$, where $c \in C$, $k \in R$, and $\equiv \in \{<, >, \leq, \geq, =\}$.
Extended states: $(s, c) \subseteq S \times R^{\|C\|}$,
with s state and
c the current values of clocks.

Distributed timed automata

Def. Composition of Timed automata
Synchronous product of timed automata
$R_i = (S_i, A_i, C_i, t_i, s_0^i), \; i = 1, \ldots, n$ is
$\|_{i=1^n} R_i = (S, A, C, t, s_0)$ with

- $S = \times_{i=1^n} S_i$
- $A = \bigcup_{i=1^n} A_i$
- $C = \bigcup_{i=1^n} C_i$
- $s_0 = (s_0^i)_{i=1}^n$
- $t \subseteq S \times A \times S \times EC \times 2^C$ such that $(s, a, s', \delta, \lambda) \in t$, iff $(s_i, a, s_i', \delta_i, \lambda_i) \in t_i$, where $s_i' = s_i$ for $a \in A_i$, $\delta = \delta_1 \wedge \delta_2$, and $\lambda = \lambda_1 \cup \lambda_2$.

Note. Regional construction is not compositional!

Interval Automata

Elementary classes of TA

- Product interval automata built by synchronous products of interval automata
- **Interval based alphabet:** $\Gamma = A \times IR$**,** with A finite alphabet and IR set of real intervals
- Definition of Interval automata
  Interval automata are automata $R = (S, \Gamma, t, I, F)$ over (symbolic) interval based alphabet.
- IA are timed automata with a single clock reset after every transitions

Product Interval Automata

- $R = (S, \Gamma, t, I, F)$ may also be viewed as weighted automaton with weights in a suitable interval semiring.
- Interval semiring: $(\mathbb{R}_{max} \times \mathbb{R}_{max}, \oplus, \otimes)$ with

$$(l_1, u_1) \otimes (l_2, u_2) = (l_1 + l_2, u_1 + u_2) \text{ and}$$

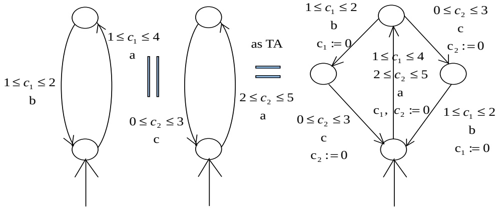
$$(l_1, u_1) \oplus (l_2, u_2) = (max(l_1, l_2), max(u_1, u_2))$$

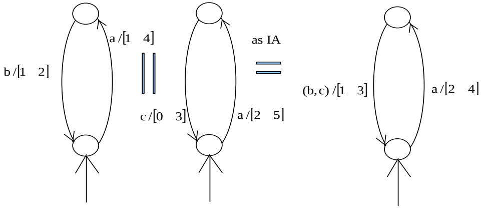
- Note that $\oplus$ is only used in composition, not in local IA (deterministic)!
- Dual addition also needed in the composition:

$$(l_1, u_1) \oplus' (l_2, u_2) = (max(l_1, l_2), min(u_1, u_2))$$

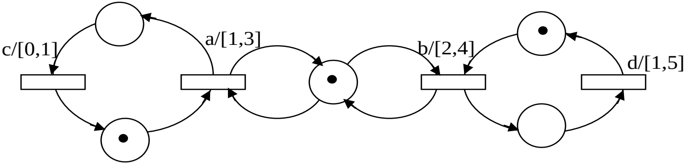
Composition of classes of timed automata

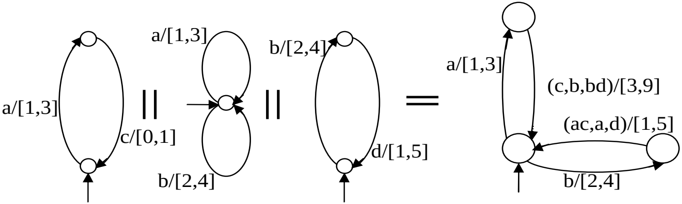
Definition. Synchronous products of interval automata.
Clocks are read and reset compatible with event distribution!





Example.





Example continued.

$$
\begin{aligned}
\mu(a) &= \mu_1(a) \otimes^t B_2(a) \otimes^t E_3 \oplus' B_1(a) \otimes^t \mu_2(a) \otimes^t E_3, \\
\mu(b) &= E_1 \otimes^t \mu_2(b) \otimes^t B_3(b) \oplus' E_1 \otimes^t B_2(b) \otimes^t \mu_3(b), \\
\mu((c,b,bd)) &= \mu_1(c) \otimes^t B_2(b) \otimes^t B_3(bd) \oplus B_1(c) \otimes^t \mu_2(b) \otimes^t B_3(bd) \oplus \\
&\quad B_1(c) \otimes^t B_2(b) \otimes^t \mu_3(bd), \\
\mu((ac,a,d)) &= \mu_1(ac) \otimes^t B_2(a) \otimes^t B_3(d) \oplus B_1(ac) \otimes^t \mu_2(a) \otimes^t B_3(d) \oplus \\
&\quad B_1(ac) \otimes^t B_2(a) \otimes^t \mu_3(d)
\end{aligned}
$$

Interpretation of extended words:
$w = acbdabdcbdac \in A^* \quad \rightarrow \quad w = a(c,b,bd)a(c,b,bd)b(ac,a,d)$
over $\mathcal{A}$.

Example coalgebraically.

Again, fundamental theorem gives for $\sigma \in \mathcal{A}^*$ and $v \in \mathcal{A}$

$$(l_1\|l_2\|l_3)(\sigma) = (l_1\|l_2\|l_3)(\sigma)(0) \oplus X(l_1\|l_2\|l_3)(\sigma)',$$

where

$$\{(l_1\|l_2\|l_3)(\sigma)\}' = (l_1\|l_2\|l_3)_{\sigma)(0)}(\sigma').$$

$(l_1\|l_2\|l_3)_v = (l_1)_{P_1(v)}\|(l_2)_{P_2(v)}\|(l_3)_{P_3(v)}$ and

$(l_1\|l_2\|l_3)[v] = l_1[P_1(v)] \otimes Bl_2[P_2(v)] \otimes Bl_3[P_3(v)] \oplus$

$Bl_1[P_1(v)] \otimes l_2[P_2(v)] \otimes Bl_3[P_3(v)] \oplus Bl_1[P_1(v)] \otimes B_2[P_2(v)] \otimes l_3[P_3(v)]$
with $\oplus$ replaced by $\oplus'$ for shared actions $v = a, b$

Outline

Concluding remarks

- Deterministic weighted automata as partial Mealy automata
- Composition of (max,+) automata
- Composition of Interval automata (PIA) and their properties
- Formulae for behavior of the synchronous product: algebraic vs. coalgebraic approach
- Supervisory control within behavioral framework
- Decentralized control of (classes) of timed automata