Open Problems in Concurrency Theory
Bertinoro, 18-21 June 2014

Causality, Revisited

Ugo Montanari
Dipartimento di Informatica
Università di Pisa

Joint work with Roberto Bruni and Matteo Sammartino
Supported in part by EU FET project ASCENS and by Italian PRIN project CINA

Information Society Technologies

Roadmap

- Observing causality
  - Causal trees
  - Causal automata
- Models with resource allocation, deallocation
  - Presheaves, coalgebras
  - Pi calculus
  - History Dependent Automata
  - From named sets to families, symmetries
  - Preasheaf models for causality
  - From causal trees to causal automata
- Conclusion

Roadmap

- Observing causality
  - Causal trees
  - Causal automata
- Models with resource allocation, deallocation
  - Presheaves, coalgebras
  - Pi calculus
  - History Dependent Automata
  - From named sets to families, symmetries
  - Preasheaf models for causality
  - From causal trees to causal automata
- Conclusion

Causal processes

Assume a set of atomic processes and their transitions

$$p_1 \xrightarrow{a} p_1 \qquad\qquad p_2 \xrightarrow{b} p_2$$

Darondeau-Degano causal semantics

$$\{1\} \Rightarrow p_1 \parallel \{2\} \Rightarrow p_2 \xrightarrow{a,\{1\}} \{1,2\} \Rightarrow p_1 \parallel \{3\} \Rightarrow p_2$$

causality vs noninterference, system maintenance

Information Society Technologies

Causal processes

$$\frac{p \xrightarrow{a} t \in \Delta}{K \Rightarrow p \xrightarrow{a,K}_{\mathrm{DD}} \{1\} \cup \delta(K) \Rightarrow t} \qquad \frac{t_1 \xrightarrow{l} t_1'}{t_1 \parallel t_2 \xrightarrow{l}_{\mathrm{DD}} t_1' \parallel \delta(t_2)}$$

$$\frac{t_1 \xrightarrow{a,K_1}_{\mathrm{DD}} t_1' \qquad t_2 \xrightarrow{\overline{a},K_2}_{\mathrm{DD}} t_2'}{t_1 \parallel t_2 \xrightarrow{\tau,K_1 \cup K_2}_{\mathrm{DD}} \eta(\delta(K_2), t_1') \parallel \eta(\delta(K_1), t_2')} \qquad \frac{t_2 \xrightarrow{l} t_2'}{t_1 \parallel t_2 \xrightarrow{l}_{\mathrm{DD}} \delta(t_1) \parallel t_2'}$$

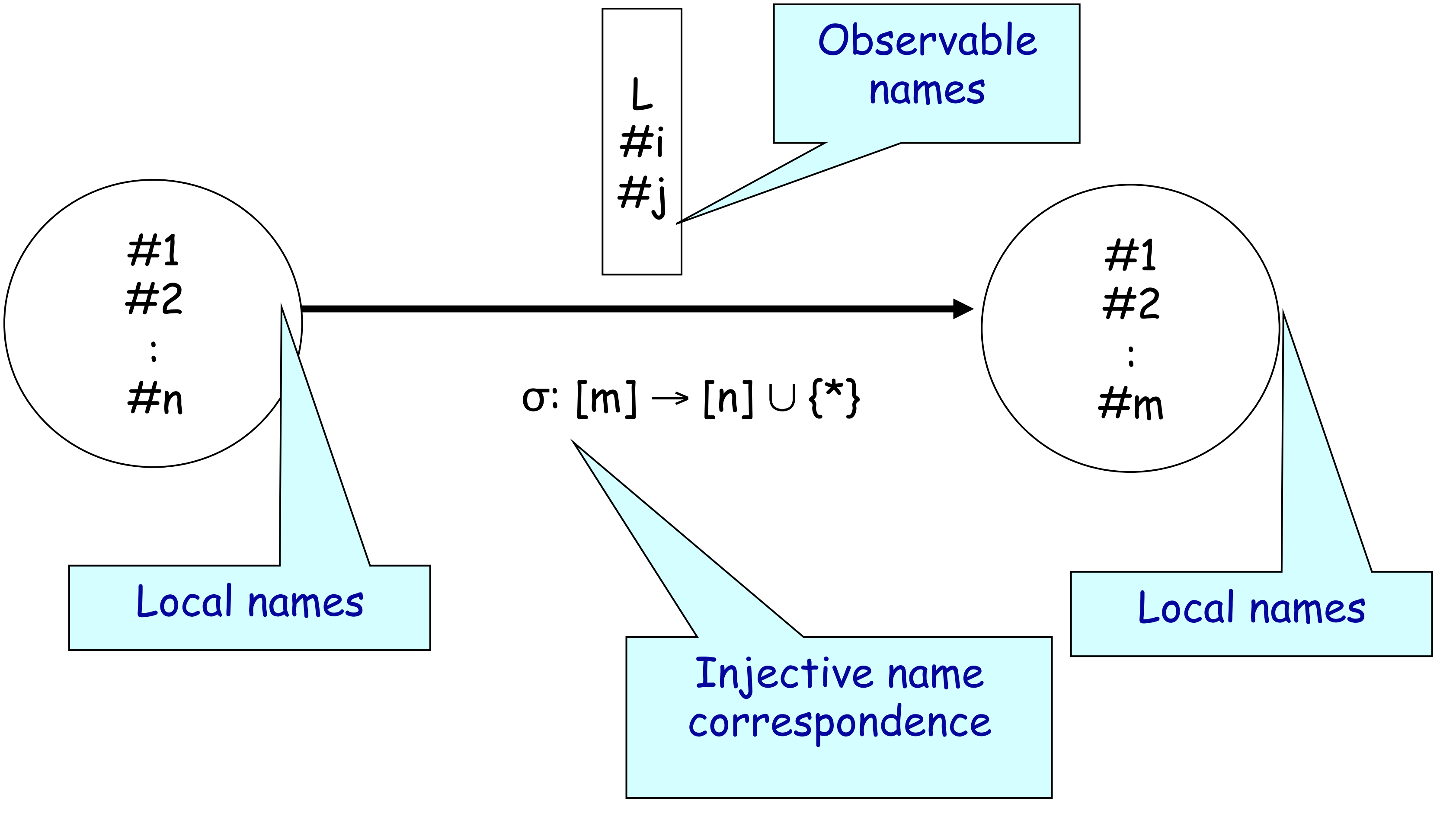
- $\delta(K)$ shifts all the causes in K by one, in order to "make room" for the new event 1; we let $\delta(K \Rightarrow p) = \delta(K) \Rightarrow p$
- $\eta(K_1, K_2)$ joins K_1 and K_2 only if $1 \in K_2$, otherwise returns K_2; we let $\eta(K_1, K_2 \Rightarrow p) = \eta(K_1, K_2) \Rightarrow p$.

infinite state-space => causal automata

Roadmap

- Observing causality
  - Causal trees
  - Causal automata
- Models with resource allocation, deallocation
  - Presheaves, coalgebras
  - Pi calculus
  - From named sets to families, symmetries
  - Preasheaf models for causality
  - From causal trees to causal automata
- Conclusion

Causal Automata: Structure of Transitions



Causal Automata

Definition 8 (causal automaton). Let $\mathcal{N}$ be a fixed infinite denumerable set of event names.
A *causal automaton* is a tuple $A = \langle Q, w, \mapsto, q_0 \rangle$ where:

- Q is a set of *states*;
- $\boxed{w : Q \to \mathcal{P}_{\mathrm{fin}}(\mathcal{N})}$ associates to each state a finite set of names;
- $\mapsto$ is a set of *transitions*; each transition has the form $\boxed{q \overset{a}{\underset{M}{\mapsto}}_{\sigma} q'}$, where:
  - $q, q' \in Q$ are the *source* and *target* states;
  - $a \in \mathrm{Act}$ is the *label*;
  - $M \subseteq w(q)$ are the *dependencies* of the transition;
  - $\boxed{\sigma : w(q') \hookrightarrow w(q) \cup \{\star\}}$ is the injective (inverse) *renaming* for the transition; the special mark $\star \notin \mathcal{N}$ is used to recognize in the target state the name corresponding to the current transition;
- $q_0 \in Q$ is the *initial state*; we require that $w(q_0) = \emptyset$.

- Montanari, U. and Pistore, M., History Dependent Verification for Partial Order Systems, in: D. Peled, V.Pratt, and G. Holzmann, Eds., Procs. Partial Order Methods in Verifications, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Vol.29, 259-272, 1996.
- Montanari, U. and Pistore, M., Minimal Transition Systems for History-Preserving Bisimulation, in: Ruediger Reischuk, Michel Morvan, Eds., STACS 97, Springer LNCS 1200, 1997, pp. 413-425.

Causal Automata Bisimulation

Definition 9 (bisimulation on causal automata). A set $\mathcal{R}$ of triples is a *causal bisimulation* for causal automata A and B if:

- whenever $\boxed{(p, \delta, q) \in \mathcal{R}}$ then $p \in Q_A$, $q \in Q_B$ and δ is a partial injective function from $w_A(p)$ to $w_B(q)$;
- $(q_{0A}, \emptyset, q_{0B}) \in \mathcal{R}$;
- whenever $(p, \delta, q) \in \mathcal{R}$ and $p \overset{a}{\underset{M}{\mapsto}}_\sigma p'$ in A then there exist some $q \overset{a}{\underset{\delta(M)}{\mapsto}}_\rho q'$ in B and some δ' such that $(p', \delta', q') \in \mathcal{R}$ and $\delta'(m) = n$ implies $\sigma(m) = \star = \rho(n)$ or $\delta(\sigma(m)) = \rho(n)$;
- whenever $(p, \delta, q) \in \mathcal{R}$ and $q \overset{a}{\underset{M}{\mapsto}}_\sigma q'$ in B then there exist some $p \overset{a}{\underset{\delta^{-1}(M)}{\mapsto}}_\rho p'$ in A and some δ' such that $(p', \delta', q') \in \mathcal{R}$ and $\delta'(m) = n$ implies $\sigma(m) = \star = \rho(n)$ or $\delta(\sigma(m)) = \rho(n)$.

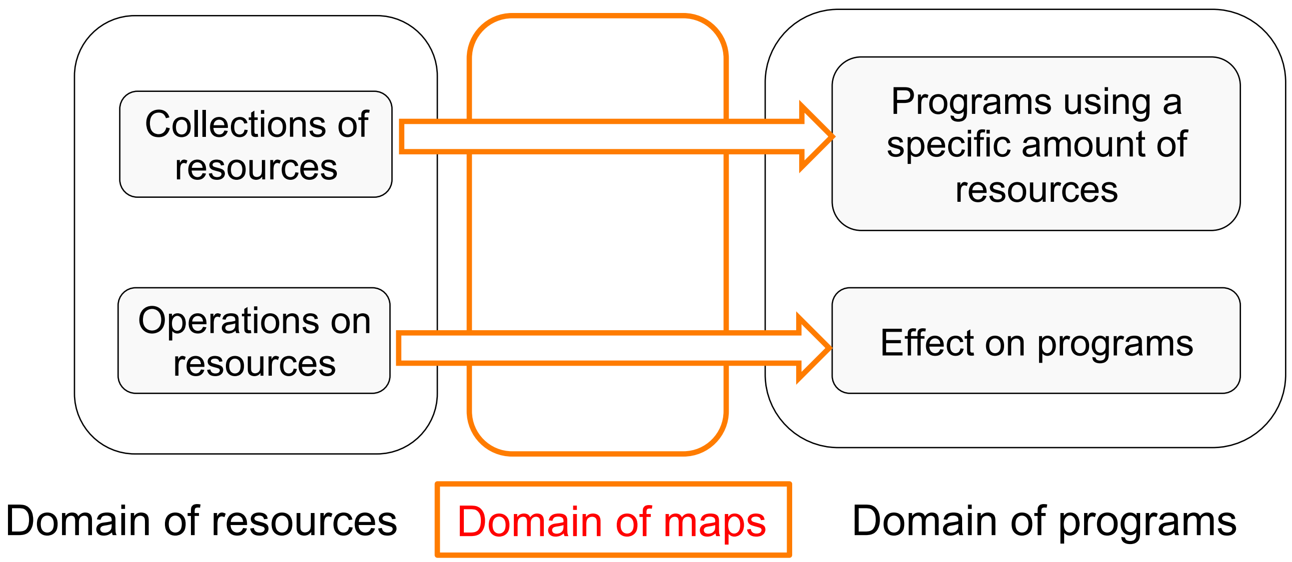
The causal automata A and B are *bisimilar*, written $A \sim_{ca} B$, if there is some bisimulation for them.

Roadmap

- Observing causality
  - Causal trees
  - Causal automata
- Models with resource allocation, deallocation
  - Presheaves, coalgebras
  - Pi calculus
  - From named sets to families, symmetries
  - Preasheaf models for causality
  - From causal trees to causal automata
- Conclusion

Models

- "Modular" models:
  - Fix the language, vary the allocation mechanisms (e.g. JVM)
  - Fix the allocation mechanisms, vary the languages (e.g. MS Common Language Runtime, CLR)
- Implementation of resources decoupled form the way they are used



Roadmap

- Observing causality
  - Causal trees
  - Causal automata
- Models with resource allocation, deallocation
  - Presheaves, coalgebras
  - Pi calculus
  - From named sets to families, symmetries
  - Preasheaf models for causality
  - From causal trees to causal automata
- Conclusion

Presheaf models

- Three independent layers
  - **Resources** as a category R, equipped with **allocation operators**

$$\delta_k : R \to R \qquad \text{(one for each type k of resource)}$$

  - **Programs** as a map (presheaf) P: R ➔ Set
  - **Behavior** as a suitable coalgebra, i.e. a categorical transition system, with states drawn from P

- Nice properties:
  - Better for binding signatures than plain sets
  - Application of known theories (universal algebras/coalgebras) is possible:
    - Denotational models
    - Operational models and minimal systems
  - Have a concrete automata-theoretic counterpart: **HD-automata**

Information Society Technologies

Categorical implementation

Domain of contexts

C

c

$f\colon c \to c'$

δ

Allocation operator

extends contexts with fresh names

$P\colon \mathbf{C} \to \mathbf{Set}$

Domain of processes

Set

$P(c) = \{p \mid c \vdash p\}$

$P(f)\colon P(c) \to P(c')$

extends f to processes

Information Society Technologies

Roadmap

- Observing causality
  - Causal trees
  - Causal automata
- Models with resource allocation, deallocation
  - Presheaves, coalgebras
  - Pi calculus
  - From named sets to families, symmetries
  - Preasheaf models for causality
  - From causal trees to causal automata
- Conclusion

Example: the π-calculus

$\mathbf{F} =$ Finite sets (of names) $n = \{1, \ldots, n\}$
+
renamings

Allocation modeled via coproducts

$$\delta(n) := \quad n \longrightarrow \boxed{n+1} \longleftarrow 1$$

embeds old names

embeds new name

Coalgebras over presheaves

Functor

$$BP = \mathscr{P}_f\left(Lab_1 \times P + Lab_2 \times P \circ \delta \right)$$

Coalgebra: (B: Set^F→ Set^F, tr: P→BP)

$$tr_c : P(c) \to \mathscr{P}_f(Lab_1(c) \times P(c) + Lab_2(c) \times P(\delta c))$$

transitions without allocation

$$n \vdash p \xrightarrow{\bar{a}b} n \vdash p'$$

transitions with allocation

$$n \vdash p \xrightarrow{\bar{a}(\star)} \delta n \vdash p'$$

Information Society
Technologies

History Dependent automata

- In presheaf semantics, names are created, but never deallocated
- Transition systems have very often infinite states
- Model checking verification impossible/difficult
- History Dependent (HD) automata allow for deallocation of names and for reuse of the same states with different names
- A single state of a HD automaton represents all the states obtained by name permutation
- HD automata are similar to causal automata, but their states have symmetries => they can be bisimilar to themselves up to a name permutation
- HD automata can be seen as coalgebras in the category of named sets
- Named sets are sets where each element is equipped with a finite set of names. Functions relate the names of arguments-results
- What is the relation between HD automata and presheaf coalgebras?

Roadmap

- Observing causality
  - Causal trees
  - Causal automata
- Models with resource allocation, deallocation
  - Presheaves, coalgebras
  - Pi calculus
  - From named sets to families, symmetries
  - Preasheaf models for causality
  - From causal trees to causal automata
- Conclusion

Information Society Technologies

Three Equivalent Structures

Categorical equivalence between

- the nominal sets by Gabbay and Pitts/permutation algebras
- the Schanuel topos (the presheaf version, precisely the full subcategory of pullback-preserving endofunctors in the presheaf category Set^I, I being the small category of finite sets of natural numbers and injective functions)
- the named sets by Montanari and Pistore (whose coalgebras are HD-automata)

Equivalence for Coalgebras

- Marcelo Fiore, Sam Staton, Information and Computation, 2006
- Fabio Gadducci, Marino Miculan, Ugo Montanari, Higher-Order and Symbolic Computation, 2006
- Vincenzo Ciancia, Ugo Montanari: A Name Abstraction Functor for Named Sets, CMCS 2008.
- Vincenzo Ciancia, Accessible Functors and Final Coalgebras for Named Sets, Ph.D. Thesis, 2008.
- Vincenzo Ciancia, Ugo Montanari: Symmetries, Local Names and Dynamic (De)-Allocation of Names. Information and Computation, 2010.

The Category of Named Sets, Revisited, Generalized

- We represent the wide pullback preserving full subcategory of $\mathbf{Set}^{\mathbf{C}}$ as the category $\mathbf{Fam}(\mathbf{Sym}(\mathbf{C})^{op})$
- $\mathbf{Sym}(\mathbf{C})$ is the category of groups of automorphisms of $\mathbf{C}$, representing the **support** and **symmetry** of an element of a presheaf
- Symmetries are the essential information that is needed to reconstruct each represented presheaf: first one reconstructs the presheaf "freely" using representables, then a quotient is made using the symmetry.

Families are indexed collections of objects of C

For named sets, C is I, the category of finite subsets of natural numbers and injections

Families vs. Presheaves

Q: what categories of presheaves can be represented as families?

1. Our answer: **small index categories of monos,** all automorphisms are iso, and **(weak) wide pullback preservation** give rise to an **equivalence of categories**

2. **[Adamek, Velebil - TAC 2008]**: **locally presentable index categories** and **weak wide pullback preservation** represent presheaves - natural transformations are not encoded. Generalises Joyal's species as representations of analytic functors.

- The two conditions are different: (1) includes coproducts of categories, (2) includes Set
  They obviously overlap (e.g. finite sets and injections).

Vincenzo Ciancia, Alexander Kurz, Ugo Montanari, Families of Symmetries as Efficient Models of Resource Binding. CMCS 2010

Roadmap

- Observing causality
  - Causal trees
  - Causal automata
- Models with resource allocation, deallocation
  - Presheaves, coalgebras
  - Pi calculus
  - From named sets to families, symmetries
  - Preasheaf models for causality
  - From causal trees to causal automata
- Conclusion

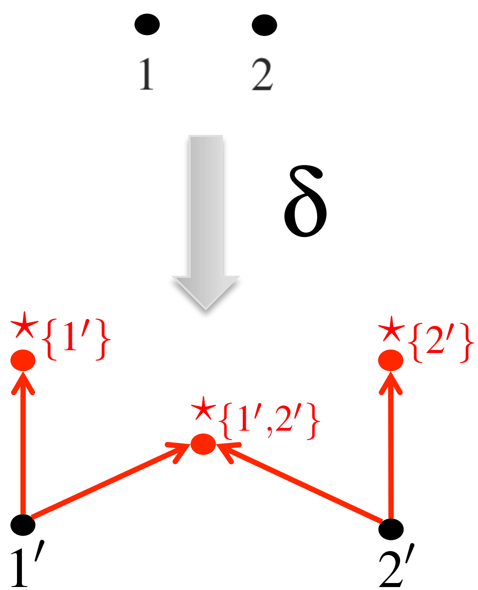
Category of causal relations

$$\mathbf{P}_m =$$

finite posets of events

+

injective functions between events that preserve and reflect partial ordering



Allocation operator

- Adds a new event for each set of causes
- Implemented through colimits

Coalgebras for causality

$$B : \mathbf{Set}^{\mathbf{P}_m} \to \mathbf{Set}^{\mathbf{P}_m}$$

$$BP(O) = \mathscr{P}_f(\ \boxed{L(O)} \times \boxed{P(\delta O)}\ )$$

Pairs (action , set of events in O)

Causal processes with additional events

Admits a final coalgebra, made of pairs

$$\boxed{O} \triangleright \boxed{T}$$

Full information about history of events

"Almost" a causal tree

Roadmap

- Observing causality
  - Causal trees
  - Causal automata
- Models with resource allocation, deallocation
  - Presheaves, coalgebras
  - Pi calculus
  - From named sets to families, symmetries
  - Preasheaf models for causality
  - From causal trees to causal automata
- Conclusion

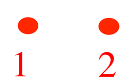
Poset-indexed LTS

Causal processes are decorated with a poset, describing the past history of events

$$O \triangleright k$$

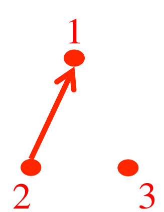
- Events in k must be a subset of those of O
- Each subprocess lists the whole history of its causes

Non-maximal events are removed from labels



$$\triangleright \{1\} \Rightarrow p_1 \parallel \{2\} \Rightarrow p_2$$

$$\Big\downarrow\ a, \{1\}$$



$$\triangleright \{1,2\} \Rightarrow p_1 \parallel \{3\} \Rightarrow p_2$$

$$\Big\downarrow\ a, \{1, \cancel{2}\}$$

Poset-indexed LTS as a coalgebra

$$\mathscr{C}(O) = \{k \mid O \triangleright k\}$$

$$\mathscr{C}(\sigma\colon O \to O') = \lambda O \triangleright k.\boxed{k\sigma\!\downarrow_{O'}}$$

$$\langle \mathscr{C}, \kappa\colon \mathscr{C} \to B\mathscr{C} \rangle$$

Applies σ to events in k and O' incorporates past events derived from the poset-indexed LTS

Still infinite-state!

Minimal processes

Given $O \triangleright k$ we can use (wide) pullbacks to compute

$supp(k) =$ Poset with all and only immediate causes of k

$seed(k) =$ version of k with only immediate causes

Existence guaranteed by $\mathscr{C}$ preserving pullbacks

An efficient model (Ciancia-Kurz-Montanari 2010)

Pullback-preserving presheaves

$$O \triangleright k$$

Poset-decorated sets

$$supp(k) \triangleright seed(k)$$

up to isomorphism

+

group of poset automorphisms that fix the state

Induces equivalence between coalgebras

Coalgebras over pullback-preserving presheaves

Coalgebras over named sets = History Dependent Automata

Minimal automaton exists, often finite

Similar to Pistore's causal automata, but automatic

Information Society Technologies

Roadmap

- Observing causality
  - Causal trees
  - Causal automata
- Models with resource allocation, deallocation
  - Presheaves, coalgebras
  - Pi calculus
  - From named sets to families, symmetries
  - Preasheaf models for causality
  - From causal trees to causal automata
- Conclusion

Operational Models with Resource Generation

- Generation of fresh resources is a basic operation in most distributed systems
  - Sessions, objects, keys, storage, links…

- We need models whose states are enriched with names
- We should be able to allocate, and possibly deallocate names

- Often more general kinds of resources

- A recent example: Matteo Sammartino PhD Thesis, Pisa, December 2013
- Resources are communication networks
- Applications to Software Defined Network
- Modeling Pastry Distributed Hash Tables
- Montanari U. and Sammartino, M., A Network-Conscious Pi-Calculus and Its Coalgebraic Semantics, TCS, to appear

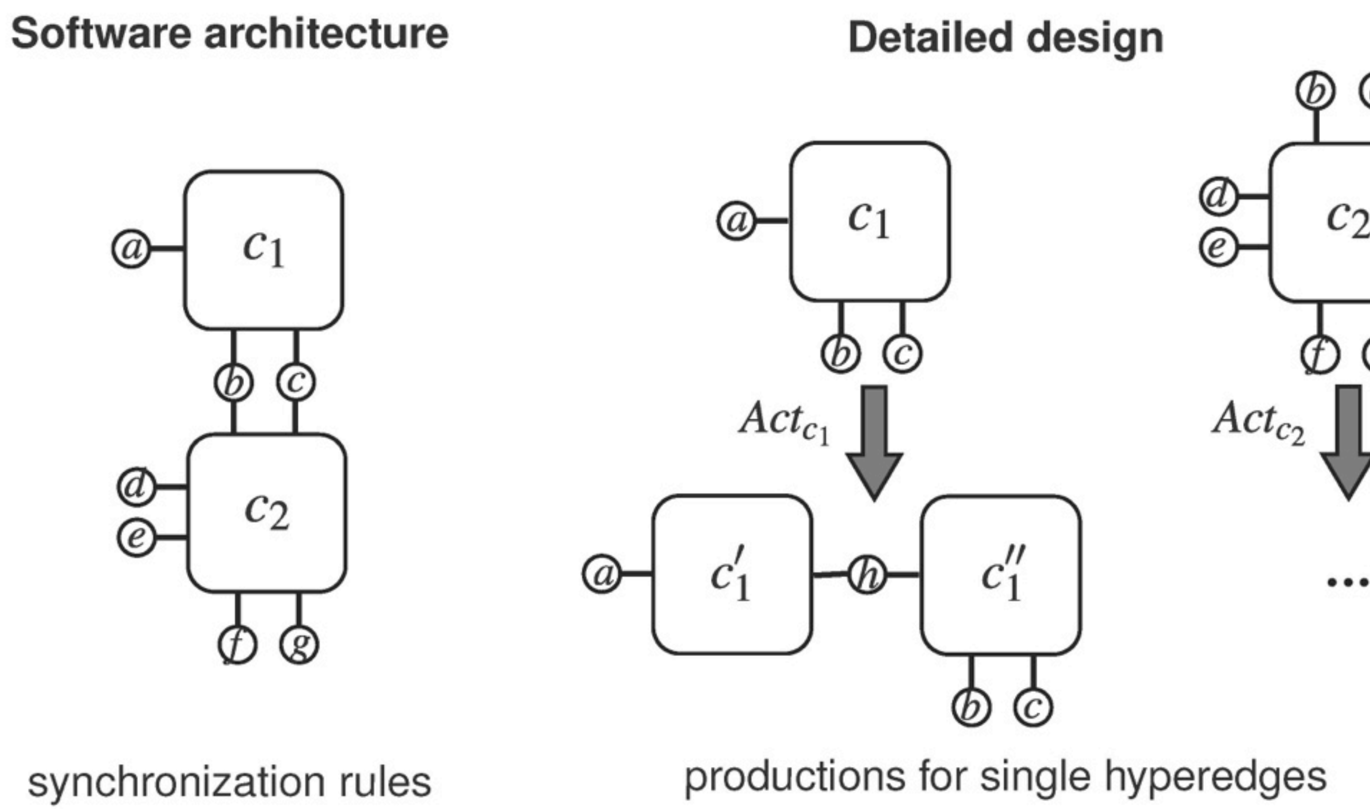
Ongoing work:

Software architectures as resources

Information Society Technologies

What’s Next: Software Architectures

Synchronized hyperedge replacement (CCS-like, i.e. without mobility) to describe



Sofware Architectures as Resources

Model of resources

- Architectures as a category of hypergraphs
- δs add new components (hyperedges) and connections (nodes)
- Presheaves index systems by their architecture

Two levels of behavior

1. *In the large*
   Algebra of parallel composition of components
   Coalgebra of component synchronization
2. *in the small*
   Syntax of sequential programs/processes
   Coalgebra of process actions

(1) + (2) = Composition of bialgebras to define the whole system

Similar structure for BI(P): **B**ehavior of atomic components + **I**nteractions among them