Asynchronous Pi Calculus

Thursday, September 29, 2011
13:27

$P ::= M \mid P_1 \mid P_2 \mid \nu x P \mid !P$

$M ::= 0 \mid \pi.P \mid M_1 + M_2$

$\pi ::= \tau \mid \bar{x}\langle y \rangle \mid x(y) \mid [x=y]\pi$

} full π Calculus → synchronization occurs on guards.

Asynchronous Pi Calculus

→ meant to more accurately model real systems with finite buffers, different delays, reordering, lossy channels

→ turns out there's an asynchronous fragment of π-calculus

→ we must re-build the guard idea to allow asynchronous operation

→ one way to do this is to just allow sends to guard processes.

$P ::= \bar{x}\langle y \rangle \mid M \mid P_1 \mid P_2 \mid \nu x P \mid !P$

$M ::= 0 \mid x(y).P \mid \tau.P \mid M_1 + M_2$

★ think about sends having already happened, or just being guarded by τ's. This way it can happen at any time, so we need to be delay insensitive

$\bar{x}\langle y \rangle + w(z).P$

$\bar{x}\langle y \rangle + \bar{x}\langle z \rangle$

} one could model loss like this, or, explicitly:

$!x(y).0 \Rightarrow$ /dev/null

$\bar{x}\langle y \rangle \mid x(z).P + M \longrightarrow P\{y/z\}$

How do we enforce some sort of ordering:

$\nu\, y,z \; (\bar{x}\langle y \rangle \mid \bar{y}\langle z \rangle \mid \bar{z}\langle a \rangle \mid R)$ where $y, z \notin fv(R)$

this enforces ordering, assuming R has all the appropriate receiver.

More complicated example:

$$\nu x,w,u\ (\overline{x}\langle w\rangle \mid w(u).(\overline{u}\langle a\rangle \mid w(z).\overline{z}\langle b\rangle))$$

(1) (2) (3) (4) (5)

$$x(w).(\overline{w}\langle u\rangle \mid u(y).\nu z\,(\overline{w}\langle z\rangle \mid z(s).Q\{^{s}/_{y}\}))$$

(1) (2) (3) (4) (5) (6)

these banana brackets mean "translation", many weird brackets

$(\!|\cdot|\!)$: Full π calculus into Async π calculus

$(\!|\cdot|\!)$
$[\![\cdot]\!]$
$\{\!\{\cdot\}\!\}$

$$(\!|\overline{x}\langle y\rangle.P|\!) \triangleq \nu w\,(\overline{x}w \mid w(u).(\overline{u}\langle y\rangle \mid [\![P]\!]))$$

$$(\!|x(z).Q|\!) \triangleq x(w).\nu u\,(\overline{w}\langle u\rangle \mid u(z).[\![Q]\!])$$

full π calculus → Async π calculus

$$P ::= \overline{x}\langle y\rangle \mid \sum_i x_i(z).P_i \mid P_1 \mid P_2 \mid \nu z\,P \mid !P$$

we'd like to express this with primitives.

→ we can do so w/ parallel composition and lots of channel communication

→ we can actually "get rid of" summation in π calculus, because we can build up to it w/ primitives → We still need 0, τ and lazy sums (M, $M+N$)

→ we'll also lose $\tilde{\tau}$- transitions as well

$(\!|\cdot|\!)$ - homo morphic except for one case

In abstract algebra, a **homomorphism** is a structure-preserving map between two algebraic structures (such as groups, rings, or vector spaces). The word *homomorphism* comes from the Greek language: *ὁμός (homos)* meaning "same" and *μορφή (morphe)* meaning "shape".

Pasted from <http://en.wikipedia.org/wiki/Homomorphism>

Definition

The definition of homomorphism depends on the type of algebraic structure under consideration. Particular definitions of homomorphism include the following:

- A group homomorphism is a homomorphism between two groups.
- A ring homomorphism is a homomorphism between two rings.
- A linear map is a homomorphism between two vector spaces.
- An algebra homomorphism is a homomorphism between two algebras.
- A functor is a homomorphism between two categories.

The common theme is that a homomorphism is a function between two algebraic objects that respects the algebraic structure.

For example, a group is an algebraic object consisting of a set together with a single binary operation, satisfying certain axioms. If G and H are groups, a **homomorphism** from G to H is a function $f: G \to H$ such that

$$f(g_1 * g_2) = f(g_1) *' f(g_2)$$

for any elements $g_1, g_2 \in G$, where * denotes the operation in G and *' denotes the operation in H.

When an algebraic structure includes more than one operation, homomorphisms are required to preserve each operation. For example, a ring possesses both addition and multiplication, and a homomorphism between two rings is a function such that

$$f(r + s) = f(r) +' f(s) \qquad \text{and} \qquad f(r * s) = f(r) *' f(s)$$

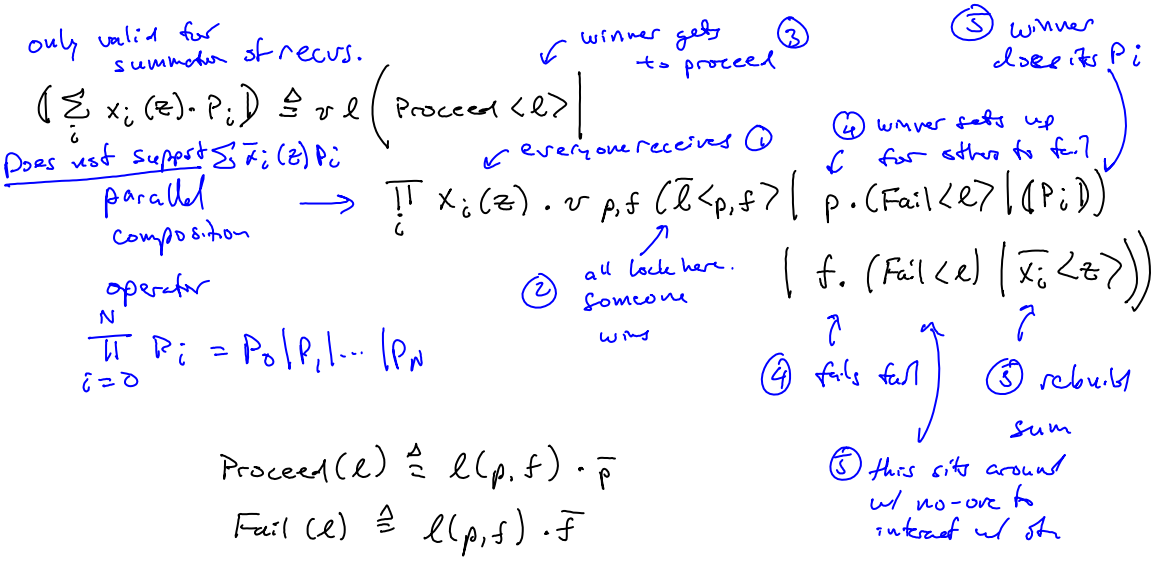
for any elements r and s of the domain ring.

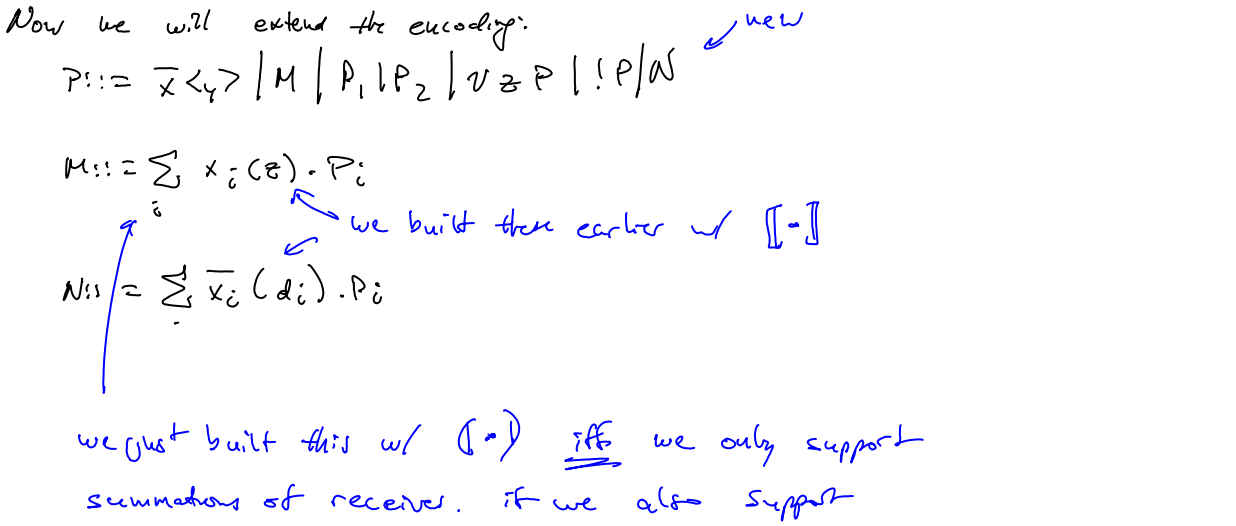
The notion of a homomorphism can be given a formal definition in the context of universal algebra, a field which studies ideas common to all algebraic structures. In this setting, a homomorphism $f: A \to B$ is a function between two algebraic structures of the same type such that

$$f(\mu_A(a_1, \ldots, a_n)) = \mu_B(f(a_1), \ldots, f(a_n))$$

for each n-ary operation μ and for all elements $a_1, \ldots, a_n \in A$.

Pasted from <http://en.wikipedia.org/wiki/Homomorphism>





summations of receives. if we also support summations of sends, the (-) definition is bogus

Ok, let's build this monotrasity to support sums of receives AND sends

"easy" case first:

$$\{\{\sum_i \bar{x}_i\langle d_i\rangle.P_i\}\} \triangleq \nu s\,(\text{Proceed}\langle s\rangle \mid \prod_i \nu a\; x_i\langle d_i, s, a\rangle.\; \nu p,f(\bar{a}\langle p,f\rangle \mid p.\{\{P_i\}\} \mid f.0))$$

$$\{\{\sum_i y_i\langle d_i\rangle.Q_i\}\} \triangleq \nu r\,(\text{Proceed}\langle r\rangle \mid$$

$$\prod_i \nu g\,(\bar{g} \mid !g.y_i(z,s,a).\nu p_1,f_1\,(\bar{r}\langle p_1,f_1\rangle \mid$$

$$p_1(\nu p_2,f_2\,(\bar{s}\langle p_2,f_2\rangle \mid p_2(\text{Fail}\langle r\rangle \mid \text{Fail}\langle s\rangle \mid \text{Proceed}\langle a\rangle \mid \{\{Q_i\}\})) \mid$$

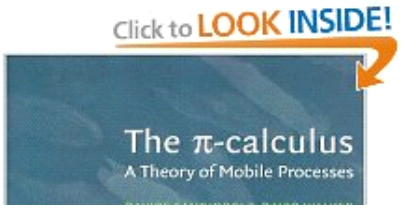
$$f_2(\text{Proceed}\langle r\rangle \mid \text{Fail}\langle s\rangle \mid \text{Fail}\langle a\rangle \mid \bar{g}))$$

$$\mid f_1(\text{Fail}\langle r\rangle \mid \bar{y}_i\langle z,s,a\rangle)))$$

this is super complicated. all it does is choose one receive to continue, and set up all the other ones to try again next "cycle"

Screen clipping taken: 9/29/2011, 14:33

π-Calculus Books

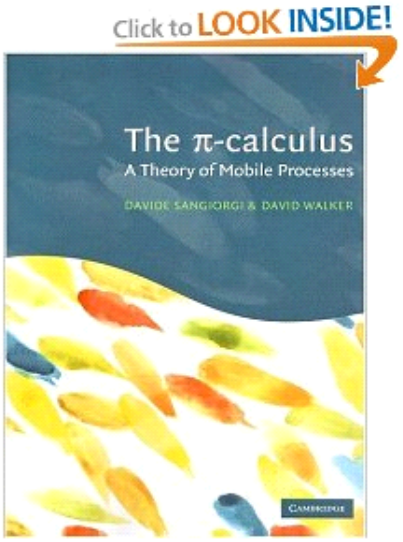


The Pi-Calculus: A Theory of Mobile Processes [Paperback]

Davide Sangiorgi (Author), David Walker (Author)

(1 customer review) | Like (0)

List Price: ~~$105.00~~



The Pi-Calculus: A Theory of Mobile Processes [Paperback]

Davide Sangiorgi (Author), David Walker (Author)

(1 customer review) | Like (0)

List Price: ~~$105.00~~

Price: $96.00 Prime

You Save: $9.00 (9%)

In Stock.

Ships from and sold by **Amazon.com**. Gift-wrap available.

Want it delivered Saturday, October 1? Order it in the next 23 hours and 26 minutes, and choose **One-Day Shipping** at checkout. Details

15 new from $88.33 **7 used** from $55.00

⇒ Prof: Very Comprehensive

- **Paperback:** 596 pages
- **Publisher:** Cambridge University Press (October 16, 2003)
- **Language:** English
- **ISBN-10:** 0521543274
- **ISBN-13:** 978-0521543279
- **Product Dimensions:** 9.4 x 7.5 x 1.6 inches
- **Shipping Weight:** 2.7 pounds (View shipping rates and policies)
- **Average Customer Review:** 5.0 out of 5 stars See all reviews (1 customer review)
- **Amazon Best Sellers Rank:** #1,112,665 in Books (See Top 100 in Books)

Pasted from <http://www.amazon.com/Pi-Calculus-Theory-Mobile-Processes/dp/0521543274>

Cool Result

Fun Fact: Bologna, Italy is π-Calculus Ville!

Palamidessi (POPL 1997)

⇒ There does not exist a translation $[\![\cdot]\!]$ (mapping) from $A\pi$ to $A\pi\text{-}\Sigma$ with

$A\pi$: regular π-calculus

$A\pi\text{-}\Sigma$: π-calculus w/o summation

the following properties:

1) $[\![P \cdot \sigma]\!] = [\![P]\!]\sigma$ (substitution on names)

2) $[\![P \mid Q]\!] = [\![P]\!] \mid [\![Q]\!]$ (parallel composition)

3) $\forall P, N \subseteq fv(P), n \in \mathbb{N}$, for all maximal computations ? our transforms

3) $\forall P, N \subseteq fv(P), n \in N$, for all maximal computations
if $p \xrightarrow{s}{}^{*} \xrightarrow{n} \xrightarrow{t}{}^{*} p'$ where $n \notin t, n \notin s$

$[\![P]\!]$ $\quad u \quad$ $[\![Q]\!]$

} our transforms $\{\{\cdot\}\}$ violate this property