Spatial and Behavioural types: safety, liveness and decidability

Lucia Acciai and Michele Boreale

Dipartimento di Sistemi e Informatica
Università degli Studi di Firenze

Lisbon, April 19–21, 2011
1 Introduction

2 Processes, types and formulae

3 The local and the global systems

4 Decidability

5 Conclusion
Logics and Types

Need to control the usage of (new) names in pi-calculus

**Spatial Logic:** suitable to
- analyze properties of systems
- describe the spatial structure of processes
- reason on distribution and concurrency

**Behavioral types:** combines static analysis and model checking
- abstract (the behavior of) processes
- simplify the analysis of concurrent message-passing processes
- properties are checked against types
- E.g. in [Igarashi, Kobayashi’01]
  - processes = pi-calculus, types = CCS
  - (global) invariant safety properties are considered
Our approach

Introduce a type system where

- processes and types share the same “shallow” spatial structure
- each block of declared names is annotated with a SL formula
- type safety: restricted processes are guaranteed to satisfy precise properties on bound names

Benefits

- properties not limited to safety invariants
- compositionality: only relevant names are considered when checking properties
Our approach

Introduce a type system where

- processes and types share the same “shallow” spatial structure
- each block of declared names is annotated with a SL formula
- type safety: restricted processes are guaranteed to satisfy precise properties on bound names

Benefits

- properties not limited to safety invariants
- compositionality: only relevant names are considered when checking properties
Outline

1. Introduction
2. Processes, types and formulae
3. The local and the global systems
4. Decidability
5. Conclusion
Processes

Pi-calculus with replicated input and guarded summation:

Prefixes  \( \alpha ::= a(\tilde{b}) \)  Input
|  \( \bar{a}\langle \tilde{b} \rangle \)  Output
|  \( \tau \)  Silent prefix

Processes  \( P ::= \sum_{i \in I} \alpha_i.P_i \)  Guarded summation
|  \( P|P \)  Parallel composition
|  \( (\nu\tilde{b})P \)  Restriction
|  \( !a(\tilde{b}).P \)  Replicated input
Types

**CCS with replicated input and guarded summation:**

Prefixes \( \mu ::= a \mid \overline{a} \mid \tau \)

Process types \( T ::= \sum_i \mu_i . T_i \)

- Guarded summation
- Parallel composition
- Restriction
- Replicated input

Channel types \( t ::= (\tilde{x} : \tilde{t})T \)
Shallow Logic (SL): examples of formulae

\text{shallow} = \text{input and output barbs are not followed by a continuation}

Race freedom:
\[ \text{NoRace}(a) \equiv \Box^* \neg H^*(\overline{a}|\overline{a}) \]

Unique receptiveness:
\[ \text{UniRec}(a) \equiv \Box^* \left( a \land \neg H^*(a|a) \right) \]

Responsiveness:
\[ \text{Resp}(a) \equiv \Box_{\neg a}^* \langle a \rangle \]

Deadlock freedom:
\[ \text{DeadFree}(a) \equiv \Box^* \left[ \left( \overline{a} \rightarrow H^*(\overline{a}|\langle a \rangle) \right) \land \left( a \rightarrow H^*(a|\langle a \rangle) \right) \right] \]
Well-annotated processes

\[ P ::= \cdots \mid (\forall \tilde{a} : \tilde{t}; \phi)P \quad \text{with} \quad \text{fn}(\phi) \subseteq \tilde{a} \]

with \( \phi \) a shallow logic formula

**Definition (well-annotated processes)**

A process \( P \in \mathcal{P} \) is *well-annotated* if whenever \( P \equiv (\forall \tilde{b})(\forall \tilde{a} : \phi)Q \) then \( Q \models \phi. \)
Remark: a "weakening" property of SL

Let $B$ with \( \text{fn}(B) = \emptyset \):

$A \models \phi \iff A \upharpoonright B \models \phi$

Necessary for soundness of scope extrusion

\[
(\nu \tilde{a} : \phi)(P \upharpoonright Q) \equiv (\nu \tilde{a} : \phi)(P \upharpoonright Q) \quad \text{if} \quad \tilde{a} \notin Q
\]

In (Caires and Cardelli’s) Spatial Logic this does not hold. E.g.

- $\neg (\neg 0 \upharpoonright \neg 0)$
- $\diamond T$
Remark: a “weakening” property of SL

Lemma

In Shallow Logic $\forall B$ with $\text{fn}(B) = \emptyset$: $A \models \phi \iff A \mid B \models \phi$

Necessary for soundness of scope extrusion

$$(\forall \tilde{a} : \phi)P \mid Q \equiv (\forall \tilde{a} : \phi)(P \mid Q) \text{ if } \tilde{a} \notin Q$$

In (Caires and Cardelli’s) Spatial Logic this does not hold. E.g.

- $\neg(\neg 0 \mid \neg 0)$
- $\Diamond T$
1 Introduction

2 Processes, types and formulae

3 The local and the global systems

4 Decidability

5 Conclusion
A “Local” Type System

Judgments: \( \Gamma \vdash_{L} P : T \)

Key rule: \((\text{T-RES})\):

\[
\frac{\Gamma, \tilde{\tilde{a}} : \tilde{\tilde{t}} \vdash P : T \quad T \downarrow_{\tilde{a}} \models \phi}{\Gamma \vdash (\nu \tilde{\tilde{a}} : \tilde{\tilde{t}}; \phi)P : (\nu \tilde{\tilde{a}} : \tilde{\tilde{t}})T}
\]

Local: in \((\text{T-RES})\) only the part of \(T\) depending on the restricted names, \(T \downarrow_{\tilde{x}}\), is taken into account - the rest is hidden

Example: \((a \cdot \tilde{b} \cdot \tilde{a} | (\nu c)(b \cdot c | \tilde{d} | \tilde{c})) \downarrow_{a} = a \cdot \tau \cdot \tilde{a} | (\nu c)(\tau \cdot c | \tau | \tilde{c})\)
A “Local” Type System

Judgments: \( \Gamma \vdash_{L} P : T \)

Key rule: \((T-\text{RES})\):

\[
\frac{\Gamma, \tilde{a} : \tilde{t} \vdash P : T \quad T \downarrow \tilde{a} \models \phi}{\Gamma \vdash (\forall \tilde{a} : \tilde{t}; \phi)P : (\forall \tilde{a} : \tilde{t})T}
\]

Local: in \((T-\text{RES})\) only the part of \(T\) depending on the restricted names, \(T \downarrow \tilde{x}\), is taken into account - the rest is hidden

Example: \(\downarrow_{a} (a \cdot b \cdot a \mid (\forall c)(b \cdot c \mid d \mid c)) = a \cdot \tau \cdot a \mid (\forall c)(\tau \cdot c \mid \tau \mid c)\)
A “Local” Type System

Judgments: $\Gamma \vdash_L P : T$

Key rule: $(T-\text{RES})$: $\frac{\Gamma, \bar{\alpha} : \bar{t} \vdash P : T \quad T \downarrow \bar{\alpha} \vdash \phi}{\Gamma \vdash (\nu \bar{\alpha} : \bar{t}; \phi) P : (\nu \bar{\alpha} : \bar{t})T}$

Local: in $(T-\text{RES})$ only the part of $T$ depending on the restricted names, $T \downarrow \bar{x}$, is taken into account - the rest is hidden

Example: $(a.b.\bar{a} | (\nu c)(b.c | d | \bar{c})) \downarrow_a = a.\tau.\bar{a} | (\nu c)(\tau.c | \tau | \bar{c})$

relevant names = newly created names
Definitions and Results

**Definition (negative formulae)**

In a negative formula each $\langle \neg \tilde{x} \rangle^*$ is under an **odd** number of $\neg$.

*Note: no limitations on other modalities!*

**Theorem (run-time soundness)**

Suppose that $\Gamma \vdash^L P : T$ and that $P$ is decorated with negative formulae of the form $\Box^* \phi$. Then $P \rightarrow^* P'$ implies that $P'$ is well-annotated.

**Race Freedom and Unique Receptiveness are negative**
Definitions and Results

Definition (negative formulae)
In a negative formula each $\langle \neg \tilde{x} \rangle^*$ is under an odd number of $\neg$

Note: no limitations on other modalities!

Theorem (run-time soundness)
Suppose that $\Gamma \vdash L P : T$ and that $P$ is decorated with negative formulae of the form $\square^* \phi$. Then $P \rightarrow^* P'$ implies that $P'$ is well-annotated.

Race Freedom and Unique Receptiveness are negative
A “Global” Type System: motivations

Type soundness does not hold for non-negative formulae like \( \text{Resp}(a) \) and \( \text{DeadFree}(a) \)

E.g.:

\[
R = (\nu a; \text{Resp}(a))(c.a|\overline{a})
\]

is well-typed for suitable \( \Gamma \). Indeed

\[
\Gamma, a \vdash_L c.a|\overline{a} : c.a|\overline{a}
\]

and

\[
(c.a|\overline{a}) \downarrow_a = \tau.a|\overline{a} \models \text{Resp}(a)
\]

but

\[
c.a|\overline{a} \not\models \text{Resp}(a)
\]

Problem: \( \text{Resp} \) on \( a \) also \textbf{depends} on a “global” name \( c \)
A “Global” Type System: motivations

Type soundness does not hold for non-negative formulae like $\text{Resp}(a)$ and $\text{DeadFree}(a)$

E.g.:

$$R = (\nu a; \text{Resp}(a))(c.a|\overline{a})$$

is well-typed for suitable $\Gamma$. Indeed

$$\Gamma, a \vdash_L c.a|\overline{a} : c.a|\overline{a}$$

and

$$(c.a|\overline{a}) \downarrow_a = \tau.a|\overline{a} \models \text{Resp}(a)$$

but

$$c.a|\overline{a} \not\models \text{Resp}(a)$$

Problem: $\text{Resp}$ on $a$ also depends on a “global” name $c$
Type soundness does not hold for non-negative formulae like $Resp(a)$ and $DeadFree(a)$

E.g.:

$$R = (\nu a; Resp(a))(c.a|\bar{a})$$

is well-typed for suitable $\Gamma$. Indeed

$$\Gamma, a \vdash_L c.a|\bar{a} : c.a|\bar{a}$$

and

$$(c.a|\bar{a}) \downarrow_a = \tau.a|\bar{a} \models Resp(a)$$

but

$$c.a|\bar{a} \not\models Resp(a)$$

**Problem:** $Resp$ on $a$ also **depends** on a “global” name $c$
A “Global” Type System

Main change:

\[ \downarrow x \text{ replaced by } \downarrow \tilde{x} \]

where \( T \downarrow \tilde{x} \) keeps the names in \( \tilde{x} \) and the causes of \( \tilde{x} \) in \( T \)

(plus some bookkeeping on names)

E.g.:

\[(c.a|\overline{a}) \downarrow a = c.a|\overline{a} \not\in Resp(a)\]

relevant names = new names + causally related free names
A “Global” Type System

Main change:

\[ \downarrow \tilde{x} \text{ replaced by } \downarrow_{\tilde{x}} \]

where \( T \downarrow_{\tilde{x}} \) keeps the names in \( \tilde{x} \) and the causes of \( \tilde{x} \) in \( T \)

(plus some bookkeeping on names)

E.g.:

\[ (c.a|\bar{a}) \downarrow_a = c.a|\bar{a} \not\in Resp(a) \]

relevant names = new names + causally related free names
Main change:

\[ \downarrow \tilde{x} \text{ replaced by } \downarrow \tilde{\tilde{x}} \]

where \( T \downarrow \tilde{x} \) keeps the names in \( \tilde{x} \) \textbf{and the causes of} \( \tilde{x} \) in \( T \)

(plus some bookkeeping on names)

E.g.:

\[ (c.a|\overline{a}) \downarrow_a = c.a|\overline{a} \not\in \text{Resp}(a) \]

relevant names = new names + causally related free names
Definitions and Results

Consider $\phi$ of the form

1. either $\Box^* \psi$ with negation not occurring underneath any $\langle \neg \tilde{y} \rangle^*$ in $\psi$

2. or $\Box^* \neg \tilde{y} \diamond \neg \psi'$, with negation not occurring in $\psi'$.

Theorem (run-time soundness)

Suppose that $\Gamma \vdash G P : T$ and that $P$ is decorated with formulae of the form (1) or (2) above. Then $P \rightarrow^* P'$ implies that $P'$ is well-annotated.

Responsiveness and Deadlock Freedom are of the form (2) and (1) respectively.
Definitions and Results

Consider \( \phi \) of the form

1. either \( \Box^* \psi \) with negation not occurring underneath any \( \langle \neg \tilde{y} \rangle^* \) in \( \psi \)
2. or \( \Box^* \neg \tilde{y} \Diamond^* \psi' \), with negation not occurring in \( \psi' \).

Theorem (run-time soundness)

Suppose that \( \Gamma \vdash_G P : T \) and that \( P \) is decorated with formulae of the form (1) or (2) above. Then \( P \rightarrow^* P' \) implies that \( P' \) is well-annotated.

Responsiveness and Deadlock Freedom are of the form (2) and (1) respectively.
Decidability of the type system

The type system is decidable provided that:

1. $\equiv$ is decidable
2. $\models$ is decidable
Decidability of the type system

The type system is decidable provided that:

1. $\equiv$ is decidable
2. $\models$ is decidable

1) $\equiv$ is decidable

From [Engelfriet & Gelsema 2004]
Decidability of the type system

The type system is decidable provided that:

1. \( \equiv \) is decidable
2. \( \vdash \) is decidable

1) \( \equiv \) is decidable

From [Engelfriet & Gelsema 2004]

2) \( \vdash \) is decidable (?)

The idea is to extend the approach in [BGZ04] for the decidability of weak barbs on CCS to handle SL
WSTS techniques for deciding “$|=\$”

Given a (decidable) preorder $\leq$ on types in $\mathcal{T}$

**Theorem ([Finkel and Schnoebelen’01])**

*Under certain conditions* for each $I \subseteq \mathcal{T}$ it is possible to compute a *finite* $X$ such that

$$\uparrow X = \text{Pred}^*(I)$$

*(finite basis of Pred}^*(I))*

Since $\llbracket \diamondsuit^* \phi \rrbracket = \text{Pred}^*(\llbracket \phi \rrbracket)$, to check $T \models \diamondsuit^* \phi$

1. set $I = \llbracket \phi \rrbracket$ above
2. check if $\exists S \in X$ s.t. $S \leq T$

$$\text{Pred}(s) = \{s' \mid s' \rightarrow s\} \quad \text{Pred}^*(s) = \{s' \mid s' \rightarrow^* s\}$$
Conditions [Finkel and Schnoebelen’01]

1. \( \mathcal{T} \) forms a **WSTS** w.r.t. (a decidable) \( \leq \)

2. \( \forall T \in \mathcal{T} \) it is possible to compute a **finite** \( Y \) s.t.
   \[ \uparrow Y = \uparrow \text{Pred}(\uparrow T) \quad \text{(effective pred-basis)} \]

3. \( \forall I(= [\phi]) \) it is possible to compute a **finite** \( Z \) s.t.
   \[ \uparrow Z = I(= [\phi]) \quad \text{(finite basis)} \]

**Our task:**
Find a preorder satisfying the three conditions above

**Our approach:**
Viewing types as forests and defining a preorder similar to Kruskal’s tree-preorder
Conditions [Finkel and Schnoebelen’01]

1. $\mathcal{T}$ forms a WSTS w.r.t. (a decidable) $\leq$

2. $\forall T \in \mathcal{T}$ it is possible to compute a finite $Y$ s.t.

   \[ \uparrow Y = \uparrow \text{Pred}(\uparrow T) \quad \text{(effective pred-basis)} \]

3. $\forall I (= \llbracket \phi \rrbracket)$ it is possible to compute a finite $Z$ s.t.

   \[ \uparrow Z = I (= \llbracket \phi \rrbracket) \quad \text{(finite basis)} \]

Our task:
Find a preorder satisfying the three conditions above

Our approach:
Viewing types as forests and defining a preorder similar to Kruskal’s tree-preorder
Preliminary definition

Fix an initial type $T_0$

Definition ($\mathcal{F}$)

$\mathcal{F} \triangleq$ the set of all terms:
- containing only subterms and restrictions of $T_0$
- having nesting depth smaller than $T_0$’s

E.g. $T_0 = (\nu a)(a.b|\overline{a}.\overline{b})$: $\begin{cases} (\nu a)(a.b|\overline{b}|a.b) \in \mathcal{F} \\ (\nu a)(\nu a)(a.b) \notin \mathcal{F} \end{cases}$
We consider types as forests where:

- **internal nodes** = restrictions
- **leaves** = prefix-guarded terms

E.g. \( T = (\nu a) (a.b | \overline{a}.b) | (\nu c) ((\nu d) c.d | \overline{c}.\overline{f}) \)

\[
\begin{align*}
(\nu a) & : a.b & (\nu c) : \overline{a}.b \\
& & (\nu d) : (c.d | \overline{c}.\overline{f}) \\
& & c.d
\end{align*}
\]
Make types a WSTS

Defining the preorder $\leq = \text{rooted tree embedding}$

\[ (\nu a) \quad (\nu c) \]
\[ \quad (\nu d) \leq \quad \]
\[ \quad \]
WSTS III: $\langle \mathcal{F}, \rightarrow, \leq \rangle$ is a WSTS

Theorem

(i) $\leq$ is a well-quasi order over $\mathcal{F}$ and (ii) $\langle \mathcal{F}, \rightarrow, \leq \rangle$ is a WSTS

Proof idea: (i) by induction on the nesting depth of restrictions of terms in $\mathcal{F}$ and by using the Higman's lemma. The base case (height = 0) relies on finiteness of guarded subterms in $T_0$. The inductive step relies on the fact that each forest can be decomposed into a finite number of subforests with smaller height.

(ii) $\langle \mathcal{F}, \rightarrow, \leq \rangle$ is a finitely branching transition system and $\leq$ is easily proved to be a computable simulation relation in $\mathcal{F}$.
Theorem

(i) $\leq$ is a well-quasi order over $\mathcal{F}$ and (ii) $\langle \mathcal{F}, \to, \leq \rangle$ is a WSTS

Proof idea: (i) by induction on the nesting depth of restrictions of terms in $\mathcal{F}$ and by using the Higman’s lemma. The base case (height = 0) relies on finiteness of guarded subterms in $T_0$. The inductive step relies on the fact that each forest can be decomposed into a finite number of subforests with smaller height.

(ii) $\langle \mathcal{F}, \to, \leq \rangle$ is a finitely branching transition system and $\leq$ is easily proved to be a computable simulation relation in $\mathcal{F}$.
WSTS III: \( \langle F, \rightarrow, \leq \rangle \) is a WSTS

**Theorem**

(i) \( \leq \) is a well-quasi order over \( F \) and (ii) \( \langle F, \rightarrow, \leq \rangle \) is a WSTS

Proof idea: (i) by induction on the nesting depth of restrictions of terms in \( F \) and by using the Higman’s lemma. The base case (height = 0) relies on finiteness of guarded subterms in \( T_0 \). The inductive step relies on the fact that each forest can be decomposed into a finite number of subforests with smaller height.

(ii) \( \langle F, \rightarrow, \leq \rangle \) is a finitely branching transition system and \( \leq \) is easily proved to be a computable simulation relation in \( F \).

NB: in CCS reductions cannot increase the nesting depth, on the contrary in pi-calculus \( (\nu b)\bar{a}\langle b \rangle | (\nu c)a(x).\bar{x}.c \rightarrow (\nu b)(\nu c)\bar{b}.c \)
Effective Pred-basis: \( \text{pb}(T) \)

\( \forall T \in \mathcal{T} \) it is possible to compute a \textbf{finite} \( Y \) s.t. \( \uparrow Y = \uparrow \text{Pred}(\uparrow T) \)
Effective Pred-basis: $\text{pb}(T)$

∀ $T \in \mathcal{T}$ it is possible to compute a finite $Y$ s.t. $\uparrow Y = \uparrow \text{Pred}(\uparrow T)$
Effective Pred-basis: $\text{pb}(T)$

∀ $T \in \mathcal{T}$ it is possible to compute a finite $Y$ s.t. $\uparrow Y = \uparrow \text{Pred}(\uparrow T)$

$G_1, G_2 = \text{prefix-guarded processes (leaves)}$
Effective Pred-basis: $pb(T)$

∀ $T \in \mathcal{T}$ it is possible to compute a [finite] $Y$ s.t. $\uparrow Y = \uparrow Pred(\uparrow T)$

$G_1, G_2 =$ prefix-guarded processes (leaves)
Effective Pred-basis: $\text{pb}(T)$

2. $\forall T \in \mathcal{T}$ it is possible to compute a **finite** $Y$ s.t. $\uparrow Y = \uparrow \text{Pred}(\uparrow T)$

**Theorem**

$\forall T \in \mathcal{T}: \text{pb}(T)$ is effective and $\uparrow \text{pb}(T) = \uparrow \text{Pred}(\uparrow T)$
Finite-basis: \( \uparrow fb(\phi) = [[\phi]] \cap \mathcal{F} \)

3. \( \forall I (= [[\phi]]) \) it is possible to compute a finite \( Z \) s.t. \( \uparrow Z = I (= [[\phi]]) \)

\[ (G = \text{prefix-guarded process (leaf) } \quad D = \text{context of parallel and restrictions}) \]

**Definition \( fb(\phi) \)**

\[
fb(a) \triangleq \{ D[G] \in \mathcal{F} \mid G \uparrow a \} \quad (\forall a)
\]
Finite-basis: \( \uparrow fb(\phi) = [\![\phi]\!] \cap \mathcal{F} \)

\[ \forall I (= [\![\phi]\!] ) \text{ it is possible to compute a finite } Z \text{ s.t. } \uparrow Z = I (= [\![\phi]\!]) \]

\( G = \) prefix-guarded process (leaf) \( \quad D = \) context of parallel and restrictions

**Definition \( fb(\phi) \)**

\[
\begin{align*}
fb(a) &\triangleq \{ D[G] \in \mathcal{F} \mid G \backslash a \} \\
fb(H^*(\phi_1|\phi_2)) &\triangleq \bigcup_{S_i \in fb(\phi_i)} \{ D[\tilde{G}_1, \tilde{G}_2] \in \mathcal{F} \mid \tilde{G}_i = \text{leaves}(S_i) \}
\end{align*}
\]
Finite-basis: \( \uparrow fb(\phi) = \llbracket \phi \rrbracket \cap \mathcal{F} \)

\( \forall I (= \llbracket \phi \rrbracket) \) it is possible to compute a finite \( Z \) s.t. \( \uparrow Z = I (= \llbracket \phi \rrbracket) \)

\( G = \) prefix-guarded process (leaf) \hspace{1cm} D = \) context of parallel and restrictions

**Definition (\( fb(\phi) \))**

\[
fb(a) \triangleq \{ D[G] \in \mathcal{F} \mid G \searrow a \}
\]

\[
fb(H^*(\phi_1 | \phi_2)) \triangleq \bigcup_{S_i \in fb(\phi_i)} \{ D[\tilde{G}_1, \tilde{G}_2] \in \mathcal{F} \mid \tilde{G}_i = \text{leaves}(S_i) \}
\]
Finite-basis: \[ \uparrow fb(\phi) = \llbracket \phi \rrbracket \cap \mathcal{F} \]

\[ 3 \quad \forall I (= \llbracket \phi \rrbracket) \text{ it is possible to compute a finite } Z \text{ s.t. } \uparrow Z = I (= \llbracket \phi \rrbracket) \]

\[ (G = \text{prefix-guarded process (leaf)} \quad - \quad D = \text{context of parallel and restrictions}) \]

**Definition (fb(\phi))**

\[ fb(a) \overset{\Delta}{=} \{ D[G] \in \mathcal{F} \mid G \ \backslash \ a \} \]

\[ fb(H^*(\phi_1 | \phi_2)) \overset{\Delta}{=} \bigcup_{S_i \in fb(\phi_i)} \{ D[\tilde{G}_1, \tilde{G}_2] \in \mathcal{F} \mid \tilde{G}_i = \text{leaves}(S_i) \} \]

\[ fb(\phi_1 \lor \phi_2) \overset{\Delta}{=} fb(\phi_1) \cup fb(\phi_2) \]
Finite-basis: \( \uparrow fb(\phi) = \llbracket \phi \rrbracket \cap \mathcal{F} \)

\( \forall I (= \llbracket \phi \rrbracket) \) it is possible to compute a finite \( Z \) s.t. \( \uparrow Z = I (= \llbracket \phi \rrbracket) \)

\( G = \) prefix-guarded process (leaf) \hspace{1cm} D = \) context of parallel and restrictions

**Definition (\( fb(\phi) \))**

\[
\begin{align*}
fb(a) & \triangleq \{ D[G] \in \mathcal{F} \mid G \downarrow a \} \\
fb(H^*(\phi_1 | \phi_2)) & \triangleq \bigcup_{S_i \in fb(\phi_i)} \{ D[\tilde{G}_1, \tilde{G}_2] \in \mathcal{F} \mid \tilde{G}_i = \text{leaves}(S_i) \} \\
fb(\phi_1 \lor \phi_2) & \triangleq fb(\phi_1) \cup fb(\phi_2) \\
fb(\diamond^* \phi) & \triangleq X \hspace{1cm} \text{s.t.} \hspace{1cm} \uparrow X = Pred^*(fb(\phi))
\end{align*}
\]

\[\cdots\]
What about $fb(\phi_1 \land \phi_2)$?

Idea:

$S_1 \in fb(\phi_1)$

$S_2 \in fb(\phi_2)$

$S = \text{least common multiple of } S_1 \text{ and } S_2$

E.g. $S_1 = a | b$, $S_2 = b | c \Rightarrow S = a | b | c$
What about $fb(\phi_1 \land \phi_2)$?

Idea:

- $S_1 \in fb(\phi_1)$
- $S_2 \in fb(\phi_2)$

$S_1$ and $S_2$ are the least common multiples of $S_1$ and $S_2$ respectively, where $S_1 = a \mid b$ and $S_2 = b \mid c$. Therefore, $S = a \mid b \mid c$. 
What about $fb(\phi_1 \land \phi_2)$?

Idea:

- $S_1 \in fb(\phi_1)$
- $S_2 \in fb(\phi_2)$
- $S = \text{"least common multiple" of } S_1 \text{ and } S_2$

E.g. $S_1 = a|b$, $S_2 = b|c \implies S = a|b|c$
Main results

Definition (monotone, anti-monotone and plain formulae)
- $\phi$ is **monotone** if it does not contain occurrences of $\neg$
- **anti-monotone** if it is of the form $\neg\psi$, with $\psi$ monotone
- $\phi$ is **plain** if it does not contain $\Diamond^*$ underneath $H^*$

Theorem (decidability on types and processes)

For any $\phi$ plain and (anti-)monotone
- $fb(\phi)$ is a computable finite basis for $[[\phi]] \cap F$
- $T \models \phi$ is decidable for any $T$
- $P \models \phi$ is decidable for any $P$ well-typed
Examples of decidable formulae

Never two concurrent outputs on $a$:

$$\text{NoRace}(a) \triangleq \neg \Diamond^* H^*(\bar{a} | \bar{a})$$

Communication on $a$ never occurs more than once:

$$\text{Linear}(a) \triangleq \neg \Diamond^* \langle a \rangle \Diamond^* \langle a \rangle$$

Resource $a$ never acquired in presence of the lock $l$:

$$\text{Lock}(a,l) \triangleq \neg \Diamond^* H^*(l | \langle a \rangle)$$
1 Introduction
2 Processes, types and formulae
3 The local and the global systems
4 Decidability
5 Conclusion
Further and related works

Further:

- Decidability: relax some constraints? Difficult:
  Known result: $\Diamond^* (a \land \neg b)$ is undecidable [Zavattaro’09]
- Quantitative behavioural types? Ongoing work

Related:

- **Behavioural types**: Acciai and Boreale’08; Chaki et al.’02; Igarashi and Kobayashi’01;
- **Decidability results in CCS**: Valencia et al.’09; Busi et al.’04
- **Spatial logics**: Caires’04
- **Undecidability results**: Kobayashi and Suto 2007
Type system

\[(T\text{-INP}) \quad \frac{\Gamma \vdash a : (\bar{x} : \bar{t}) T \quad fn(\bar{t}) \cup fn(T) \setminus \bar{x} = a, \quad \Gamma, \bar{x} : \bar{t} \vdash P : T|T'} \quad \bar{x} \notin fn(T')}{\Gamma \vdash a(\bar{x}) . P : a^a . T'}\]

\[(T\text{-OUT}) \quad \frac{\Gamma \vdash a : (\bar{x} : \bar{t}) T \quad \Gamma \vdash \bar{b} : \bar{t} \quad \Gamma \vdash P : S}{\Gamma \vdash \bar{a}(\bar{b}) . P : \bar{a} . (T[\bar{b}/\bar{x}]|S)}\]

\[(T\text{-RES}) \quad \frac{\Gamma, a : t \vdash P : T \quad a = fn(t)}{\Gamma \vdash (\nu a : t) P : (\nu a^a) T}\]

\[(T\text{-PAR}) \quad \frac{\Gamma \vdash P : T \quad \Gamma \vdash Q : S}{\Gamma \vdash P|Q : T|S}\]

\[(T\text{-SUM}) \quad |I| \neq 1 \quad \forall i \in I : \Gamma \vdash \alpha_i . P_i : \mu_i . T_i\]

\[(T\text{-EQ}) \quad \frac{\Gamma \vdash P : T \quad T \equiv S}{\Gamma \vdash P : S}\]

\[(T\text{-REP}) \quad \frac{\Gamma \vdash a(\bar{x}) . P : a^a . T}{\Gamma \vdash !a(\bar{x}) . P : !a^a . T}\]

\[(T\text{-TAU}) \quad \frac{\Gamma \vdash P : T}{\Gamma \vdash \tau . P : \tau . T}\]


Example: Unique Receptiveness (a liveness property)

⇒ Local Type System

\[ UniRec(a) \trianglerighteq \square^*(a \land \neg H^*(a|a)) \]

\[ P = (\nu a, b, c ; UniRec(a))Q \]

\[ Q = ( (\bar{c}\langle a \rangle | a + b(x).x) | c(y).\bar{b}\langle y \rangle) \]

is well-typed. Indeed, for a suitable \( \Gamma \):

\[ \Gamma, a, b, c \vdash_L Q : T \trianglerighteq \bar{c}.\bar{b}.a | a + b | c \]

with

\[ T \downarrow_{a,b,c} = T \models UniRec(a) \]

hence well-typed by \((T-RES)\)
Example: Unique Receptiveness (a liveness property)

\[ UniRec(a) \triangleq \Box^*(a \land \neg H^*(a|a)) \]

\[ P = (\forall a, b, c ; UniRec(a))Q \]

\[ Q = ((\bar{c}\langle a \rangle | a + b(x).x) | c(y).\bar{b}\langle y \rangle) \]

is well-typed. Indeed, for a suitable \( \Gamma \):

\[ \Gamma, a, b, c \vdash_L Q : T \triangleq \bar{c}.\bar{b}.a | a + b | c \]

with

\[ T \downarrow_{a,b,c} = T \models UniRec(a) \]

hence well-typed by \((T-Res)\)
Example: Unique Receptiveness (a liveness property)

⇒ Local Type System

\[ UniRec(a) \triangleq \square^*(a \land \neg H^*(a|a)) \]

\[ P = (\nu a, b, c ; UniRec(a))Q \]

\[ Q = ( (\bar{c}\langle a \rangle \mid a + b(x).x) \mid c(y).\bar{b}\langle y \rangle) \]

is well-typed. Indeed, for a suitable \( \Gamma \):

\[ \Gamma, a, b, c \vdash_L Q : T \triangleq \bar{c}.\bar{b}.a \mid a + b \mid c \]

with

\[ T \downarrow_{a,b,c} = T \models UniRec(a) \]

hence well-typed by \((T-RES)\)
Example: Responsiveness

⇒ Global Type System

\[ \text{Resp}(a) \triangleq \Box^*_a \Diamond^* \langle a \rangle \]

\[ P = (\nu a : \text{Resp}(a))(\overline{c}\langle a \rangle) | Q \]

\[ Q = !c(x). (\overline{x} | x) | \overline{c}\langle b \rangle \]

is well-typed. Indeed, for a suitable \( \Gamma \):

\[ \Gamma \vdash_G \overline{c}\langle a \rangle | Q : \overline{c} | (\overline{a} | a) | !c | \overline{c} | (\overline{b} | b) \triangleq T \]

and

\[ T \Downarrow_a = \overline{c} | (\overline{a} | a) | !c | \overline{c} | (\tau | \tau) \vdash \text{Resp}(a) \]

hence well-typed by \((T-\text{RES})\)
Example: Responsiveness

⇒ Global Type System

\[ \text{Resp}(a) \triangleq \sqcap^*_a \Diamond^*(a) \]

\[ P = (\nu a : \text{Resp}(a))(\overline{c}(a))|Q \]

\[ Q = !c(x).(x|x)|\overline{c}(b) \]

is well-typed. Indeed, for a suitable \( \Gamma \):

\[ \Gamma \vdash_G \overline{c}(a)|Q : \overline{c}.(\overline{a}|a)|!c|\overline{c}.(\overline{b}|b) \triangleq T \]

and

\[ T \Downarrow_a = \overline{c}.(\overline{a}|a)|!c|\overline{c}.(\tau|\tau) \models \text{Resp}(a) \]

hence well-typed by \((T\text{-RES})\)
Example: Responsiveness

⇒ Global Type System

\[ \text{Resp}(a) \triangleq \Box^* a \diamond^* \langle a \rangle \]

\[ P = (\forall a : \text{Resp}(a))(\overline{c}(a))|Q \]

\[ Q = !c(x).(\overline{x}|x)|\overline{c}\langle b \rangle \]

is well-typed. Indeed, for a suitable \( \Gamma \):

\[ \Gamma \vdash_G \overline{c}\langle a \rangle | Q : \overline{c}(\overline{a}|a)|!c|\overline{c}.(\overline{b}|b) \triangleq T \]

and

\[ T \Downarrow_a = \overline{c}.(\overline{a}|a)!c|\overline{c}.(\tau|\tau) \models \text{Resp}(a) \]

hence well-typed by (T-RES)
Shallow Logic (SL)

\[ \phi ::= T \]

\[ [\neg \phi] = \mathcal{U} \setminus [\phi] \]

\[ [\phi \lor \phi] = [\phi_1 \lor \phi_2] = [\phi_1] \cup [\phi_2] \]

\[ [\phi \land \phi] = [\phi_1 \land \phi_2] = [\phi_1] \cap [\phi_2] \]

\[ [a] = \{ A \mid A \downarrow a \} \]

\[ [\bar{a}] = \{ A \mid A \downarrow \bar{a} \} \]

\[ [[\phi_1 \mid \phi_2]] = \{ A \mid \exists A_1, A_2 : A \equiv A_1 \upharpoonright A_2, A_1 \in [\phi_1], A_2 \in [\phi_2] \} \]

\[ [[H^*\phi]] = \{ A \mid \exists \bar{a}, B : A \equiv (\bar{a})B, \bar{a} \# \phi, B \in [\phi] \} \]

\[ [[\langle a \rangle \phi]] = \{ A \mid \exists B : A \xrightarrow{\langle a \rangle} B, B \in [\phi] \} \]

\[ [[\langle \bar{a} \rangle^* \phi]] = \{ A \mid \exists \sigma, B : A \xrightarrow{\sigma} B, \mathcal{N} \setminus \bar{a} \# \sigma, B \in [\phi] \} \]

\[ [[\langle - \bar{a} \rangle^* \phi]] = \{ A \mid \exists \sigma, B : A \xrightarrow{\sigma} B, \bar{a} \# \sigma, B \in [\phi] \} \]
Shallow Logic (SL)

\[ \phi ::= T \]

\[ [[T]] = U \]

\[ \neg \phi \]

\[ [[\neg \phi]] = U \setminus [[\phi]] \]

\[ \phi \lor \phi \]

\[ [[\phi_1 \lor \phi_2]] = [[\phi_1]] \cup [[\phi_2]] \]

\[ \phi \land \phi \]

\[ [[\phi_1 \land \phi_2]] = [[\phi_1]] \cap [[\phi_2]] \]

\[ a \]

\[ [[a]] = \{ A \mid A \downarrow a \} \]

\[ \bar{a} \]

\[ [[\bar{a}]] = \{ A \mid A \downarrow \bar{a} \} \]

\[ \phi \mid \phi \]

\[ [[\phi_1 \mid \phi_2]] = \{ A \mid \exists A_1, A_2 : A \equiv A_1 \upharpoonright A_2, A_1 \in [[\phi_1]], A_2 \in [[\phi_2]] \} \]

\[ H^* \phi \]

\[ [[H^* \phi]] = \{ A \mid \exists \tilde{a}, B : A \equiv (\tilde{\nu} \tilde{a})B, \tilde{a} \# \phi, B \in [[\phi]] \} \]

\[ \langle a \rangle \phi \]

\[ [[\langle a \rangle \phi]] = \{ A \mid \exists B : A \xrightarrow{\langle a \rangle} B, B \in [[\phi]] \} \]

\[ \langle \tilde{a} \rangle^{*} \phi \]

\[ [[\langle \tilde{a} \rangle^{*} \phi]] = \{ A \mid \exists \sigma, B : A \xrightarrow{\sigma} B, \mathcal{N} \setminus \tilde{a} \# \sigma, B \in [[\phi]] \} \]

\[ \langle - \tilde{a} \rangle^{*} \phi \]

\[ [[\langle - \tilde{a} \rangle^{*} \phi]] = \{ A \mid \exists \sigma, B : A \xrightarrow{\sigma} B, \tilde{a} \# \sigma, B \in [[\phi]] \} \]