

Constraint Logic Programming

Sylvain Soliman and François Fages
{Sylvain.Soliman,Francois.Fages}@inria.fr

INRIA – Project-Team CONTRAINTES

MPRI C-2-4-1 Course – September 2009 - February 2010

Part III: CLP - Operational and Fixpoint Semantics

- 1 Operational Semantics
 - CSLD resolution
 - Observables
- 2 Fixpoint Semantics
 - Fixpoint Preliminaries
 - Fixpoint Semantics of Successes
 - Fixpoint Semantics of Computed Answers
- 3 Program Analysis
 - Abstract Interpretation
 - Constraint-based Model Checking

Part IV: Logical Semantics

- 4 Logical Semantics of CLP(\mathcal{X})
 - Soundness
 - Completeness
- 5 Automated Deduction
 - Proofs in Group Theory
- 6 CLP(λ)
 - λ -calculus
 - Proofs in λ -calculus
- 7 Negation as Failure
 - Finite Failure
 - Clark's Completion
 - Soundness w.r.t. Clark's Completion
 - Completeness w.r.t. Clark's Completion

Part V: Practical CLP Programming

- 8 CLP implementation, the WAM
- 9 Optimizing CLP
- 10 Summing up

Part VI: Concurrent Constraint Programming

11 Introduction

- Syntax
- CC vs. CLP

12 Operational Semantics

- Transitions
- Properties
- Observables

13 Examples

- append
- merge
- $CC(\mathcal{FD})$

Part VII: CC - Denotational Semantics

- 14 Deterministic Case
 - Syntax
 - I/O Function
 - Terminal Stores
- 15 Constraint Propagation
 - Closure Operators
 - Chaotic Iteration
- 16 Non-deterministic Case
 - Problems
 - Blind Choice
 - Example: merge
- 17 Sequentiality

Non-deterministic $CC(\mathcal{X})$ with Local Choice (2)

Let $\llbracket \cdot \rrbracket : \mathcal{D} \times A \rightarrow \mathcal{P}(\mathcal{P}(\mathcal{C}))$ be the least fixpoint (for \subseteq) of

$$\begin{aligned} \llbracket \mathcal{D}.c \rrbracket &= \{\uparrow c\} \\ \llbracket \mathcal{D}.c \rightarrow A \rrbracket &= \{\mathcal{C} \setminus \uparrow c\} \cup \{\uparrow c \cap X \mid X \in \llbracket \mathcal{D}.A \rrbracket\} \\ \llbracket \mathcal{D}.A \parallel B \rrbracket &= \{X \cap Y \mid X \in \llbracket \mathcal{D}.A \rrbracket, Y \in \llbracket \mathcal{D}.B \rrbracket\} \\ \llbracket \mathcal{D}.A + B \rrbracket &= \llbracket \mathcal{D}.A \rrbracket \cup \llbracket \mathcal{D}.B \rrbracket \\ \llbracket \mathcal{D}.\exists xA \rrbracket &= \{\{d \mid \exists xc = \exists xd, c \in X\} \mid X \in \llbracket \mathcal{D}.A \rrbracket\} \\ \llbracket \mathcal{D}.p(\vec{x}) \rrbracket &= \llbracket \mathcal{D}.A[\vec{x}/\vec{y}] \rrbracket \end{aligned}$$

Theorem 1 ([MFP97])

For any process $\mathcal{D}.A$,

$$\mathcal{O}_{ts}(\mathcal{D}.A; c) = \{d \mid \text{there exists } X \in \llbracket \mathcal{D}.A \rrbracket \text{ s.t. } d = \min(\uparrow c \cap X)\}.$$

Part VIII

CC and Linear Logic

Part VIII: CC and Linear Logic

- 18 CC - Logical Semantics
 - Intuitionistic
 - Linear
 - Soundness
 - Completeness

- 19 Must Properties
 - Definition
 - Soundness
 - Completeness

- 20 Program Analysis
 - Equivalence
 - Phase Semantics and Theorem Proving

Logical Semantics of CC?

- CC calculus is sound but not complete w.r.t. CLP logical semantics interpreting *asks* as *tells*
- Interpreting $ask(c \rightarrow A)$ as logical implication leads to identify **CC transitions** with **logical deductions**:

$$left \rightarrow \frac{c \vdash_c d}{c \wedge (d \rightarrow A^\dagger) \vdash c \wedge A^\dagger} \quad \frac{p(\vec{x}) \vdash_{\mathcal{D}} A^\dagger}{c \wedge p(\vec{x}) \vdash c \wedge A^\dagger}$$

(reverses the arrow of CLP interpretation...)

- To distinguish between successes and accessible stores agents shouldn't "disappear" by the \rightarrow rule:

Logical Semantics of CC?

- CC calculus is sound but not complete w.r.t. CLP logical semantics interpreting *asks* as *tells*
- Interpreting $ask(c \rightarrow A)$ as logical implication leads to identify **CC transitions** with **logical deductions**:

$$left \rightarrow \frac{c \vdash_c d}{c \wedge (d \rightarrow A^\dagger) \vdash c \wedge A^\dagger} \quad \frac{p(\vec{x}) \vdash_{\mathcal{D}} A^\dagger}{c \wedge p(\vec{x}) \vdash c \wedge A^\dagger}$$

(reverses the arrow of CLP interpretation...)

- To distinguish between successes and accessible stores agents shouldn't "disappear" by the **weakening** rule:

$$leftW \frac{\Gamma \vdash c}{\Gamma, A^\dagger \vdash c}$$

Linear Logic

- Introduced by Jean-Yves Girard in 1986 as a new *constructive* logic without the asymmetry of intuitionistic logic (sequent calculus with symmetric left and right sides)
- Logic of **resource consumption**

$$A \otimes A \not\vdash_{LL} A$$

$$A \otimes (A \multimap B) \vdash_{LL} B$$

$$A \otimes (A \multimap B) \not\vdash_{LL} A \otimes B$$

- $!A$ provides arbitrary duplication (unbounded throwable resource)

$$!A \otimes (A \multimap B) \vdash_{LL} !A \otimes B \vdash_{LL} B$$

- Sequent calculus **without weakening and contraction**

Intuitionistic Linear Logic

Multiplicatives

$$\frac{\Gamma, A, B \vdash C}{\Gamma, A \otimes B \vdash C} \quad \frac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B} \quad \frac{\Gamma \vdash A \quad \Delta, B \vdash C}{\Delta, \Gamma, A \multimap B \vdash C} \quad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B}$$

Additives

$$\frac{\Gamma, A \vdash C \quad \Gamma, B \vdash C}{\Gamma, A \oplus B \vdash C} \quad \frac{\Gamma \vdash A}{\Gamma \vdash A \oplus B} \quad \frac{\Gamma \vdash B}{\Gamma \vdash A \oplus B}$$

$$\frac{\Gamma, A \vdash C}{\Gamma, A \& B \vdash C} \quad \frac{\Gamma, B \vdash C}{\Gamma, A \& B \vdash C} \quad \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \& B}$$

Constants

$$\frac{\Gamma \vdash A}{\Gamma, \mathbf{1} \vdash A} \quad \vdash \mathbf{1} \quad \perp \vdash \quad \frac{\Gamma \vdash}{\Gamma \vdash \perp} \quad \Gamma \vdash \top \quad \Gamma, \mathbf{0} \vdash A$$

Intuitionistic Linear Logic (cont.)

Axiom - Cut

$$A \vdash A \quad \frac{\Gamma \vdash A \quad \Delta, A \vdash B}{\Delta, \Gamma \vdash B}$$

Bang

$$\frac{\Gamma, A \vdash B}{\Gamma, !A \vdash B} \quad \frac{\Gamma, !A, !A \vdash B}{\Gamma, !A \vdash B} \quad \frac{\Gamma \vdash B}{\Gamma, !A \vdash B} \quad \frac{! \Gamma \vdash A}{! \Gamma \vdash !A}$$

Quantifiers

$$\frac{\Gamma, A[t/x] \vdash B}{\Gamma, \forall x A \vdash B} \quad \frac{\Gamma \vdash A}{\Gamma \vdash \forall x A} \quad x \notin \text{fv}(\Gamma)$$

$$\frac{\Gamma, A \vdash B}{\Gamma, \exists x A \vdash B} \quad x \notin \text{fv}(\Gamma, B) \quad \frac{\Gamma \vdash A[t/x]}{\Gamma \vdash \exists x A}$$

Intuit. Linear Logic = the Logic of CC agents

Translation:

$$\begin{array}{lll} (c \rightarrow A)^\dagger = c \multimap A^\dagger & (A \parallel B)^\dagger = A^\dagger \otimes B^\dagger & \text{tell}(c)^\dagger = !c \\ (A + B)^\dagger = A^\dagger \& B^\dagger & (\exists x A)^\dagger = \exists x A^\dagger & p(\vec{x})^\dagger = p(\vec{x}) \\ & (X; c; \Gamma)^\dagger = \exists X (!c \otimes \Gamma^\dagger) & \end{array}$$

Axioms: $!c \vdash !d$ for all $c \vdash_c d$ $p(\vec{x}) \vdash A^\dagger$ for all $p(\vec{x}) = A \in \mathcal{D}$

Intuit. Linear Logic = the Logic of CC agents

Translation:

$$\begin{array}{lll}
 (c \rightarrow A)^\dagger = c \multimap A^\dagger & (A \parallel B)^\dagger = A^\dagger \otimes B^\dagger & \text{tell}(c)^\dagger = !c \\
 (A + B)^\dagger = A^\dagger \& B^\dagger & (\exists x A)^\dagger = \exists x A^\dagger & p(\vec{x})^\dagger = p(\vec{x}) \\
 (X; c; \Gamma)^\dagger = \exists X (!c \otimes \Gamma^\dagger)
 \end{array}$$

Axioms: $!c \vdash !d$ for all $c \vdash_c d$ $p(\vec{x}) \vdash A^\dagger$ for all $p(\vec{x}) = A \in \mathcal{D}$

Soundness and Completeness

If $(c; \Gamma) \rightarrow_{CC} (d; \Delta)$ then $c^\dagger \otimes \Gamma^\dagger \vdash_{ILL(c, \mathcal{D})} d^\dagger \otimes \Delta^\dagger$.

If $A^\dagger \vdash_{ILL(c, \mathcal{D})} c$ then *there exists a **success store** d such that $(\text{true}; A) \rightarrow_{CC} (d; \emptyset)$ and $d \vdash_c c$.*

If $A^\dagger \vdash_{ILL(c, \mathcal{D})} c \otimes \top$ then *there exists an **accessible store** d such that $(\text{true}; A) \rightarrow_{CC} (d; \Gamma)$ and $d \vdash_c c$.*

Soundness

Theorem 2 (Soundness of transitions)

Let $(X; c; \Gamma)$ and $(Y; d; \Delta)$ be CC configurations.

If $(X; c; \Gamma) \equiv (Y; d; \Delta)$ then $(X; c; \Gamma)^\dagger \dashv\vdash_{ILL(\mathcal{C}, \mathcal{D})} (Y; d; \Delta)^\dagger$.

If $(X; c; \Gamma) \longrightarrow (Y; d; \Delta)$ then $(X; c; \Gamma)^\dagger \vdash_{ILL(\mathcal{C}, \mathcal{D})} (Y; d; \Delta)^\dagger$.

Proof.

By induction on \equiv . Immediate.

By induction on \longrightarrow .

The choice operator $+$ is translated by the additive conjunction $\&$, which expresses “may” properties: $A \& B \vdash A$ and $A \& B \vdash B$. \square

Completeness I

Theorem 3 (Observation of successes)

Let A be a CC agent and c be a constraint.

If $A^\dagger \vdash_{ILL(\mathcal{C}, \mathcal{D})} c$, then there exists a constraint d such that $(\emptyset; 1; A) \longrightarrow (X; d; \emptyset)$ and $\exists X d \vdash_{\mathcal{C}} c$.

Proof.

By induction on a sequent calculus proof π of

$$A_1^\dagger, \dots, A_n^\dagger \vdash_{ILL(\mathcal{C}, \mathcal{D})} \phi$$

where the A_i 's are agents and ϕ is either a constraint or a procedure name. □

Completeness II

Recall that \top is the additive true constant neutral for \otimes .

Theorem 4 (Observation of accessible stores)

Let A be a CC agent and c be a constraint.

*If $A^\dagger \vdash_{ILL(\mathcal{C}, \mathcal{D})} c \otimes \top$, then c is a store accessible from A ,
i.e. there exist a constraint d and a multiset Γ of agents such that
 $(\emptyset; 1; A) \longrightarrow (X; d; \Gamma)$ and $\exists Xd \vdash_{\mathcal{C}} c$.*

Proof.

The proof uses the first completeness theorem, and proceeds by induction for the right introduction of the tensor connective in $c \otimes \top$. □

Observing “must” Properties

Properties true on **all branches** on the derivation tree.

Redefine the operational semantics by a rewriting relation on **frontiers**, i.e. multisets of configurations

Blind choice

$$\langle (X; c; A + B), \Phi \rangle \Longrightarrow \langle (X; c; A), (X; c; B), \Phi \rangle$$

Tell

$$\langle (X; c; \text{tell}(d), \Gamma), \Phi \rangle \Longrightarrow \langle (X; c \wedge d; \Gamma), \Phi \rangle$$

Ask

$$\frac{c \vdash_c d}{\langle (X; c; d \rightarrow A, \Gamma), \Phi \rangle \Longrightarrow \langle (X; c; A, \Gamma), \Phi \rangle}$$

Procedure calls

$$\frac{(p(\vec{y}) = A) \in \mathcal{D}}{\langle (X; c; p(\vec{y}), \Gamma), \Phi \rangle \Longrightarrow \langle (X; c; A, \Gamma), \Phi \rangle}$$

Translating the Frontier Calculus in LL with

Translate

$$(A + B)^\ddagger =$$

$$\langle (X; c; A), \Phi \rangle^\ddagger =$$

same translation for the other operations

Theorem 5 (Soundness of transitions)

Let Φ and Ψ be two frontiers.

If $\Phi \equiv \Psi$ then $(\Phi)^\ddagger \Vdash_{ILL(C,D)} (\Psi)^\ddagger$.

If $\Phi \Longrightarrow \Psi$ then $\Phi^\ddagger \vdash_{ILL(C,D)} \Psi^\ddagger$.

Translating the Frontier Calculus in LL with \oplus

Translate

$$(A + B)^{\ddagger} = A^{\ddagger} \oplus B^{\ddagger}$$

$$\langle (X; c; A), \Phi \rangle^{\ddagger} = \exists X (c^{\ddagger} \otimes A^{\ddagger}) \oplus \Phi^{\ddagger}$$

same translation for the other operations

Theorem 5 (Soundness of transitions)

Let Φ and Ψ be two frontiers.

If $\Phi \equiv \Psi$ then $(\Phi)^{\ddagger} \Vdash_{ILL(C,D)} (\Psi)^{\ddagger}$.

If $\Phi \Longrightarrow \Psi$ then $\Phi^{\ddagger} \vdash_{ILL(C,D)} \Psi^{\ddagger}$.

Completeness III for “must” Properties

Theorem 6 (Observation of frontiers' accessible stores)

Let A be a CC agent and c be a constraint.

If $A^\ddagger \vdash_{ILL(C,D)} c \otimes \top$

then $\langle (\emptyset; 1; A) \rangle \Longrightarrow \langle (X_1; d_1; \Gamma_1), \dots, (X_n; d_n; \Gamma_n) \rangle$ with

$\forall j \exists X_j d_j \vdash_c c$

Theorem 7 (Observation of frontiers' success stores)

Let A be an CC agent and c be a constraint.

If $A^\ddagger \vdash_{ILL(C,D)} c$

then $\langle (\emptyset; 1; A) \rangle \Longrightarrow \langle (X_1; d_1; \emptyset), \dots, (X_n; d_n; \emptyset) \rangle$ with $\forall j \exists X_j d_j \vdash_c c$

Logical Equivalence of CC programs

Let $P = \mathcal{D}.A$ be a $CC(\mathcal{C})$ process.

Corollary 8

If $P^\dagger \dashv\vdash_{ILL(\mathcal{C}, \mathcal{D})} P'^\dagger$
then $\mathcal{O}_{ss}(P) = \mathcal{O}_{ss}(P')$ (same set of success stores)
and $\mathcal{O}_{as}(P) = \mathcal{O}_{as}(P')$ (same set of accessible stores).

Corollary 9

If $P^\ddagger \dashv\vdash_{ILL(\mathcal{C}, \mathcal{D})} P'^\ddagger$
then P and P' have the same set of accessible stores on all
branches
and the same success frontiers.

Proving Properties of CC Programs

- Proving *logical equivalence* of CC programs with the sequent calculus of LL:
 - focusing proofs (deterministic rules for the additives first)
 - lazy splitting (input/output contexts for the multiplicatives)
- Proving *safety properties* of CC programs with the *phase semantics* of LL [FRS98]

Soundness gives $\Gamma \vdash_{ILL} A$ implies $\forall \mathbf{P} \forall \eta \mathbf{P}, \eta \models (\Gamma \vdash A)$.

$\exists \mathbf{P}, \eta$, s.t. $\mathbf{P}, \eta \not\models (\Gamma \vdash A)$ implies $\Gamma \not\vdash_{ILL_{C,D}} A$.

Proposition 10

To prove a safety property $(c, A) \dashv\rightarrow (d, B)$, it is enough to show that \exists a phase space \mathbf{P} , a valuation η , and an element $a \in \eta((c, A)^\dagger)$ such that $a \notin \eta((d, B)^\dagger)$.

Implementations of LL Sequent Calculi

- Forum [Miller&al.] specification languages based on LL
- LO [Andreoli] Property of “focusing proofs” in LL
- Lolli [Cervesato Hodas Pfenning] Search for “Uniform proofs”
- Lygon [Harland Winikoff] Linear Logic Programming language

Problem of lazy splitting:

$$\frac{\vdash A, \Gamma \quad \vdash B, \Delta}{\vdash A \otimes B, \Gamma, \Delta} (\otimes)$$

First idea:

$$\frac{\vdash A - (\Gamma, \Delta); \Delta \quad \vdash B, \Delta}{\vdash A \otimes B, \Gamma, \Delta} (\otimes)$$

- problems with the rules for ! and for \top ...
- stacks are necessary

Part IX

LCC

Part IX: LCC

- 21 LCC
 - Syntax
 - Operational Semantics
- 22 Examples
 - Dining Philosophers
 - Other examples
 - Indexicals
- 23 Logical Semantics
 - Intuitionistic Linear Logic
 - Phase Semantics
 - Example
- 24 Modules
 - First class closures
 - Encoding modules
 - Code protection

Linear Constraint Systems $(\mathcal{C}, \vdash_{\mathcal{C}})$

\mathcal{C} is a set of formulas built from V, Σ with logical operators: $1, \otimes, \exists$ and $!$;

$\vdash_{\mathcal{C}} \subseteq \mathcal{C} \times \mathcal{C}$ defines the non-logical axioms of the constraint system.

$\vdash_{\mathcal{C}}$ is the least subset of $\mathcal{C}^* \times \mathcal{C}$ containing $\vdash_{\mathcal{C}}$ and closed by:

$$\begin{array}{c}
 c \vdash c \quad \frac{\Gamma, c \vdash d \quad \Delta \vdash c}{\Gamma, \Delta \vdash d} \quad \vdash 1 \quad \frac{\Gamma \vdash c}{\Gamma, 1 \vdash c} \\
 \\
 \frac{\Gamma \vdash c_1 \quad \Delta \vdash c_2}{\Gamma, \Delta \vdash c_1 \otimes c_2} \quad \frac{\Gamma, c_1, c_2 \vdash c}{\Gamma, c_1 \otimes c_2 \vdash c} \quad \frac{\Gamma \vdash c[t/x]}{\Gamma \vdash \exists x c} \quad \frac{\Gamma, c \vdash d}{\Gamma, \exists x c \vdash d} \quad x \notin \text{fv}(\Gamma, d) \\
 \\
 \frac{\Gamma, c \vdash d}{\Gamma, !c \vdash d} \quad \frac{! \Gamma \vdash d}{! \Gamma \vdash !d} \quad \frac{\Gamma \vdash d}{\Gamma, !c \vdash d} \quad \frac{\Gamma, !c, !c \vdash d}{\Gamma, !c \vdash d}
 \end{array}$$

A **synchronization constraint** is a constraint not appearing in $\vdash_{\mathcal{C}}$

Same agents and observables as CC

Processes $P ::= \mathcal{D}.A$

Declarations $\mathcal{D} ::= p(\vec{x}) = A, \mathcal{D} \mid \epsilon$

Agents $A ::= \text{tell}(c) \mid \forall \vec{x}(c \rightarrow A) \mid A \parallel A \mid A + A \mid \exists xA \mid p(\vec{x})$

- observing the set of **success stores**,
- observing the set of **terminal stores** (successes and suspensions),
- observing the set of **accessible stores**,

Same agents and observables as CC

Processes $P ::= \mathcal{D}.A$

Declarations $\mathcal{D} ::= p(\vec{x}) = A, \mathcal{D} \mid \epsilon$

Agents $A ::= \text{tell}(c) \mid \forall \vec{x}(c \rightarrow A) \mid A \parallel A \mid A + A \mid \exists xA \mid p(\vec{x})$

- observing the set of **success stores**,

$$\mathcal{O}_{ss}(\mathcal{D}.A; c) = \{\exists \vec{x}d \in \mathcal{C} \mid (\emptyset; c; A) \longrightarrow^* (\vec{x}; d; \epsilon)\}$$

- observing the set of **terminal stores** (successes and suspensions),

- observing the set of **accessible stores**,

Same agents and observables as CC

Processes $P ::= \mathcal{D}.A$

Declarations $\mathcal{D} ::= p(\vec{x}) = A, \mathcal{D} \mid \epsilon$

Agents $A ::= \text{tell}(c) \mid \forall \vec{x}(c \rightarrow A) \mid A \parallel A \mid A + A \mid \exists xA \mid p(\vec{x})$

- observing the set of **success stores**,

$$\mathcal{O}_{ss}(\mathcal{D}.A; c) = \{\exists \vec{x}d \in \mathcal{C} \mid (\emptyset; c; A) \longrightarrow^* (\vec{x}; d; \epsilon)\}$$

- observing the set of **terminal stores** (successes and suspensions),

$$\mathcal{O}_{ts}(\mathcal{D}.A; c) = \{\exists \vec{x}d \in \mathcal{C} \mid (\emptyset; c; A) \longrightarrow^* (\vec{x}; d; \Gamma) \dashv\rightarrow\}$$

- observing the set of **accessible stores**,

Same agents and observables as CC

Processes $P ::= \mathcal{D}.A$

Declarations $\mathcal{D} ::= p(\vec{x}) = A, \mathcal{D} \mid \epsilon$

Agents $A ::= \text{tell}(c) \mid \forall \vec{x}(c \rightarrow A) \mid A \parallel A \mid A + A \mid \exists xA \mid p(\vec{x})$

- observing the set of **success stores**,

$$\mathcal{O}_{ss}(\mathcal{D}.A; c) = \{\exists \vec{x}d \in \mathcal{C} \mid (\emptyset; c; A) \longrightarrow^* (\vec{x}; d; \epsilon)\}$$

- observing the set of **terminal stores** (successes and suspensions),

$$\mathcal{O}_{ts}(\mathcal{D}.A; c) = \{\exists \vec{x}d \in \mathcal{C} \mid (\emptyset; c; A) \longrightarrow^* (\vec{x}; d; \Gamma) \nrightarrow\}$$

- observing the set of **accessible stores**,

$$\mathcal{O}_{as}(\mathcal{D}.A; c) = \{\exists \vec{x}d \in \mathcal{C} \mid (\emptyset; c; A) \longrightarrow^* (\vec{x}; d; B)\}$$

Linear-CC(\mathcal{C}) Transitions

Tell $(X; c; \text{tell}(d), \Gamma) \longrightarrow (X; c \otimes d; \Gamma)$

Ask
$$\frac{c \vdash_c d \otimes e[\vec{t}/\vec{y}]}{(X; c; \forall \vec{y}(e \rightarrow A), \Gamma) \longrightarrow (X; d; A[\vec{t}/\vec{y}], \Gamma)}$$

Call
$$\frac{(p(\vec{y}) = A) \in \mathcal{D}}{(X; c; p(\vec{y}), \Gamma) \longrightarrow (X; c; A, \Gamma)}$$

Choice $(X; c; A + B, \Gamma) \longrightarrow (X; c; A, \Gamma)$
 $(X; c; A + B, \Gamma) \longrightarrow (X; c; B, \Gamma)$

Congr.
$$\frac{z \notin \text{fv}(A)}{\exists y A \equiv \exists z A[z/y]} \quad A \parallel B \equiv B \parallel A \quad A \parallel (B \parallel C) \equiv (A \parallel B) \parallel C$$

Bibliography I



Patrick Cousot and Radhia Cousot.

Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints.

In *POPL'77: Proceedings of the 6th ACM Symposium on Principles of Programming Languages*, pages 238–252, New York, 1977. ACM Press. Los Angeles.



Frank S. de Boer, Maurizio Gabbrielli, and Catuscia Palamidessi.

Proving correctness of constraint logic programming with dynamic scheduling.

In *Proceedings of SAS'96*, LNCS 1145. Springer-Verlag, 1996.



François Fages, Paul Ruet, and Sylvain Soliman.

Phase semantics and verification of concurrent constraint programs.

In *Proceedings of the 13th Annual IEEE Symposium on Logic In Computer Science*, pages 141–152, Indianapolis, 1998. IEEE Computer Society.



Kim Marriott Moreno Falaschi, Maurizio Gabbrielli and Catuscia Palamidessi.

Confluence in concurrent constraint programming.

Theoretical Computer Science, 183(2):281–315, 1997.



Vijay A. Saraswat, Martin C. Rinard, and Prakash Panangaden.

Semantic foundations of concurrent constraint programming.

In *POPL'91: Proceedings of the 18th ACM Symposium on Principles of Programming Languages*, 1991.