Progetto GR 2016-17

Verifica formale di modelli e programmi basata sulla trasformazione di clausole di Horn con vincoli

Emanuele De Angelis

Università degli Studi `G. d'Annunzio' Chieti-Pescara

joint work with
F. Fioravanti (UdA), M. C. Meo (UdA),
A. Pettorossi (U. Tor Vergata), and M. Proietti (IASI-CNR)

Convegno GNCS – Montecatini Terme, 14–16 febbraio 2018

Automatic Verification of (software) artifacts

Providing a **proof** that an artifact (for instance, a model or a program) satisfies its specification.

Use a **mathematical formalism** to:

- model artifacts, and
- derive specifications as theorems.

Constrained Horn Clauses (CHCs)

First order formulas of the form

$$\mathbf{B_1} \wedge \ldots \wedge \mathbf{B_n} \wedge \mathbf{c} \rightarrow \mathbf{H}$$

where:

- $\mathbf{B_1},\ldots,\mathbf{B_n}$, and **H** are atomic formulas, and
- **c** is a formula in a theory of constraints.

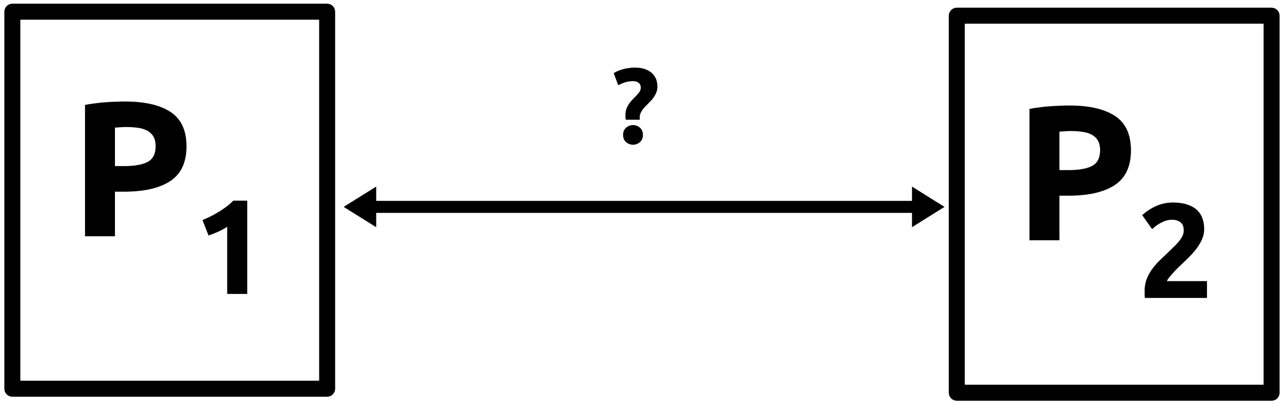
Formulas are universally quantified in front.

We use the syntax of Logic Programming

$$\mathbf{H} \leftarrow \mathbf{c}\,,\mathbf{B_1},\ldots,\mathbf{B_n} \qquad (\,\textbf{Head} \leftarrow \textbf{Body}\,)$$

Relational Verification

Proving relations between programs



Program Equivalence

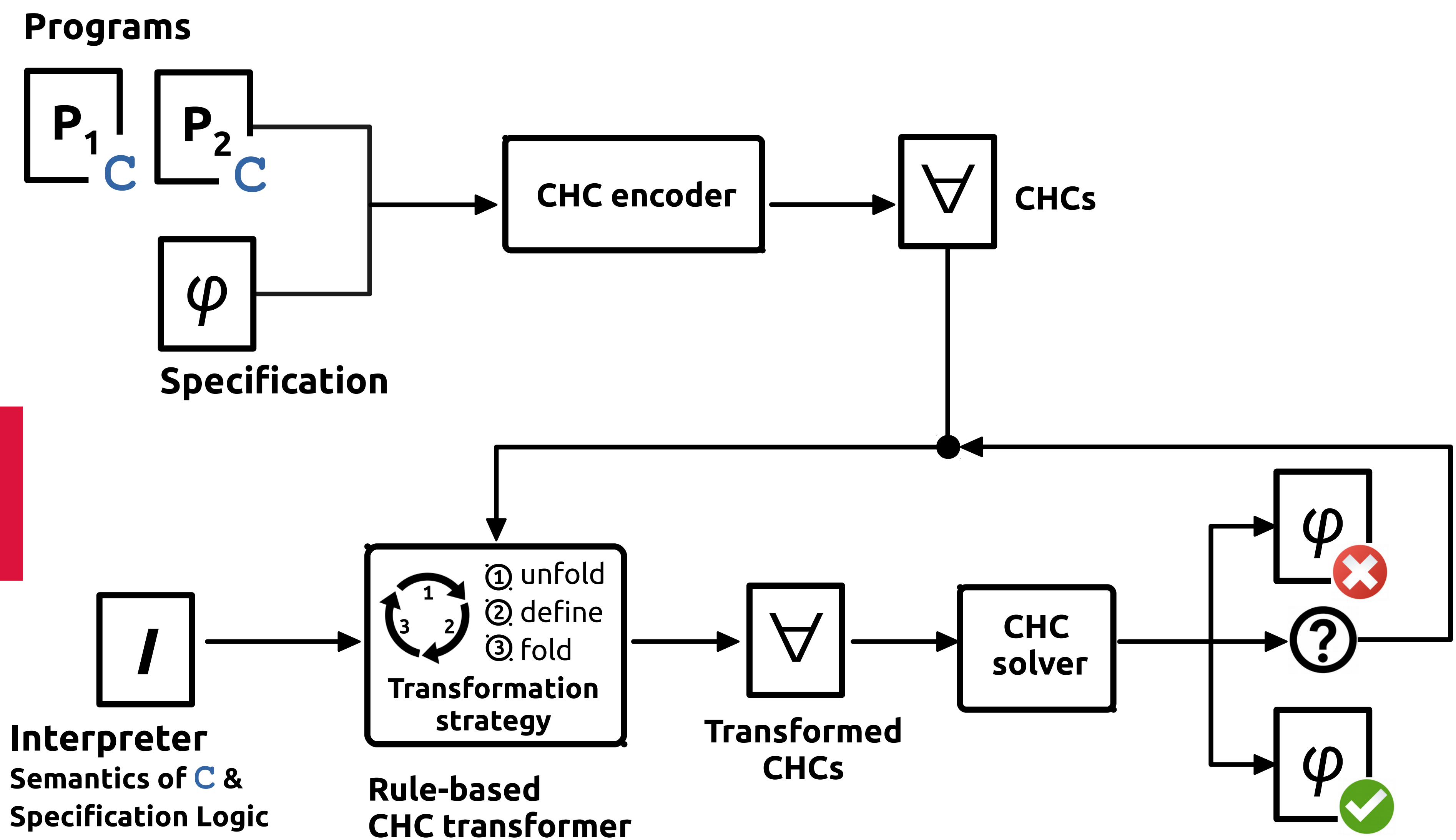
If $\mathbf{P}_1$ terminates on the input $\mathbf{i}_1$ producing $\mathbf{o}_1$ &
$\mathbf{P}_2$ terminates on the input $\mathbf{i}_2$ producing $\mathbf{o}_2$ &
$\mathbf{i}_1$ equals to $\mathbf{i}_2$

then

$\mathbf{o}_1$ equals to $\mathbf{o}_2$

Verification of Relational Properties

using transformation of CHCs



Example

P1

```
void sum_upto() {
  z1=f(x1);
}
int f(int n1){
  int r1;
  if (n1 <= 0) {
    r1 = 0;
  } else {
    r1 = f(n1 - 1) + n1;
  }
  return r1;
}
```

non-tail recursive

P2

```
void prod() {
  z2 = g(x2,y2);
}
int g(int n2, int m2){
  int r2;
  r2=0;
  while (n2 > 0) {
    r2 += m2;
    n2--;
  }
  return r2;
}
```

iterative

global variables of **P1**: $\{\mathtt{x1}, \mathtt{z1}\}$

global variables of **P2**: $\{\mathtt{x2}, \mathtt{y2}, \mathtt{z2}\}$

$$\mathtt{z1} = \sum_{\mathtt{i}=0}^{\mathtt{x1}} \mathtt{i}$$

$$\mathtt{z2} = \mathtt{x2} \times \mathtt{y2}$$

Relational property $\{\mathtt{x1} = \mathtt{x2},\ \mathtt{x2} \leq \mathtt{y2}\}\ \mathtt{sum\_upto} \sim \mathtt{prod}\ \{\mathtt{z1} \leq \mathtt{z2}\}$

Specifying relational properties using CHCs

The relational property $\{\varphi\}$ **P1** $\sim$ **P2** $\{\psi\}$ is translated into the clause

$$post(X', Y') \leftarrow pre(X, Y), p1(X, X'), p2(Y, Y')$$

pre-relation	φ	$pre(X, Y)$
input/output relation	**P1**	$p1(X, X')$
input/output relation	**P2**	$p2(Y, Y')$
post-relation	ψ	$post(X', Y')$

Linear Integer Arithmetic constraints

Relational property: $\{\texttt{x1} = \texttt{x2},\ \texttt{x2} \leq \texttt{y2}\}$ $\texttt{sum\_upto} \sim \texttt{prod}$ $\{\texttt{z1} \leq \texttt{z2}\}$

CHC translation: $\boxed{Z1 \leq Z2} \leftarrow \boxed{X1 = X2,\ X2 \leq Y2},\ su(X1, Z1),\ pr(X2, Y2, Z2)$

Interpreter (a glimpse)

Operational semantics of the programming language

$$\underline{prog(X, X')} \leftarrow \underline{initConf(C, X)}, \boxed{reach(C, C')}, \underline{finalConf(C', X')}$$

input/output relation

initial C and final C' configurations

$cf(cmd(Label, Command), Environment)$

$$\boxed{\begin{array}{l} reach(C, C) \\ reach(C, C2) \leftarrow \boxed{tr(C, C1)}, reach(C1, C2) \end{array}}$$

x=e;

$$\boxed{\begin{array}{l} tr(cf(cmd(L, asgn(\ V, expr(E))), Env), cf(cmd(L1,\ S), Env1)) \leftarrow \\ \quad eval(E, Env, V), update(Env, X, V, Env1), nextlab(L, L1), at(L1,\ S) \end{array}}$$

Interpreters & CHC specialization

Take advantage of static information, that is,

- actual programs
- relational property

to customize the interpreter.

By specializing the interpreter w.r.t. the static input, we get CHCs with <u>no references to</u>

- **reach**
- **tr**
- complex terms representing **configurations**

Input/Output relation of P1

```
void sum_upto() {
  z1 = f(x1);
}
int f(int n1) {
 int r1;
 if (n1<=0) {
   r1 = 0;
 } else {
   r1 = f(n1-1) + n1;
 }
 return r1;
}
```

su(X,Z') ← **f**(X,Z')

f(X,Z) ← N≤0, Z=0

f(N,Z) ← N≥1, N1=N-1, Z=R+N, **f**(N1,R)

input/output relation

Satisfiability of CHCs

$\{\varphi\}$ **P1** $\sim$ **P2** $\{\psi\}$

$$\boxed{Z1 \leq Z2} \leftarrow X1 = X2,\ X2 \leq Y2, su(X1, Z1),\ pr(X2, Y2, Z2)$$

$$\left.\begin{aligned}
&su(X,Z) \leftarrow f(X,Z)\\
&f(N,Z) \leftarrow N \leq 0,\ Z = 0\\
&f(N,Z) \leftarrow N \geq 1,\ N1 = N - 1,\ Z = R + N,\ f(N1,R)
\end{aligned}\right\} \textbf{P1}$$

$$\left.\begin{aligned}
&pr(X,Y,Z) \leftarrow W = 0,\ g(X,Y,W,Z)\\
&g(N,P,R,R) \leftarrow N \leq 0\\
&g(N,P,R,R2) \leftarrow N \geq 1,\ N1 = N - 1,\ R1 = P + R,\ g(N1,P,R1,R2)
\end{aligned}\right\} \textbf{P2}$$

prove the **validity** of a relational property **reduces** to
prove the **satisfiability** of CHCs

$$false \leftarrow X1 = X2,\ X2 \leq Y2,\ \boxed{Z1 > Z2,}\ su(X1, Z1),\ pr(X2, Y2, Z2)$$

Satisfiability of CHCs

$$
\begin{array}{l}
\mathit{false} \leftarrow X1 = X2,\ X2 \leq Y2,\ Z1 > Z2,\ su(X1, Z1),\ pr(X2, Y2, Z2) \\
su(X, Z) \leftarrow f(X, Z) \\
f(N, Z) \leftarrow N \leq 0,\ Z = 0 \\
f(N, Z) \leftarrow N \geq 1,\ N1 = N - 1,\ Z = R + N,\ f(N1, R) \\
pr(X, Y, Z) \leftarrow W = 0,\ g(X, Y, W, Z) \\
g(N, P, R, R) \leftarrow N \leq 0 \\
g(N, P, R, R2) \leftarrow N \geq 1,\ N1 = N - 1,\ R1 = P + R,\ g(N1, P, R1, R2)
\end{array}
$$

State-of-the-art solvers for CHCs with **Linear Integer Arithmetic** (**LIA**) look for **models** of **single atoms**: su and pr

Hence, LIA solvers should discover quadratic relations:

$$Z1' = X1 \times (X1 - 1)/2 \qquad Z2' = X2 \times Y2$$

Predicate pairing transformation

"Solution 1": use a solver for non-linear integer arithmetic

drawback: satisfiability of constraints is **undecidable** (decide satisfiability of Diophantine equations)

Solution 2: **predicate pairing** transformation

- composes the predicates f and g into a new predicate fg equivalent to their **conjunction**
- objective: discover linear relations among variables occurring in f and g may help solvers in proving the satisfiability of CHCs

Satisfiability of CHCs

Transformed CHCs

$$
\begin{aligned}
&false \leftarrow N \le Y,\ W=0,\ Z1 > Z2,\ fg(N,Z1,Y,W,Z2) \\
&fg(N,Z1,Y,Z2,Z2) \leftarrow N \le 0,\ Z1=0 \\
&fg(N,Z1,Y,W,Z2) \leftarrow N \ge 1,\ N1=N-1,\ Z1=R+N,\ M=Y+W, \\
&\qquad\qquad\qquad\qquad fg(N1,R,Y,M,Z2)
\end{aligned}
$$

Predicate Pairing makes it possible to **infer linear relations** among variables in the conjunction fg of predicates f and g

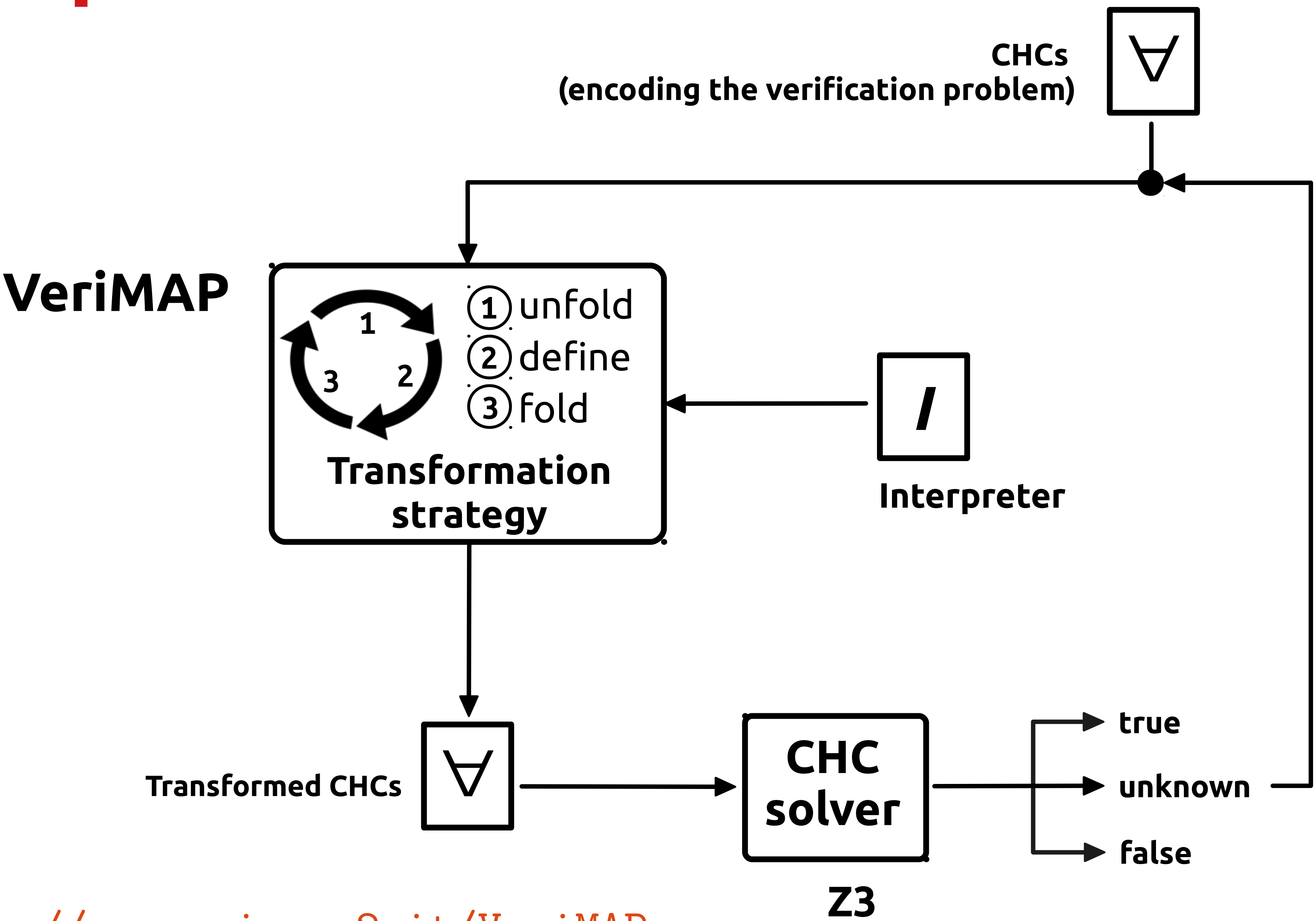
$$fg(N,Z1,Y,W,Z2) \ \leftarrow f(N,Z1),\ g(N,Y,W,Z2)$$

Whenever $W=0$ the conjunction fg enforces the linear constraint

$$(N > Y) \vee (Z1 \le Z2)$$

Hence the satisfiability of the first clause

Implementation

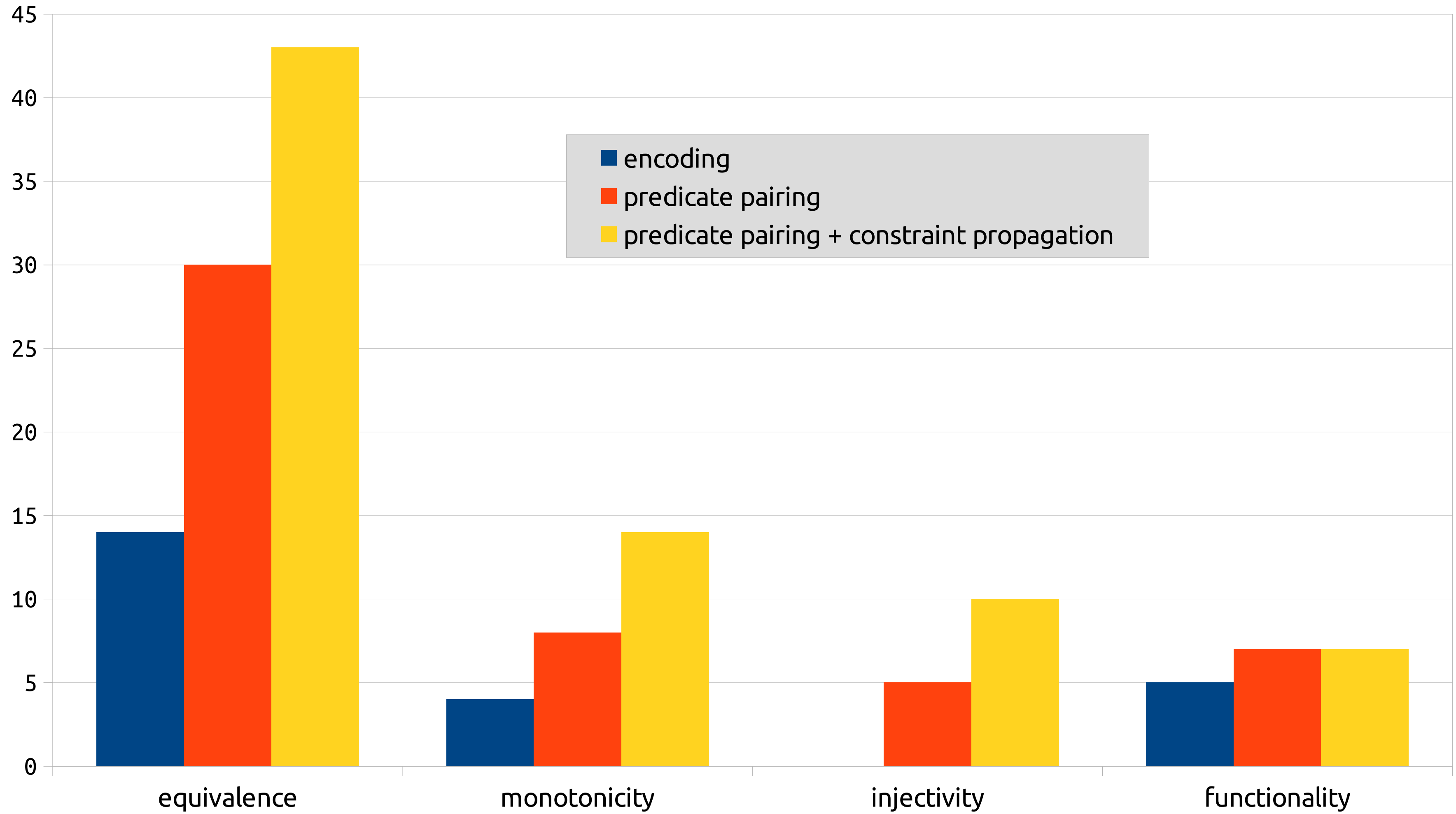


http://map.uniroma2.it/VeriMAP

Results

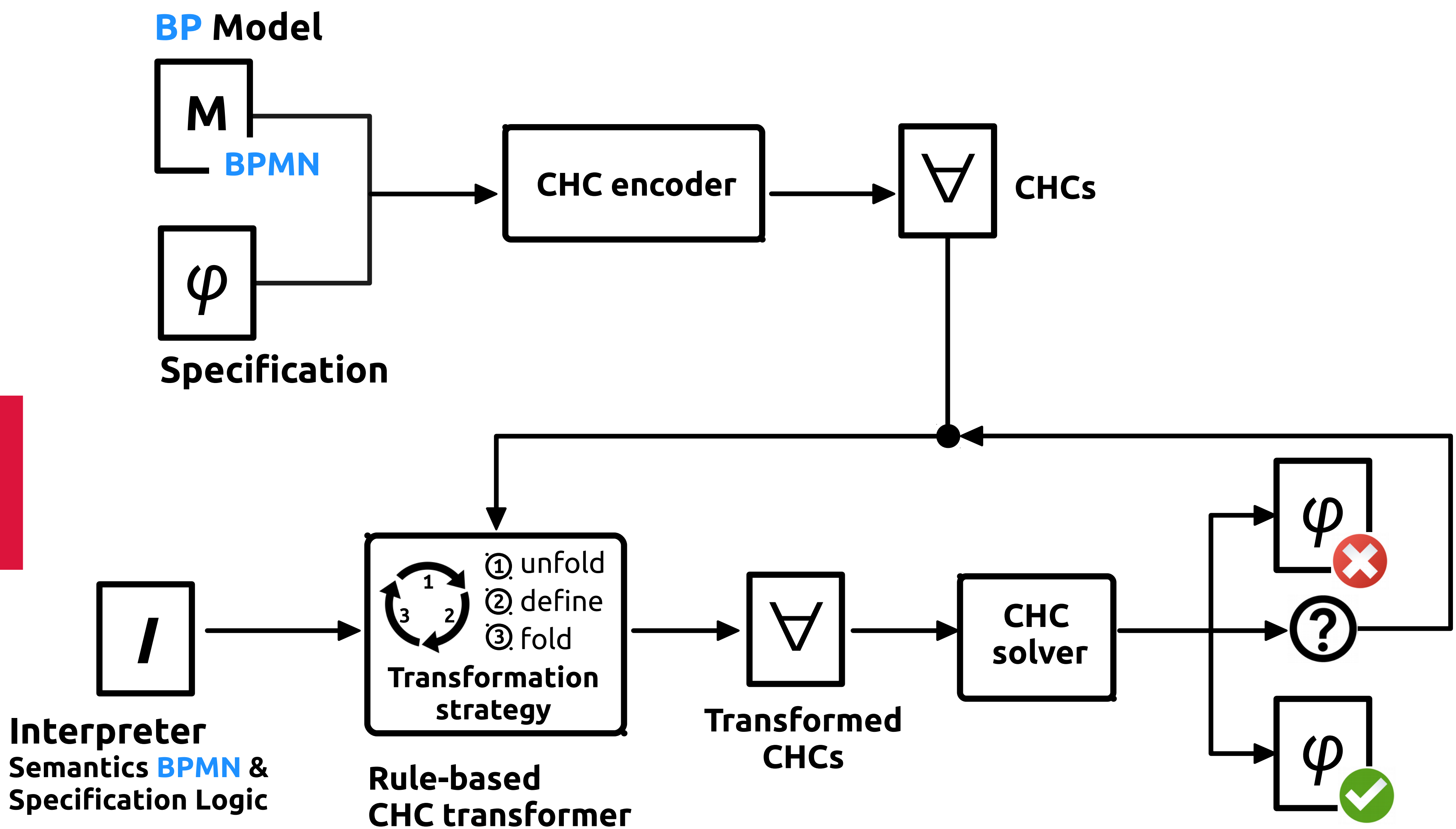
Equivalence	p1(X,X'), p2(Y,Y'), X = Y → X' = Y'
Monotonicity	p(X,X'), p(Y,Y'), X ≤ Y → X' ≤ Y'
Injectivity	p(X,X'), p(Y,Y'), X'= Y' → X = Y
Functionality	p(X,f(X),X'), p(Y,f(Y),Y'), X = Y → X' = Y'

Relational properties



Verification of Models

using transformation of CHCs

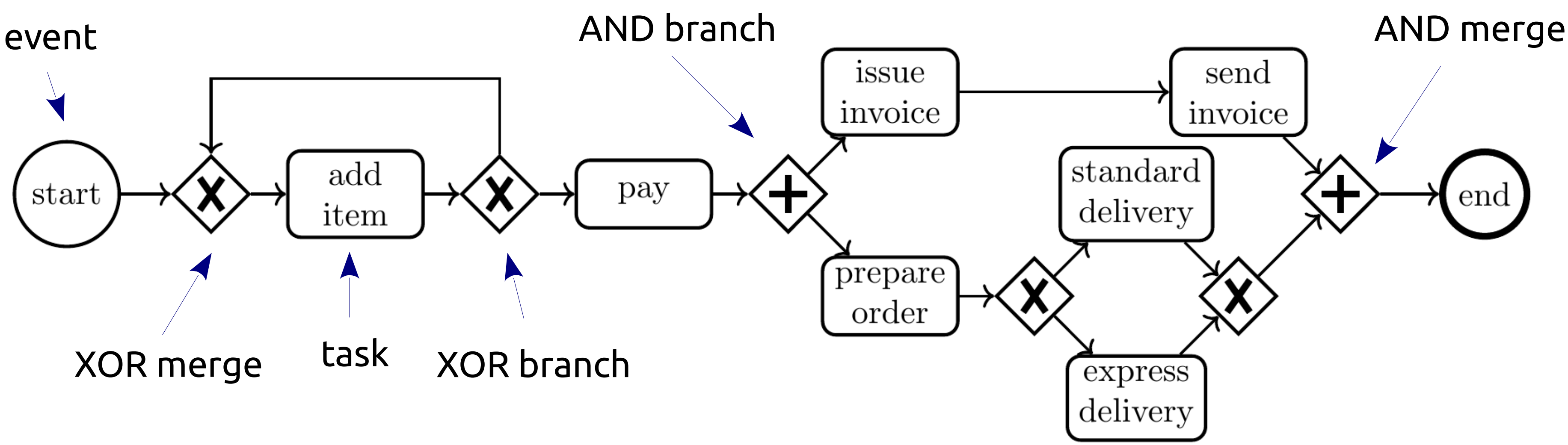


Business Process

A Business Process (BP) **coordinates** the activities of an organization towards a business goal.

A BP can be represented using the **Business Process Modeling Notation**.

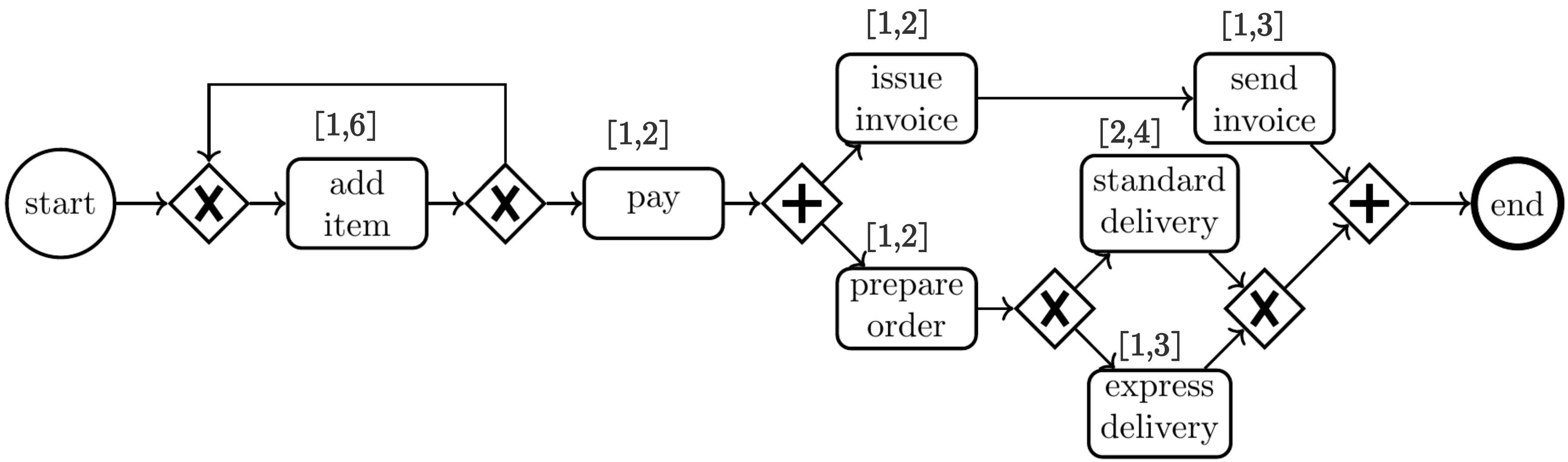
Purchase Order
A customer adds one or more items to the shopping cart and pays. Then, the vendor sends the invoice and delivers the order.



No quantitative time information (such as the durations of tasks).

Time-aware Business Process

Specify intervals of task duration $\mathbf{D} \in [\mathbf{dmin}, \mathbf{dmax}] \subset \mathbb{N}$



Properties

- **Reachability**
  The time to reach 'end' from 'start' is less than T.
- **Controllability**
  It is possible to determine the durations of some (controllable) tasks so that a given reachability property holds.

Interpreter (a glimpse)

Semantics of Business Process Modeling Notation

Reachability

$$reach(S, S, U, C).$$
$$reach(S, S2, U, C) \leftarrow tr(S, S1, U, C),\ reach(S1, S2, U, C)$$

where tr encodes the semantics of BPMN (Interpeter I).

Reachability Property

$$reachProp(U, C) \leftarrow c(T, U, C),\ reach(init,\ fin(T),\ U,\ C)$$

where $c(T, U, C)$ is a constraint.

Controllability Property

Weak $\quad I \cup LIA \models \forall U.adm(U) \rightarrow \exists C\ reachProp(U, C)$

Strong $\quad I \cup LIA \models \exists C\, \forall U.\ adm(U) \rightarrow reachProp(U, C)$

where $adm(U)$ iff the durations in U belong to the given intervals.

Applying CHCs solvers

Validity of weak and strong controllability properties

- **cannot be proved** by CHC solvers over LIA (such as Z3), because of complex terms occurring in the interpreter
- **cannot be proved** by CLP systems, because of $\exists\forall$ and $\forall\exists$
- CHC solvers and CLP system **may not terminate**, because of recursive definition of reach

Transformation techniques can be applied

- to get CHCs with no complex terms, and
- to avoid expensive quantifier elimination by reducing the problem of verifying controllability to the problem of verifying simpler properties where quantification is restricted to LIA constraints

Conclusions

A method for **proving** correctness of (software) artifacts

- **Independent of the formalism** used to represent the **artifact** and its **specification**

  The only language specific element is the **interpreter**

- **Improves effectiveness** of state-of-the art **CHC solvers**

Future work:

- more formalisms (programming and modeling languages)
- more properties

Publications

- Emanuele De Angelis, Fabio Fioravanti, Alberto Pettorossi, Maurizio Proietti: *Program Verification using Constraint Handling Rules and Array Constraint Generalizations*. Fundamenta Informaticae, vol. 150(1): 73-117 (2017)
- Emanuele De Angelis, Fabio Fioravanti, Alberto Pettorossi, Maurizio Proietti: *Semantics-based generation of verification conditions via program specialization*. Science of Computer Programming, 2017, 147: 78-108
- Emanuele De Angelis, Fabio Fioravanti, Maria Chiara Meo, Alberto Pettorossi, Maurizio Proietti: *Verification of Time-Aware Business Processes using Constrained Horn Clauses*. LOPSTR 2016: 38-55
- Emanuele De Angelis, Fabio Fioravanti, Maria Chiara Meo, Alberto Pettorossi, Maurizio Proietti: *Verifying Controllability of Time-Aware Business Processes*. RuleML+RR 2017: 103-118
- Emanuele De Angelis, Fabio Fioravanti, Alberto Pettorossi, Maurizio Proietti: *Predicate Pairing for Program Verification*. TPLP (2017 – in press)
- Emanuele De Angelis, Fabio Fioravanti, Alberto Pettorossi, Maurizio Proietti: *Enhancing Predicate Pairing with Abstraction for Relational Verification*. LOPSTR 2017 (to appear)