PIC2LNT: Model Transformation for Model Checking an Applied Pi-Calculus

Radu Mateescu and Gwen Salaün
Inria Grenoble – Rhône-Alpes and LIG / Convecs

http://convecs.inria.fr
Motivation

- Pi-calculus [Milner-Parrow-Walker-92]
  - Formalism for describing concurrent mobile processes
  - Various extensions proposed in two decades

- Difficult to provide and maintain analysis tools
  - Mobility Workbench (MWB) [Victor-Moller-94]

- Alternative solution
  - Translation to a value-passing process algebra (LNT)
  - Use of analysis tools for concurrent systems (CADP)

  \[\Rightarrow\text{easy extension to an applied pi-calculus by adding LNT data types and functions}\]
PIC: an applied pi-calculus
(syntax of behaviour terms)

\[
B ::= \text{nil} \\
| \ P (E_1, \ldots, E_n) \\
| \ \tau . B \\
| \ 'C <E_1, \ldots, E_n> . B \\
| \ C (X_1:T_1, \ldots, X_n:T_n) . B \\
| \ [E ] B \\
| \ ! k B \\
| \ (\text{new } C_1, \ldots, C_n) B \\
| \ \text{var } X_1:T_1 := E_1, \ldots, X_n:T_n := E_n \ \text{in } B \ \text{end var} \\
| \ B_1 + B_2 \\
| \ B_1 \mid B_2
\]

empty
agent call
silent prefix
emission
reception
guard
bounded replication
channel creation
variable definition
choice
parallel composition
Restrictions
(for finite-state verification)

- Finite control fragment [Dam-94]
  - No recursion through parallel composition
    \[ P = Q \parallel P \]

- Bounded replication
  - No “bang”
    \[ !P \]

- Recursion through channel creation allowed
  - But beware of infinite state space
    \[ P = (\text{new } c) \ 'a <c> . P \]
## Pi-calculus vs. LNT

**[Mateescu-Salaun-10]**

### Differences

<table>
<thead>
<tr>
<th>Pi-calculus</th>
<th>LNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary rendez-vous</td>
<td>Multi-way rendez-vous</td>
</tr>
<tr>
<td>Unidirectional</td>
<td>Bidirectional</td>
</tr>
<tr>
<td>communication</td>
<td>communication</td>
</tr>
<tr>
<td>Mobile channels</td>
<td>Static channels</td>
</tr>
<tr>
<td>Dynamic creation of</td>
<td>Static network</td>
</tr>
<tr>
<td>processes</td>
<td>of processes</td>
</tr>
<tr>
<td>Names only</td>
<td>Constructed datatypes</td>
</tr>
<tr>
<td>Action prefix</td>
<td>Symmetric sequential</td>
</tr>
<tr>
<td></td>
<td>compo.</td>
</tr>
</tbody>
</table>

### Similarities

- Choice, recursion
- Binary parallel composition
Translation from PIC to LNT

PIC2LNT 2.0

Diagram showing the process flow from PIC to LNT, including
- .pic
- .lib
- pic2lnt_dyn.tnt
- LNT2LOTOS
- .lotos
- CÆSAR & CÆSAR.ADT
- OPEN/CÆSAR API
- .mcl
- .ren
- EVALUATOR 4.0
- yes / no & diagnostic

Legend:
- : input/output
- : reference
- : input code
- : intermediate code

TACAS 2013 – Rome (Italy)
Running example
(architecture)

User

ask

Desk

dial

Service

notify

static channels

mobile channel
Running example
(PIC code)

Main =

(new ask, notify) (User (ask) | Desk (ask, notify) | Service (notify))

User (ask) =

ask (c) . 'echo <ask, c> . 'c <1 of Nat> . 'echo <c, req, 1 of Nat> .
c (r:Nat) . 'echo <c, res, r> . User (ask)

Service (notify) =

notify (c) . c (n:Nat) . 'c <n + 10> . Service (notify)

Desk (ask, notify) =

(new dial) Desk_2 (ask, notify, dial)

Desk_2 (ask, notify, c) =

'ask <c> . 'notify <c> . Desk_2 (ask, notify, c)
LTS construction

**PIC2BCG:**

- Generate the LTS (in BCG format) of a PIC specification
- Rename the labels in pi-calculus style
Verification

Evaluator 4.0:
- On-the-fly model checking
- MCL properties (data-based modal mu-calculus)
- Label hiding and renaming

Safety property:
\[
[ \text{(not \{ASK !"DIAL"\})}* . \{DIAL !"REQ" ?n:Nat\} ] \text{ false}
\]

Liveness property:
\[
[ \text{true}* . \{DIAL !"REQ" ?n:Nat\} ] < \{DIAL !"RES" !n+10\} > \text{true}
\]
Conclusion

PIC: an applied pi-calculus (+ LNT data types)
- Syntax and semantics
- PIC2LNT 2.0 translator $\rightarrow$ connection to CADP
- Used for teaching concurrency (Saarland University)

Future work:
- Application domains
  - Systems biology
  - Cryptographic protocols
  - Cloud computing