Parameterised Session Types

Communication Patterns Through the Looking Glass of Session Types

Andi Bejleri
Imperial College London
Behavioural Types- April 21, 2011
Session Types: What do they offer

- Intuitive, light-weight type annotation
  - Syntax of simply names and arrows, no question marks or bangs
  - One type for n processes

- A mode of organising structured communications from a global point of view
  - Consequently, describe the order of communications (part of programs logic)

- Efficient type-checking strategy of processes through projection of global types onto participants
Example: Buyer-Store-Credit Agency

Sequence diagram

Global type

Branching Causality

Buyer → Store :<String>.
Store → Credit Agency :<Int>.
Credit Agency → Store :

Failed:

Approved:

Message Causality

Processes
A chatting room

Session

Process

Private channels

Global type

Shared names

\( a[1, 2, 3, 4] \)

Keys = 1, 2, …
Semantics
Communication patterns

- Describe simple and elegant structured interactions
  - Ring, Star, Tree, 2D-Mesh, Hypercube
  - Used in: parallel algorithms, data exchange protocols, web services

- Simulation of the n-body algorithm over the Ring pattern
  - First iteration: send particles to the neighbor on its left and calculate forces exerted within their particles.
  - i-th iteration: forward particles received in i-1 iteration, calculate forces of them and receive the next particles set
Guaranteeing communication-safety for communication patterns I

• Communication patterns and global types describe structured interactions
  ◦ Global types serve not only as a blue print of the system’s architecture but also as a type system that guarantees communication-safety
Guaranteeing communication-safety for communication patterns II

- Communication patterns describe structured interactions of an arbitrary number of participants
  - In the n-body algorithm (for another implementation) one needs 80 processes for 64 particles and 110 for 128 to obtain a good speed up.

- Unable to express sessions where the number of participants is known only at run-time
  - In parallel algorithms, the number of participants depends on the size of the problem instance
Parameterised Session Types

1. Parameterise participants
2. Iterate over parameterised causalities that abstract repetitive behavior of pattern
3. Compose sequentially global types

Gödel’s $R$— primitive recursive function
Encoded as an iterator, for $(i=n-1; i>0; i--)G; G'$
The Ring communication pattern

R n → 0:<U>.end
λ i. λ X.n-i-1 → n-i:<U>.X

R 2 → 0:<U>.end
λ i. λ X.2-i-1 → 2-i:<U>.X

Ring of length 3

iterate

0 → 1:<U>.
1 → 2:<U>.
2 → 0:<U>.
end
Roles

- Blueprint that describes the nature of a communication pattern and the behavior that all run-time processes will share
- Concept similar to classes
In our calculus (not complete):

\[ \text{Starter} = y!\langle W[1], 1\rangle; y?\langle W[n], x\rangle; R \]

Communication abstraction of processes/objects that will be generated at runtime

Abstraction of neighbors

In Java (not complete):

```java
public class Starter{
   public Starter(int port_l, String host_r, int port_r){
      // Set up the sockets for the pattern...
      PrintWriter out = null;
      BufferedReader in = null;
      try{
         serverSocket = new ServerSocket(port_l);
         clientSocket = new Socket(host_r, port_r);
         out = ... // Init. the output stream on clientSocket
         in = ... // Init. the input stream on serverSocket
         // Exchange messages with neighbors
         out.println("1");
         String m = in.readLine();
         ... // Close streams and sockets
      }
   }
}
```
Efficient type-checking through projection of global types onto the principals

\[ G = R \ n \rightarrow 0 : \langle U \rangle \ \text{end} \]
\[ \lambda i. \lambda X. n - i - 1 \rightarrow n - i : \langle U \rangle X \ n \]

\[ W[0] = \langle W[1], U \rangle; ?\langle W[n], U \rangle; \text{end} \]
\[ W[j + 1] = ?\langle W[j], U \rangle; !\langle W[j + 2], U \rangle; \text{end} \]
\[ W[n] = ?\langle W[n - 1], U \rangle; !\langle W[0], U \rangle; \text{end} \]

\[ \lambda n: \{ n > 1 \} . R \]
Projection

- Projection

\[
p \rightarrow p': \langle U \rangle G \uparrow q = \begin{cases} 
\langle p' \{ p = q \}, U \rangle (p); ? \langle p' = q \}, U \rangle (p'); G \uparrow q & \text{if } C \leq p = q \text{ and } C \leq p' = q \\
\langle p' \{ p = q \}, U \rangle (p); G \uparrow q & \text{if } C \leq p = q \\
\langle p' \{ p' = q \}, U \rangle (p'); G \uparrow q & \text{if } C \leq p' = q \\
G \uparrow q & \text{otherwise}
\end{cases}
\]

- Ring pattern

  - Middle worker \( W[j+1] \) (1..n-1) appear in both sides \( n-i-1 \rightarrow n-i \rightarrow \ldots 1 \rightarrow 2:<U> \rightarrow 2 \rightarrow 3:<U> \rightarrow \ldots \),
Sorting

- From projection of the Ring global type on $W[j+1]$: 

$$!\langle W[j+2], U\rangle(W[n-i-1]); ?\langle W[j], U\rangle(W[n-i])$$

- Sort the sequence of actions on their appearance in $G$ 

$$?\langle W[1], U\rangle;!\langle W[3], U\rangle$$

- Sorting of the sequence of actions on the participants of $G$ who represent them, $W[n-i-1], W[n-i]$
Static type system

- Values of parameters range over infinite sets of parameters

\[ G = R\ n \rightarrow 0 : \langle U \rangle \ \text{end} \]

\[ \lambda i. \lambda X. n - i - 1 \rightarrow n - i : \langle U \rangle X \ n \]

\[ \lambda n : \{ n > 1 \} \ . \ P \]
Control index calculation in MPI

- Index calculation in global types is simpler than the one in roles
  - A direct advantage of the global representation of interactions

\[
G = R \end
\]

\[
\lambda i. \lambda X i \rightarrow 2^* i + 1 : \langle U \rangle.
\]

\[
i \rightarrow 2^* i + 2 \cdot \langle U \rangle \cdot X 2^n - 1
\]

\[
OddLeaf = a[2^*(2^{n-1} + i - 1) + 1](y) \cdot y ?(2^{n-1} + i - 1, z); S
\]
Final remarks

- A system that expresses sessions of an arbitrary number of processes through the role idiom, preserving:
  - An intuitive, light-weight type annotation
  - A global description of structured interactions
  - Efficient type-checking strategy: projection of global types onto participants and sorting of actions in the role types
  - A non-conservative type system that allows parameters to range over an infinite set of values
References

- Parameterised Session Types: Communication patterns through the looking glass of session types. Dissertation (To appear sometime in May-June)

  - Practical Parameterised Session Types. ICFEM 2010
  - Synchronous Multiparty Session Types. PLACES 2008.

http://www.doc.ic.ac.uk/~ab406/