RADU POPESCU

CURRY ON! BARCELONA CONFERENCE TRIP REPORT

CURRY ON! JUNE 19-20 2017

"A new and unusual non-profit conference focused on programming languages & emerging challenges in industry."

CURRY (FOOD)

CURRYING (<- HASKELL CURRY)

For `f(x,y), x in Z, y in R`, go from

```
f: ZxR -> Z
```

to

```
f: Z -> (R -> Z)
```

In Haskell REPL:

```
Prelude> let f :: Int -> Double -> Int; f a x = …

Prelude> let f’ = f 2

Prelude> :type f'

f' :: Double -> Int
```

CURRY ON! JUNE 19-20 2017

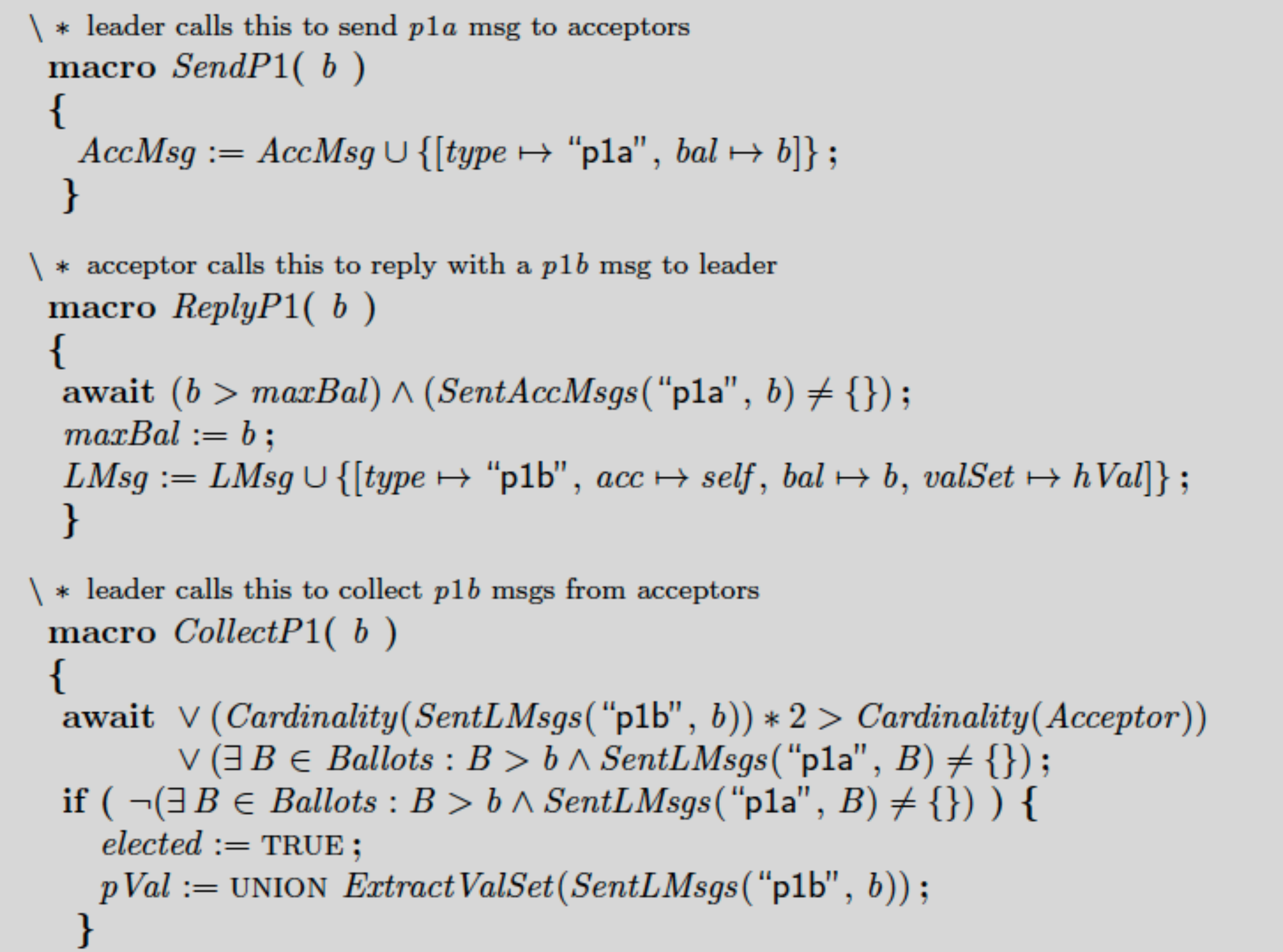
- http://www.curry-on.org/2017/
- 3rd edition
- Academia + Industry
- Colocated with ECOOP and PLDI
- All talks are on YouTube: https://www.youtube.com/channel/UC-WICcSW1k3HsScuXxDrp0w

FOUR KEYNOTES

- Martin Odersky (Scala) - What to leave implicit
- Brian Goetz (Java) - FP is dead: Long live FP
- José Valim (Elixir) - Idioms for building distributed fault-tolerant application with Elixir
- Roberto Ierusalimschy (Lua) - Scripting with Lua

RON PRESSLER, “THE PRACTICE AND THEORY OF TLA+”

TLA+: a specification language created by Leslie Lamport



```
\* leader calls this to send $p1a$ msg to acceptors
 macro $SendP1(\ b\ )$
 {
   $AccMsg := AccMsg \cup \{[type \mapsto \text{“p1a”},\ bal \mapsto b]\}$ ;
  }

\* acceptor calls this to reply with a $p1b$ msg to leader
 macro $ReplyP1(\ b\ )$
 {
  await $(b > maxBal) \land (SentAccMsgs(\text{“p1a”},\ b) \neq \{\})$ ;
  $maxBal := b$ ;
  $LMsg := LMsg \cup \{[type \mapsto \text{“p1b”},\ acc \mapsto self,\ bal \mapsto b,\ valSet \mapsto hVal]\}$ ;
  }

\* leader calls this to collect $p1b$ msgs from acceptors
 macro $CollectP1(\ b\ )$
 {
  await $\lor (Cardinality(SentLMsgs(\text{“p1b”},\ b)) * 2 > Cardinality(Acceptor))$
        $\lor (\exists B \in Ballots : B > b \land SentLMsgs(\text{“p1a”},\ B) \neq \{\})$ ;
  if ( $\neg(\exists B \in Ballots : B > b \land SentLMsgs(\text{“p1a”},\ B) \neq \{\})$ ) {
     $elected :=$ TRUE ;
     $pVal :=$ UNION $ExtractValSet(SentLMsgs(\text{“p1b”},\ b))$ ;
   }
```

MATT MIGHT, WINNING THE WAR ON ERROR, SOLVING THE HALTING PROBLEM AND CURING CANCER

- Program analysis tools used for genomic analysis
- Precision medicine
- Has actually found treatment for 5 genetic conditions
- Obama

JEAN YANG, PREVENTING INFORMATION LEAKS BY CONSTRUCTION

“Policy-agnostic programming addresses the issue of unintentional information leaks by factoring out the implementation of information flow security from other functionality”

JOHN HUGHES, DON'T WRITE TESTS!

- QuickCheck - randomised property testing
- For C++11: https://github.com/emil-e/rapidcheck

```
#include <rapidcheck.h>

#include <vector>
#include <algorithm>

int main() {
  rc::check("double reversal yields the original value",
            [](const std::vector<int> &l0) {
              auto l1 = l0;
              std::reverse(begin(l1), end(l1));
              std::reverse(begin(l1), end(l1));
              RC_ASSERT(l0 == l1);
            });

  return 0;
}
```

NIKO MATSAKIS, PUTTING OWNERSHIP TO USE

- “Rust is a systems programming language that runs blazingly fast, prevents segfaults, and guarantees thread safety. “
- Good tutorial of ownership, borrowing and RAII in Rust

TENSORFLOW

- Martin Abadi, Machine learning systems with privacy and for privacy: TensorFlow & PATE-G
- Compiled machine learning: Accelerated Linear Algebra (XLA) for TensorFlow

SYLVAN CLEBSCH, PONY LANGUAGE

- “Pony is an open-source, object-oriented, actor-model, capabilities-secure, high performance programming language.”
- Like Erlang, but native (uses LLVM)
- Capabilities are an interesting alternative to Rust’s ownership model

ROBERTO IERUSALIMSCHY, SCRIPTING WITH LUA

- Simplicity without hand-waving
- Constructing higher level features out of a small set of core features
- Pragmatic design
- Non partisan (OO vs FP)

CONSIDERABLE INTEREST IN THE CERN

- Data analysis
- Exploratory programming
- ROOT, Swan, Cling!
- Case studies

NEXT EDITION

- Curry On 2018, Amsterdam
- www.curry-on.org
- YouTube: https://www.youtube.com/channel/UC-WICcSW1k3HsScuXxDrp0w