Review

Amer Diwan

Topics by popular demand

• Midterm-2 continuation example
• Method invocations in various languages
The problem: exceptions without exceptions

- proc main() { f(); g(); }
  proc f() {
    try
      h();
      print "Normal return to f"
    except e => print "Just caught e in f"
    end
  }
  proc g() {
    try
      h();
      print "Normal return to g"
    except e => print "Just caught e in g"
    end
  }
  proc h() { raise e; }

Approaches

- Have an exception continuation which is thrown with the id of the exception
  - approach in solution

- Have one continuation for each exception and one for “normal”
  - approach in these slides
Transforming “h”

- `proc h() { raise e; }
- `proc h(nk, ek) { throw ek; }

Transforming “g” and “f”

- `proc g() {
  try
    h();
    print “Normal return to g”
  except e => print “Just caught e in g”
  end
}
- `proc g(nk, ek) {
  (callcc(fn gnk => callcc(fn gek =>
    h(gnk, gek));
    print “Just caught e in g”;
    throw nk)
  print “Normal return to g”;
  throw nk)
- f is almost the same as g...
Transforming main

- **proc** `main()` { `f()`; `g()`; }
- **fun** `main()` = (`callcc(fn k => f(k, k));
callcc(fn k => g(k, k));`
- To really complete this example, `main()` should create and pass a failure continuation in case except `e` is thrown.
  - Left as an exercise!

Method invocations in Smalltalk

![Diagram of method invocations in Smalltalk](image-url)
Method invocations in Smalltalk and Self

- Smalltalk:

- Self:
  - Look at slots in current object. If not found then delegate to parent

Method invocations in C++, Modula-3, Java

- Modula-3:
  - straightforward v-tables

- C++:
  - v-tables plus deltas

- Java:
  - v-table plus some hashing mechanism to pick the right v-table
Other questions?