Comp 311
Principles of Programming Languages
Lecture 20
Run-time Environment Representations II

Corky Cartwright

October 15, 2010

Review

- In Algol-like languages, the collection of environments that exist at any point during a computation is embedded in the machine *control stack* supporting (recursive) procedure calls.

- When the frames of the control stack are used in this way, they are called *activation records*.

- In each activation record, a pointer called the *static link* points to the environment parent of the record. Similarly, a pointer called the *dynamic link* points to the preceding stack frame (activation record).

- The static link is used for looking up bindings in the environment.

- The dynamic link is used to return control from the current procedure to its caller (whose local variables may not be accessible from the current frame).

Example I

Consider the following Scheme program to reverse a list:

```scheme
(define rev (lambda (l)
  (local
    [(define revhelp
      (lambda (tl acc)
        (if (empty? tl) acc
            (revhelp (rest tl) (cons (first tl) acc)))))]
    (revhelp l empty))))
```

What happens when

```scheme
(rev '(0 1 2))
```

is called?

Example II

Consider the following Scheme program to reverse a list:

```
(define lookup (lambda (sym env)
  (local
    [(define lookup-help
        (lambda (env)
          (cond [(empty? env) null]
                [(eq? sym (pair-var (first env))
                  (pair-val (first env))]
                [else (lookup-help (rest env) tl)])))]
    (lookup-help env))))
```

What happens when

```
(lookup 'a (cons (make-pair 'a 5) null))
```

is called?

Snapshots

On board.

Exceptions

- Exceptions were not included in Algol 60 or most of its successors (Pascal, Algol W, C). But the Algol 60 run-time stack can easily handle the Java **try/catch** construct.

- How does exception handling work? The activation record must include a **catch** table for the active **catch** (assuming one exists) listing the caught exception classes (types) and their handlers (the bodies of the **catch** clauses). (A **catch** is active if control is within the corresponding **try** block.)

- When an exception is thrown the executing code (interpreter or compiled code) searches back through the dynamic chain --- popping exited frames off the stack --- to find the first matching catch clause.