classmate

Date

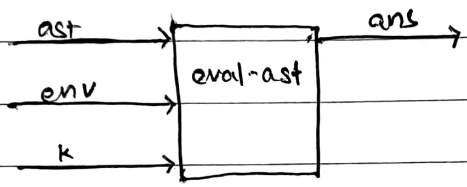
Page

CONTINUATION PASSING INTERPRETERS

e ::= n | b | (e e...) | (if e e e) | (fun (x...) e)
(assume ((x e)...) e) | x.

semantic domains

Value continuations:



answer :: ret (val) + err(str)

simple vs unbounded-time function:

```
(+ x 5)          (f x 5)
   ↓                ↓
(k (+ x 5))      (f' x 5 k)
```

```
(define eval-ast/k
  (λ (a env k)
    (cases ast a
      [num (n) (k n)]
      ...
      [id (x) (lookup-env/k env x k)]
```

```
[ifte (test then else)
  (eval-ast/k test env)
    (λ (v)
      (if (boolean? v)
          (eval-ask/k (if v then else) env k)
          (top-k* (format "~a" v))))

[assume (binds body)
  (eval-ast-binds/k binds env
    (λ (rib)

[fun (formals body)
  (k (make-closure formals body env)))

[app (rator rands)
  rator → rands.

e ::= ... | (abort e) | (break) | (resume v)
```

classmate

Date 14.11.2019

Page 1

CONTINUATION PASSING INTERPRETERS

① abort ③ try/catch ⑤ letcc

② break / resume ④ throw

```
> (+ 3 (abort 5))
  5
> (+ 3 (break 5)
                ↓
5
> (resume)
8
> (* resume * 7)
10
```

```
(define eval-ast/k
 (λ (a env k)
   [abort (a)
        (eval-ast/k (a) (env) *top-k*)]
   [break (a)
    (eval-ast/k (a) env
      (λ (v)
        (set! *resume*
          (match-lambda
            [() (k v)]
            [(list w) (k w)]
          )
        (*top-k* v)))]],
```

(resume)

(resume 4)

e ::= | (try e v e)
try block, id, catch.
| (throw e)

(define eval-ast/k
(λ (a env k ex-k)
[throw (a)
(eval-ast/k a env ex-k ex-k)
[try (a body ex-id handler)
(eval-ast/k body env k
(λ (exn)
(let ([new-env (extended-env
(list ex-id) (list exn))])
(eval-ast/k handler new-env k ex-k))))].

(eval-ast a *top-env* *top-k* *top-ex-k*).

(define *top-k*
(λ (v) v))

(define *top-ex-k*
(λ (v) "uncaught exception")))

coroutines, threads → continuation program.

letcc.

(letcc (k) e)

(+ 4 (letcc k (k 5)))

(+ 4 (letcc (k 2 (k 5))))

(define eval-ast/k

[letcc (sym body) extended-env
(let ([new-env (list sym)
(list (continuation-proc k)))

(eval-ast/k body new-env k ex-k)

(define apply-proc/k
(λ (p args k ex-k)
[primitive ...
[closure ...
[continuation-proc (k1)
~~(app-proc/k k1 args)~~
(apply k1 args)]]

nov 14-16 baker expo hitex

nov 10 maker shrine

aluminium x1

sardar patel. apj abdul kalam.