8-1)

$M, N ::= X \mid b \mid \lambda X.M \mid (o^n\ M \ldots) \mid M\ N$

$V ::= b \mid \lambda X.M$

β: $(\lambda X.M)\,V \rightarrow M[X \leftarrow V]$

Δ: $o^n\ b \ldots \rightarrow \delta(o^n, b, \ldots)$

$v = \beta_v \cup \Delta$

$$\frac{M \rightarrow N}{E[M] \rightarrow E[N]}$$

$E ::= \square \mid E\ N \mid V\ E \mid o^n\ V \ldots E\ M \ldots$

CC-machine $eval(m) = \langle m, \square \rangle \mapsto_{cc}^{*}$ (code → context)

$St = \langle M, E \rangle$

parsing
1. $\langle (M\ N), E \rangle \mapsto_{cc} \langle M, E[(\square\ N)] \rangle$
2. $\langle (V\ N), E \rangle \mapsto_{cc} \langle N, E[(V\ \square)] \rangle$

work happens
3. $\langle ((\lambda X.M)\ V), E \rangle \mapsto_{cc} \langle M[X \leftarrow V], E \rangle$

unparsing
6. $\langle V, E[(U\ \square)] \rangle \mapsto_{cc} \langle (U\ V), E \rangle$
7. $\langle V, E[(\square\ N)] \rangle \mapsto_{cc} \langle (V\ N), E \rangle$

$\langle ((\lambda X.((\lambda Y.Y)\ X))\ 7), \square \rangle$

~~$\mapsto_{cc}$ $\langle ((\lambda Y.Y)\ X)[X \leftarrow 7] \rangle$~~

$\langle (((\lambda X.(\lambda Y.Y))\ 7)\ ((\lambda X.8)\ 9)), \square \rangle$

1 $\langle (\lambda X.(\lambda Y.Y))\ 7, (\square\ ((\lambda X.8)\ 9)) \rangle$

3 $\langle (\lambda Y.Y), (\square\ ((\lambda X.8)\ 9)) \rangle$

7 $\langle (\lambda Y.Y)\ ((\lambda X.8)\ 9), \square \rangle$

2 $\langle ((\lambda X.8)\ 9), ((\lambda Y.Y)\ \square) \rangle$

3 $\langle 8, ((\lambda Y.Y)\ \square) \rangle$

6. $\langle (\lambda Y.Y)\ 8, \square \rangle$

3 $\langle 8, \square \rangle \Rightarrow 8$

8-2 SCC-machine (S = "simplified")

$St = \langle M, E \rangle$

1'. $\langle (M\ N), E \rangle \longmapsto_{scc} \langle M, E[(■\ N)] \rangle$

2' = 2+7. $\langle V, E[(■\ N)] \rangle \longmapsto_{scc} \langle N, E[(V\ ■)] \rangle$

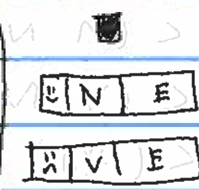
3' = 3+6. $\langle V, E[(\lambda X.M)\ ■] \rangle \longmapsto_{scc} \langle M[X \leftarrow V], E \rangle$

$=_v$ $\rightarrow_v$ $\longmapsto_v$ $\longmapsto_{cc}$ $\longmapsto_{scc}$ $\longmapsto_{ck}$

1	3	7	2	3	6	3	= cc
1'	1'	3'	2'	3'			= scc

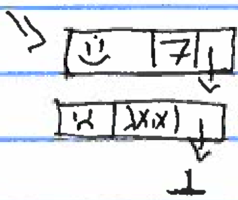
E[M] = search inside E for a leaf and replace with M

E = ■ — new Hole()
| (E N) — new Fun(Ctxt E, Term N)
| (V E) — new Arg(Term V, Ctxt E)



((λX.X) (■ 7))

||



Hole → top
Fun(M,E) → [E] → [☺|N] → was ~~...~~ inside
Arg(V,E) → E → ~~[☹|V]~~

CK-machine $St = \langle M, K \rangle$ K = end
eval(M) = ⟨M, end⟩ | fn(N, K)
| ar(V, K)
| pr(oⁿ, ⟨V...⟩, ⟨N...⟩, K)

1. $\langle (M\ N), K \rangle \longmapsto_{ck} \langle M, fn(N,K) \rangle$
2. $\langle V, fn(N,K) \rangle \longmapsto_{ck} \langle N, ar(V,K) \rangle$
3. $\langle V, ar(\lambda X.M, K) \rangle \longmapsto_{ck} \langle M[X \leftarrow V], K \rangle$

frame(K)

4. $\langle o^n\ M\ N \ldots, K \rangle \longmapsto_{ck} \langle M, pr(o^n, \langle\rangle, \langle N \ldots \rangle, K) \rangle$
5. $\langle V, pr(o^n, \langle U \ldots \rangle, \langle M\ N \ldots \rangle, K) \rangle \longmapsto_{ck} \langle M, pr(o^n, \langle U \ldots V \rangle, \langle N \ldots \rangle, K) \rangle$
6. $\langle b_n, pr(o^n, \langle b_1 \ldots b_{n-1} \rangle, \langle\rangle, K) \rangle \longmapsto_{ck} \langle \delta(o^n, b_1 \ldots b_n), K \rangle$

3-3

$\langle (((\lambda X.(\lambda Y.Y))\ 7)\ ((\lambda X.8)\ 9)) , end \rangle$

1 $\langle ((\lambda X.(\lambda Y.Y))\ 7) , fn((\lambda X.8)\ 9), end) \rangle$

1 $\langle (\lambda X.(\lambda Y.Y)) , fn(7, \square, \square) \rangle$

2. $\langle 7 , ar((\lambda X.(\lambda Y.Y)), \square) \rangle$

3. $\langle (\lambda X.(\lambda Y.Y))[X \leftarrow 7] = (\lambda Y.Y) , \square \rangle$

2. $\langle ((\lambda X.8), 9) , ar(\lambda Y.Y, end) \rangle$

1. $\langle (\lambda X.8) , fn(9, \square) \rangle$

2. $\langle 9 , ar(\lambda X.8, \square) \rangle$

3. $\langle \lambda X.8[X \leftarrow 9] = 8 , ar(\lambda Y.Y, end) \rangle$

2 $\langle \lambda Y.Y[Y \leftarrow 8] = 8 , end \rangle \Longrightarrow 8$

What is the k-part? $k = end$

context - backwards | $fn(N, k)$

or, context as a stack | $ar(V, k)$

```
30: int f () {
34:   g()
38:   h()
40:   ret 1; }
```

30: → place to go to (A)

34:	38	40	A
A	A	A	
38	40	jmp A	
g	h		

"ret" in asm = decrement stack and jump to stack+1

```
int f (int x) {
  int y = 1;
  return z(y)); }
```

$\langle \Omega , k \rangle \mapsto_k^* \langle \Omega , k \rangle \mapsto_{ck}^* \langle \Omega , f(k) \rangle$

```
main ...
  f (10);
```

"proper" tail-calling

"tail-call optimization"