CS 185, Lecture Notes, Continuations.

To define the semantics of continuations, a small-step evaluation relation is useful:

\[ C ::= \ast | C \; e | v \; C | C \; op \; e | v \; op \; C | if \; C \; then \; e_1 \; else \; e_2 | (C, e) | (v, C) | C.1 | C.2 \]

n We have small step rules like:

\[ C[(\lambda x. e) \; v] \sim C[[v/x]e] \]
\[ C[if \; true \; then \; e_1 \; else \; e_2] \sim C[e_1] \]
\[ C[(v_1, v_2).1] \sim v_1 \]

For continuations, we add to our contexts:

\[ C ::= \ldots | callcc \; C | throw \; C \; e | throw \; v \; C \]

And we then take these rules:

\[ C[callcc \; v] \sim C[v \; (\lambda x. C[x])] \]
\[ C[throw \; v \; v'] \sim (v \; v') \]

For example:

1. \( 5 + callcc \; (\lambda k. (2 + throw \; k \; (3 * 4))) \sim \)
2. \( 5 + ((\lambda k. (2 + throw \; k \; (3 * 4))) \; \lambda x. 5 + x) \sim \)
3. \( 5 + (2 + throw \; (\lambda x. 5 + x) \; (3 * 4)) \sim \)
4. \( 5 + (2 + throw \; (\lambda x. 5 + x) \; 12) \sim \)
5. \( (\lambda x. 5 + x) \; 12 \sim \)
6. \( 5 + 12 \sim \)
7. \( 17 \)

Notice that in the transition from (4) to (5), we throw away the surrounding context \( 5 + (2 + \ast) \). This is due to the small-step rule for \( throw \).