Adding Delimited and Composable Control to a Production Programming Environment

Matthew Flatt University of Utah

Gang Yu Institute of Software, Chinese Academy of Sciences

Robert Bruce Findler University of Chicago

Matthias Felleisen Northeastern University

Web Servlet with Continuations

```
(define (paper-search-servlet)
  (let ([terms (get-search-terms)])
    (find-paper terms)))
```

Web Servlet with Continuations

send back HTML form, then
wait for answer as new request

```
(define (paper-search-servlet)
  (let ([terms (get-search-terms)])
    (find-paper terms)))
```

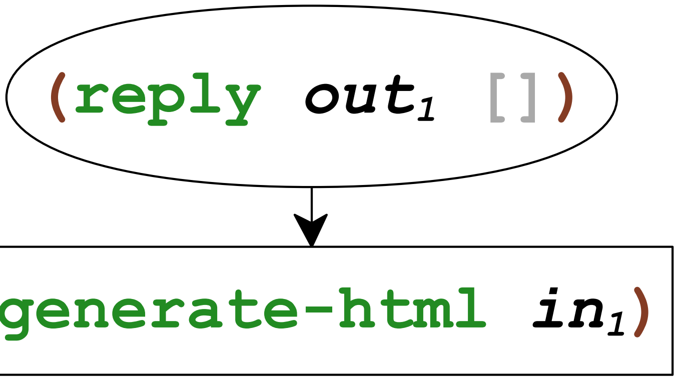
Implementing a Web Server

```
(serve out₁ in₁)
```

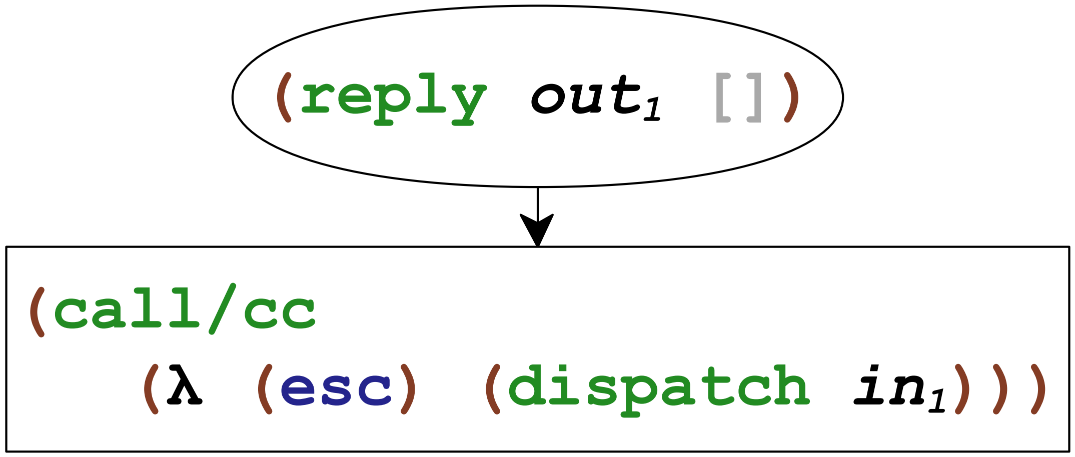
Implementing a Web Server

```
(reply out₁ (generate-html in₁))
```

Implementing a Web Server



Implementing a Web Server

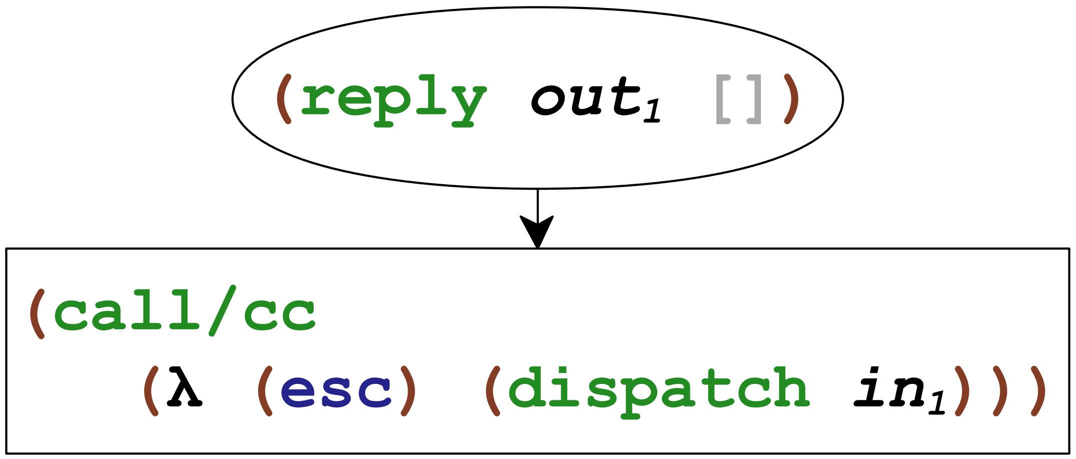


```
(reply out₁ [])
```

```
(call/cc
    (λ (esc) (dispatch in₁)))
```

Implementing a Web Server

```
esc = (reply out₁ [])
```

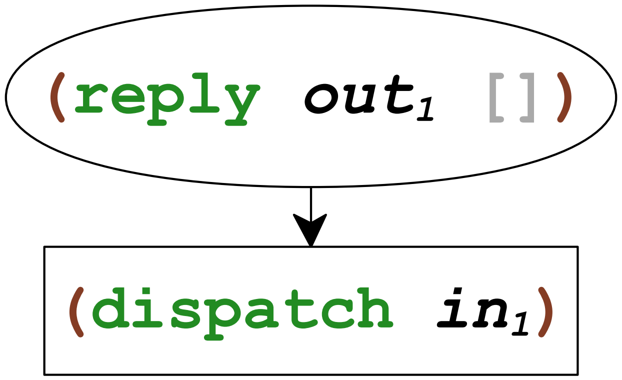


```
(reply out₁ [])
```

```
(call/cc
   (λ (esc) (dispatch in₁)))
```

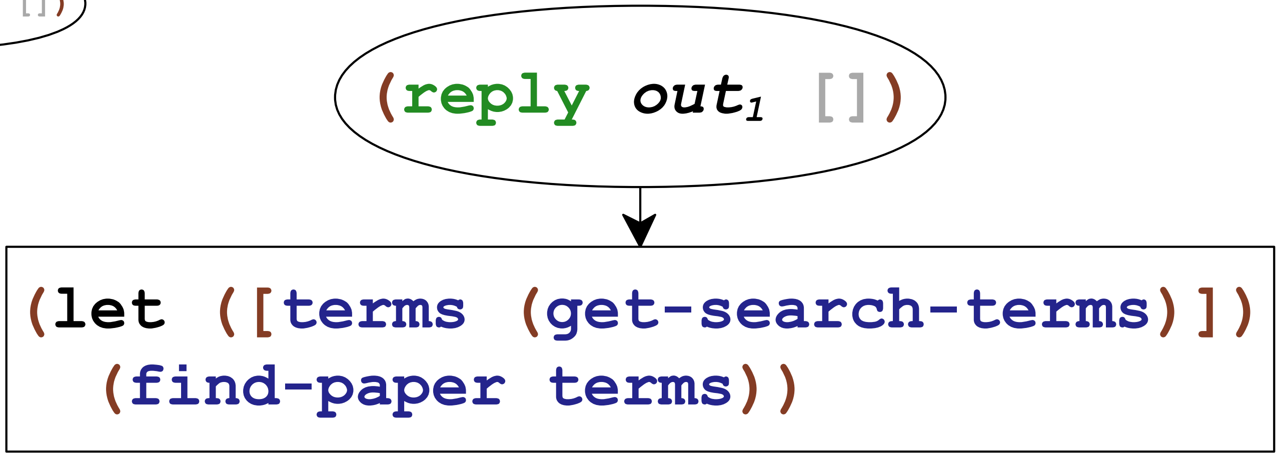
Implementing a Web Server

`esc =` `(reply out₁ [])`



Implementing a Web Server

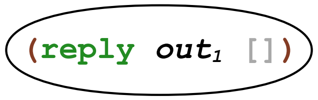
esc = (reply out_1 [])

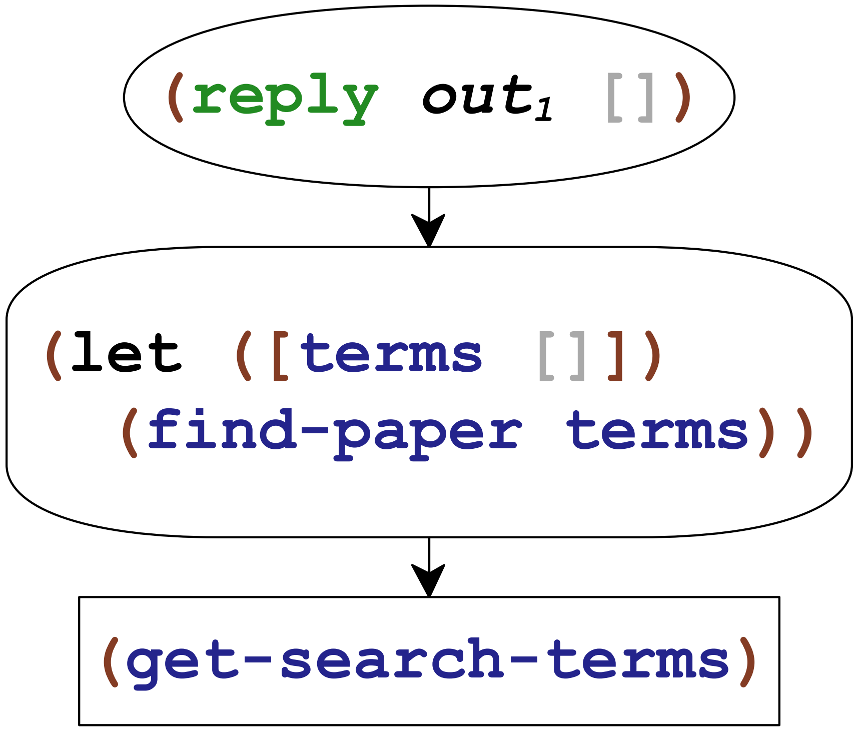


(reply out_1 [])

```
(let ([terms (get-search-terms)])
  (find-paper terms))
```

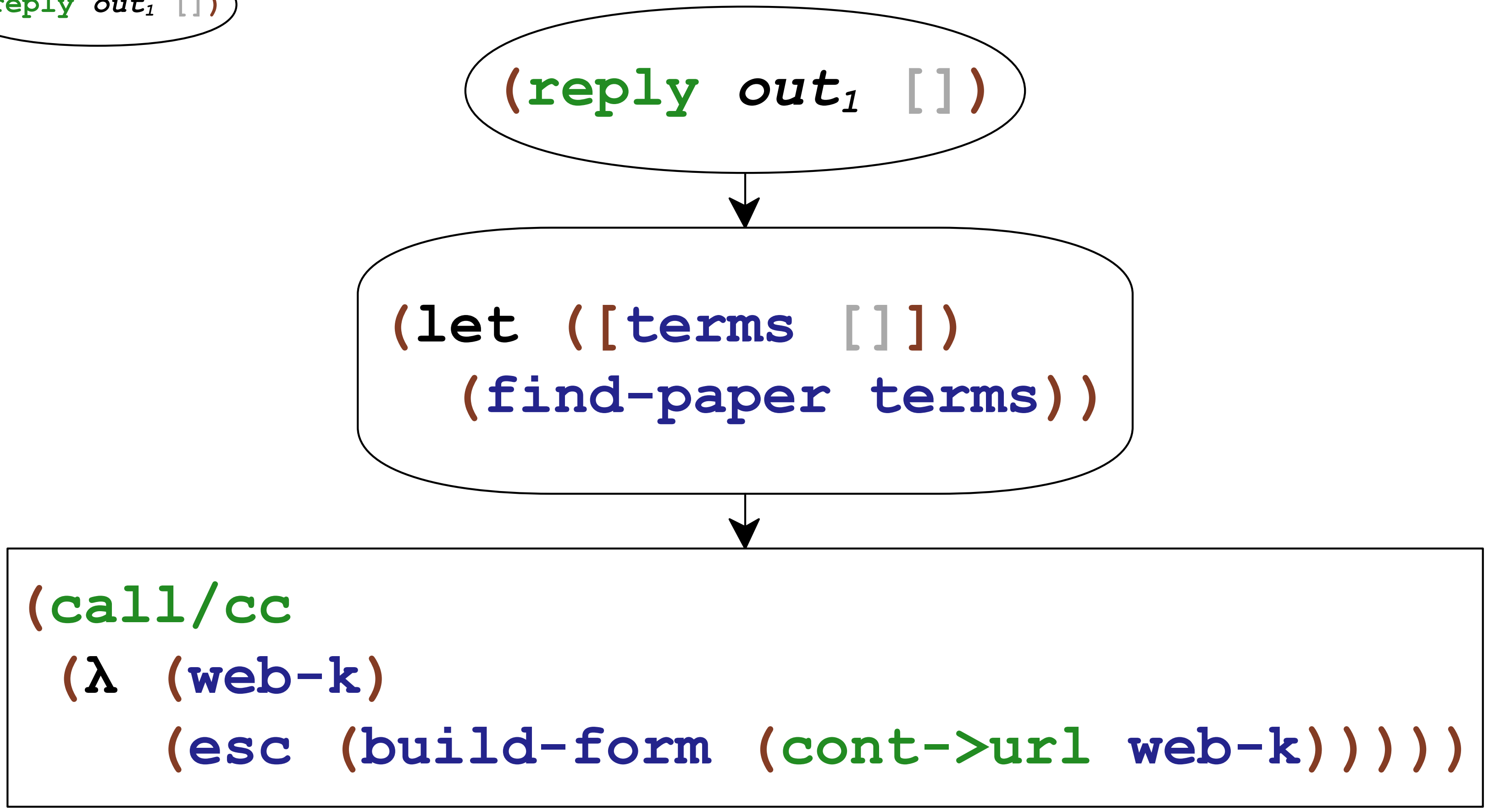
Implementing a Web Server

`esc =` 



Implementing a Web Server

```
esc = (reply out₁ [])
```

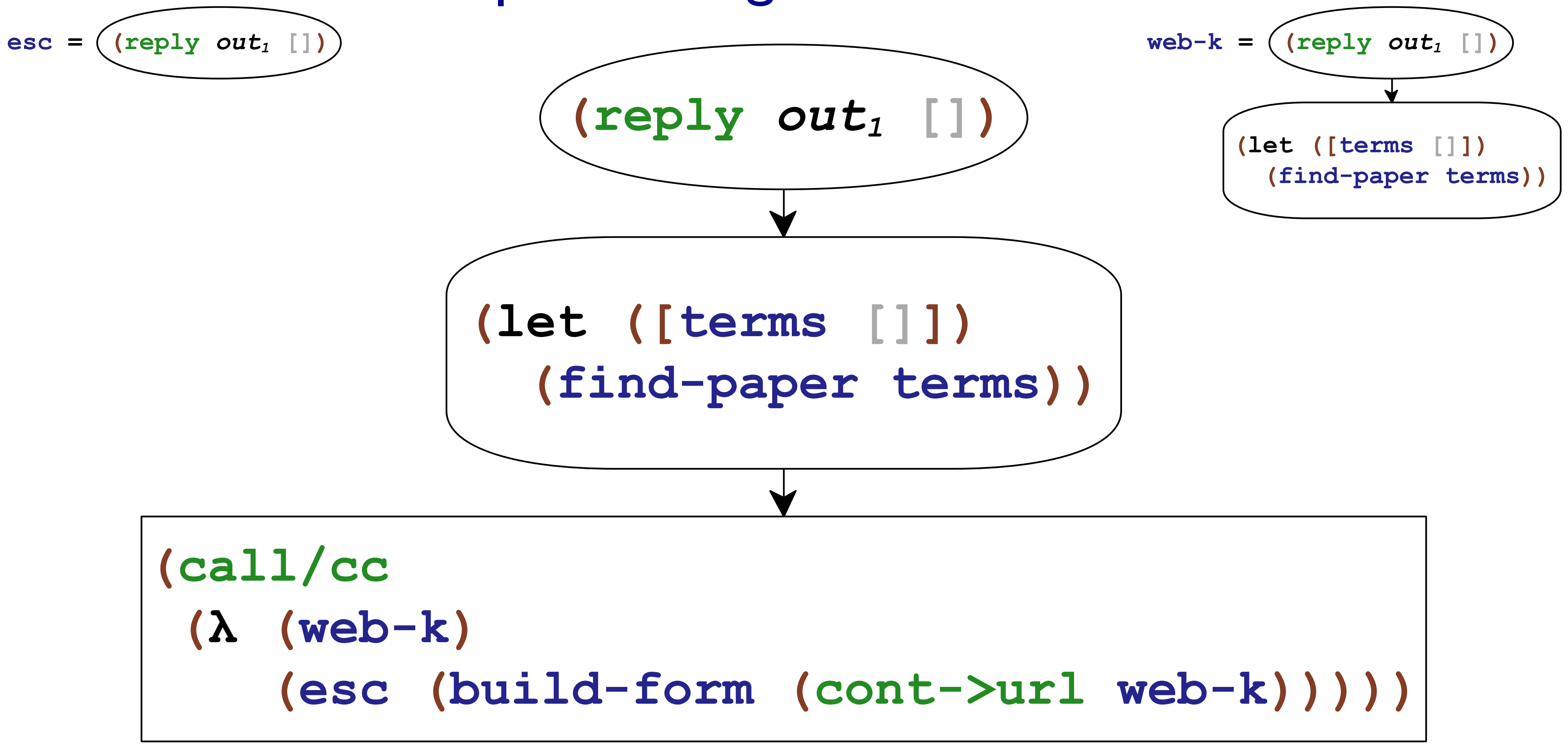


```
(reply out₁ [])
```

```
(let ([terms []])
  (find-paper terms))
```

```
(call/cc
 (λ (web-k)
   (esc (build-form (cont->url web-k)))))
```

Implementing a Web Server



```
esc = (reply out_1 [])
```

```
web-k = (reply out_1 [])
          ↓
        (let ([terms []])
          (find-paper terms))
```

```
(reply out_1 [])
```

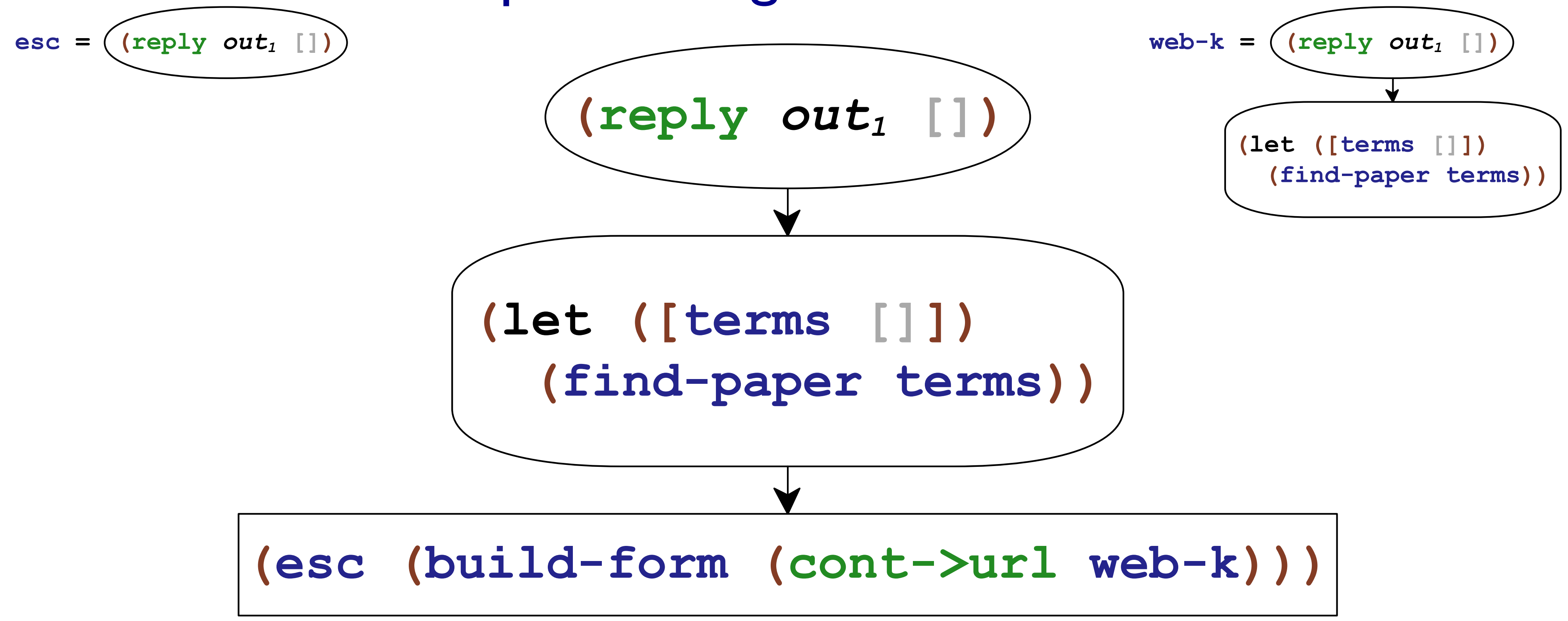
↓

```
(let ([terms []])
  (find-paper terms))
```

↓

```
(call/cc
 (λ (web-k)
   (esc (build-form (cont->url web-k)))))
```

Implementing a Web Server



```
esc = (reply out₁ [])
```

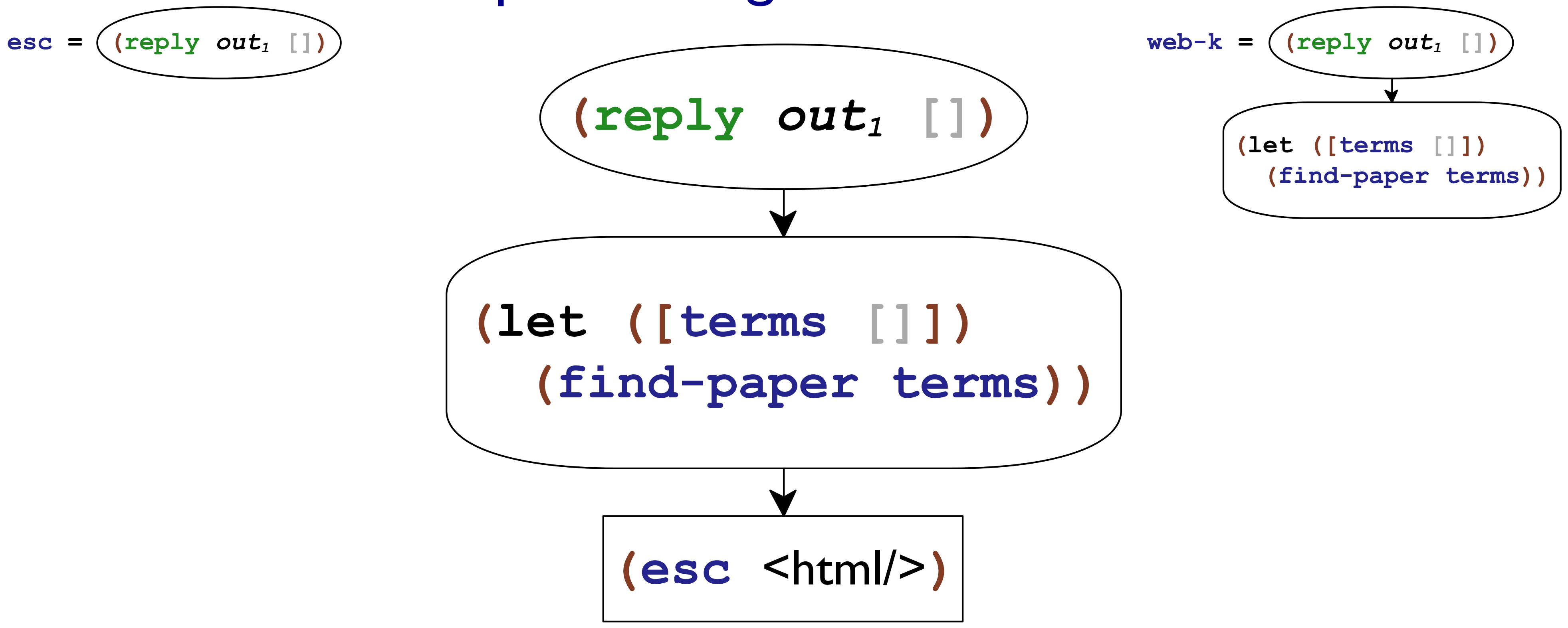
```
web-k = (reply out₁ [])
        (let ([terms []])
          (find-paper terms))
```

```
(reply out₁ [])
```

```
(let ([terms []])
  (find-paper terms))
```

```
(esc (build-form (cont->url web-k)))
```

Implementing a Web Server



Implementing a Web Server

esc = (reply out_1 [])

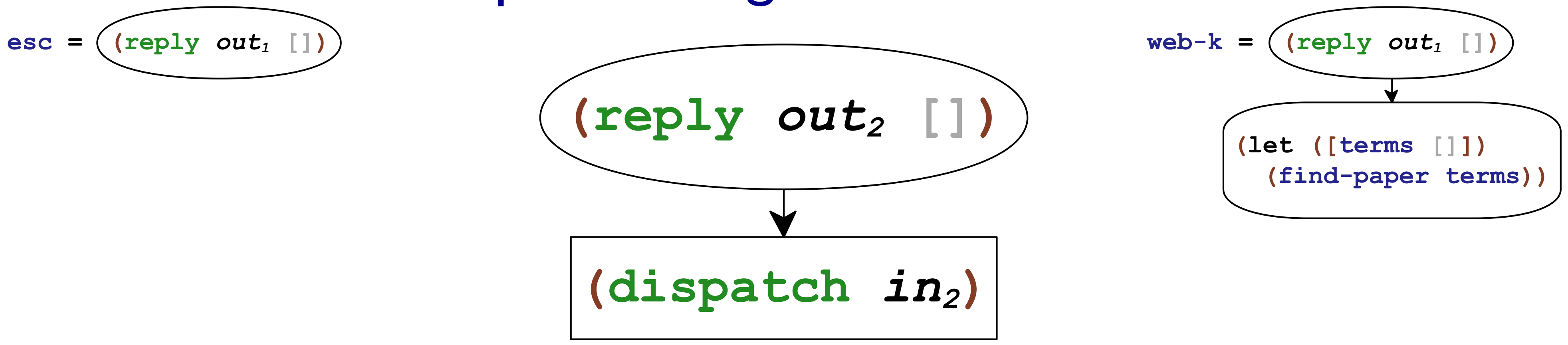
(reply out_1 <html/>)

web-k = (reply out_1 [])

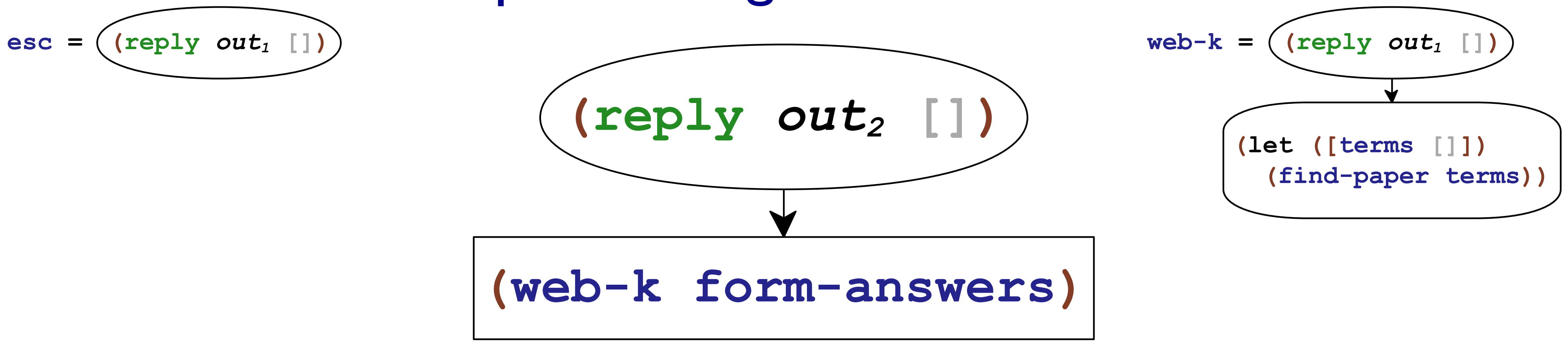
$\downarrow$

```
(let ([terms []])
  (find-paper terms))
```

Implementing a Web Server



Implementing a Web Server



```
esc = (reply out₁ [])
```

```
(reply out₂ [])
```

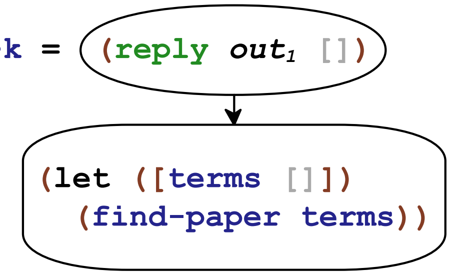
```
(web-k form-answers)
```

```
web-k = (reply out₁ [])
```

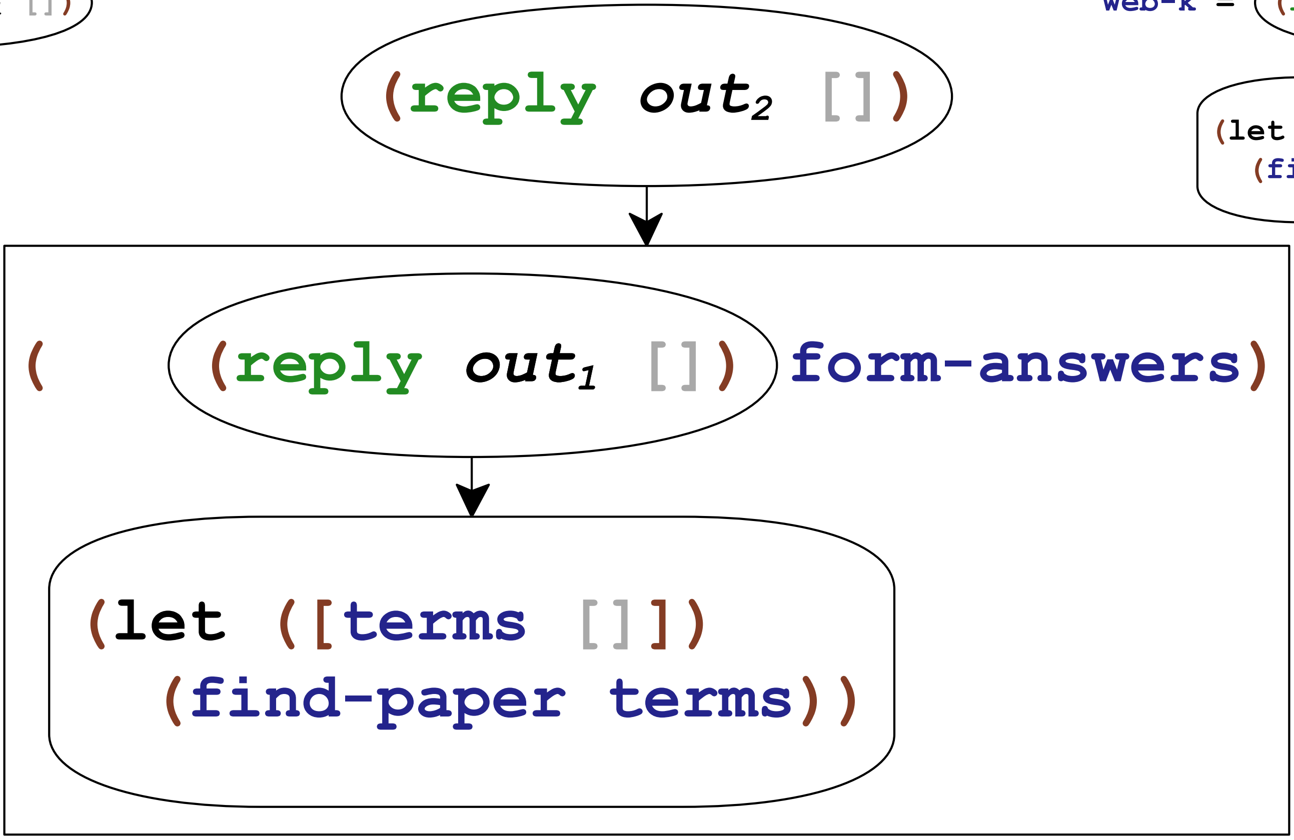
```
(let ([terms []])
  (find-paper terms))
```

Implementing a Web Server

```
esc = (reply out₁ [])
```



```
web-k = (reply out₁ [])

(let ([terms []])
  (find-paper terms))
```

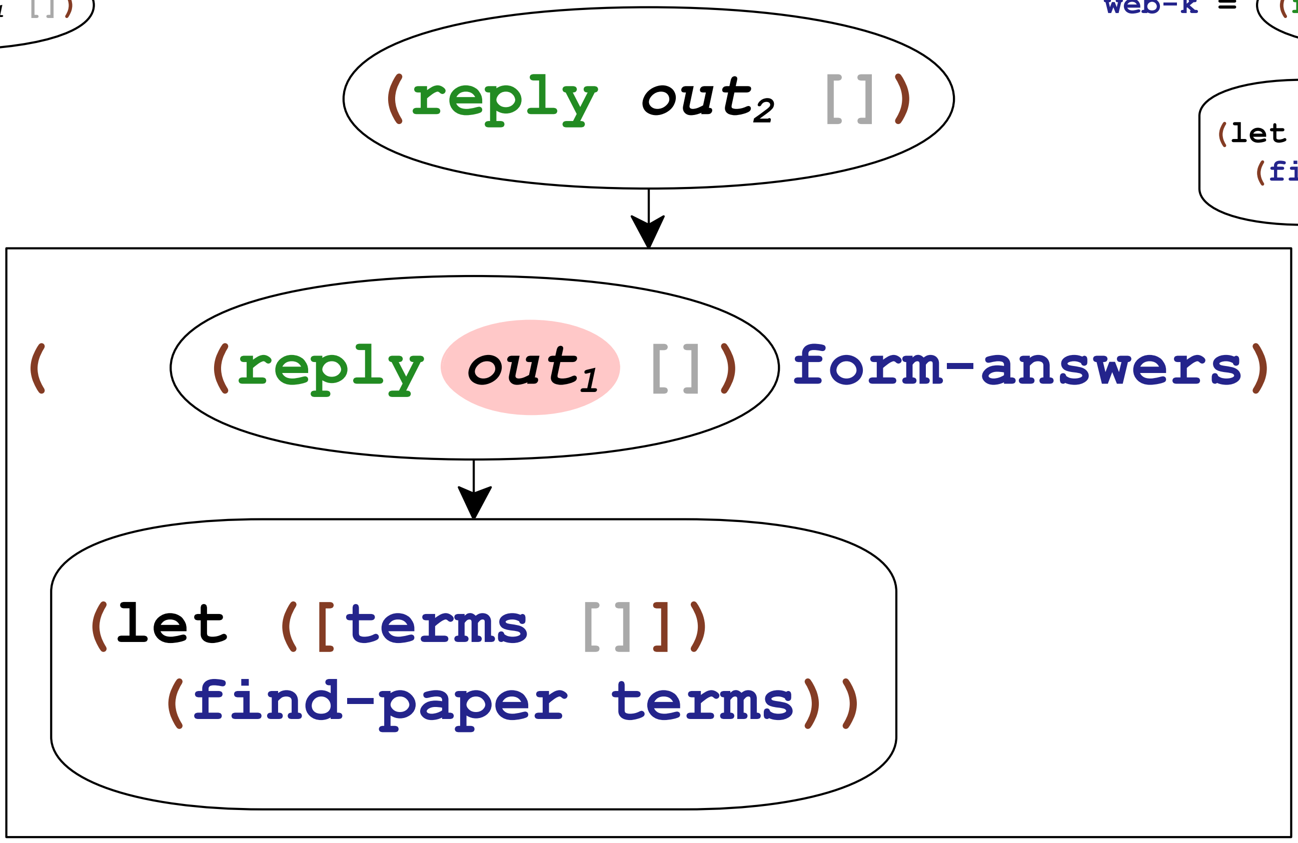


```
(reply out₂ [])

(    (reply out₁ []) form-answers)

(let ([terms []])
  (find-paper terms))
```

Implementing a Web Server

```
esc = (reply out₁ [])
```

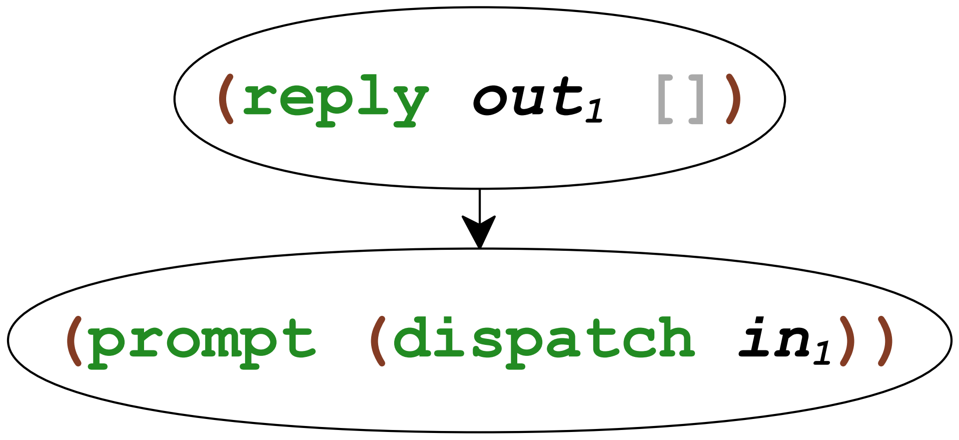
```
web-k = (reply out₁ [])
        (let ([terms []])
          (find-paper terms))
```



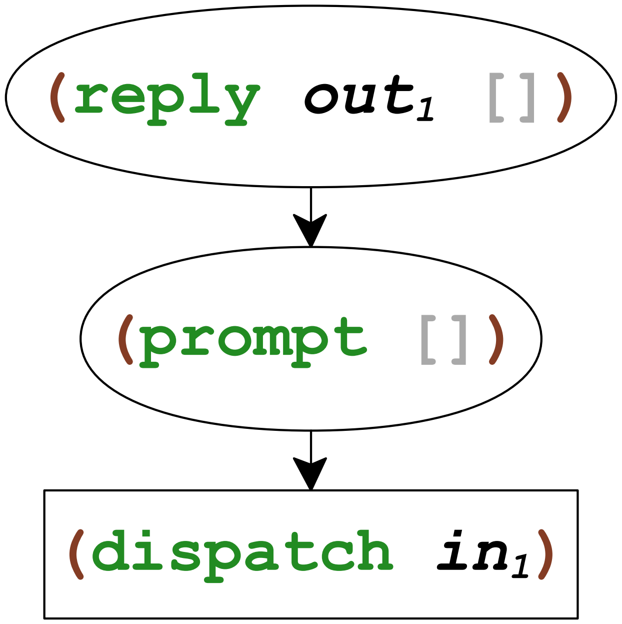
```
(reply out₂ [])
```

```
(    (reply out₁ []) form-answers)
     (let ([terms []])
       (find-paper terms))
```

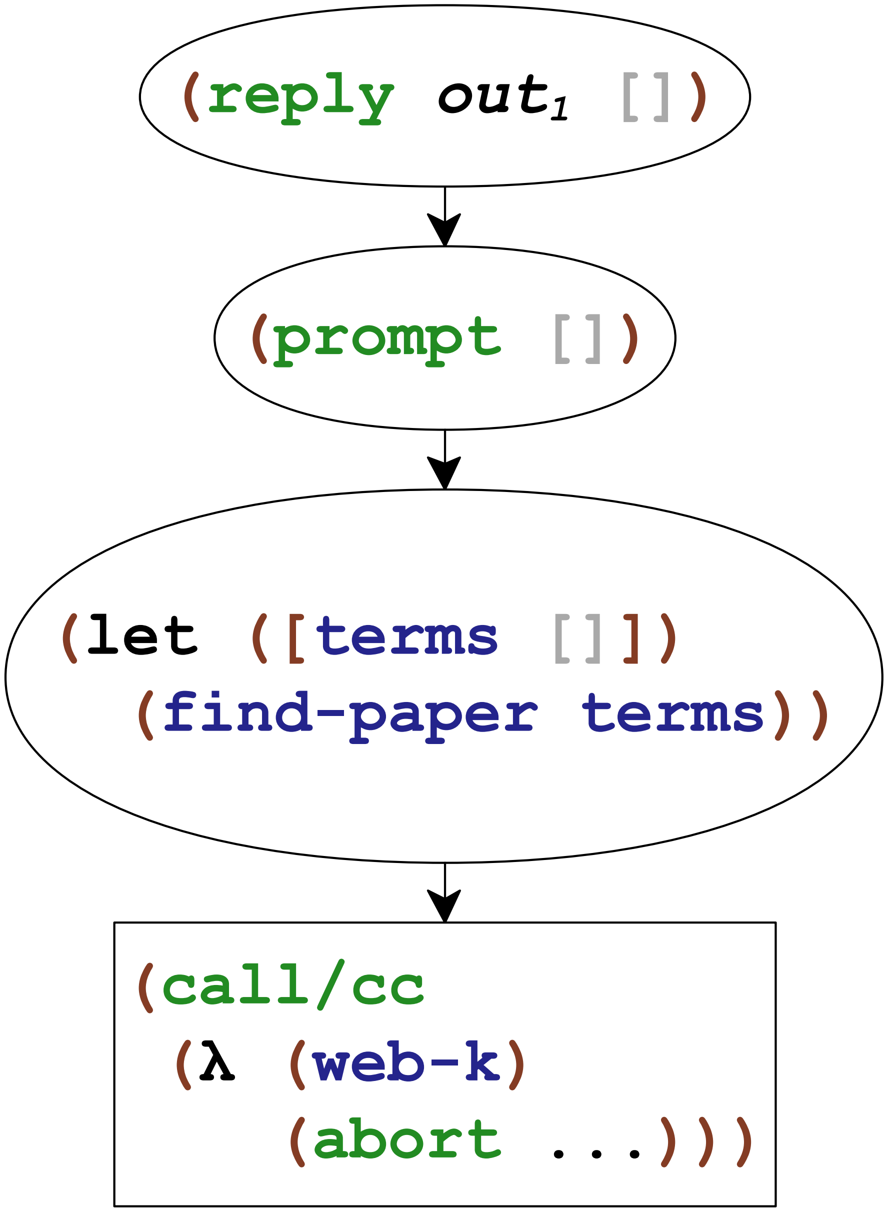
Web Server with Prompt



Web Server with Prompt



Web Server with Prompt



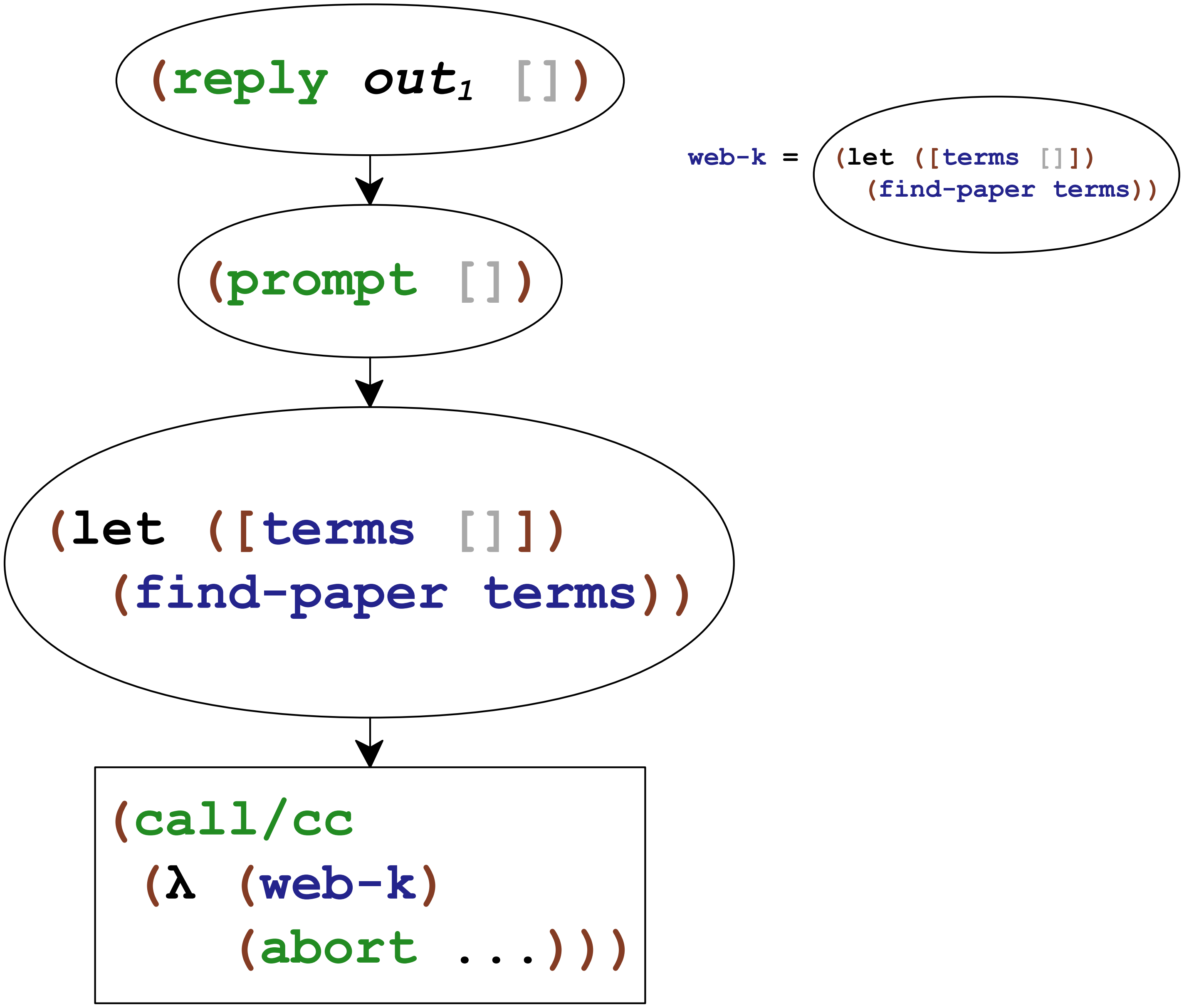
```
(reply out₁ [])
```

```
(prompt [])
```

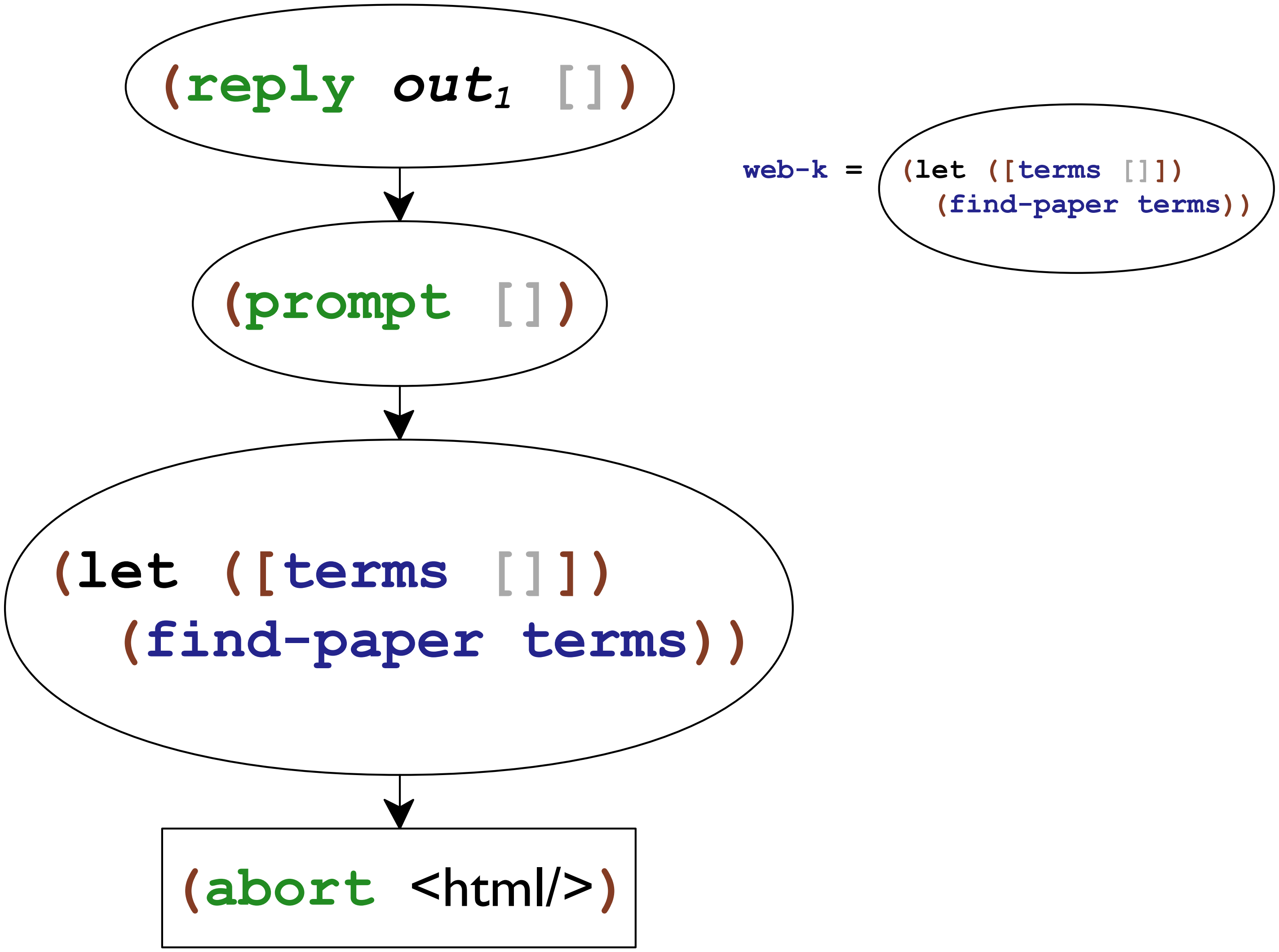
```
(let ([terms []])
  (find-paper terms))
```

```
(call/cc
 (λ (web-k)
    (abort ...)))
```

Web Server with Prompt



Web Server with Prompt



```
(reply out₁ [])
```

```
(prompt [])
```

```
(let ([terms []])
  (find-paper terms))
```

```
(abort <html/>)
```

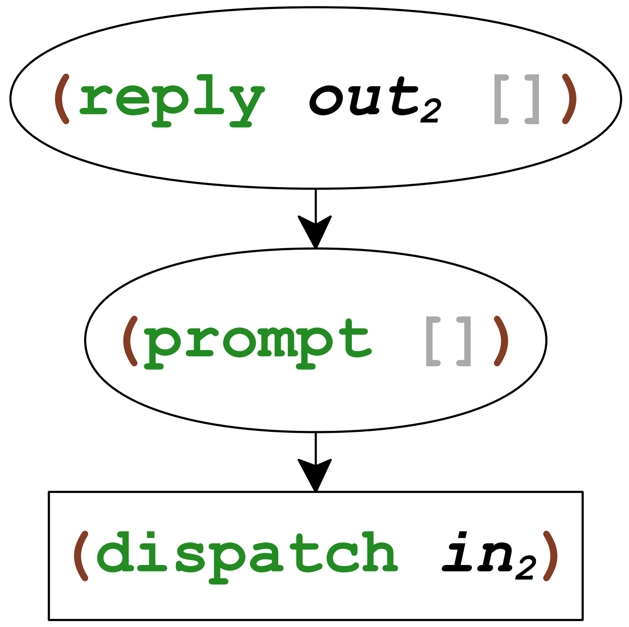
```
web-k = (let ([terms []])
          (find-paper terms))
```

Web Server with Prompt

```
(reply out₁ <html/>)
```

```
web-k = (let ([terms []])
          (find-paper terms))
```

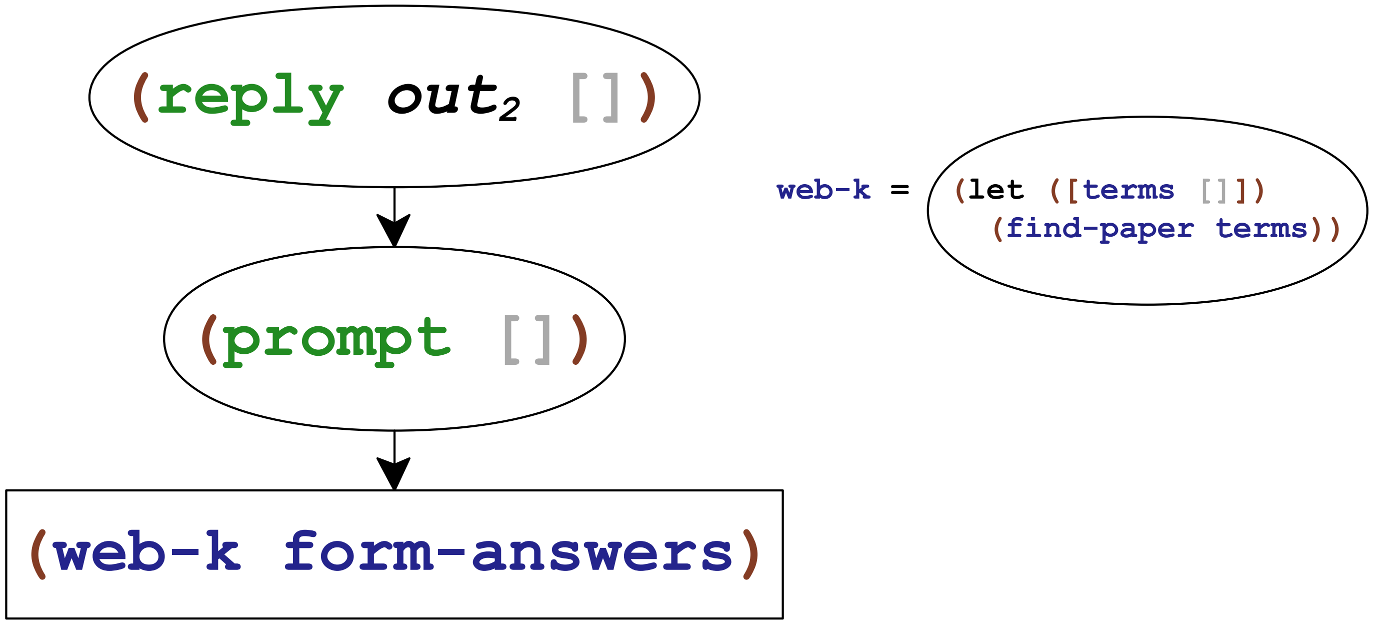
Web Server with Prompt



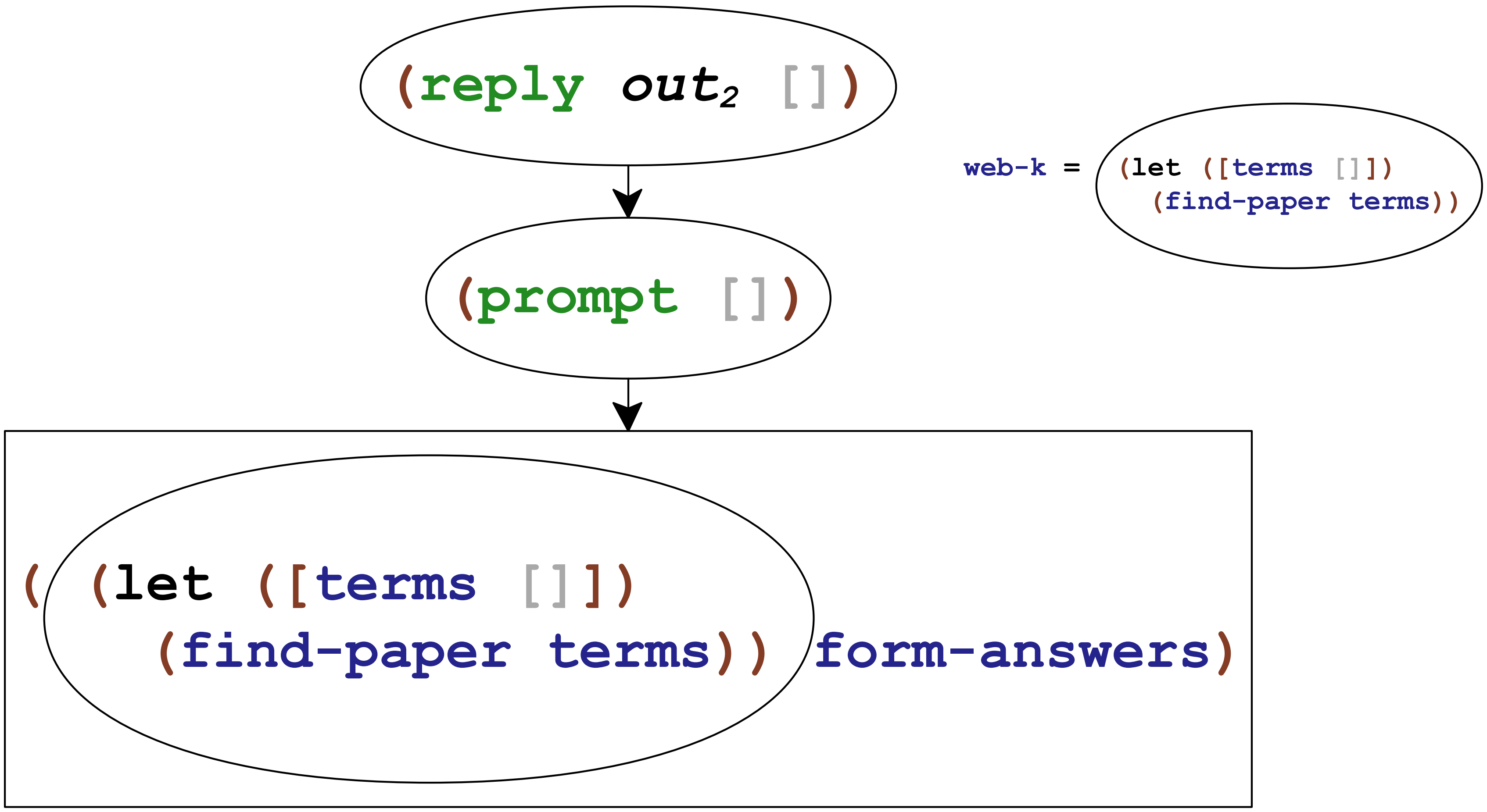
web-k =

```
(let ([terms []])
  (find-paper terms))
```

Web Server with Prompt

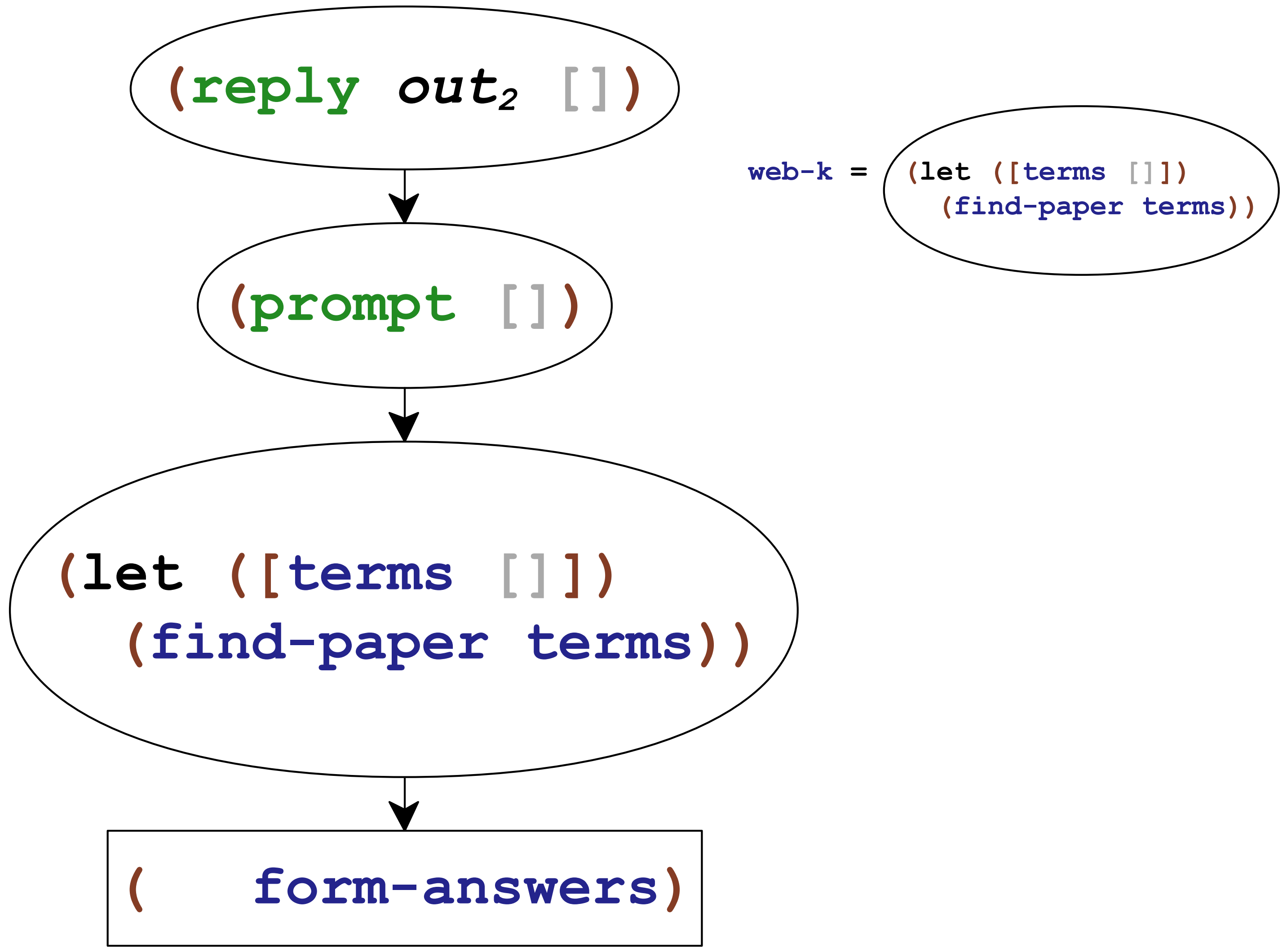


Web Server with Prompt

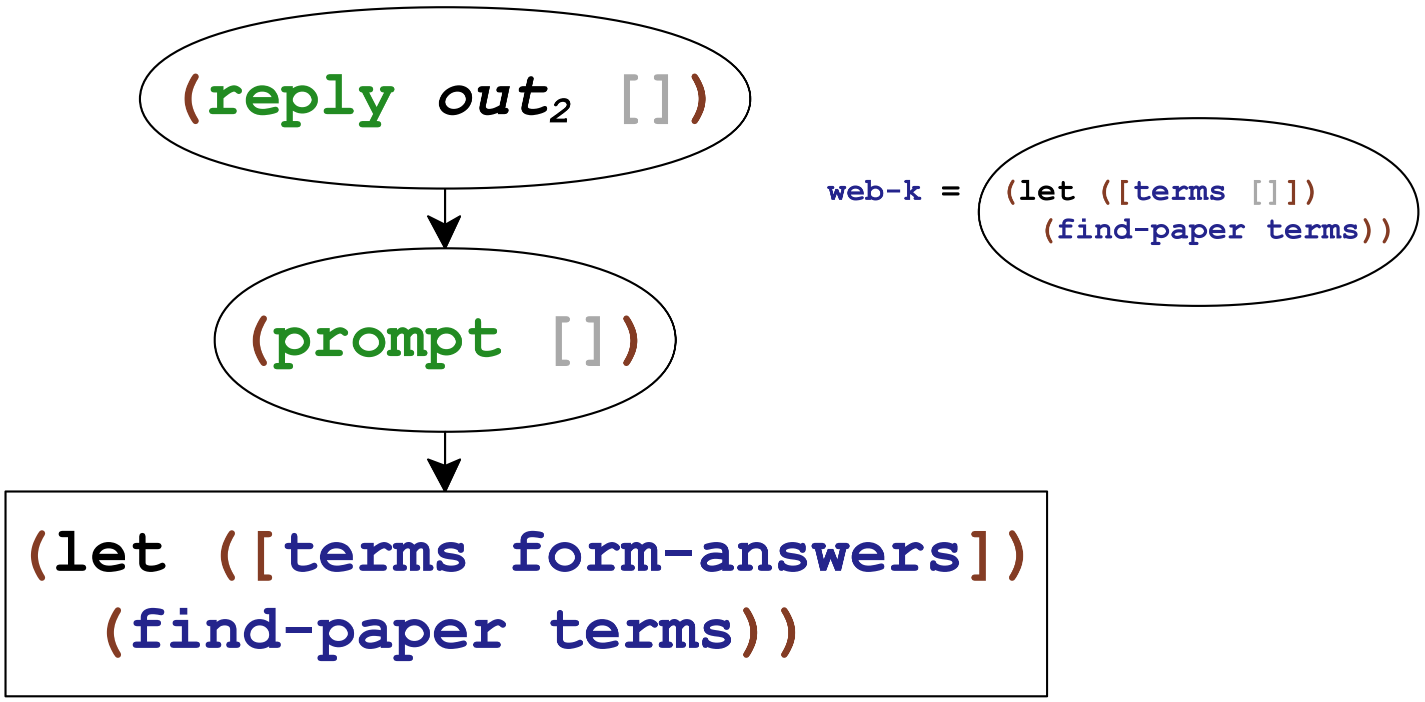


(reply *out*₂ [])
(prompt [])
((let ([terms []]) (find-paper terms)) form-answers)
web-k = (let ([terms []]) (find-paper terms))

Web Server with Prompt



Web Server with Prompt



(reply out_2 [])

(prompt [])

```
(let ([terms form-answers])
  (find-paper terms))
```

```
web-k = (let ([terms []])
          (find-paper terms))
```

Rolling Your Own vs. Language Extension

Delimited control via `call/cc` doesn't work right with

- exceptions
- dynamic binding
- `dynamic-wind`

Our goal: delimited control integrated with existing Scheme & Racket constructs

Papers on Delimited Continuations

Felleisen 88
Hieb and Dybvig 90
Queinnec and Serpette 91
Queinnec 93
Wadler 94
Rehof and Sørensen 94
Gunter et al. 95
Rehof 01
Kameyama and Hasegawa 03
Shan 04
Kiselyov et al. 06
Biernacki et al. 06

Danvy and Filinski 90
Sitaram and Felleisen 90
Sitaram 93
Moreau and Queinnec 94
deGroote 94
Gunter et al. 95
Thielecke 97
Gasbichler and Sperber 02
Ariola et al. 04
Saurin 05
Dybvig et al. 06

Papers on Design

Felleisen 88

Danvy and Filinski 90

Hieb and Dybvig 90

Sitaram and Felleisen 90

Queinnec and Serpette 91

Sitaram 93

Moreau and Queinnec 94

Gunter et al. 95

Kiselyov et al. 06

Dybvig et al. 06

Papers on Implementation

Gasbichler and Sperber 02
Dybvig et al. 06

Implementations (circa 2006)

Implementations (Now)

Balance

exceptions

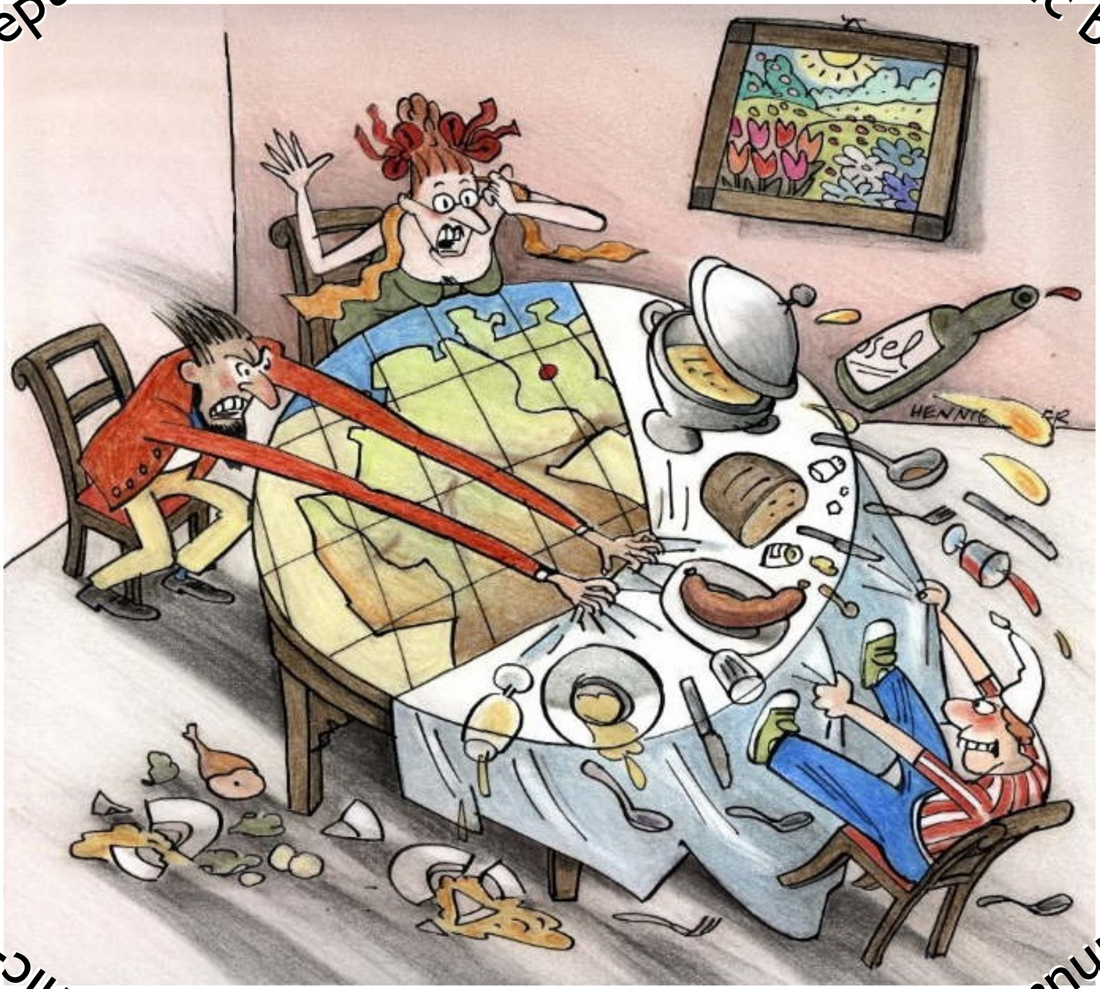
dynamic binding

prompts

abort

dynamic-wind

continuations



Contributions

- Comprehensive design
- Formal model
- Implementation

Contributions

- Comprehensive design

$$\Uparrow$$

graphical intuition

$$\Downarrow$$

- Formal model

- Implementation

Contributions

- Comprehensive design

$\Updownarrow$ *graphical intuition*

- Formal model

$\Updownarrow$ *random testing*

- Implementation

Notation

$(v_1\ (((\lambda\ (x)\ x)\ v_3)\ v_2))$

Notation

```
(v₁ (((λ (x) x) v₃) v₂))
```

Notation

```
(v₁ (((λ (x) x) v₃) v₂))
```

$$\boxed{\texttt{((}\lambda\ \texttt{(x) x)}\ v_3\texttt{)}}$$

Notation

`(v`$_1$` (((λ (x) x) v`$_3$`) v`$_2$`))`

$$\boxed{\texttt{((λ (x) x) } v_3\texttt{)}}$$

Notation

$(v_1\ (((\lambda\ (x)\ x)\ v_3)\ v_2))$

$([]\ v_2)$

$\boxed{((\lambda\ (x)\ x)\ v_3)}$

Notation

`(`v_1` (((λ (x) x) `v_3`) `v_2`))`

$\bigcirc$ `([] `v_2`)`

$\boxed{\texttt{((λ (x) x) } v_3\texttt{)}}$

Notation

$(v_1\ (((\lambda\ (x)\ x)\ v_3)\ v_2))$

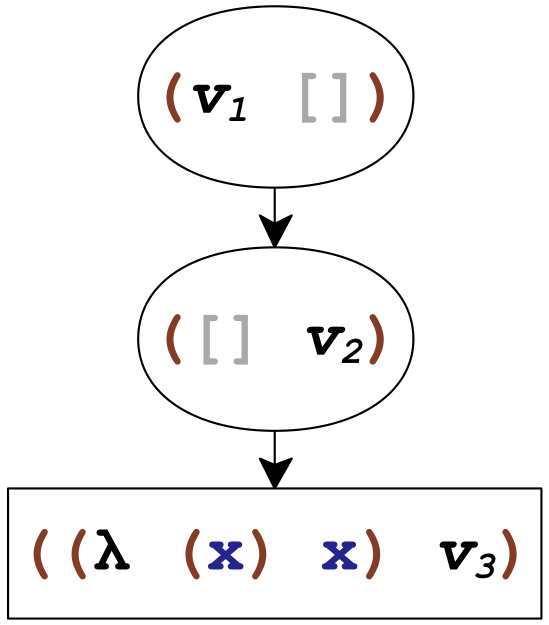
$(v_1\ [\,])$

$([\,]\ v_2)$

$\boxed{((\lambda\ (x)\ x)\ v_3)}$

Notation

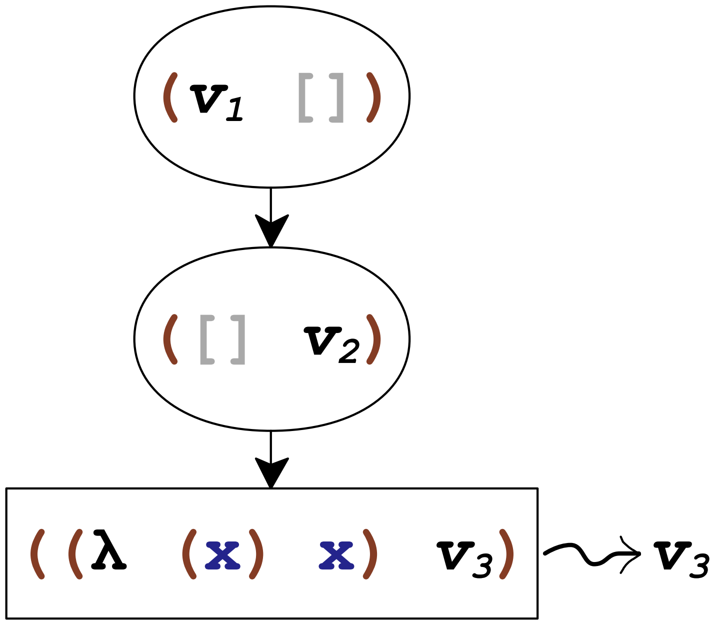
$$(v_1\ (((\lambda\ (x)\ x)\ v_3)\ v_2))\ =$$



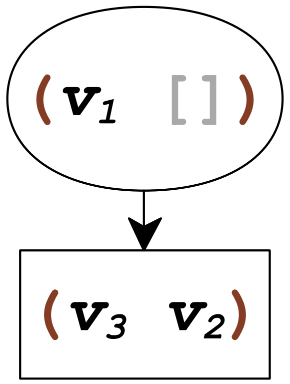
Reductions

$$\begin{tikzcd}
(v_1\ [\,]) \arrow[d] \\
([\,]\ v_2) \arrow[d] \\
\boxed{((\lambda\ (x)\ x)\ v_3)}
\end{tikzcd}$$

Reductions



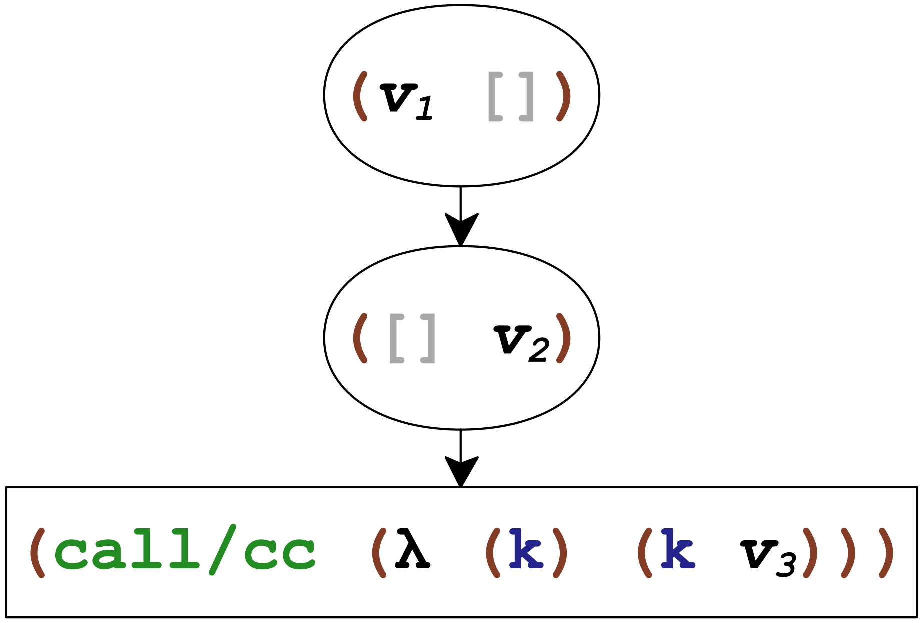
Reductions



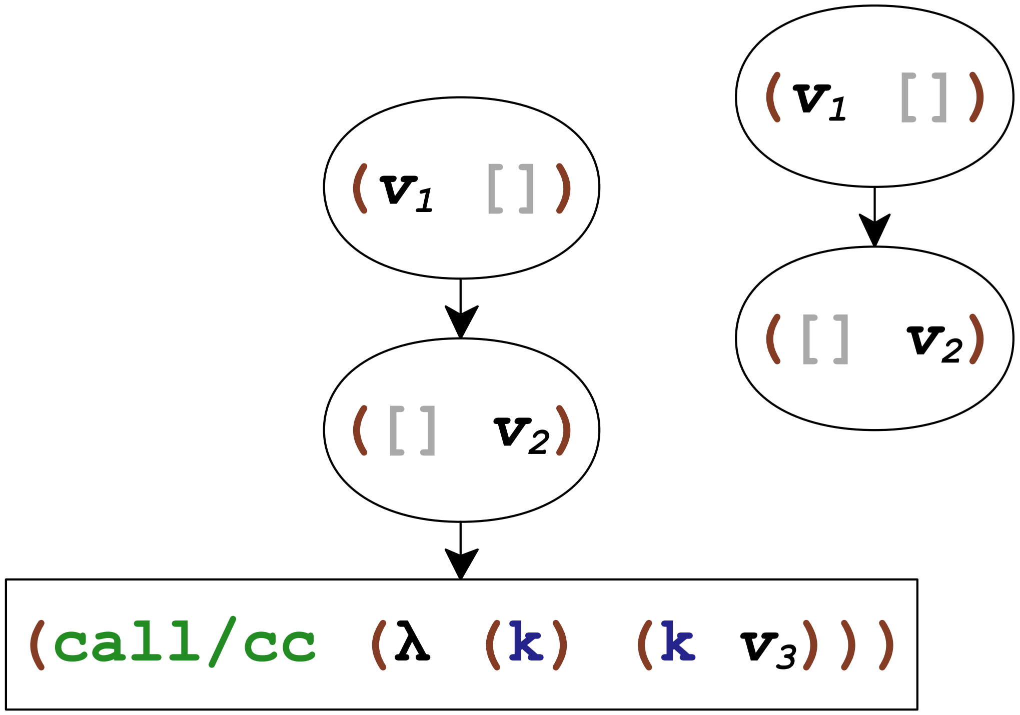
Reductions

$$\begin{tikzcd}
(v_1\ [\,]) \arrow[d] \\
([\,]\ v_2) \arrow[d] \\
\boxed{((\lambda\ (x)\ x)\ v_3)}
\end{tikzcd}$$

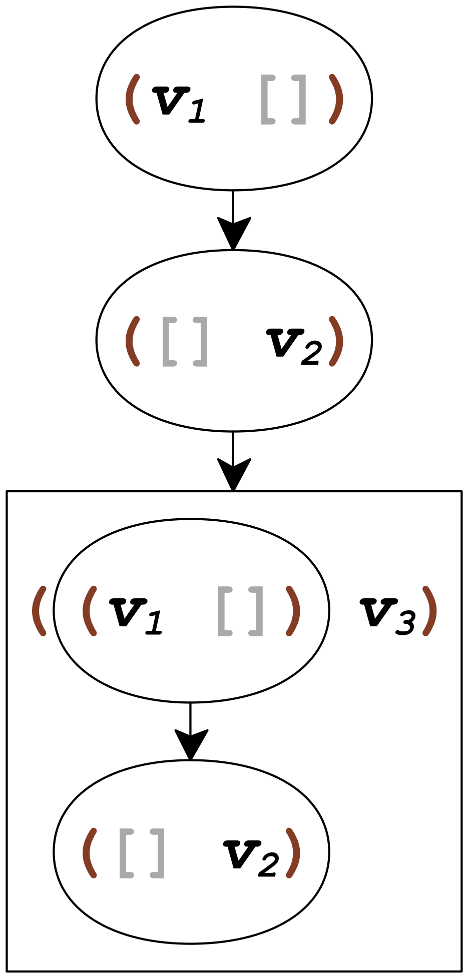
Continuations



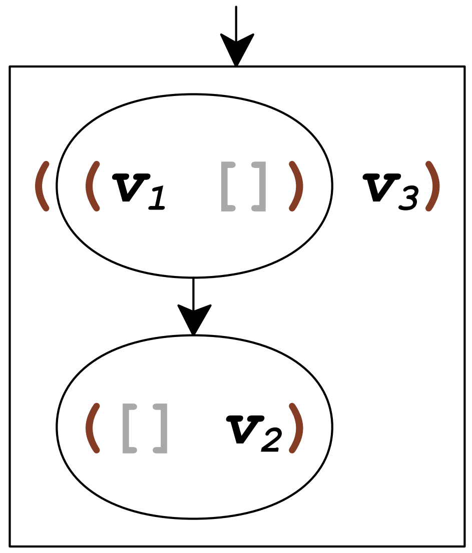
Continuations



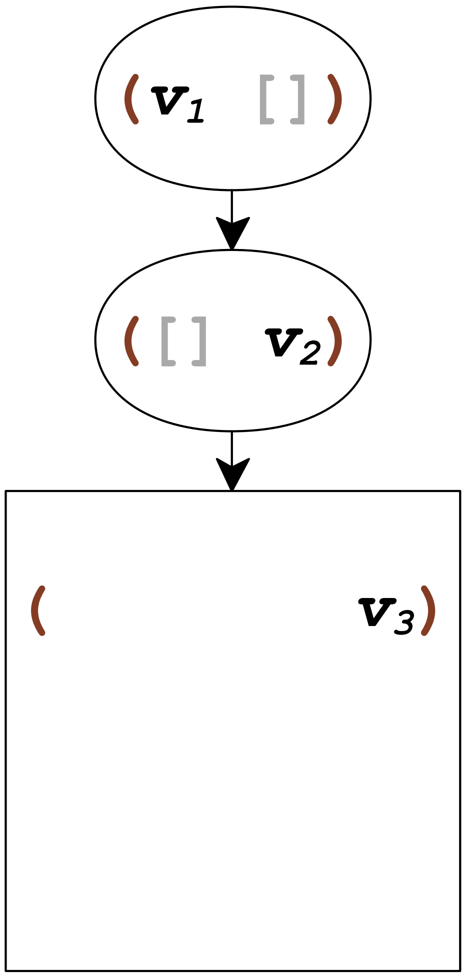
Continuations



Continuations



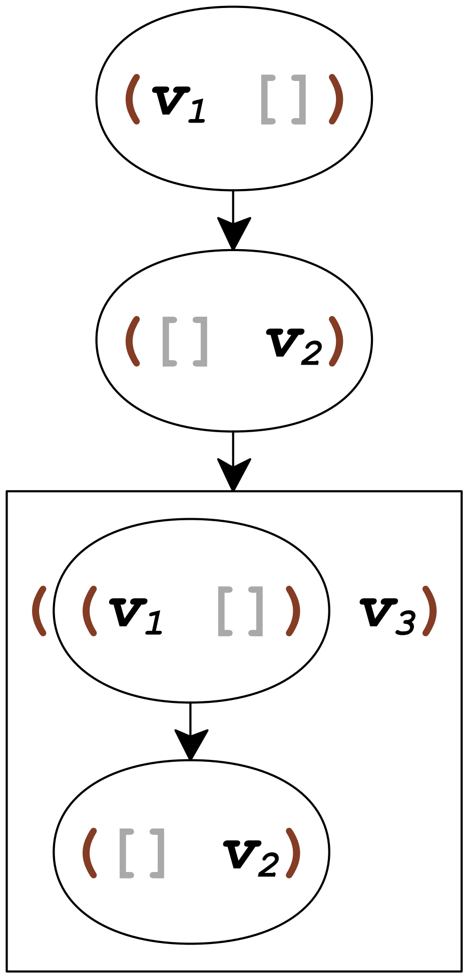
Continuations



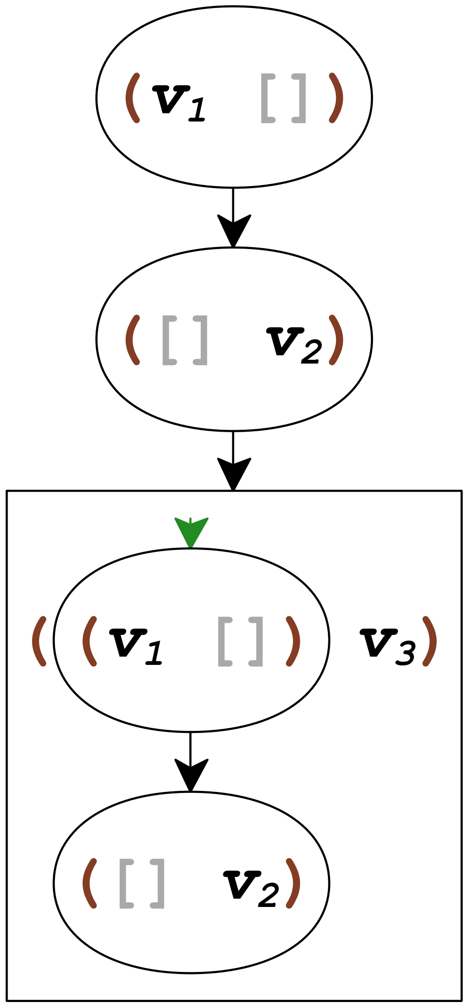
Continuations

$$\begin{tikzcd}
(v_1\ \ [\,]) \arrow[d] \\
\boxed{(v_3\ \ v_2)}
\end{tikzcd}$$

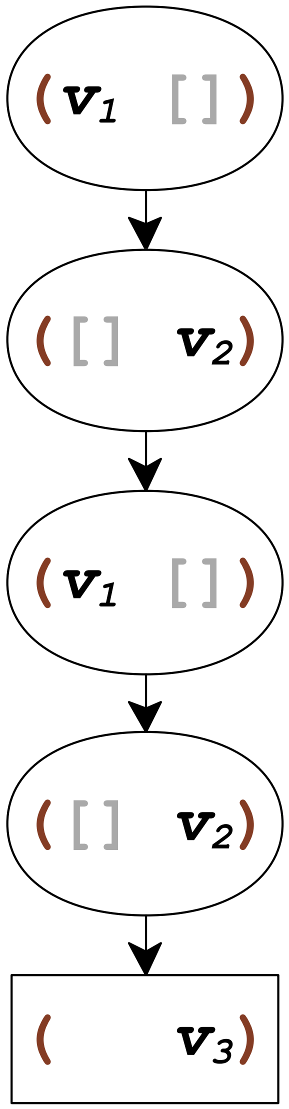
Continuations



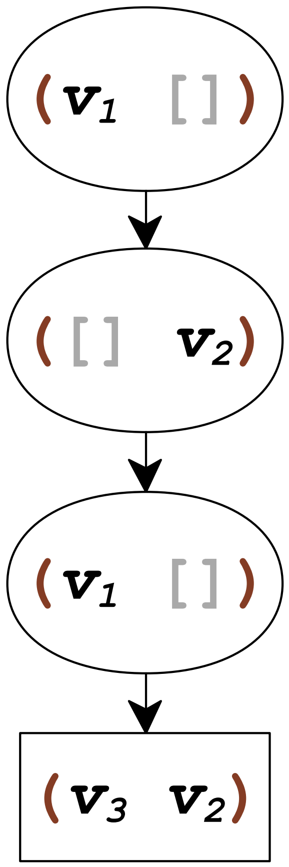
Composable Continuations



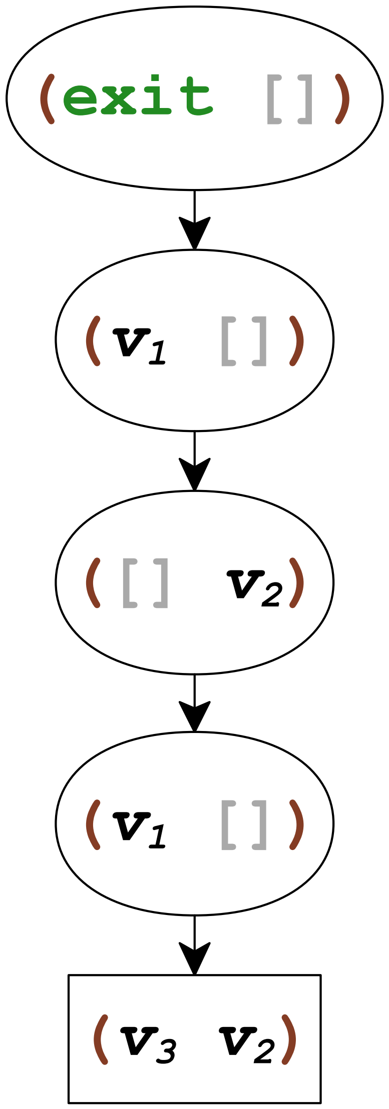
Composable Continuations



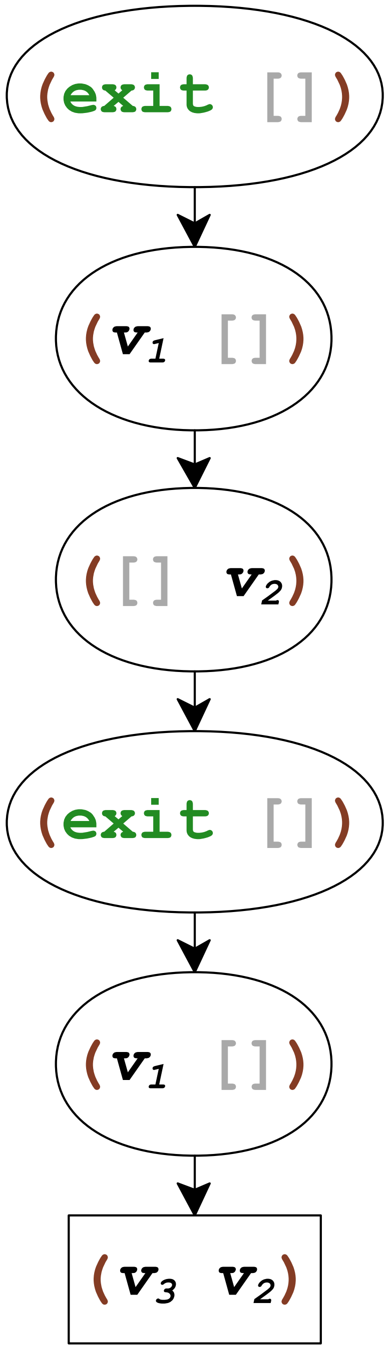
Composable Continuations



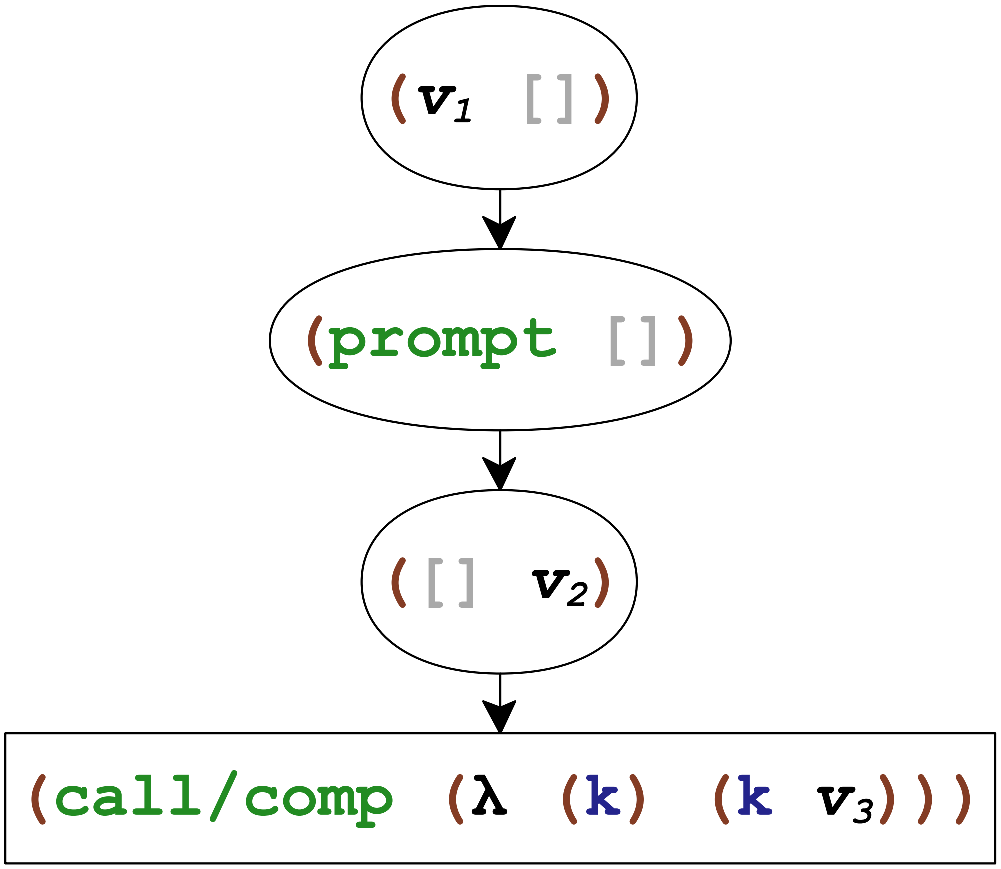
Composable Continuations



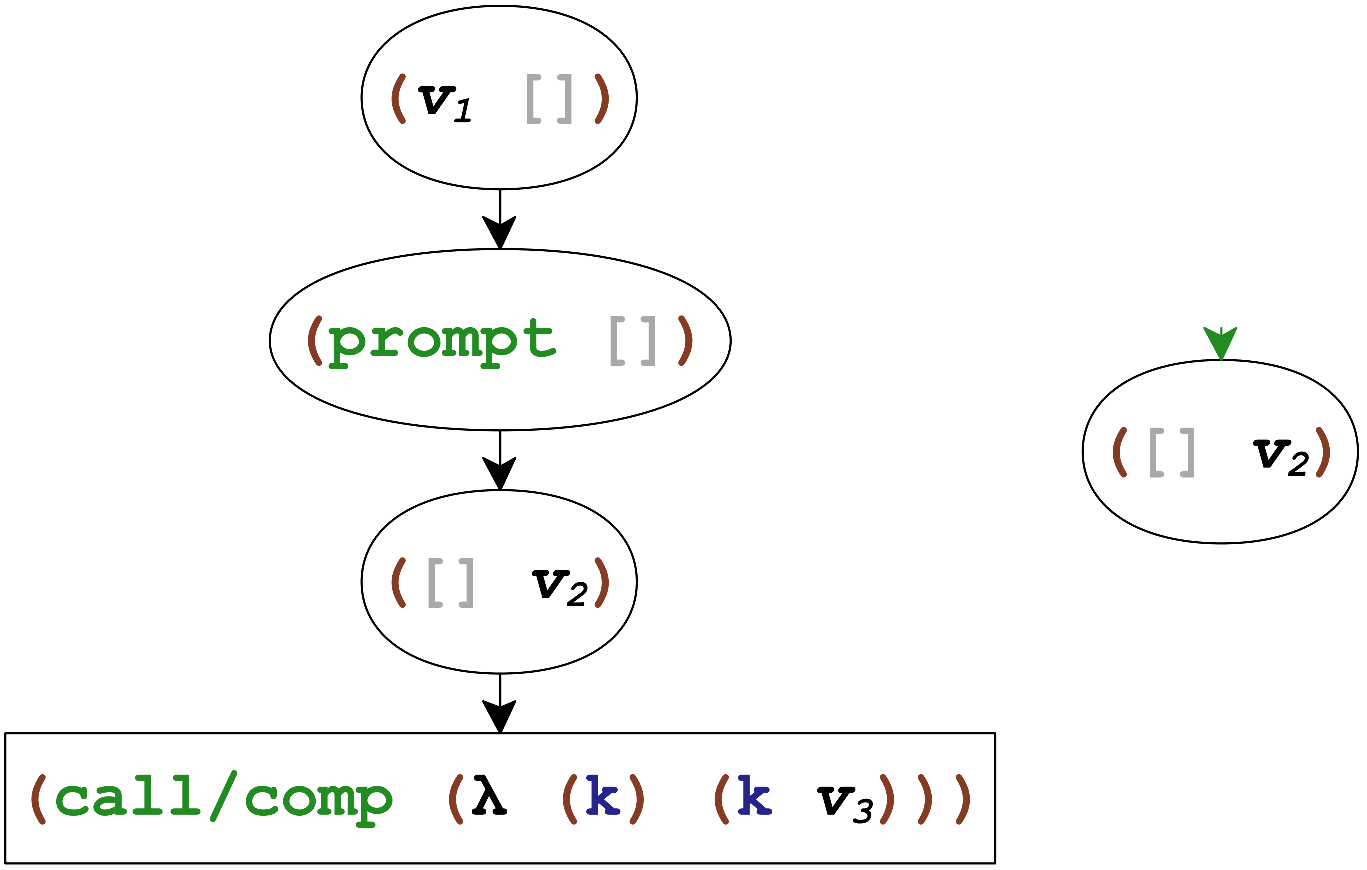
Composable Continuations



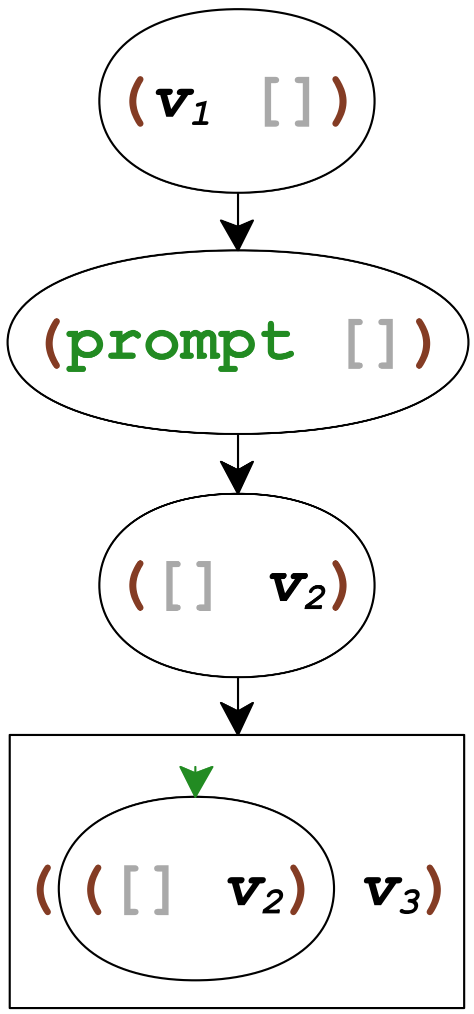
Delimited Capture



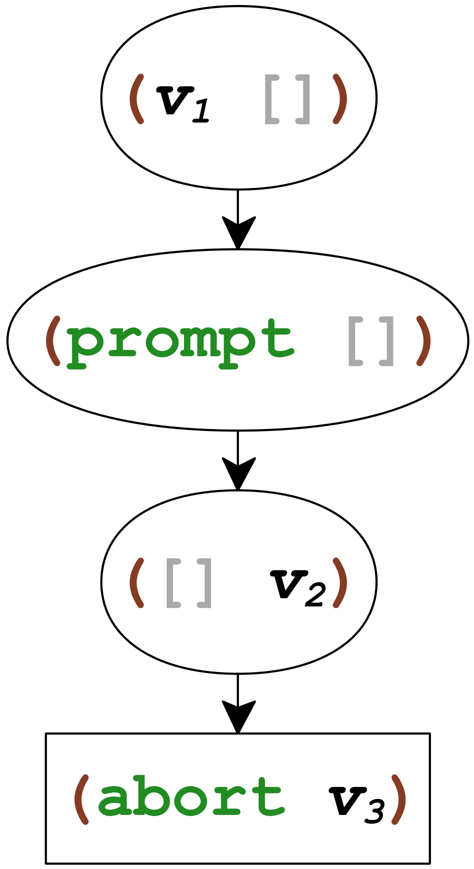
Delimited Capture



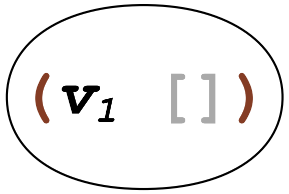
Delimited Capture



Delimited Abort



Delimited Abort



v_3

Delimited Abort

$$\boxed{(\mathbf{v}_1 \quad \mathbf{v}_3)}$$

Splitting Capture and Abort

- **`call/comp`** : capture current continuation
- **`abort`** : abort current continuation

- $\mathcal{F}$: capture and abort current continuation

Splitting Capture and Abort

- `call/comp` : capture current continuation
- `abort` : abort current continuation

- $\mathcal{F}$: capture and abort current continuation

```
𝓕 = (λ (f) (call/comp
               (λ (k) (abort (λ () (f k))))))
```

Splitting Capture and Abort

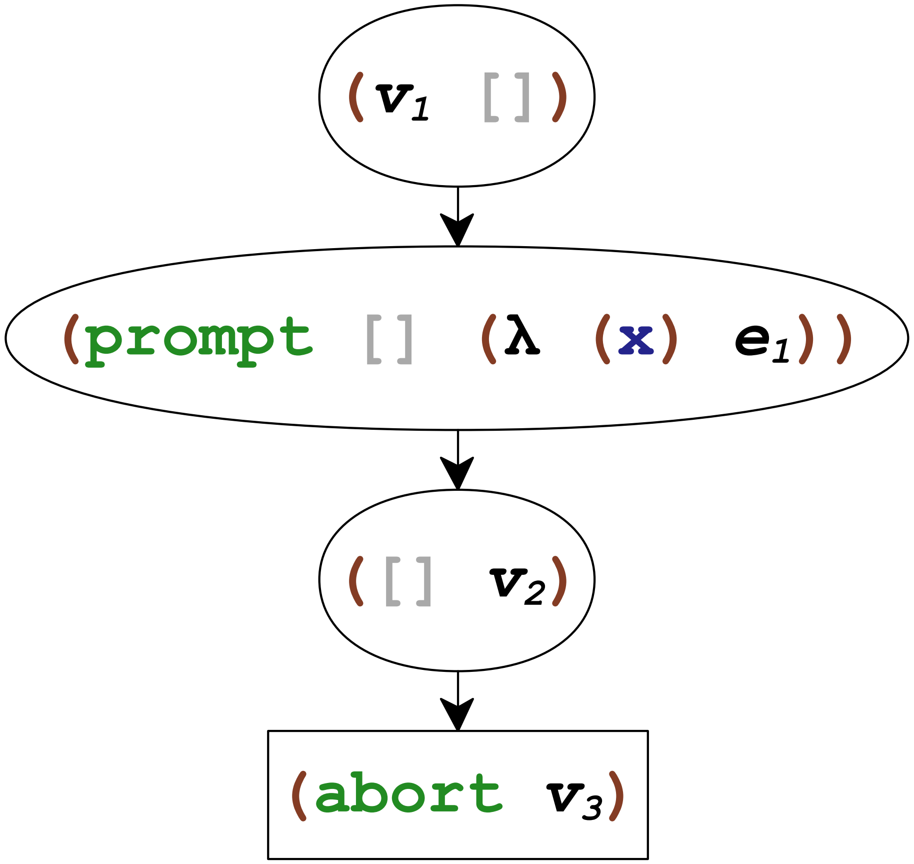
- `call/comp` : capture current continuation
- `abort` : abort current continuation

- $\mathcal{F}$: capture and abort current continuation

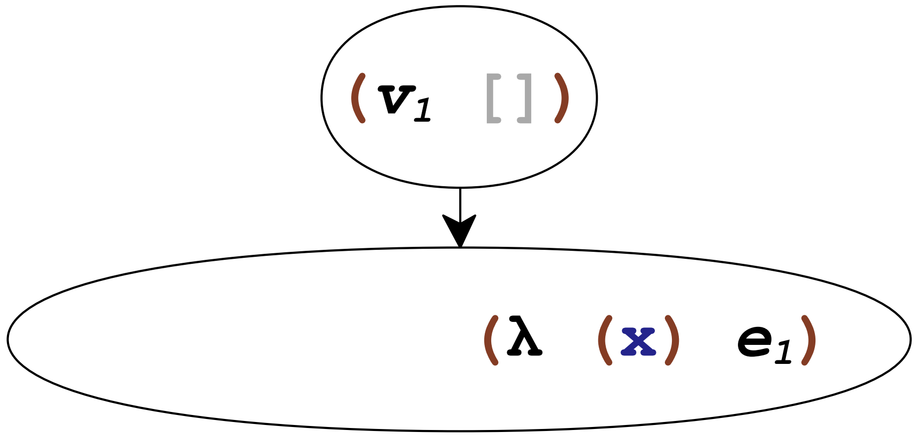
```
𝓕 = (λ (f) (call/comp
              (λ (k) (abort (λ () (f k))))))
```

delay `(f k)` until after abort

Prompt with Handler

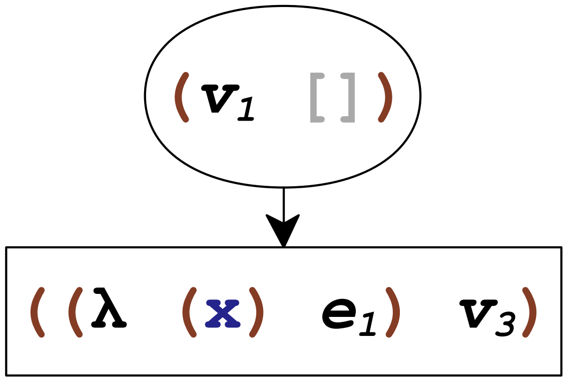


Prompt with Handler



v_3

Prompt with Handler



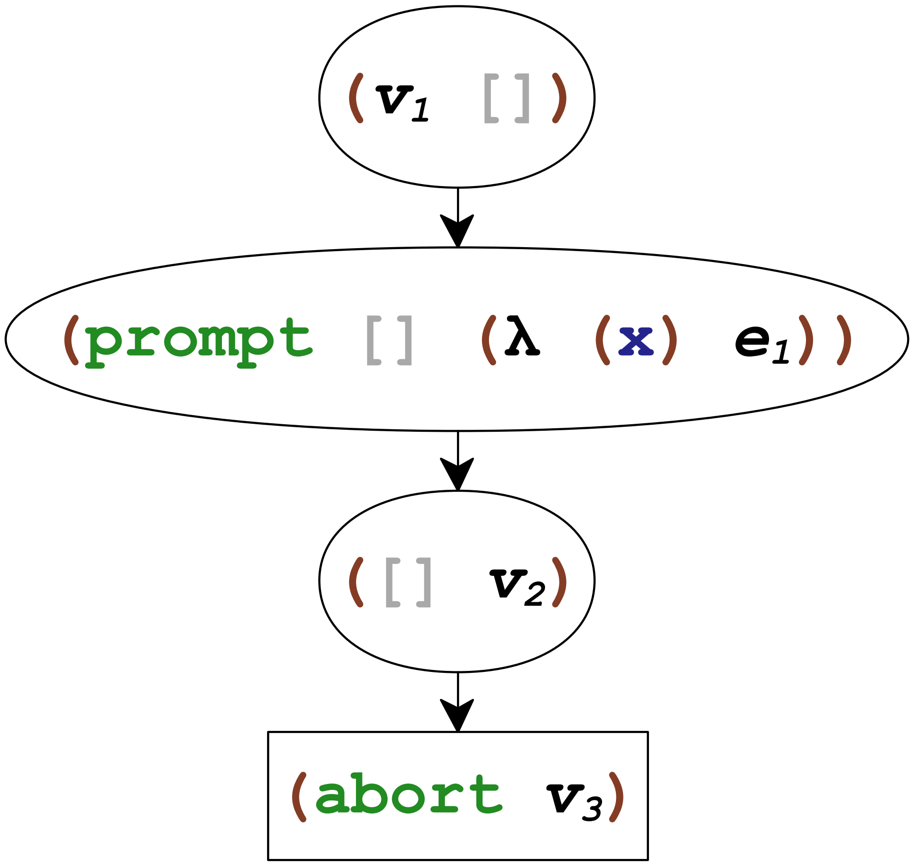
Prompt with Handler

$$\begin{tikzcd}
\texttt{(}v_1\ \texttt{[])} \arrow[d] \\
\boxed{\texttt{((}\lambda\ \texttt{(x)}\ e_1\texttt{)}\ v_3\texttt{)}}
\end{tikzcd}$$

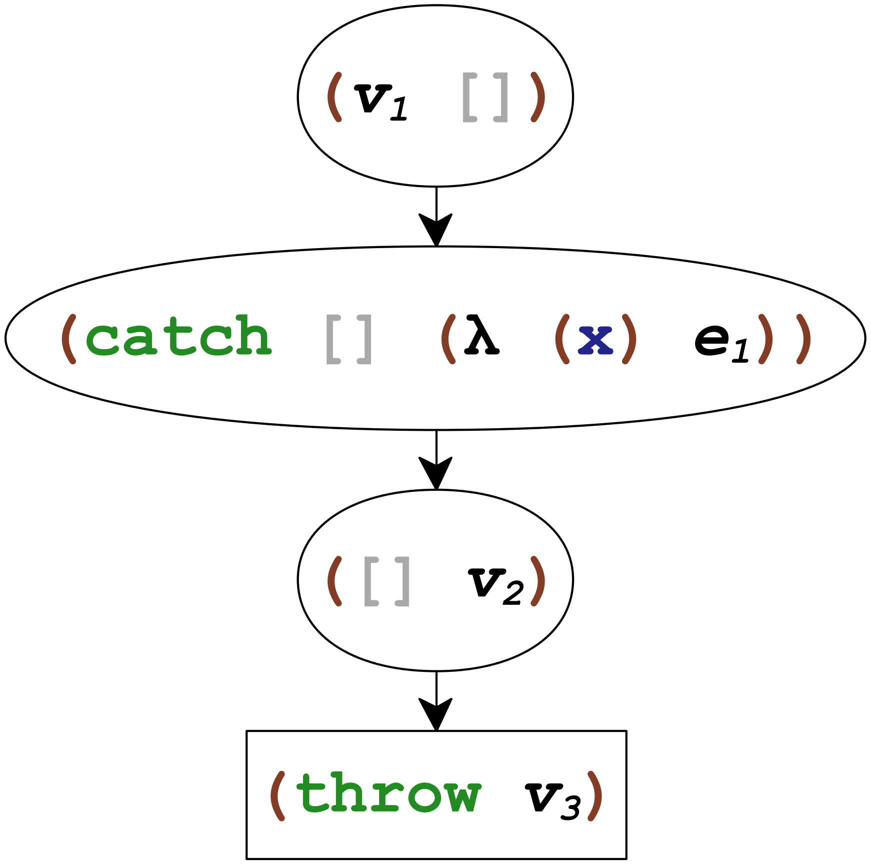
$$\texttt{(prompt } e_1\texttt{)} = \texttt{(prompt } e_1\ \texttt{(}\lambda\ \texttt{(t) (t)))}$$

$$\begin{aligned}\mathcal{F} = \texttt{(}\lambda\ \texttt{(f) (call/comp}& \\ &\texttt{(}\lambda\ \texttt{(k) (abort (}\lambda\ \texttt{() (f k))))))}\end{aligned}$$

Prompt with Handler



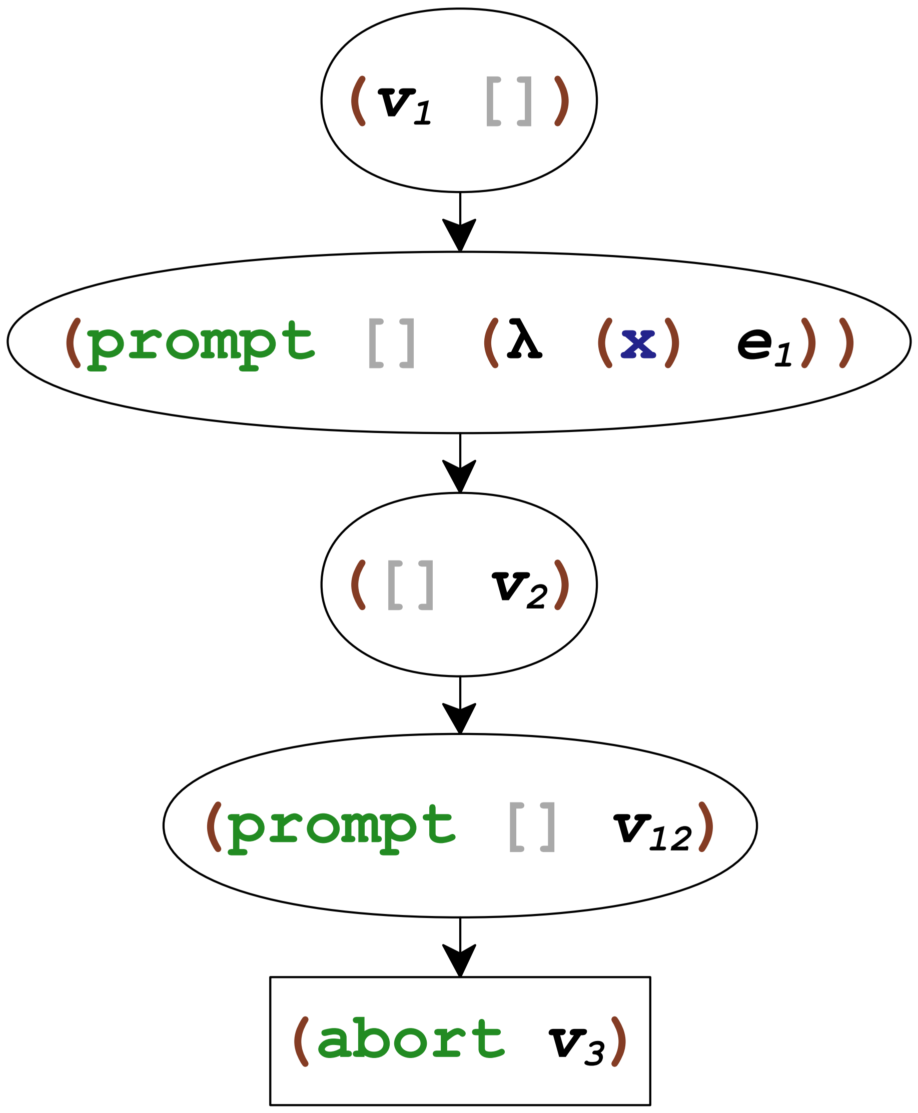
Catch and Throw



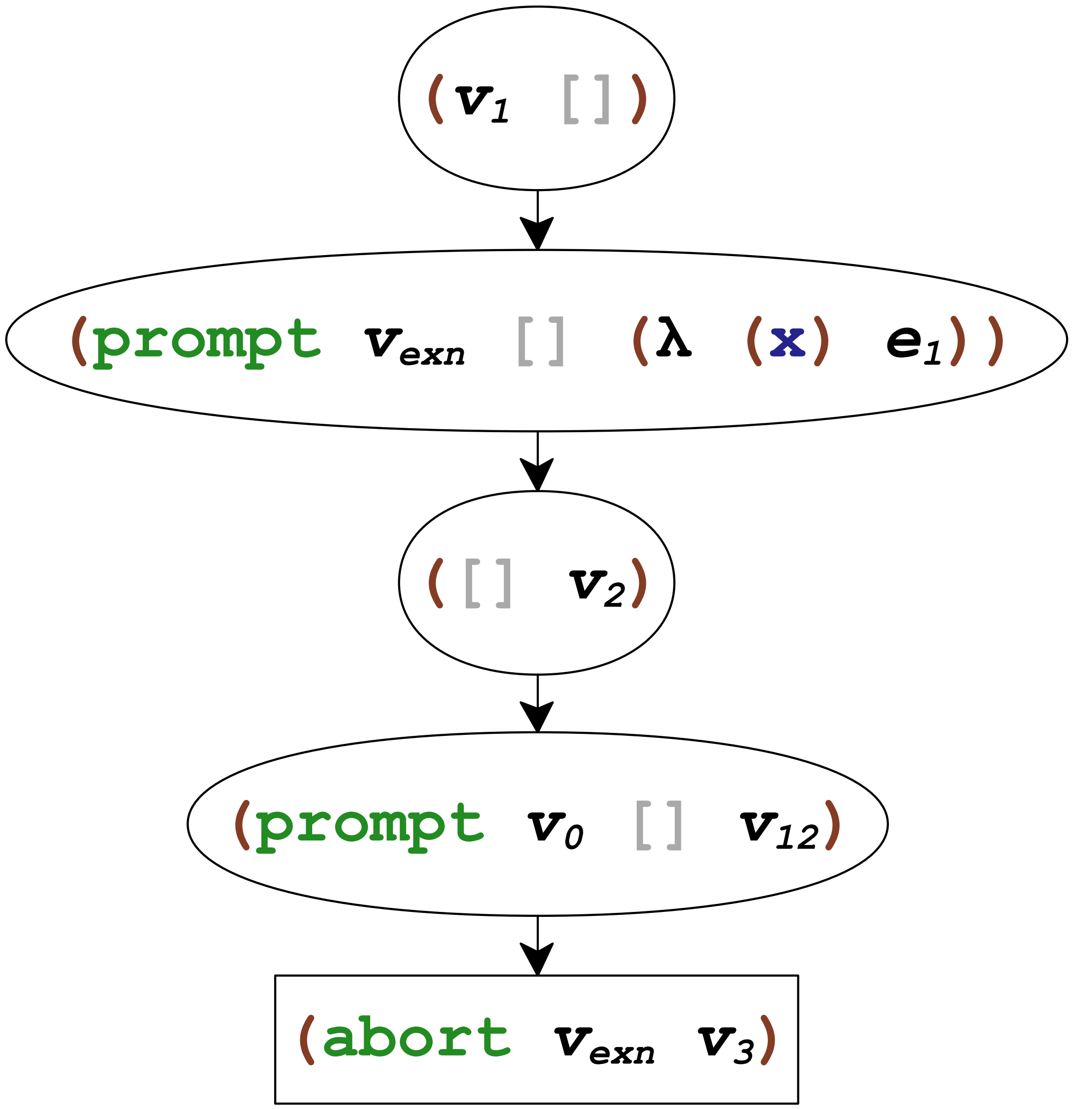
Catch and Throw

$$\begin{tikzcd}
(v_1\ [\,]) \arrow[d] \\
(\texttt{catch}\ [\,]\ (\lambda\ (x)\ e_1)) \arrow[d] \\
([\,]\ v_2) \arrow[d] \\
(\texttt{prompt}\ [\,]\ v_{12}) \arrow[d] \\
\boxed{(\texttt{throw}\ v_3)}
\end{tikzcd}$$

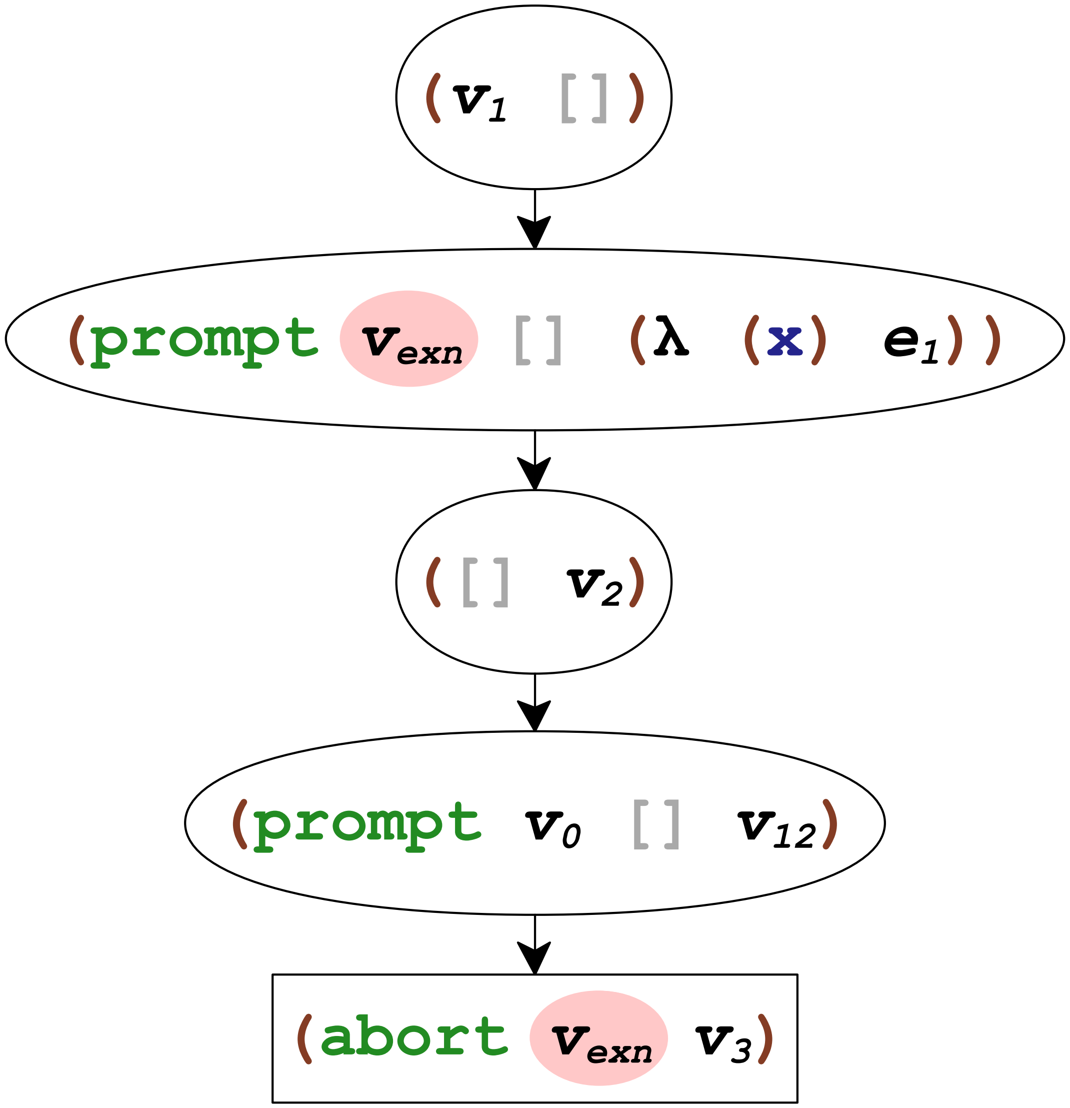
Catch and Throw



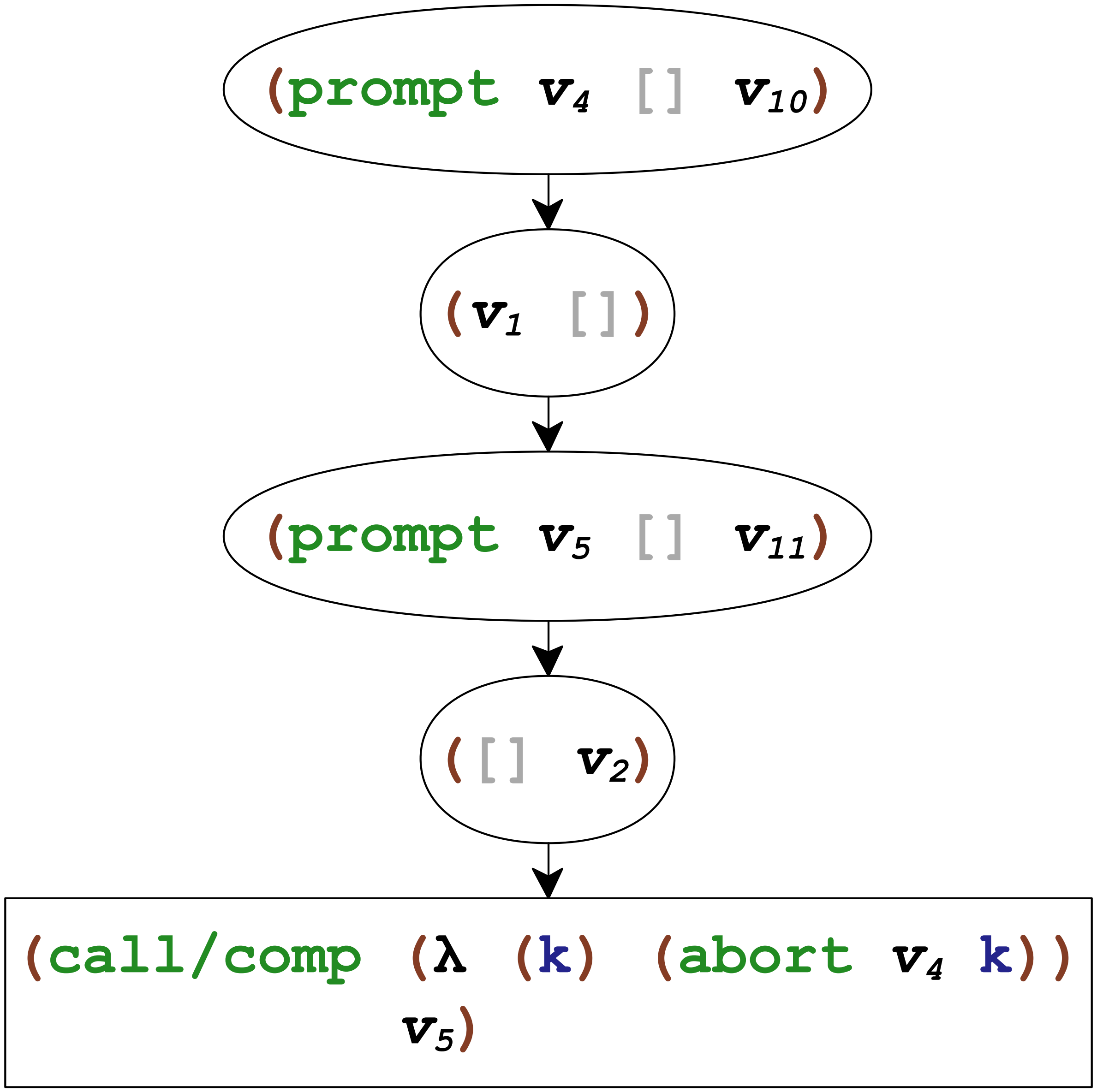
Catch and Throw



Tagged Prompts

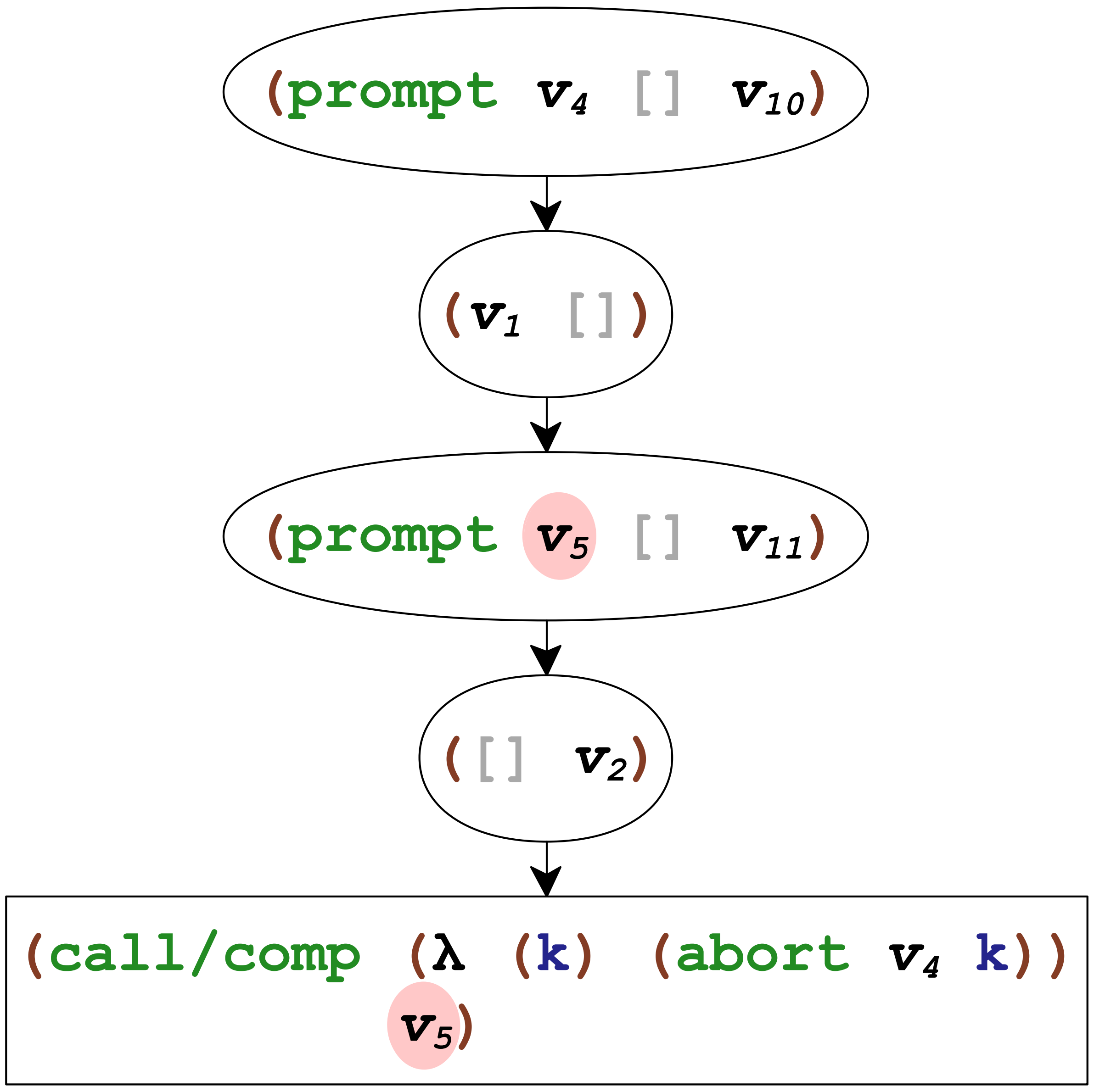

```
(v1 [])
(prompt vexn [] (λ (x) e1))
([] v2)
(prompt v0 [] v12)
(abort vexn v3)
```

Tagged Prompts

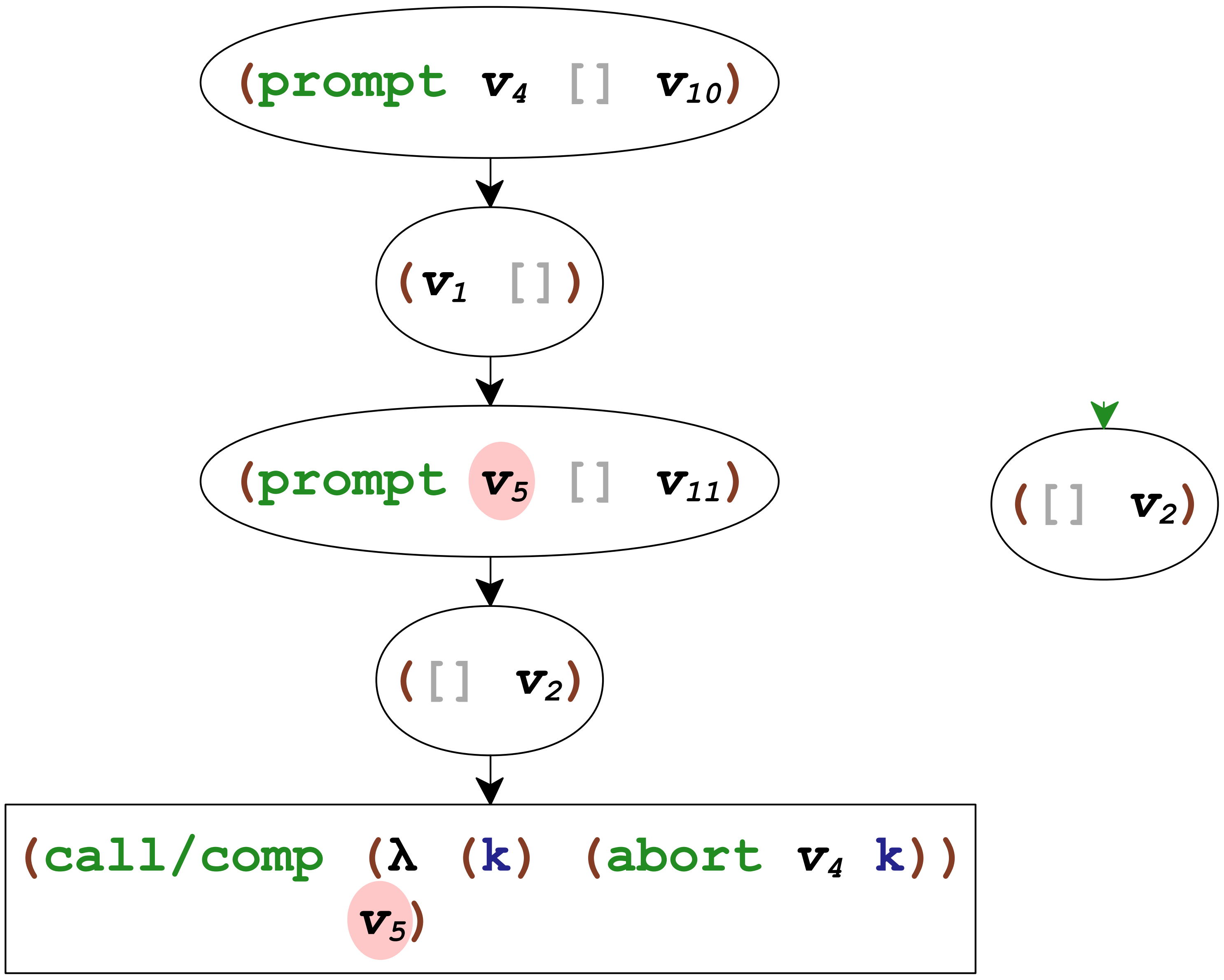


```
(prompt v4 [] v10)
(v1 [])
(prompt v5 [] v11)
([] v2)
(call/comp (λ (k) (abort v4 k))
           v5)
```

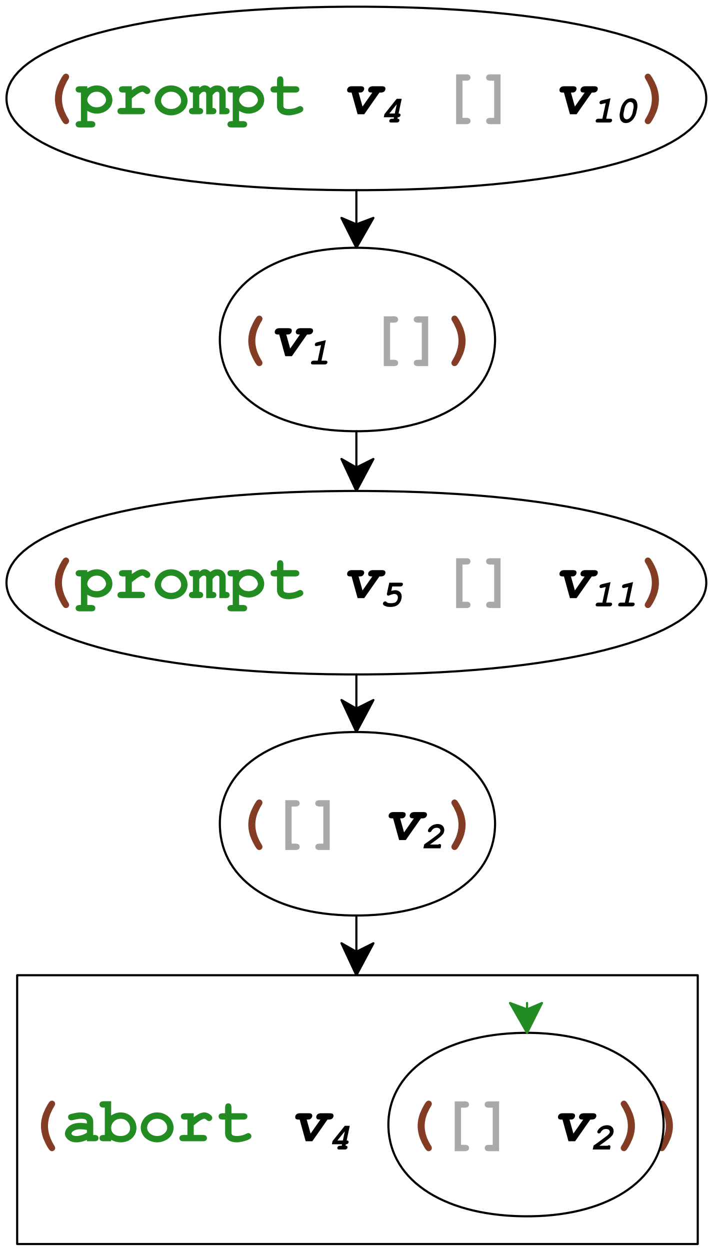
Tagged Prompts


(prompt v_4 [] v_{10})
(v_1 [])
(prompt v_5 [] v_{11})
([] v_2)
(call/comp (λ (k) (abort v_4 k))
v_5)

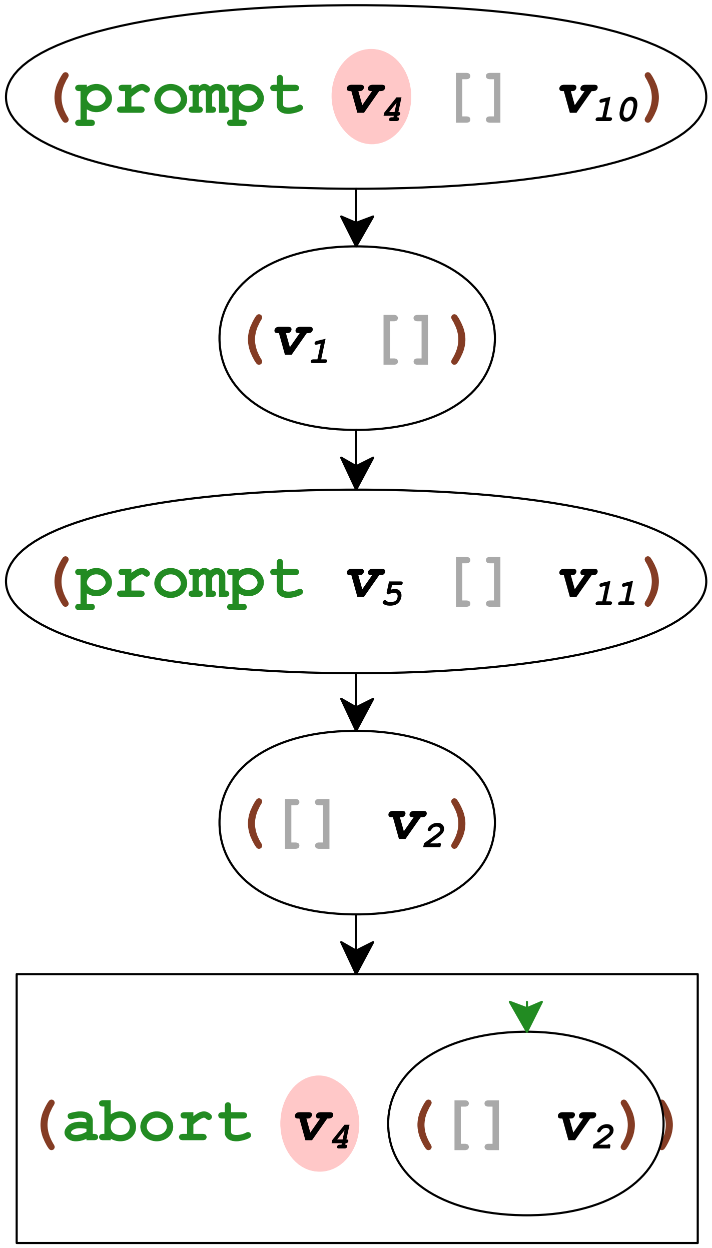
Tagged Prompts



Tagged Prompts

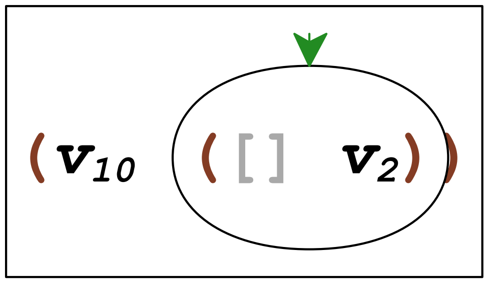


Tagged Prompts



Tagged Prompts

Tagged Prompts



Continuations Summary

- `prompt` with handler and tag
- `abort` with tag
- `call/comp` with tag
- continuation composition
- plain `call/cc` uses default tag

[Sitaram PLDI'93]

Dynamic Binding

```
(let ([user (get-login)])
  (search user (find-user-files user)))
```

Dynamic Binding

```
(let ([user (get-login)])
  (call-with-user
   user
    (λ () (search (find-user-files)))))
```

Dynamic Binding

```
(let ([user (get-login)])
  (call/cm
   'user user
   (λ () (search (find-user-files)))))
```

Dynamic Binding

```
(let ([user (get-login)])
  (call/cm
   'user user
   (λ () (search (find-user-files)))))

(define (find-user-files)
  ....
  (current-mark 'user))
```

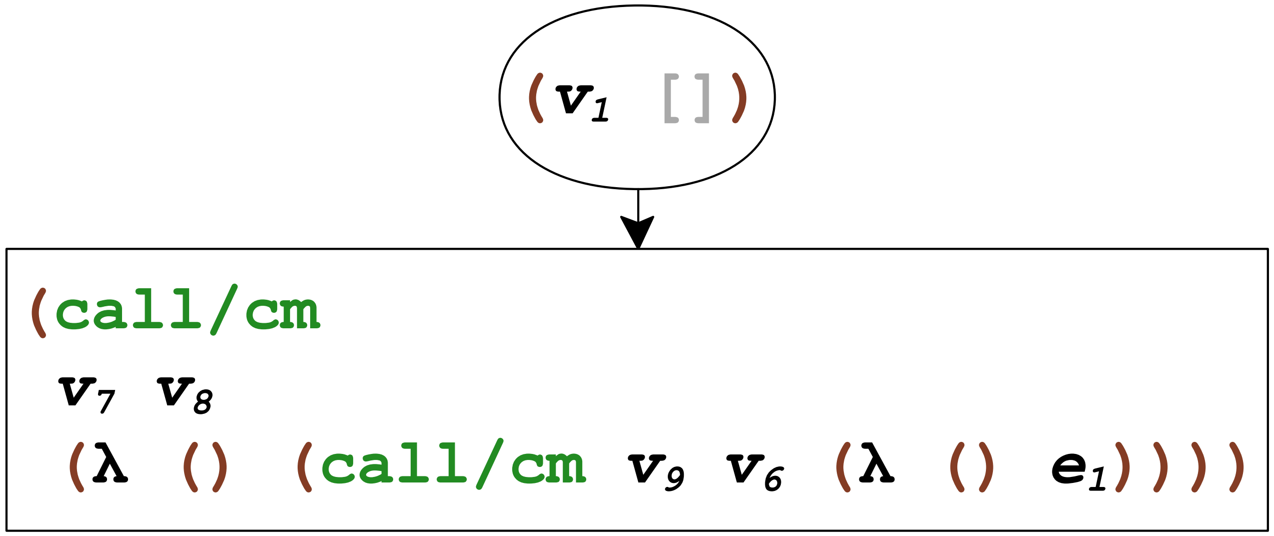
Dynamic Binding

```
(let ([user (get-login)])
  (call/cm
   'user user
   (λ () (search (find-user-files)))))

(define (find-user-files)
  ....
  (first (current-marks 'user))
```

Stack Inspection

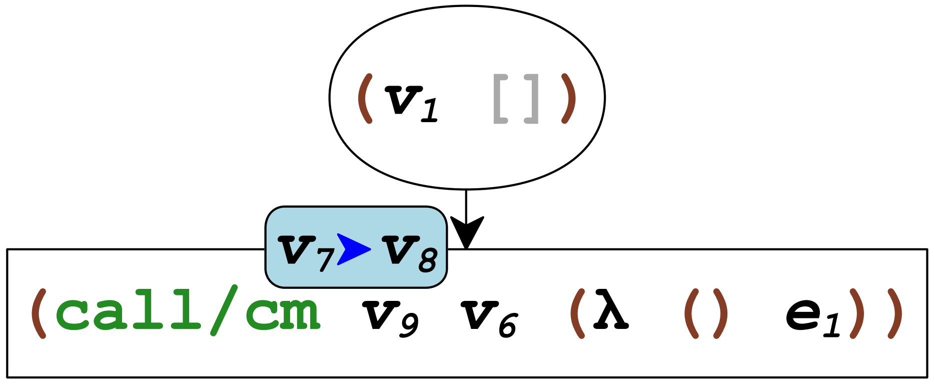
```
(define (search files)
  (call/cm
   'context 'search
   (lambda ()
     .... (make-database files) ....)))

(define (make-database files)
  (call/cm
   'context 'make-database
   (lambda ()
     .... (map open-input-file files) ....)))

(define (open-input-file file)
  (if (file-exists? file)
      ....
      (raise .... (current-marks 'context))))
```

Continuation Marks

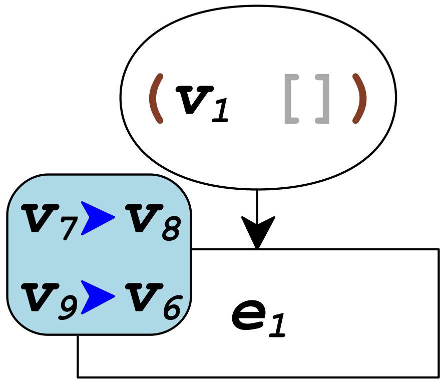


```
(call/cm
 v7 v8
 (λ () (call/cm v9 v6 (λ () e1))))
```

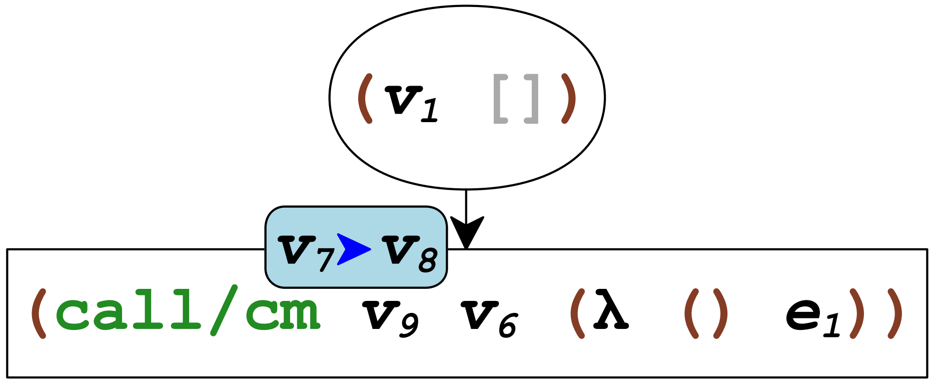
Continuation Marks



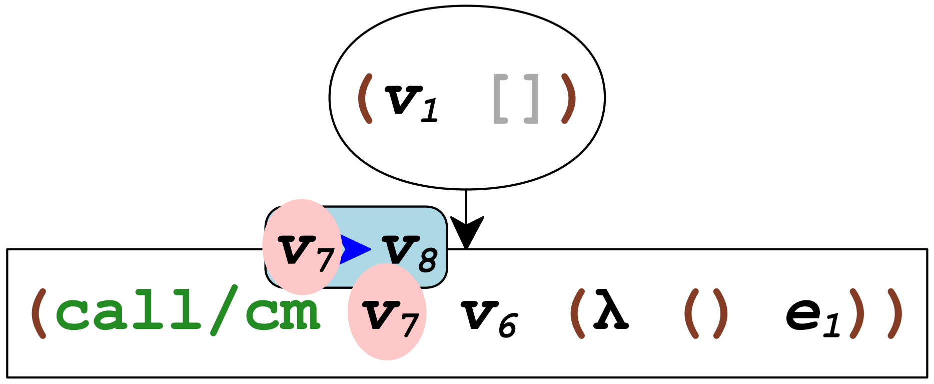
Continuation Marks



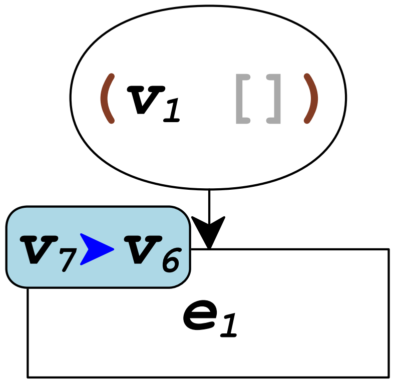
Continuation Marks



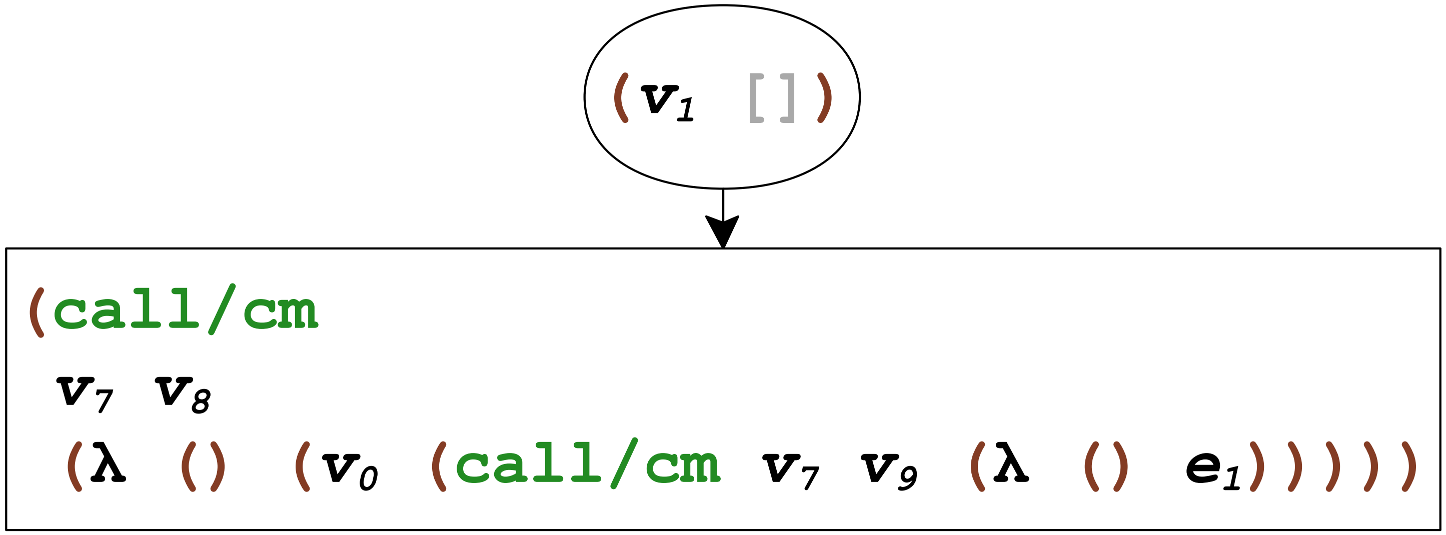
Continuation Marks



Continuation Marks



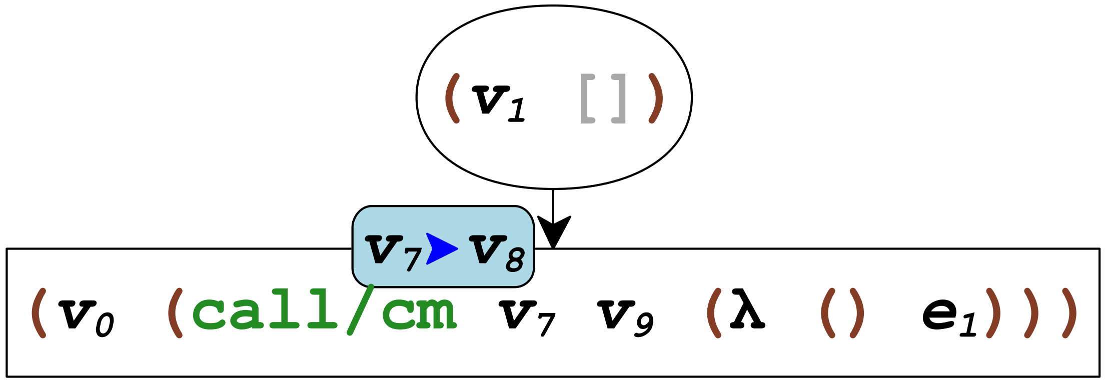
Continuation Marks



```
(v1 [])
```

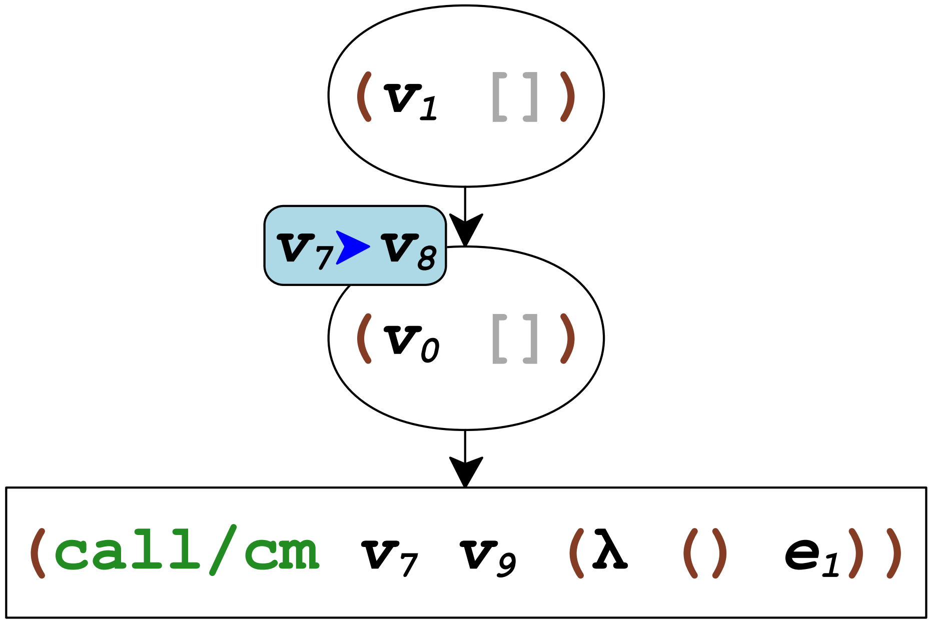
```
(call/cm
 v7 v8
 (λ () (v0 (call/cm v7 v9 (λ () e1)))))
```

Continuation Marks

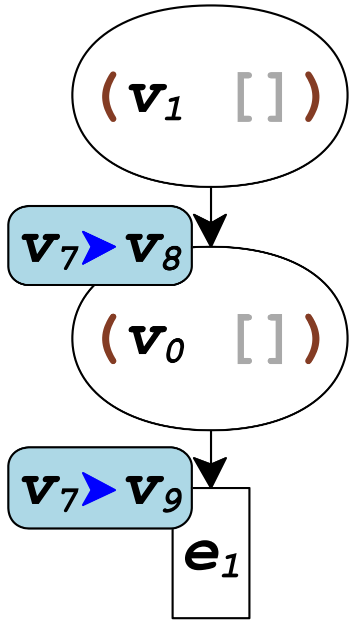


$(v_1 \; [\,])$
$v_7 \blacktriangleright v_8$
$(v_0 \; (\texttt{call/cm} \; v_7 \; v_9 \; (\lambda \; () \; e_1)))$

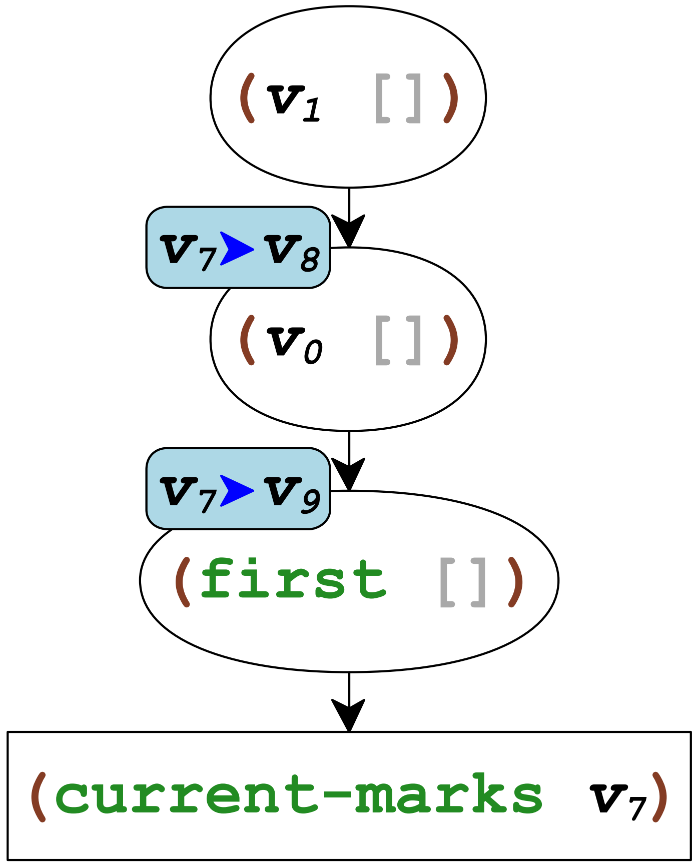
Continuation Marks



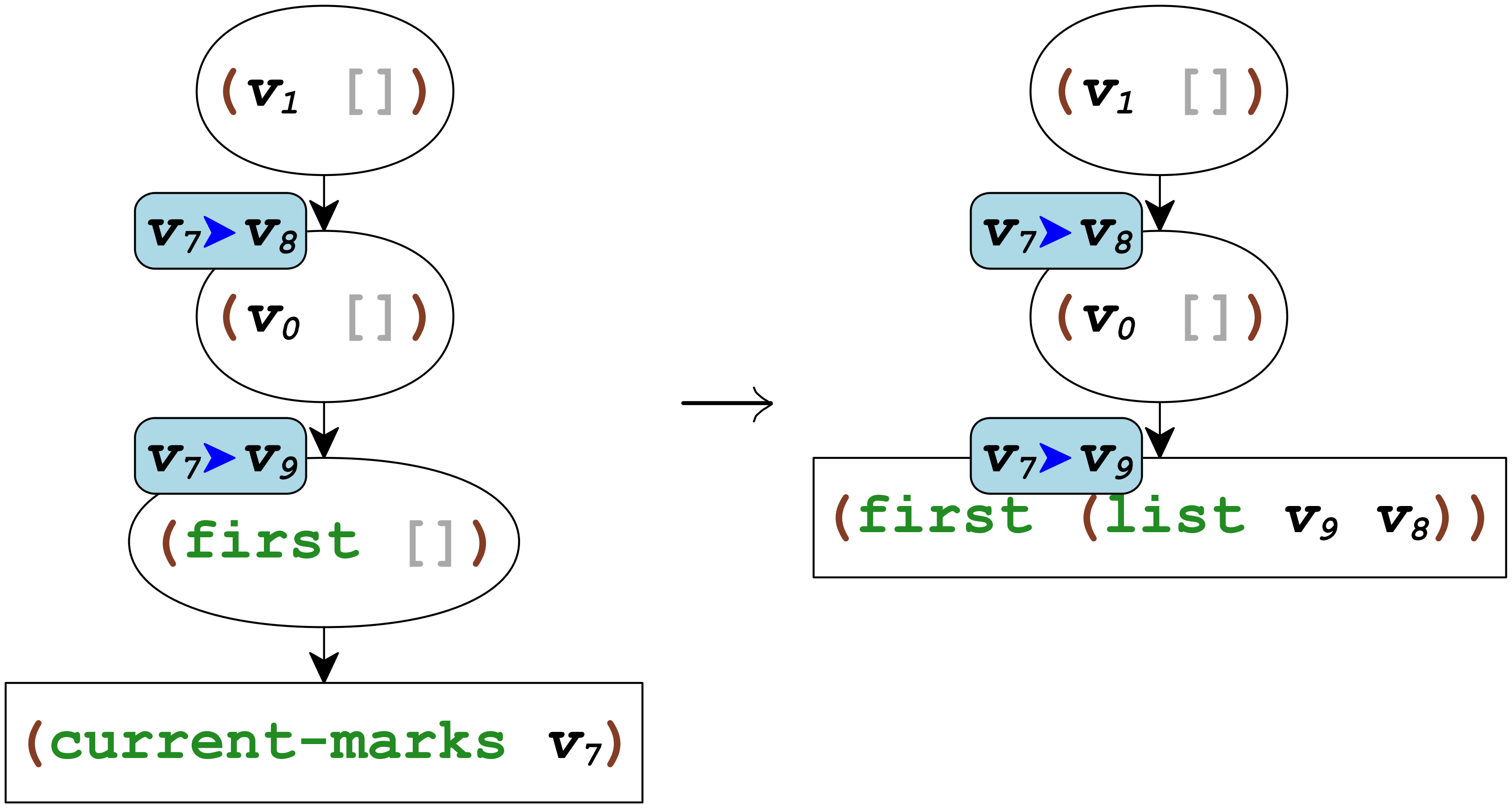
Continuation Marks



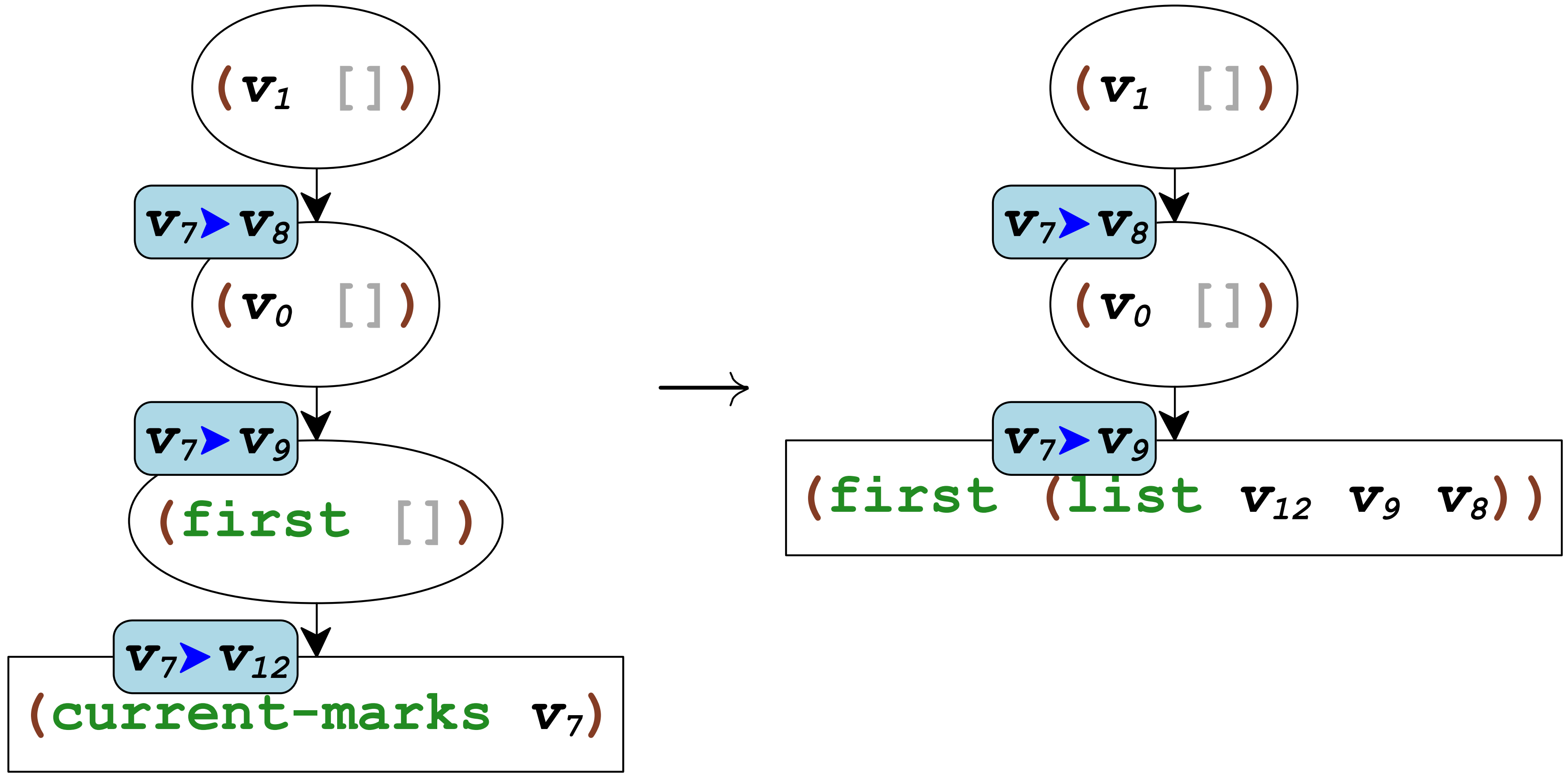
Continuation Marks



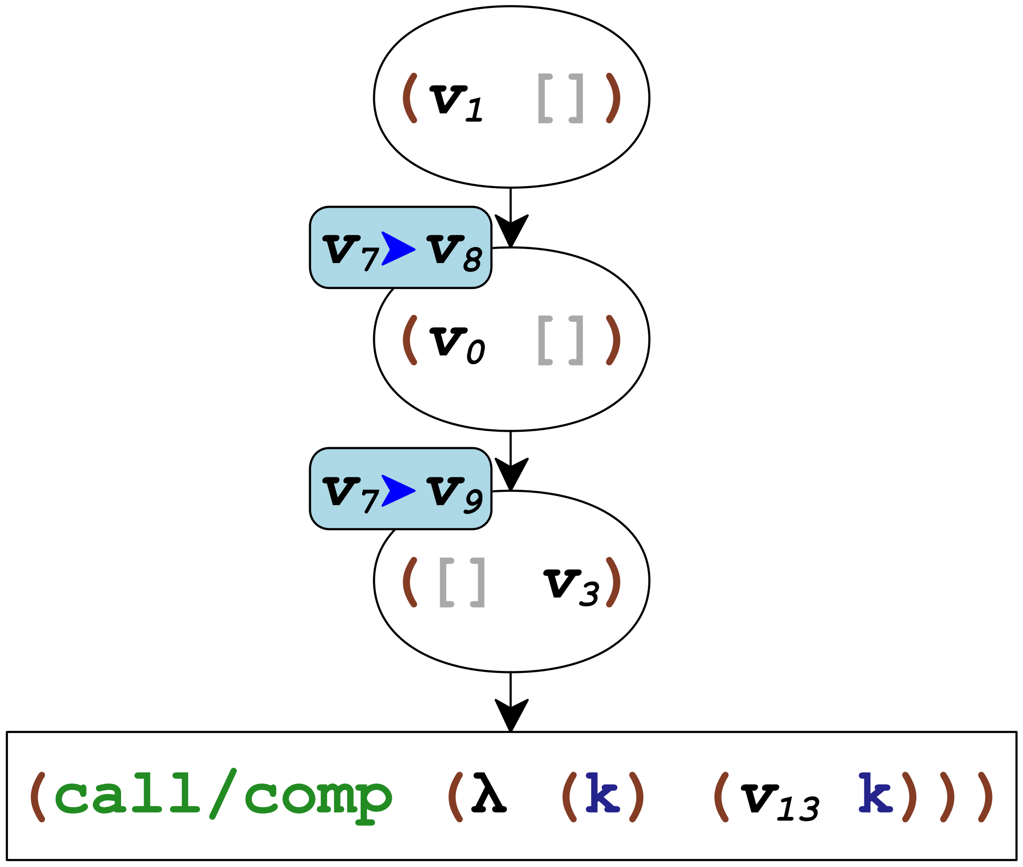
Continuation Marks



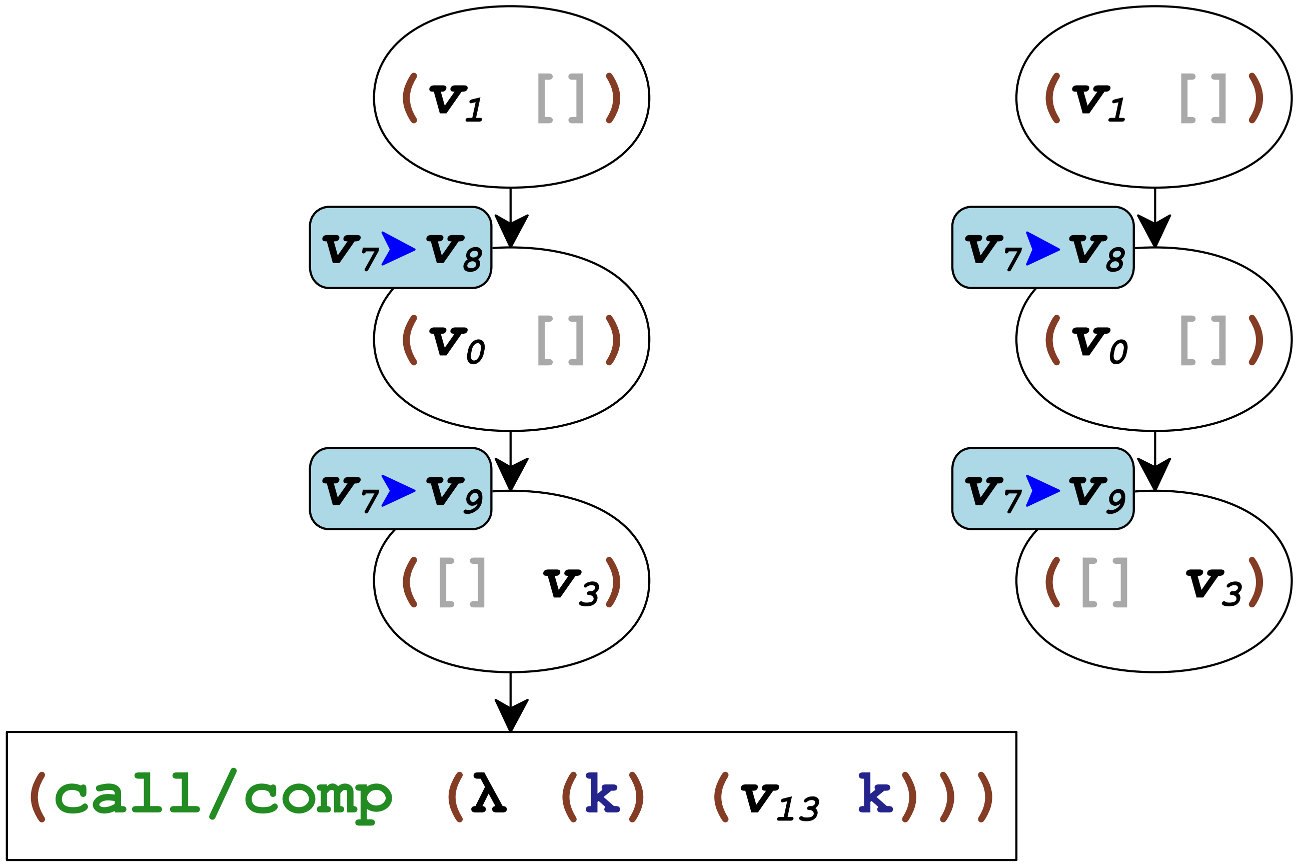
Continuation Marks



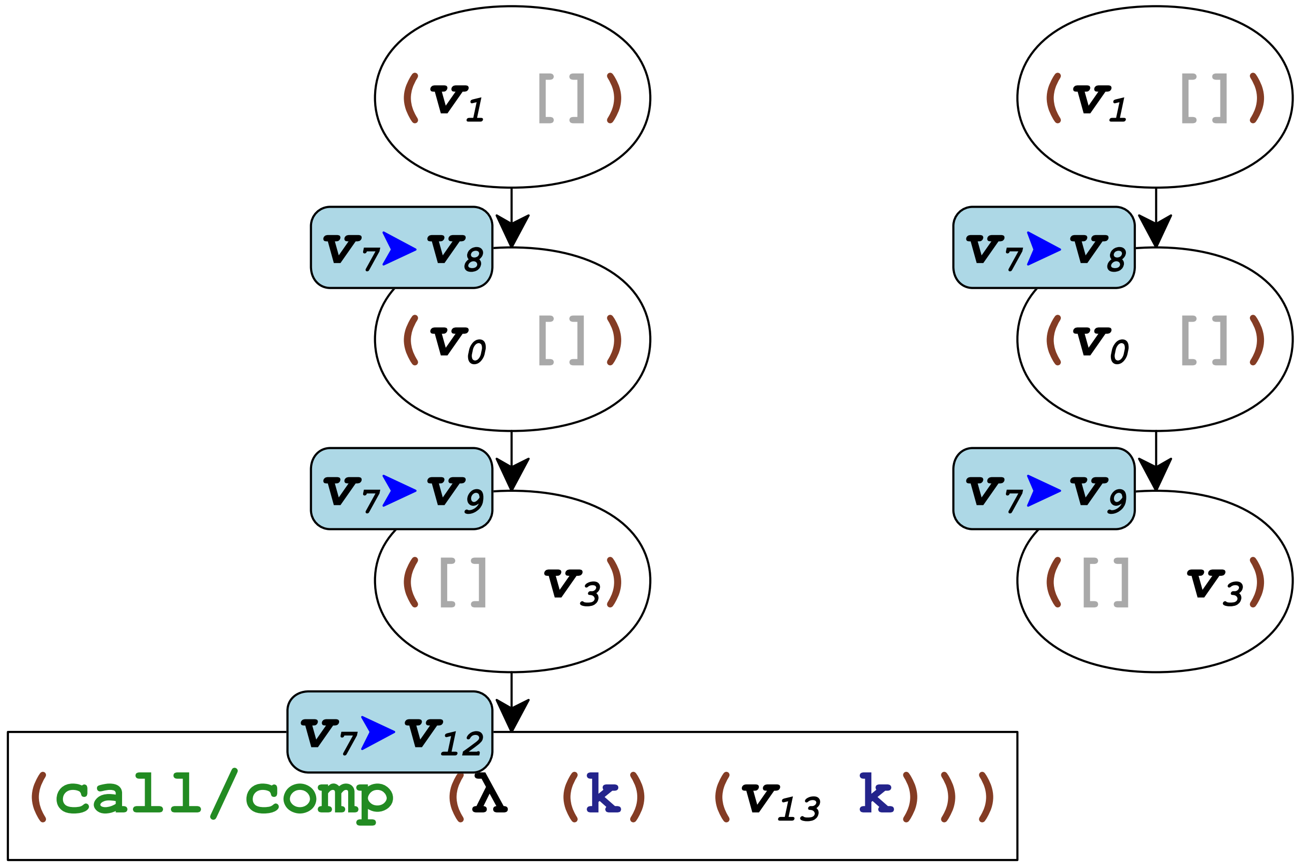
Capturing Marks



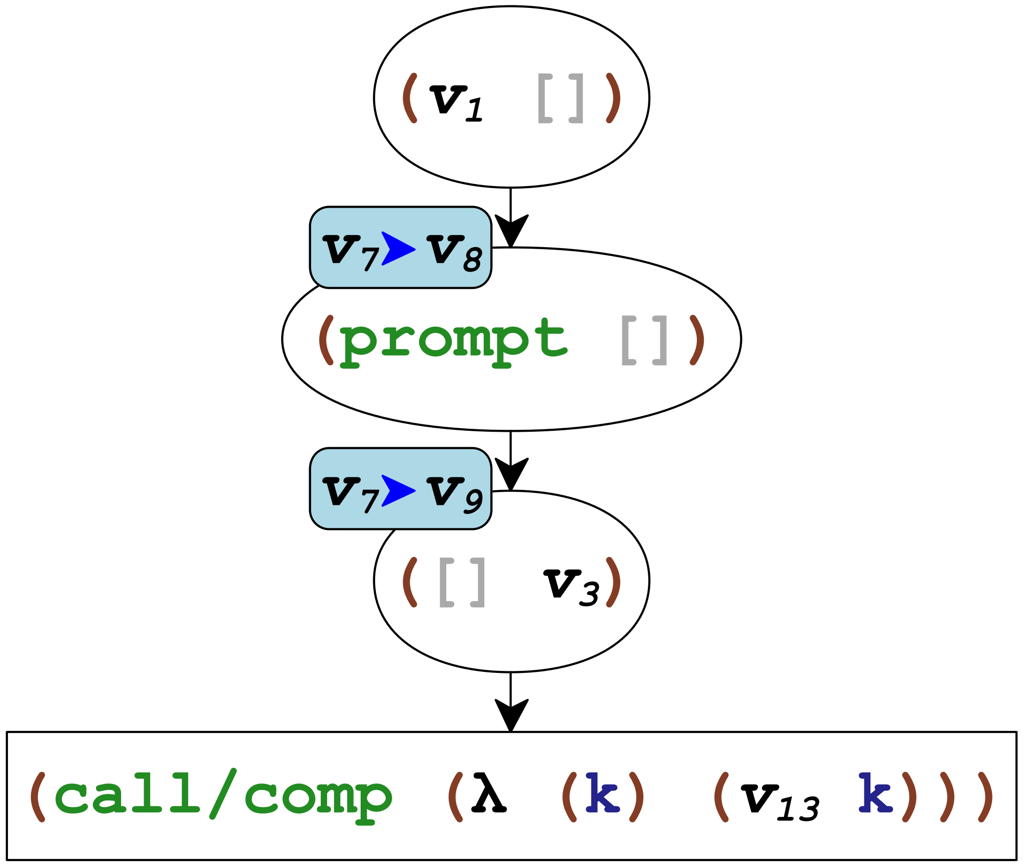
Capturing Marks



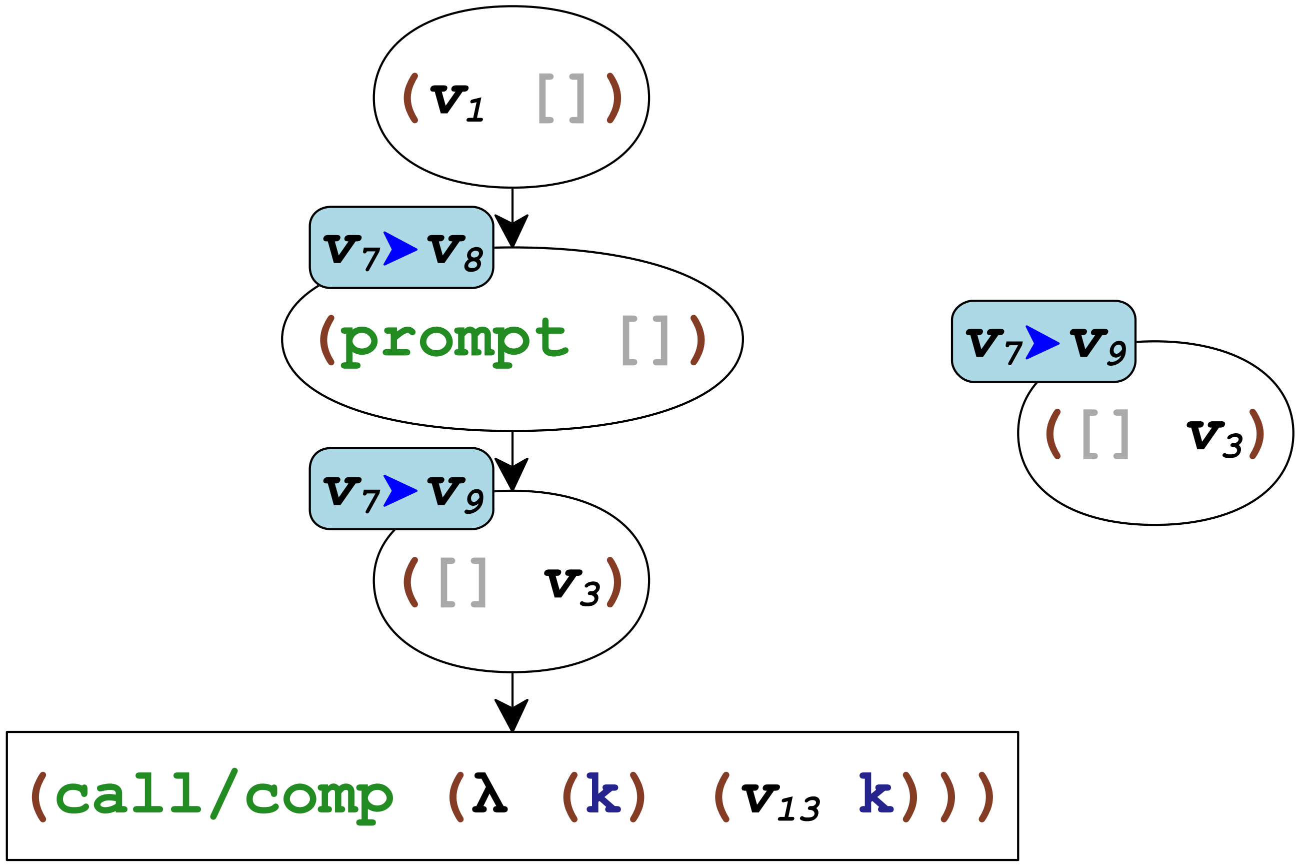
Capturing Marks



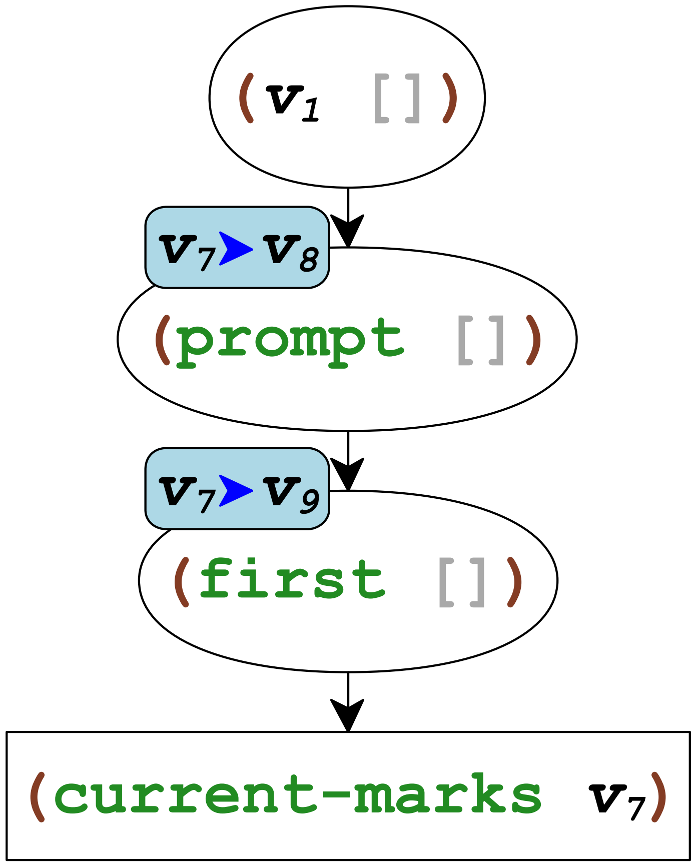
Capturing Marks



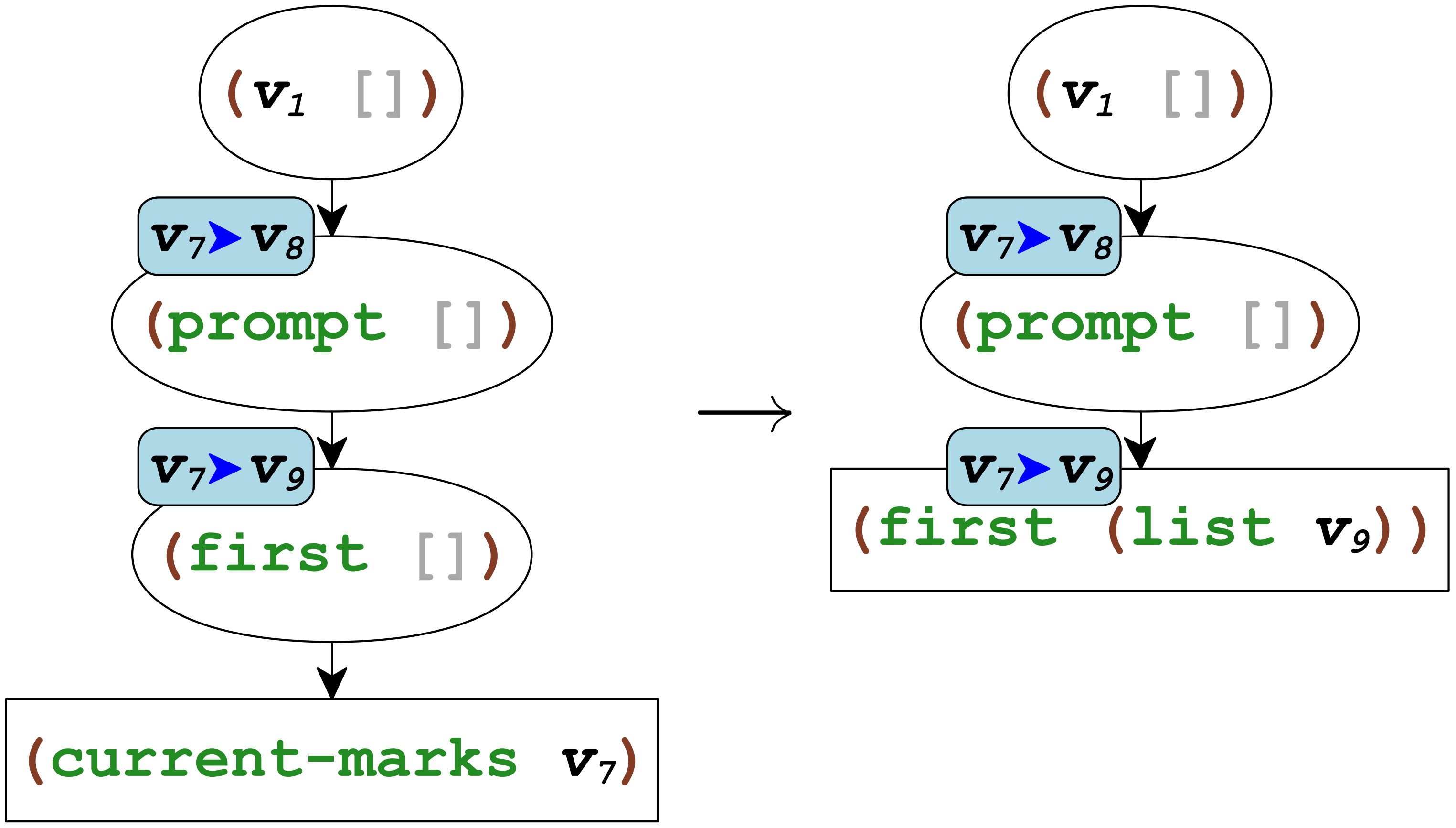
Capturing Marks



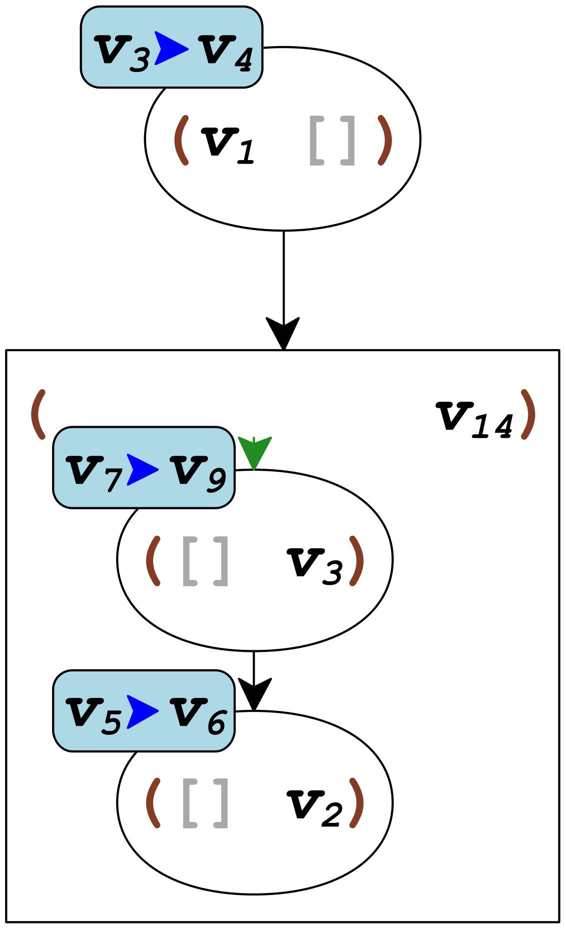
Getting Delimited Marks



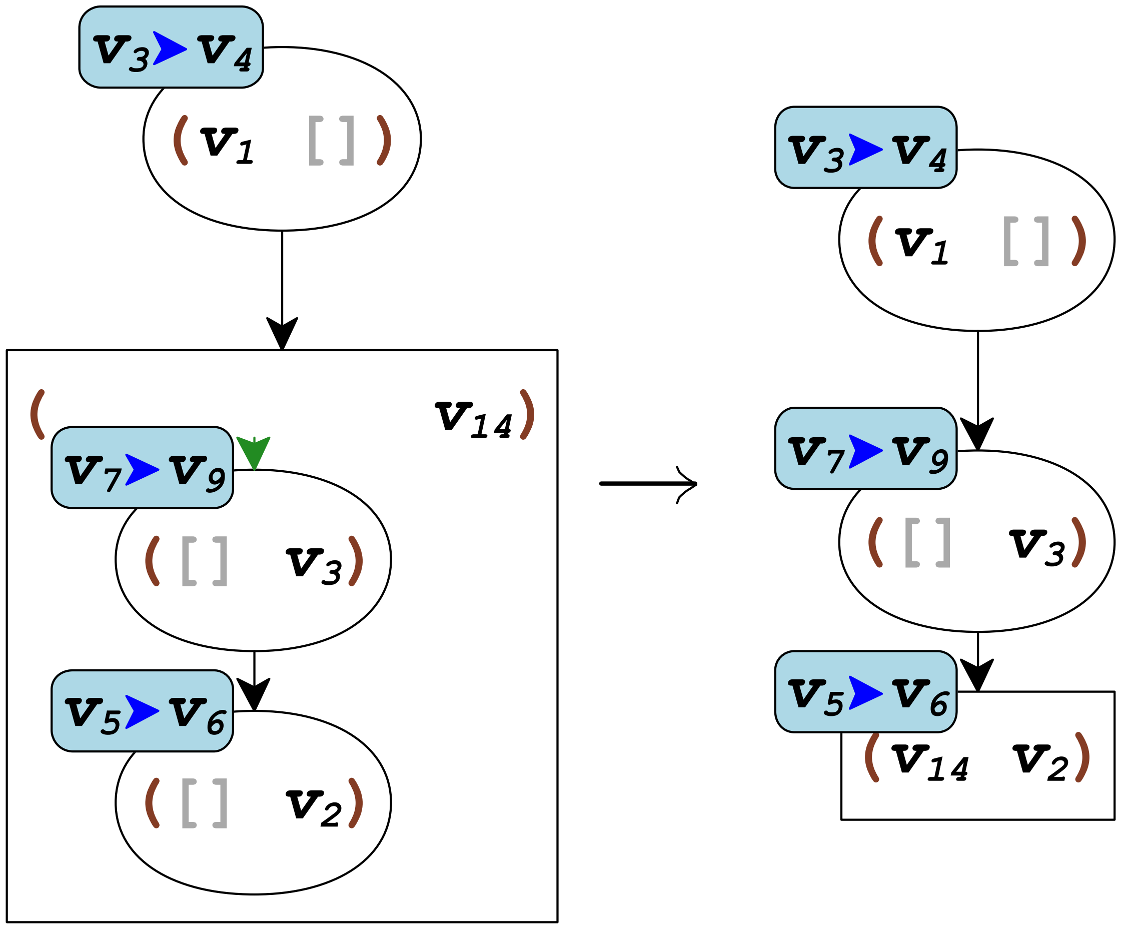
Getting Delimited Marks



Restoring Marks

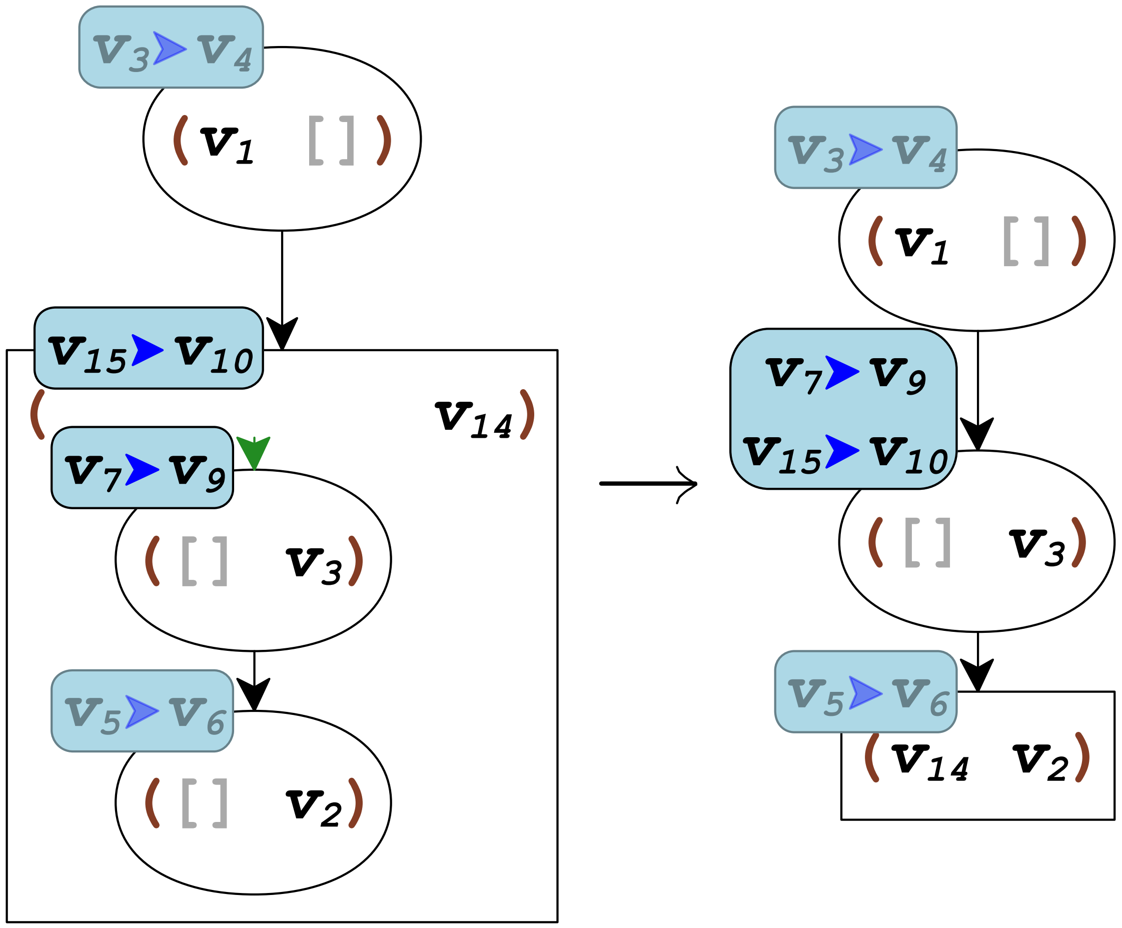

$v_3 \blacktriangleright v_4$
$(v_1 \ [\,])$
$($
$v_{14})$
$v_7 \blacktriangleright v_9$
$([\,] \ v_3)$
$v_5 \blacktriangleright v_6$
$([\,] \ v_2)$

Restoring Marks

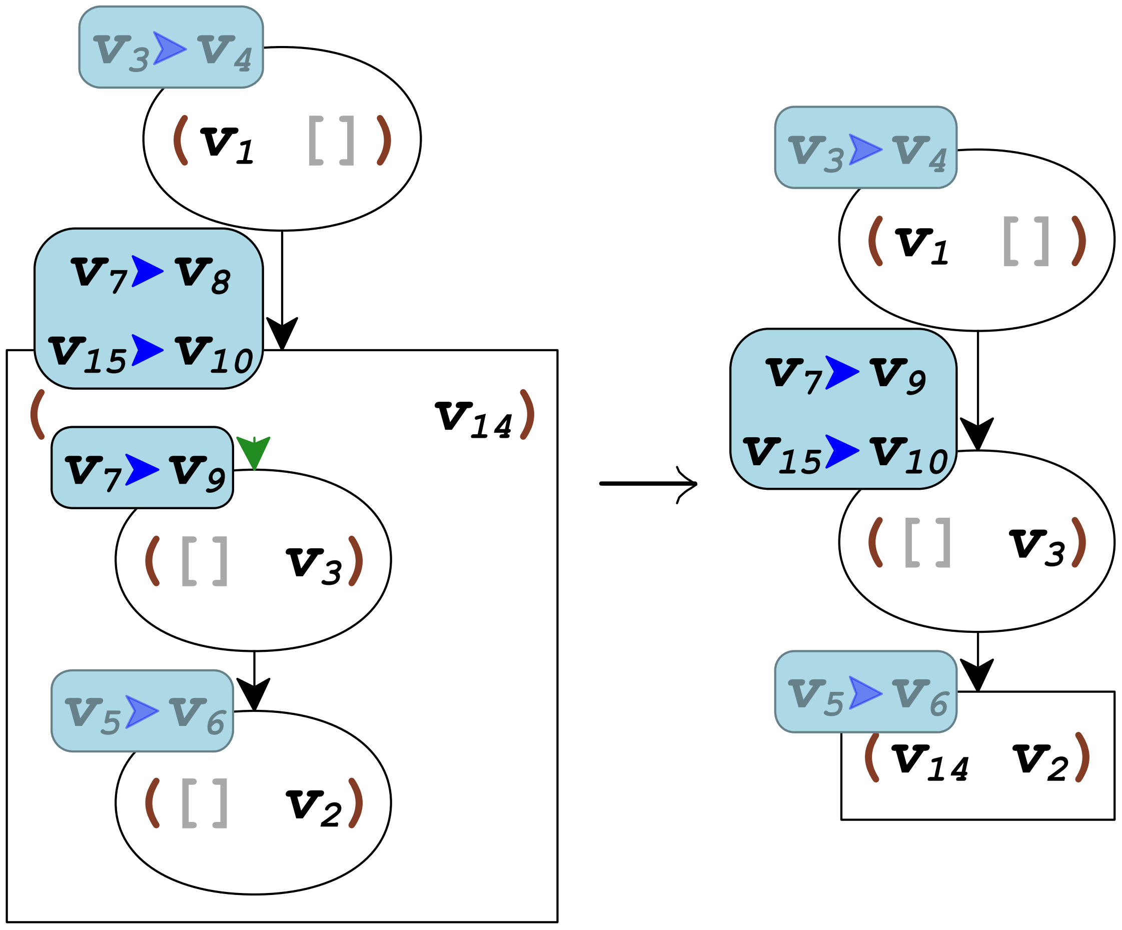


$v_3 \blacktriangleright v_4$
$(v_1 \; [\,])$
$(\quad v_{14})$
$v_7 \blacktriangleright v_9$
$([\,] \; v_3)$
$v_5 \blacktriangleright v_6$
$([\,] \; v_2)$
$\longrightarrow$
$v_3 \blacktriangleright v_4$
$(v_1 \; [\,])$
$v_7 \blacktriangleright v_9$
$([\,] \; v_3)$
$v_5 \blacktriangleright v_6$
$(v_{14} \; v_2)$

Restoring Marks



Restoring Marks



$v_3 \blacktriangleright v_4$
(v_1 [ ])
$v_7 \blacktriangleright v_8$
$v_{15} \blacktriangleright v_{10}$
(v_{14})
$v_7 \blacktriangleright v_9$
([ ] v_3)
$v_5 \blacktriangleright v_6$
([ ] v_2)
$v_3 \blacktriangleright v_4$
(v_1 [ ])
$v_7 \blacktriangleright v_9$
$v_{15} \blacktriangleright v_{10}$
([ ] v_3)
$v_5 \blacktriangleright v_6$
(v_{14} v_2)

Dynamic Binding Summary

- `call/cm` to add marks
- `current-marks` to get marks, up to a tag
- capture marks in `call/comp`
- splice marks in continuation composition

see also [Kiselyov et al. ICFP'06]

Side Effects and Control

```
(define (with-resource work-thunk)
  (begin
    (grab-resource)
    (work-thunk)
    (release-with-resource)))
```

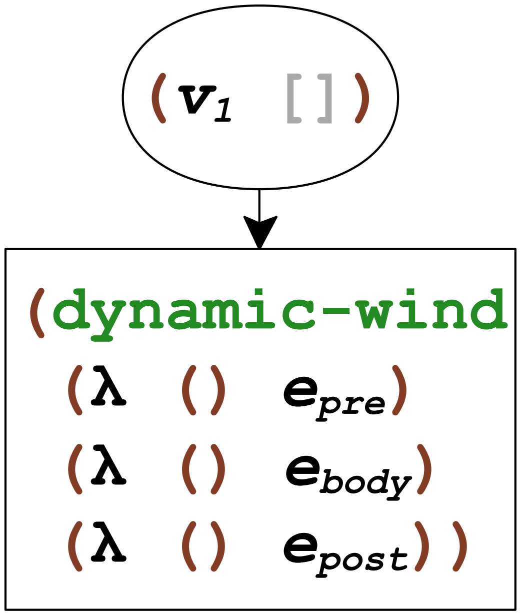
Dynamic Wind

```
(define (with-resource work-thunk)
  (dynamic-wind
   (λ () (grab-resource))
   (λ () (work-thunk))
   (λ () (release-with-resource))))
```

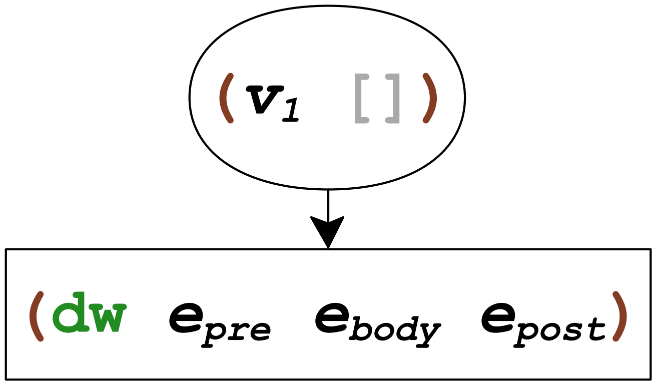
Dynamic Wind

```
(dynamic-wind
 (λ () e_pre)
 (λ () e_body)
 (λ () e_post))
```

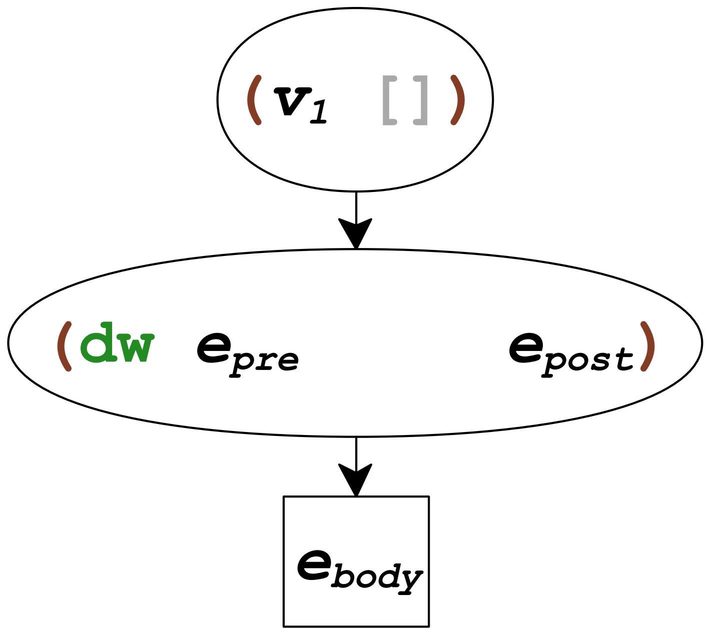
Dynamic Wind



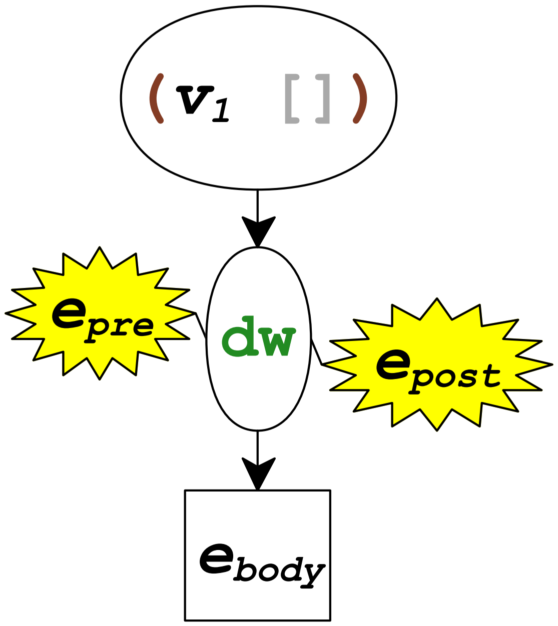
Dynamic Wind



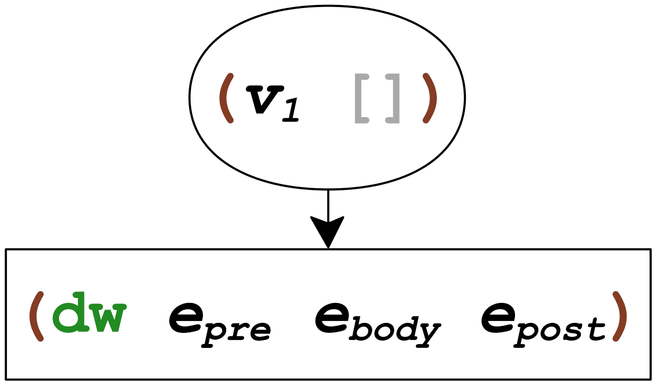
Dynamic Wind



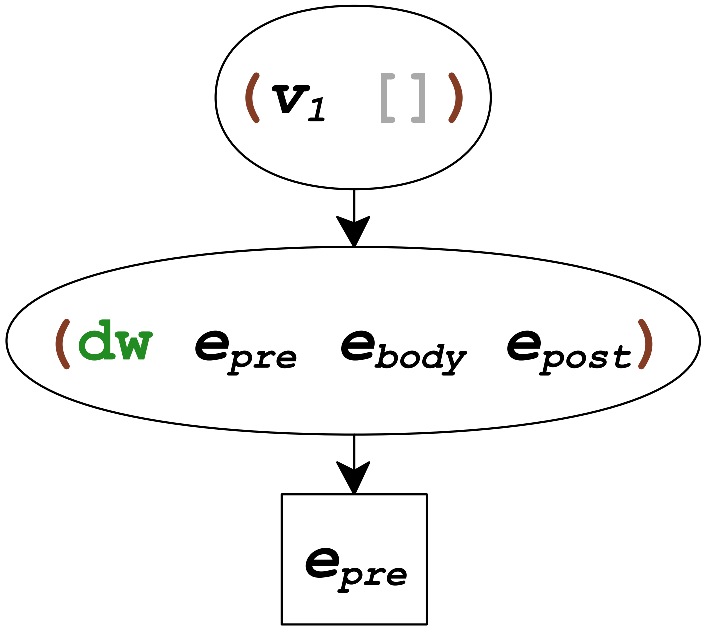
Dynamic Wind



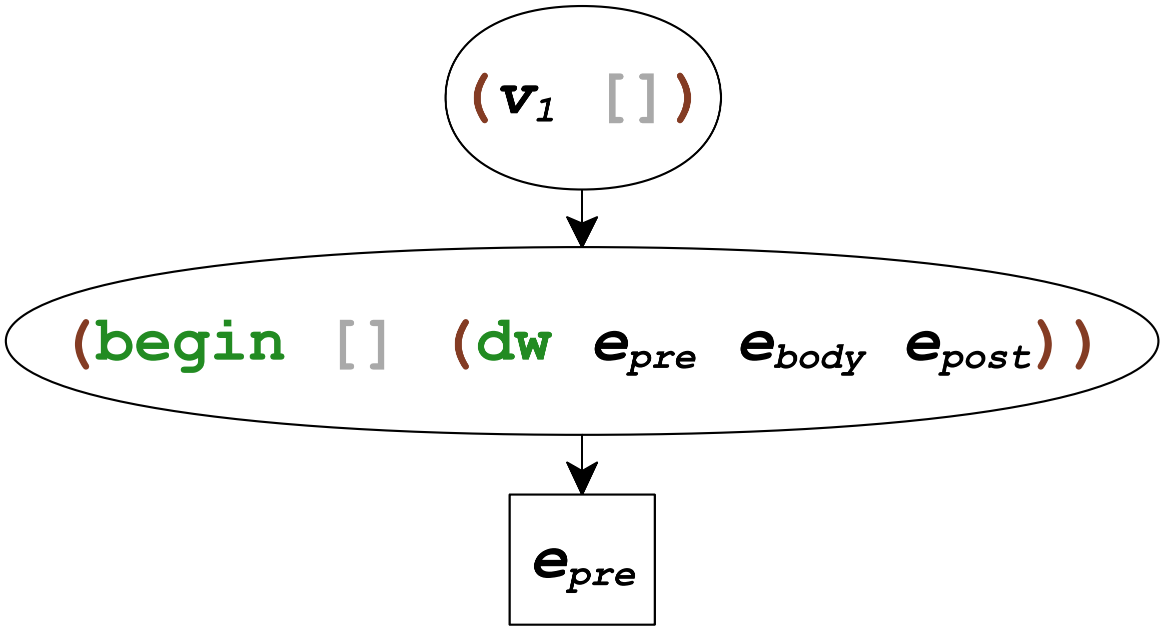
Dynamic Wind Evaluation



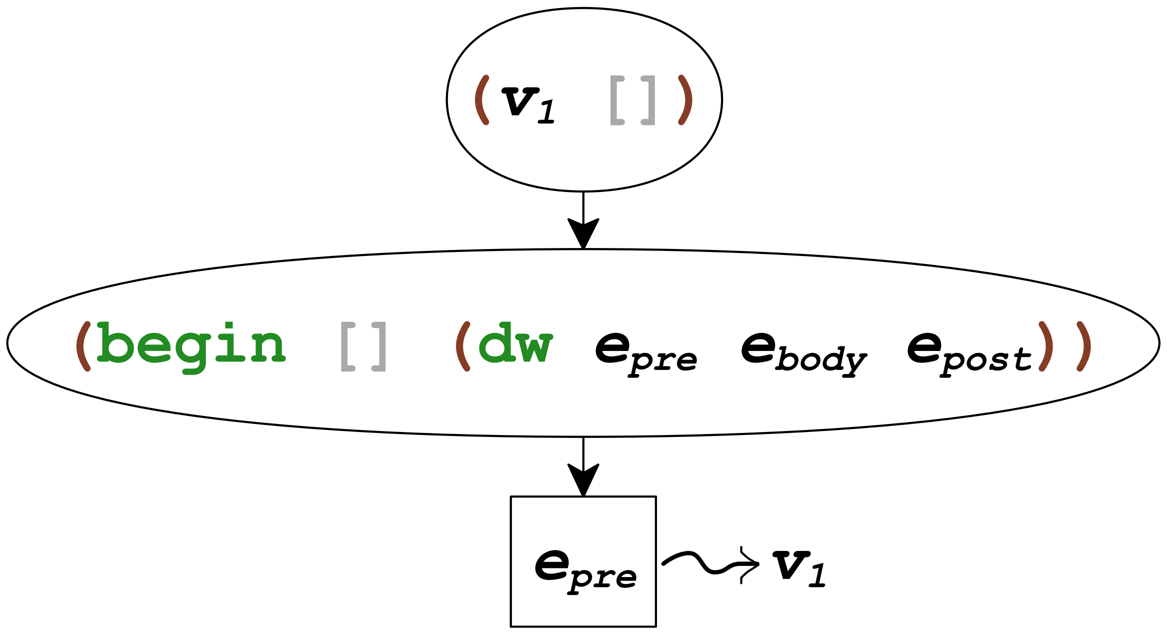
Dynamic Wind Evaluation



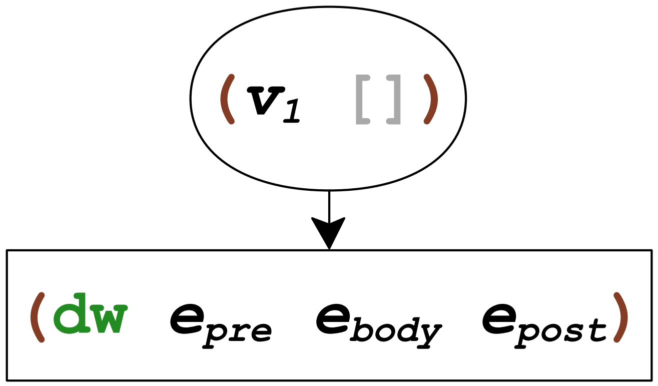
Dynamic Wind Evaluation



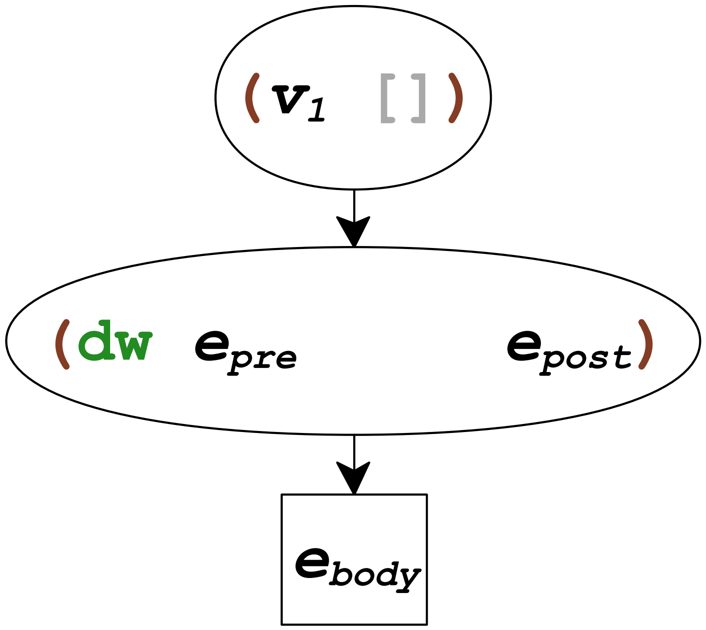
Dynamic Wind Evaluation



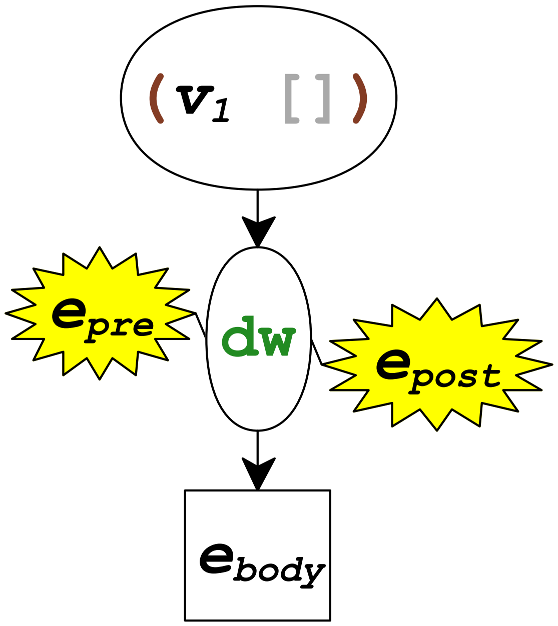
Dynamic Wind Evaluation



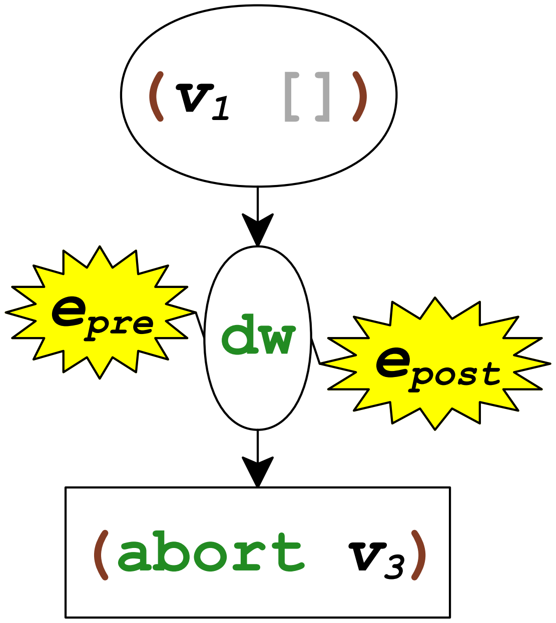
Dynamic Wind Evaluation



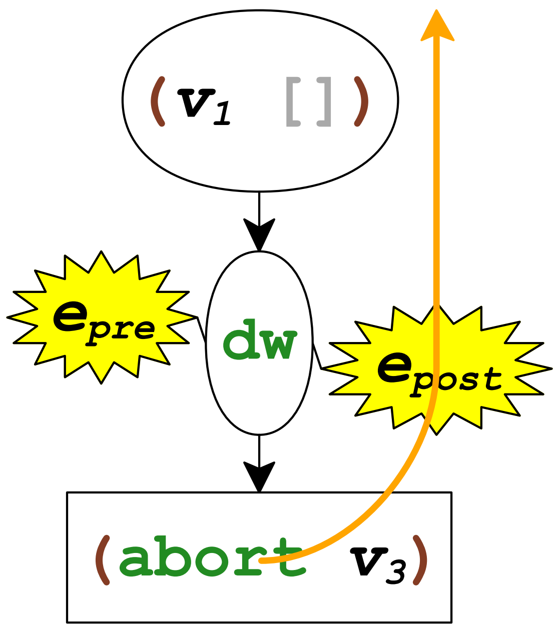
Dynamic Wind and Jumps



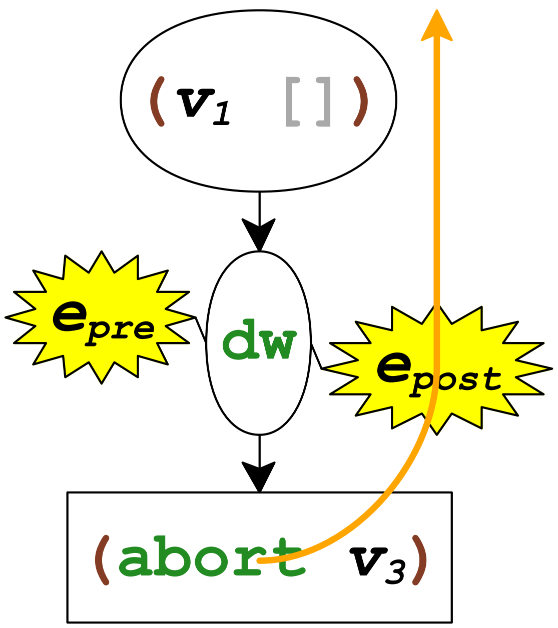
Dynamic Wind and Jumps

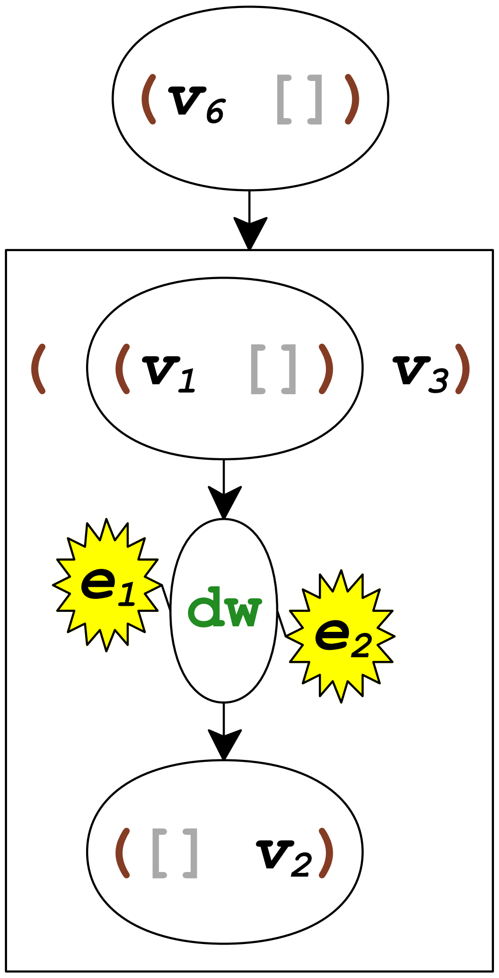


Dynamic Wind and Jumps

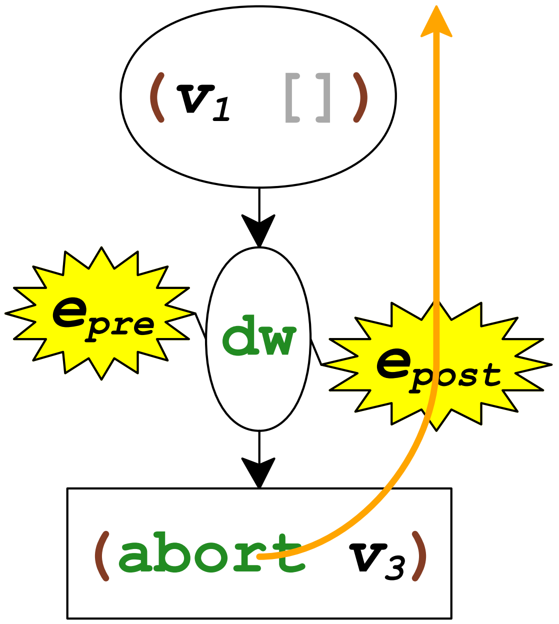


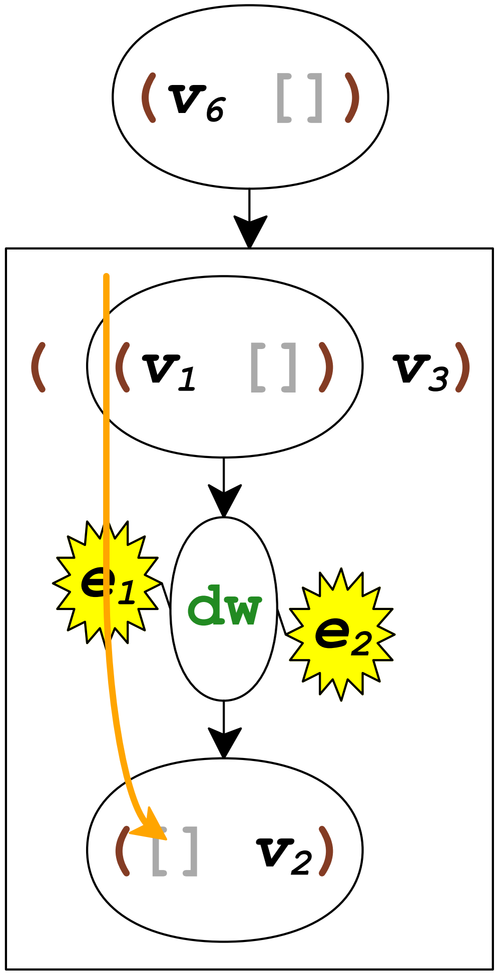
Dynamic Wind and Jumps



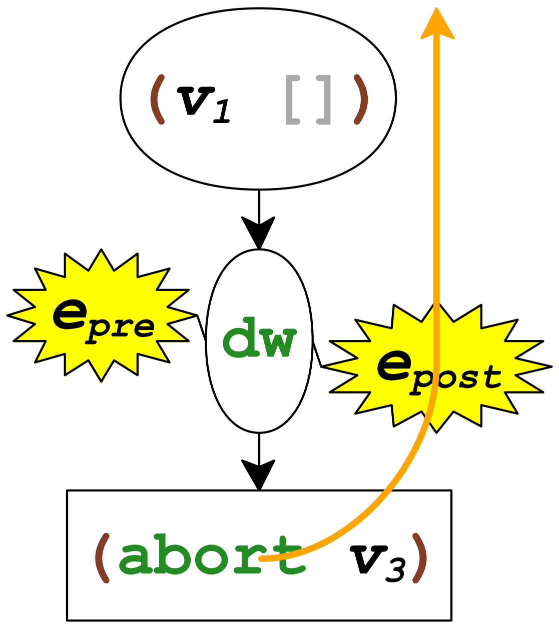


Dynamic Wind and Jumps

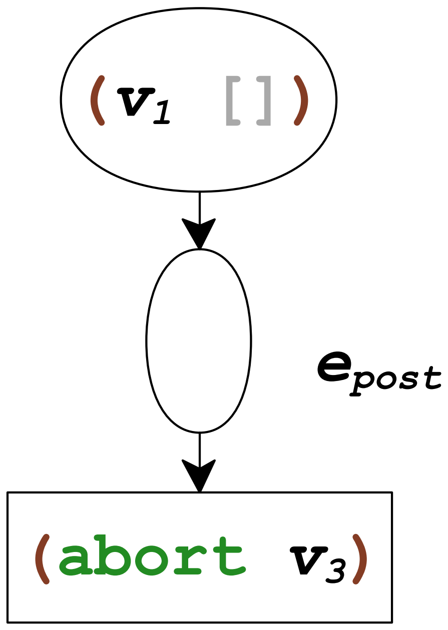




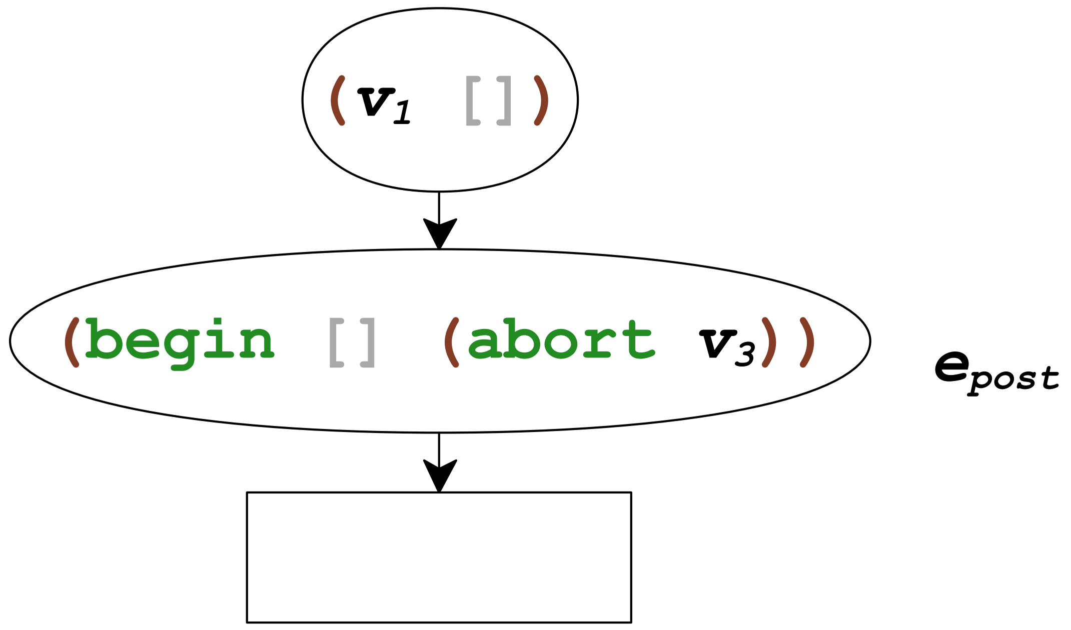
Dynamic Wind and Abort Evaluation



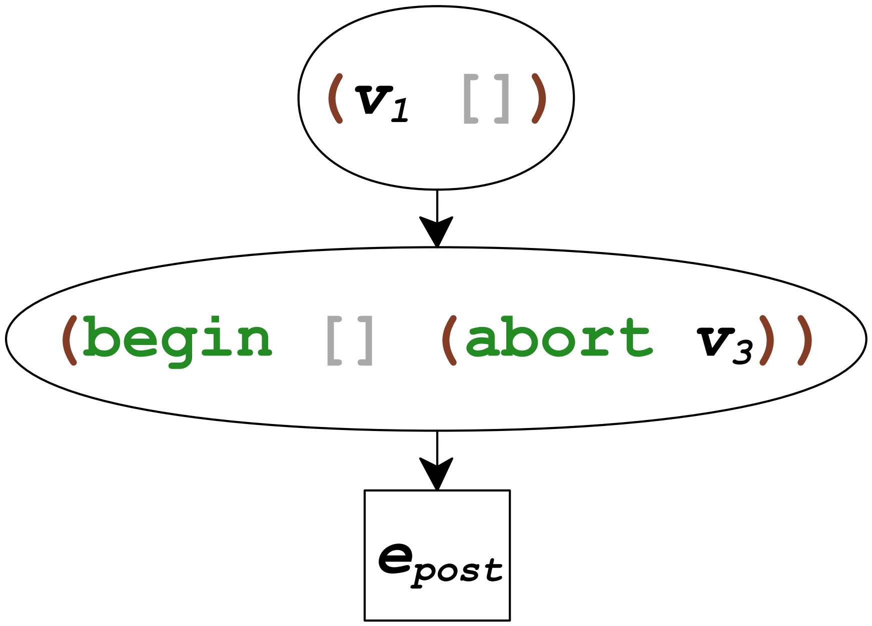
Dynamic Wind and Abort Evaluation



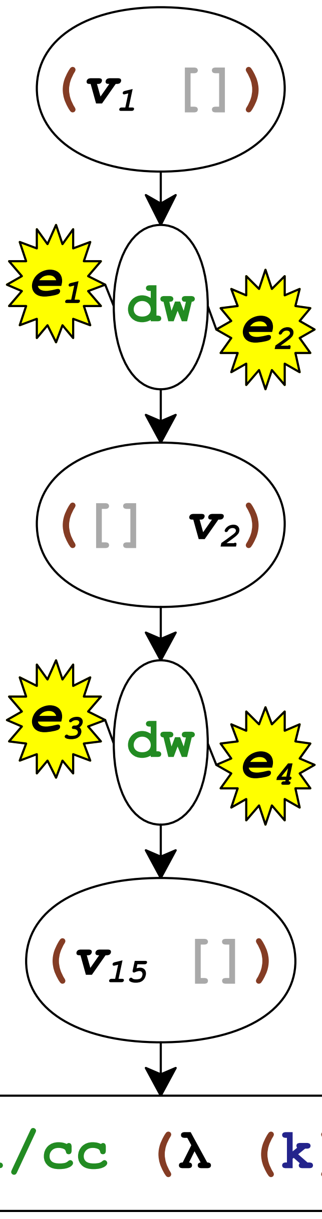
Dynamic Wind and Abort Evaluation



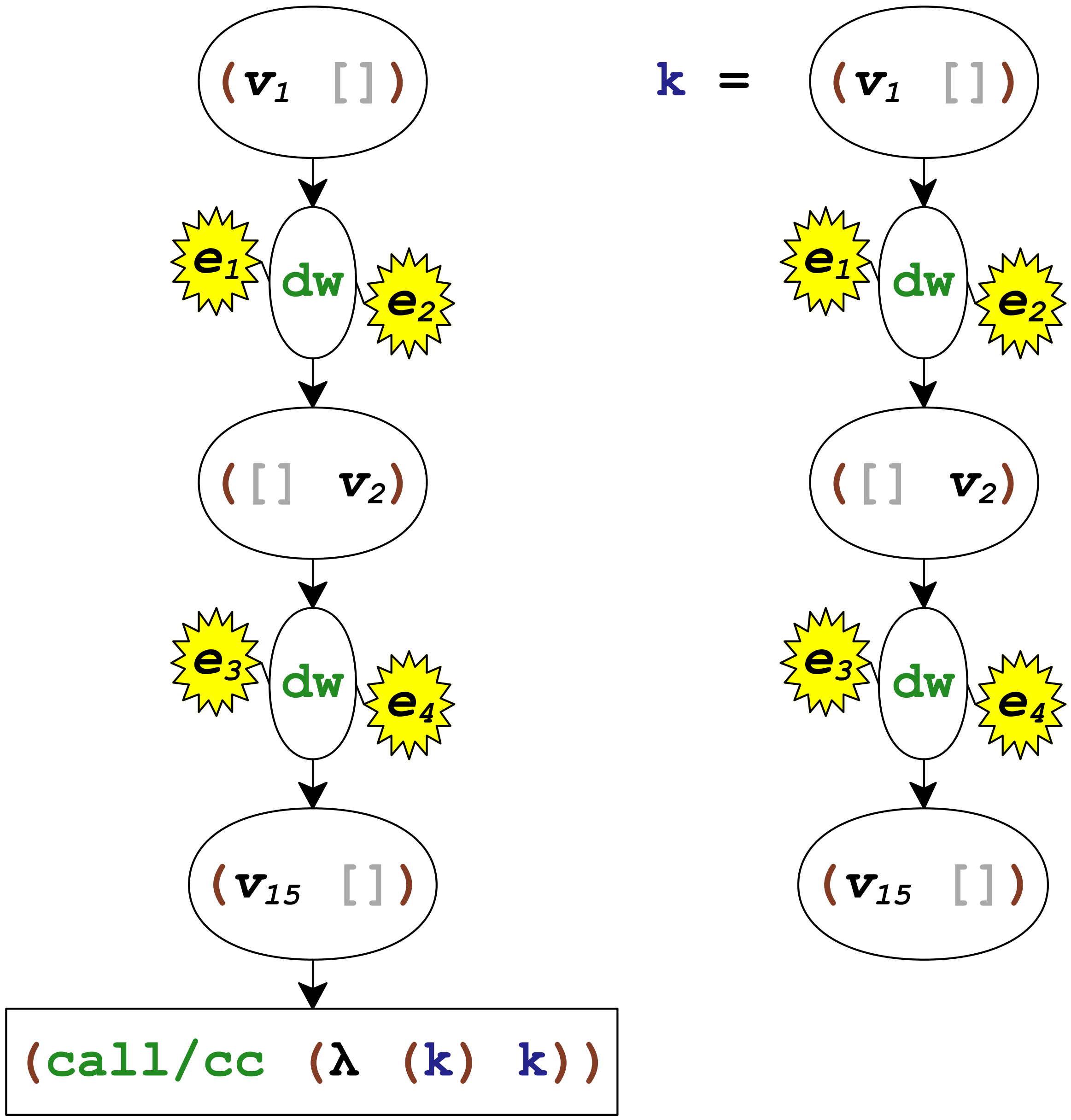
Dynamic Wind and Abort Evaluation



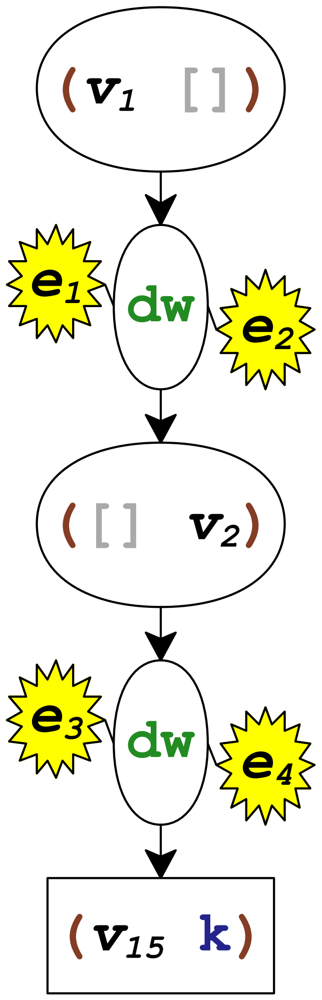
Dynamic Wind and Call/cc

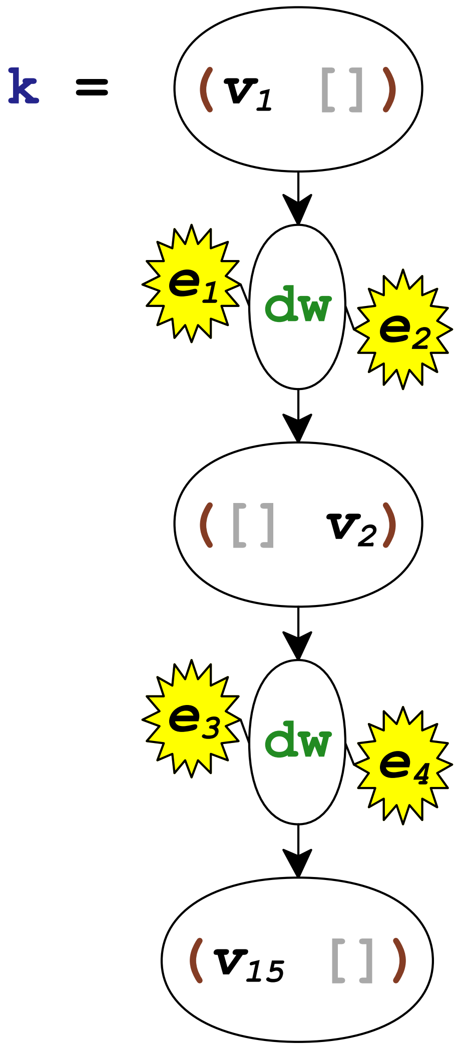


Dynamic Wind and Call/cc

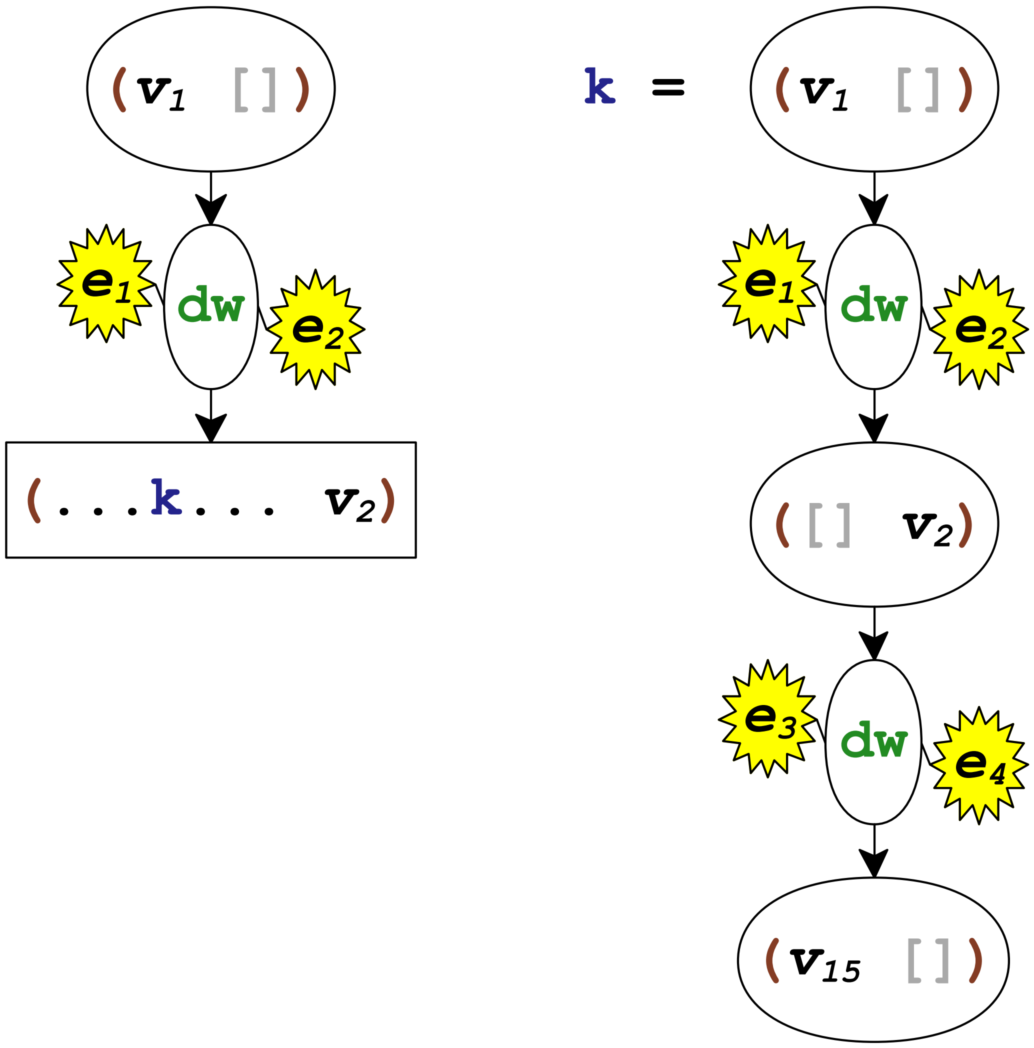


Dynamic Wind and Call/cc

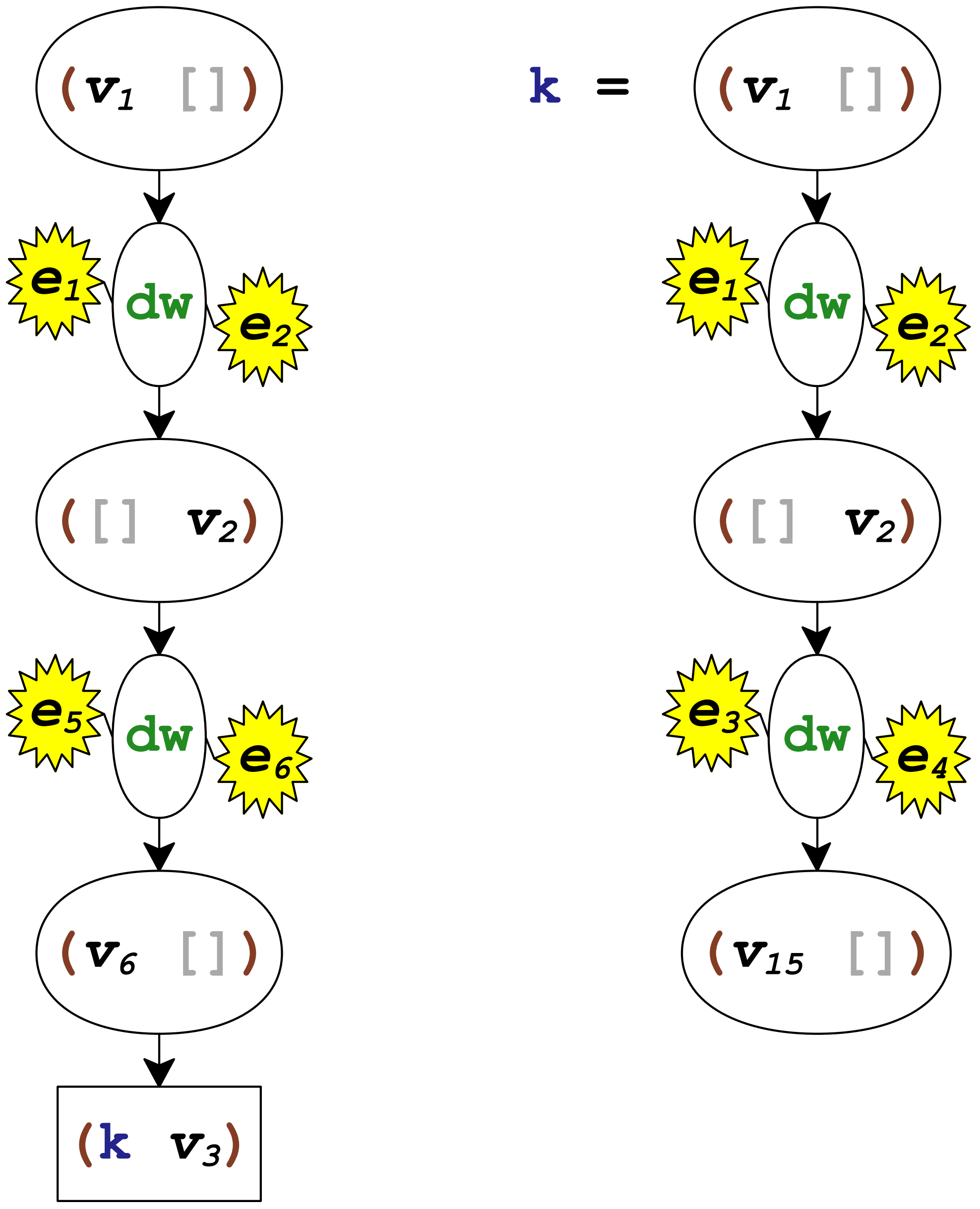




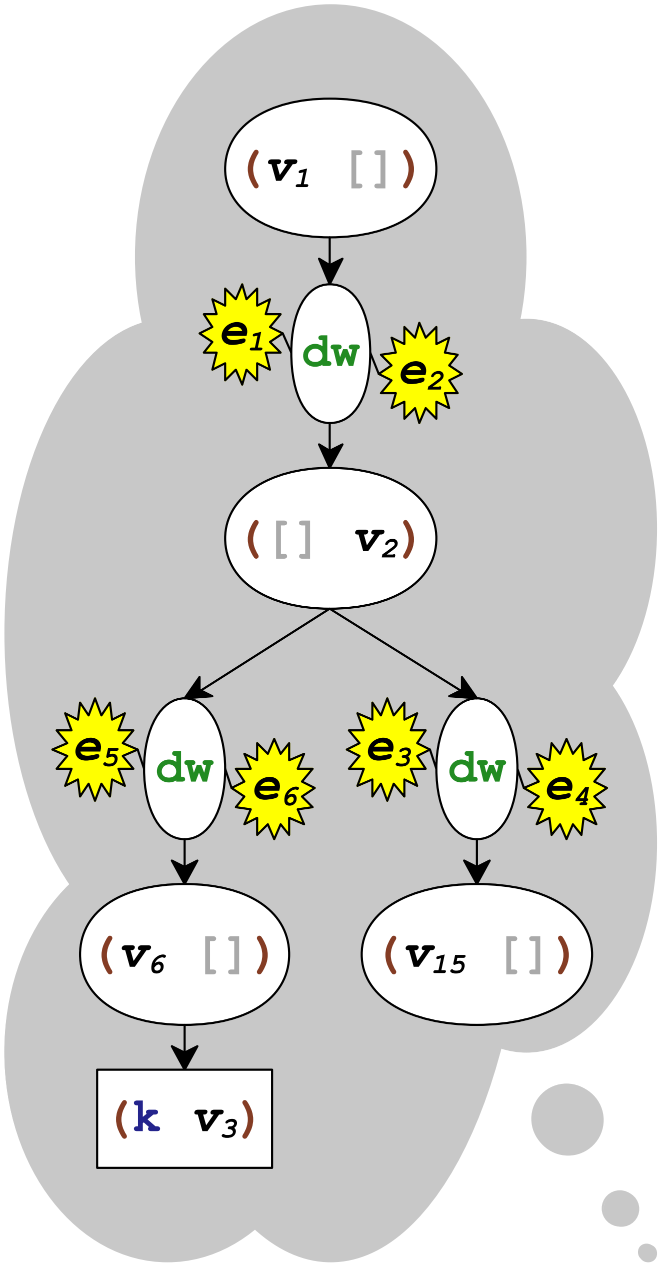
Dynamic Wind and Call/cc

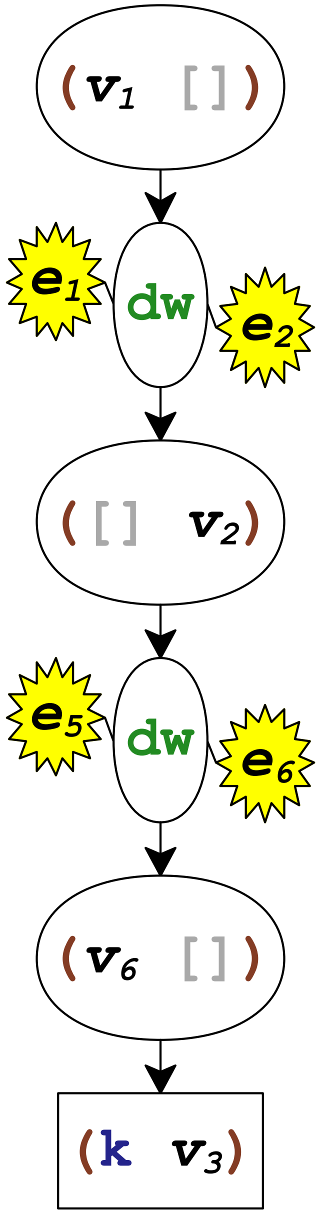


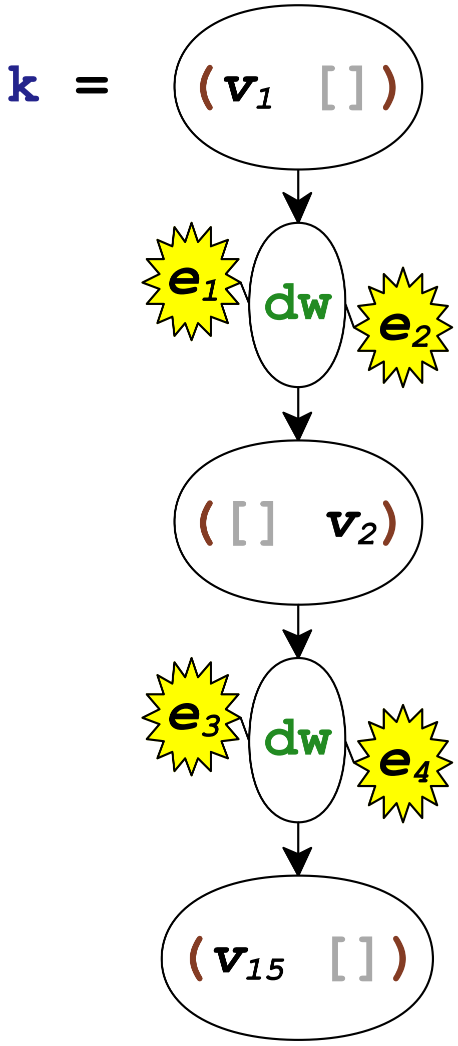
Dynamic Wind and Call/cc



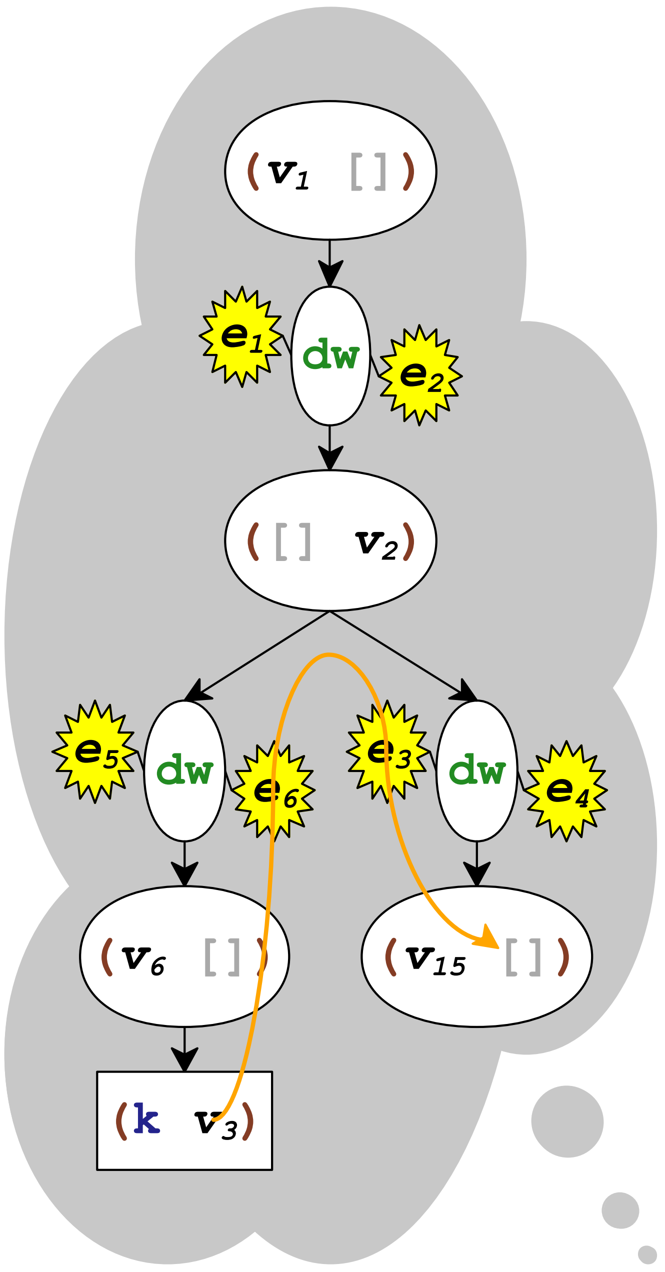
Dynamic Wind and Call/cc

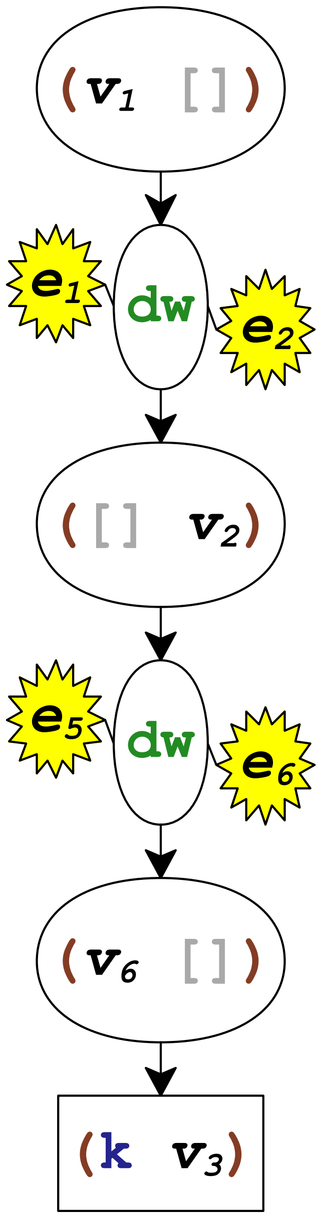


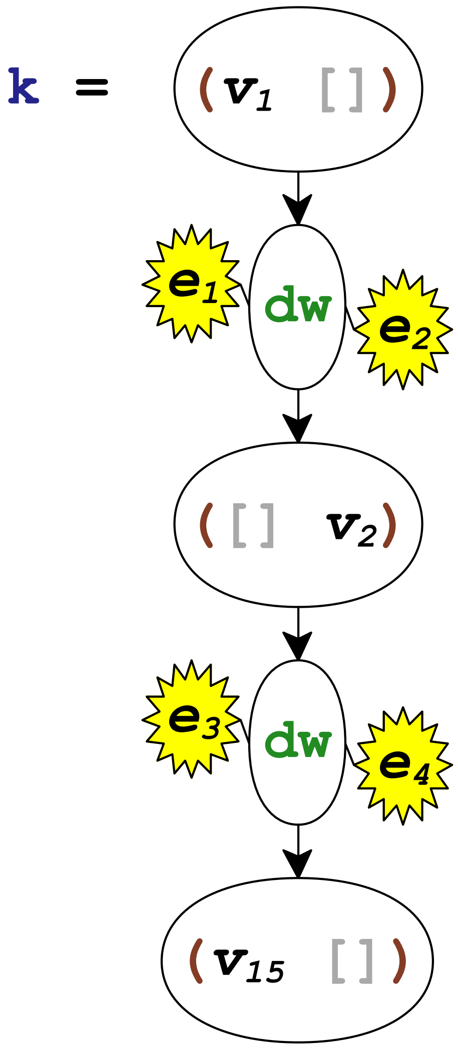




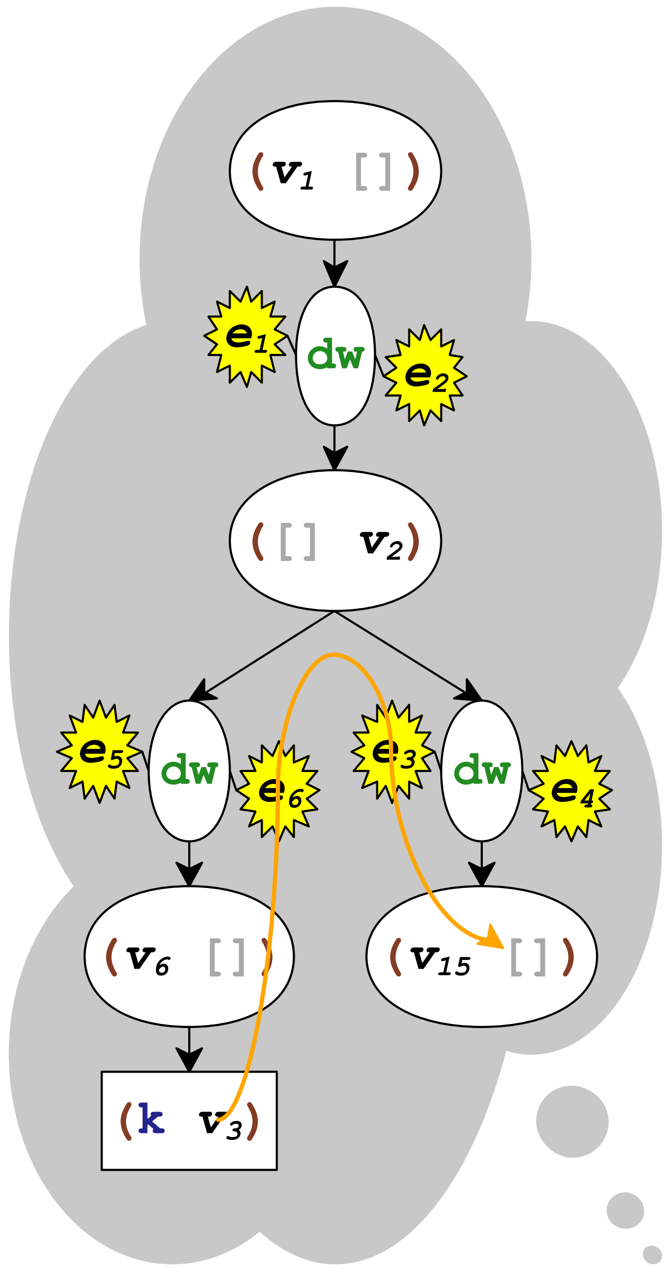
Dynamic Wind and Call/cc

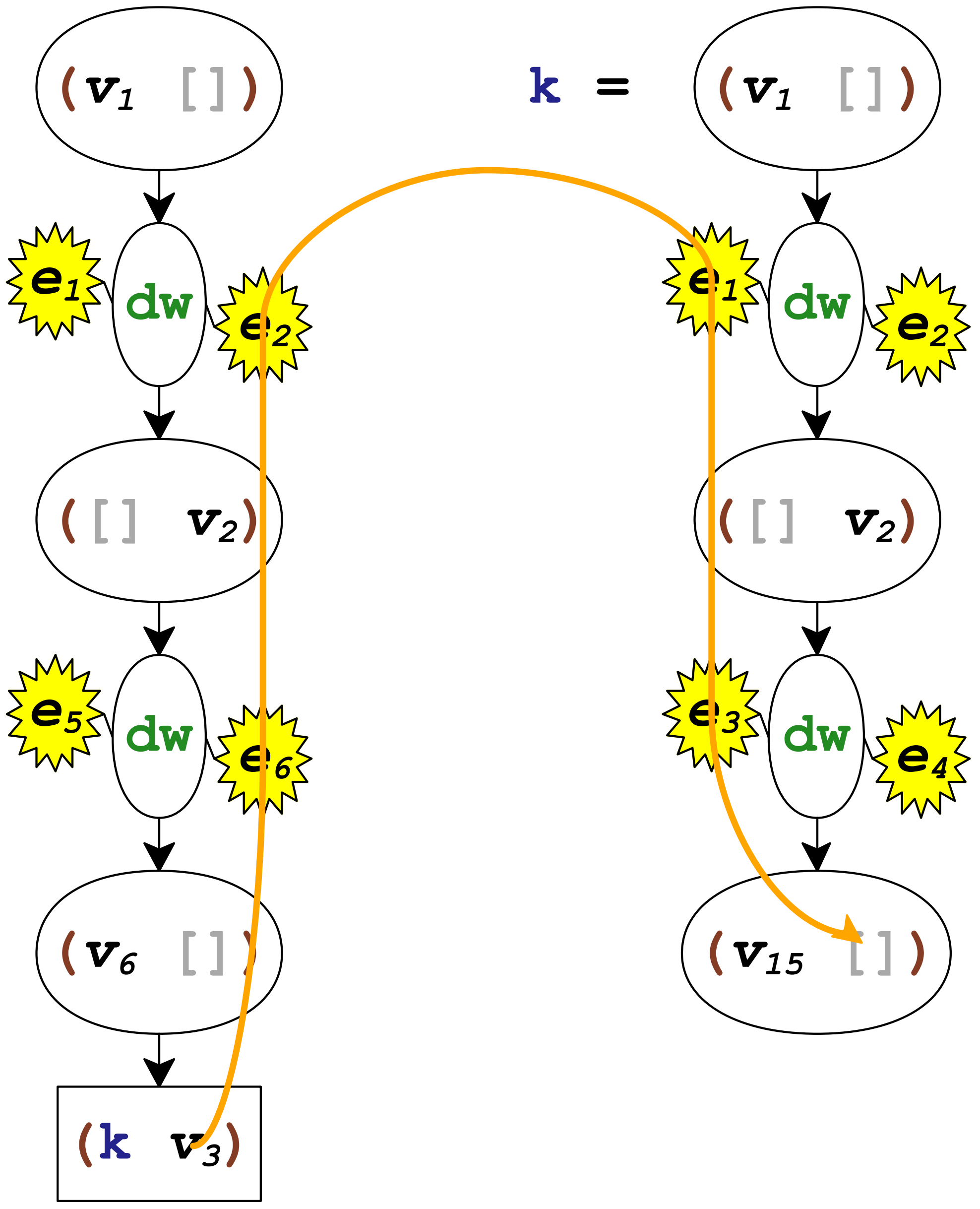




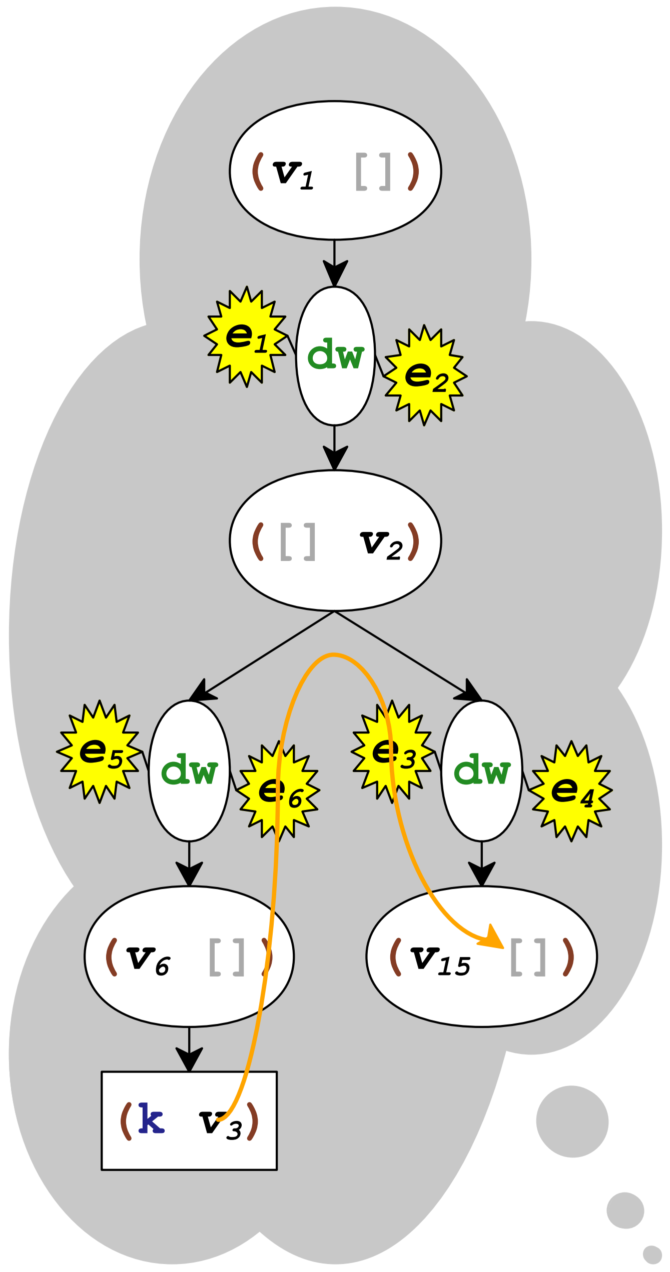


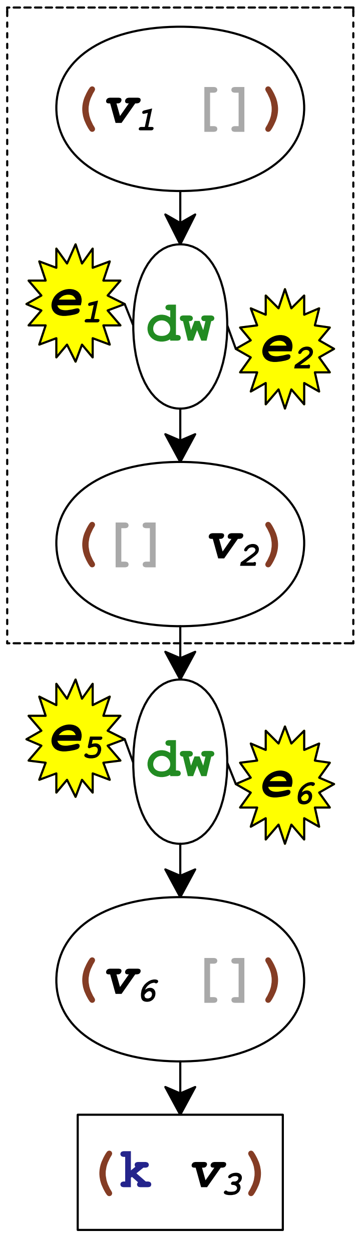
Dynamic Wind and Call/cc

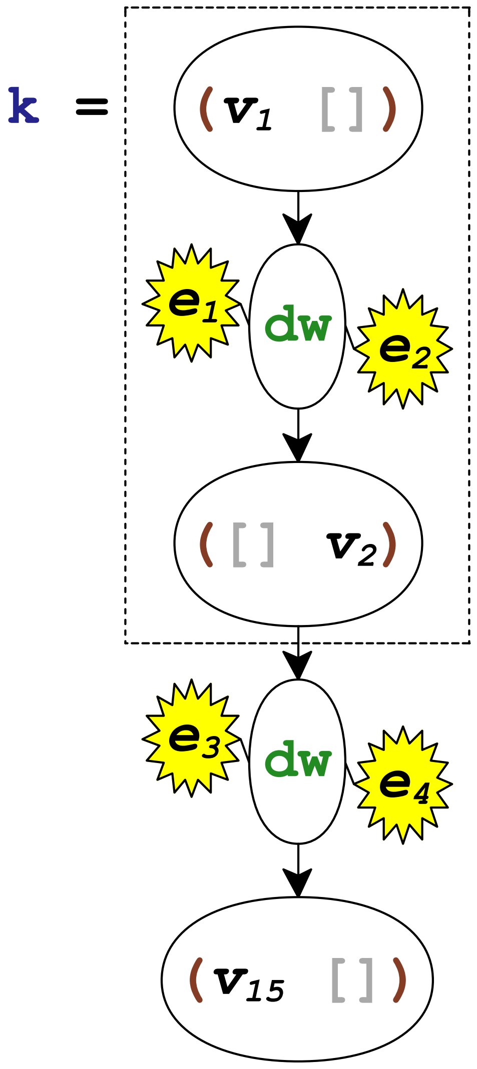




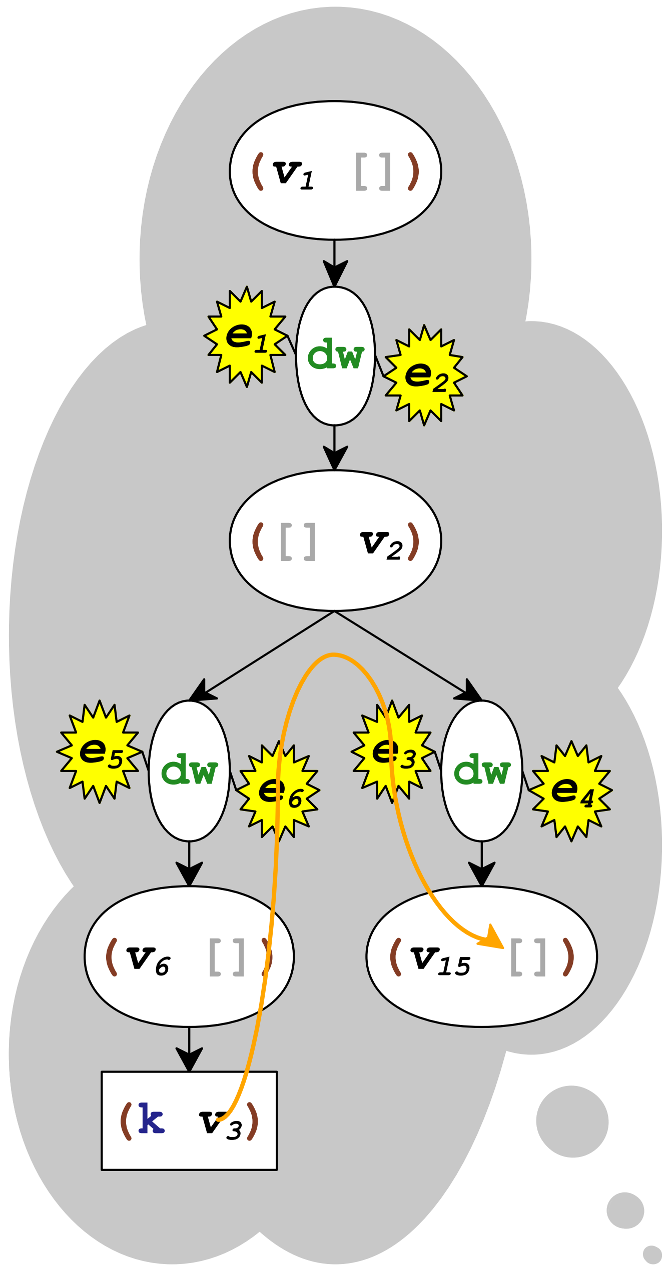
Dynamic Wind and Call/cc

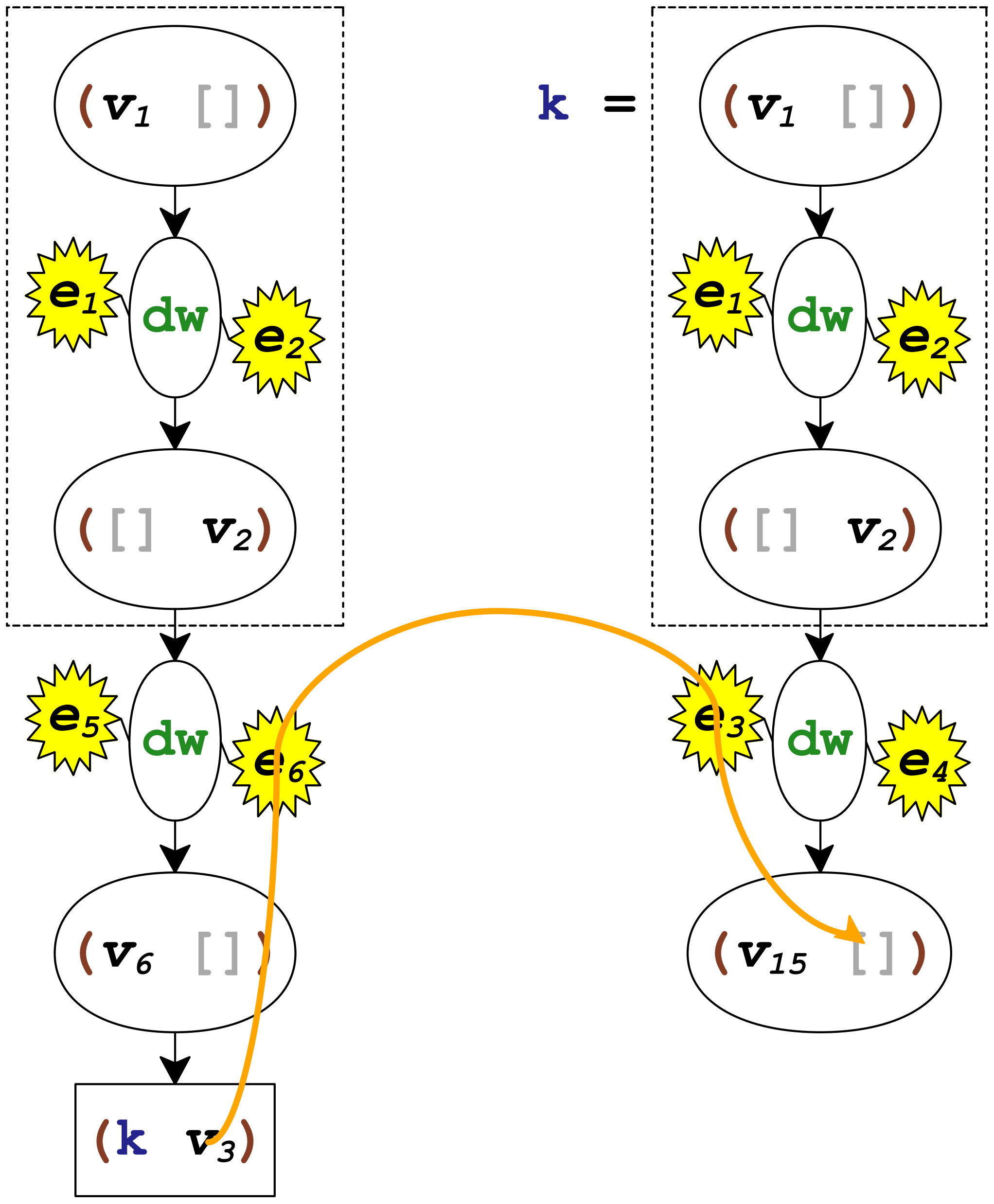




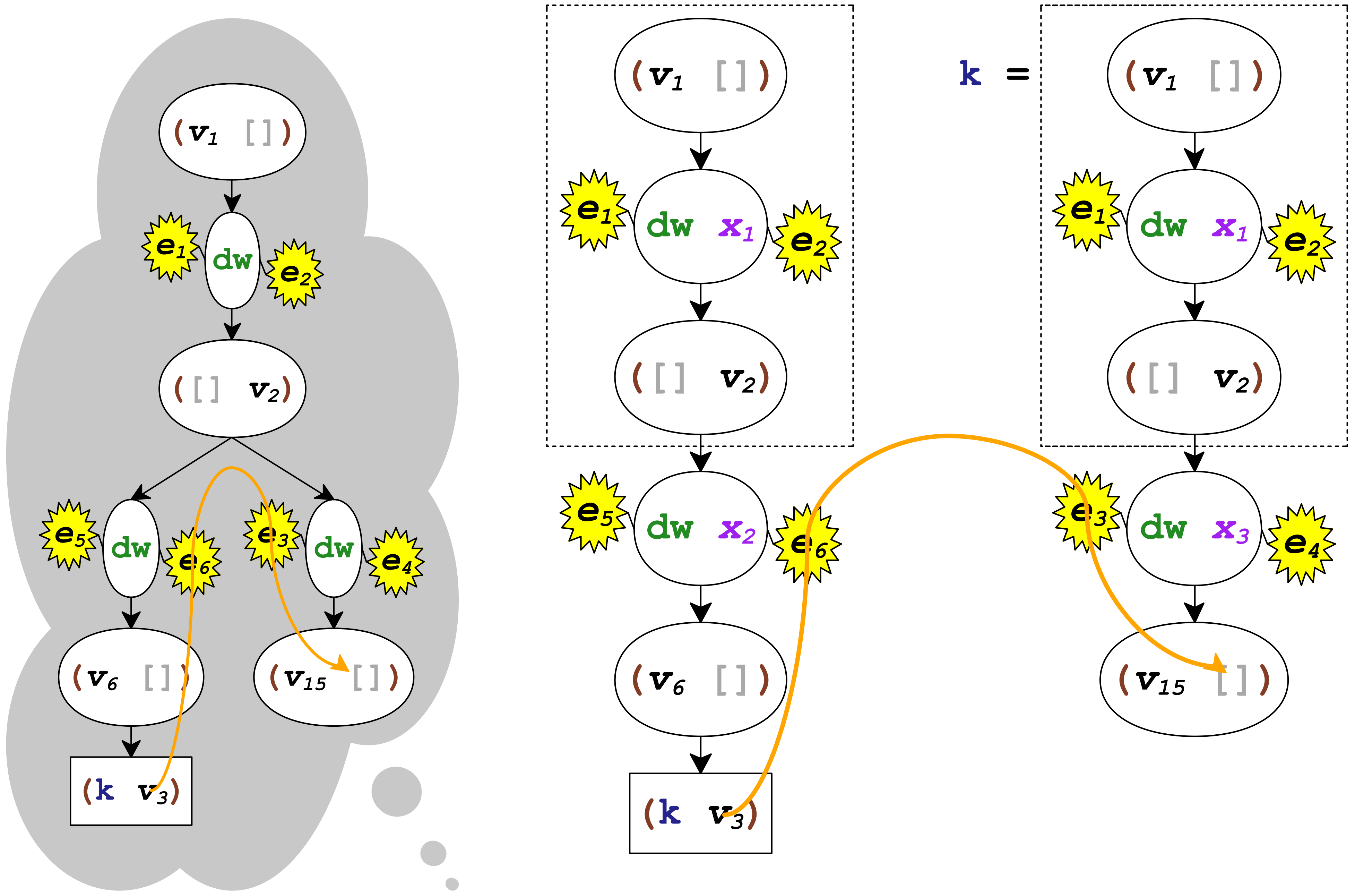


Dynamic Wind and Call/cc

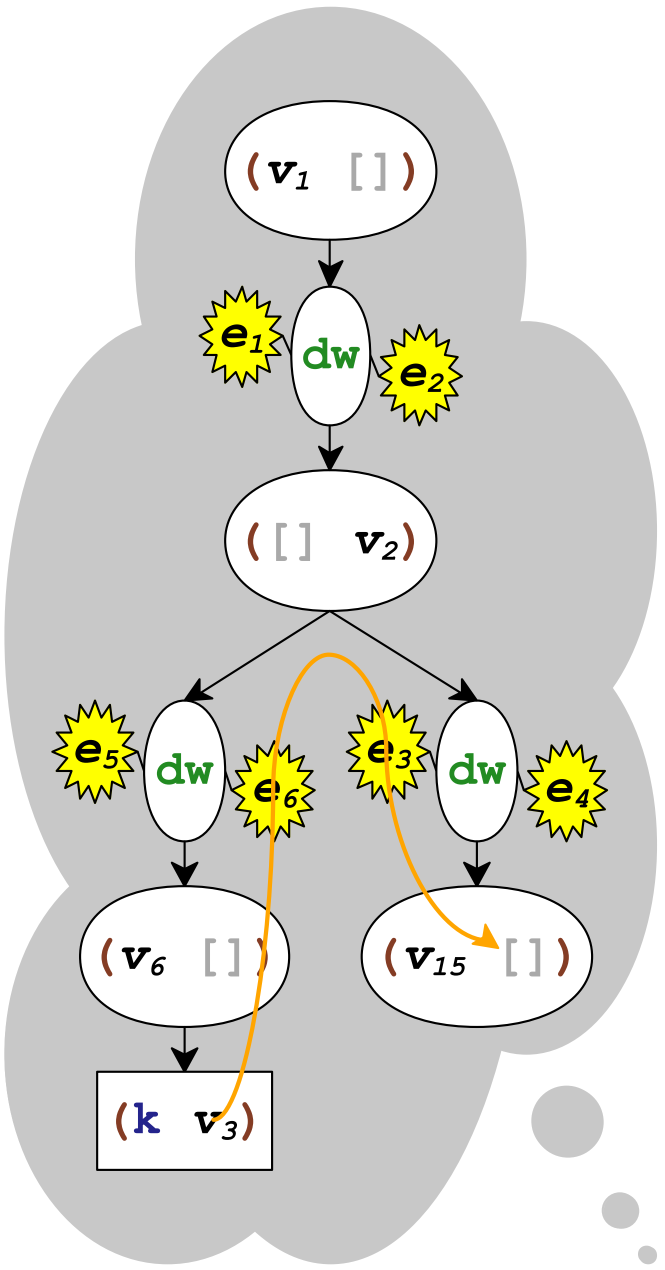


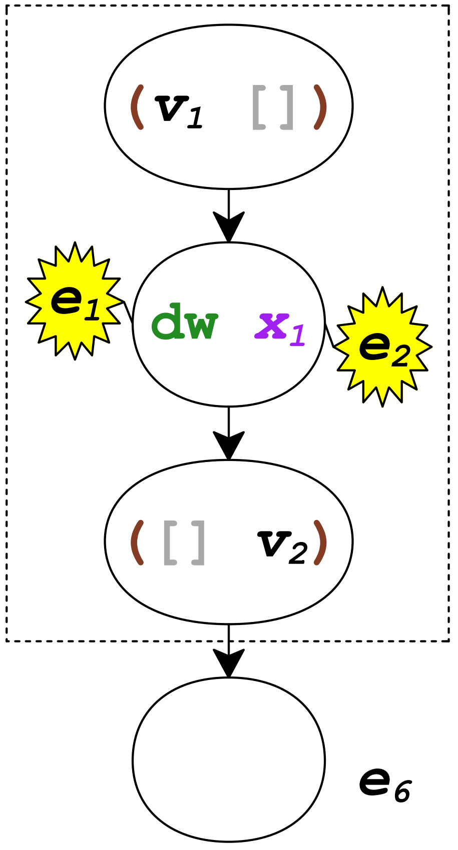


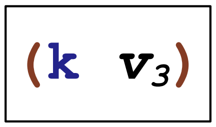
Dynamic Wind and Call/cc

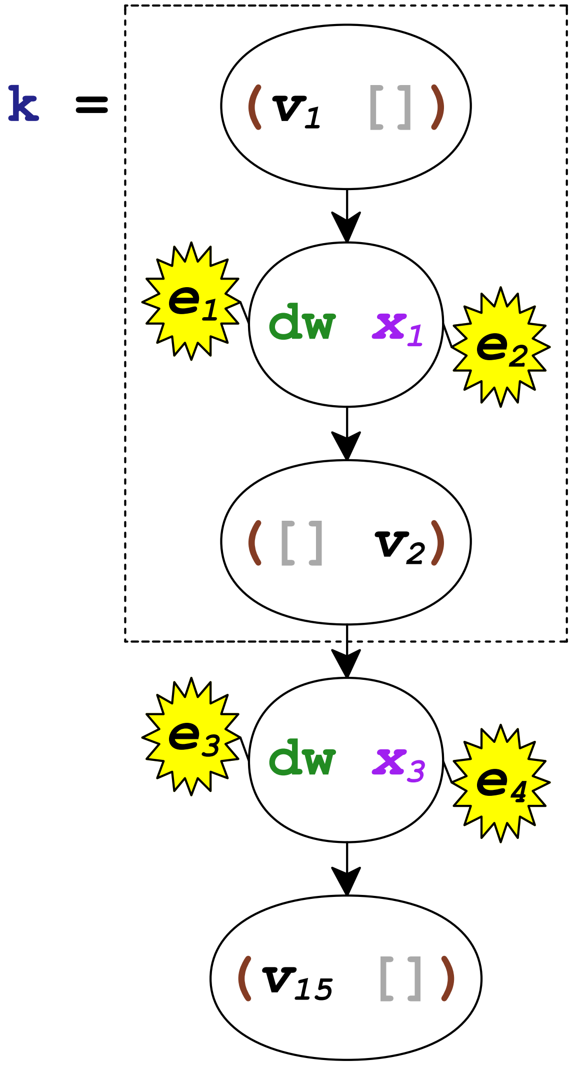


Dynamic Wind and Call/cc

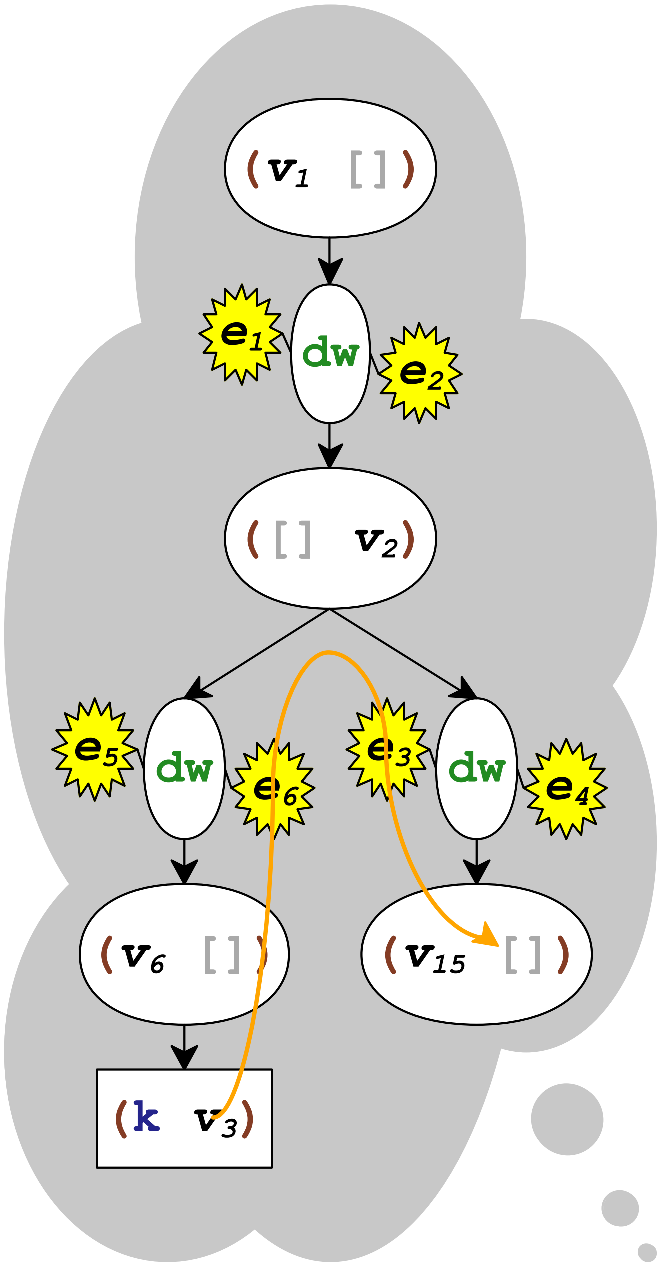


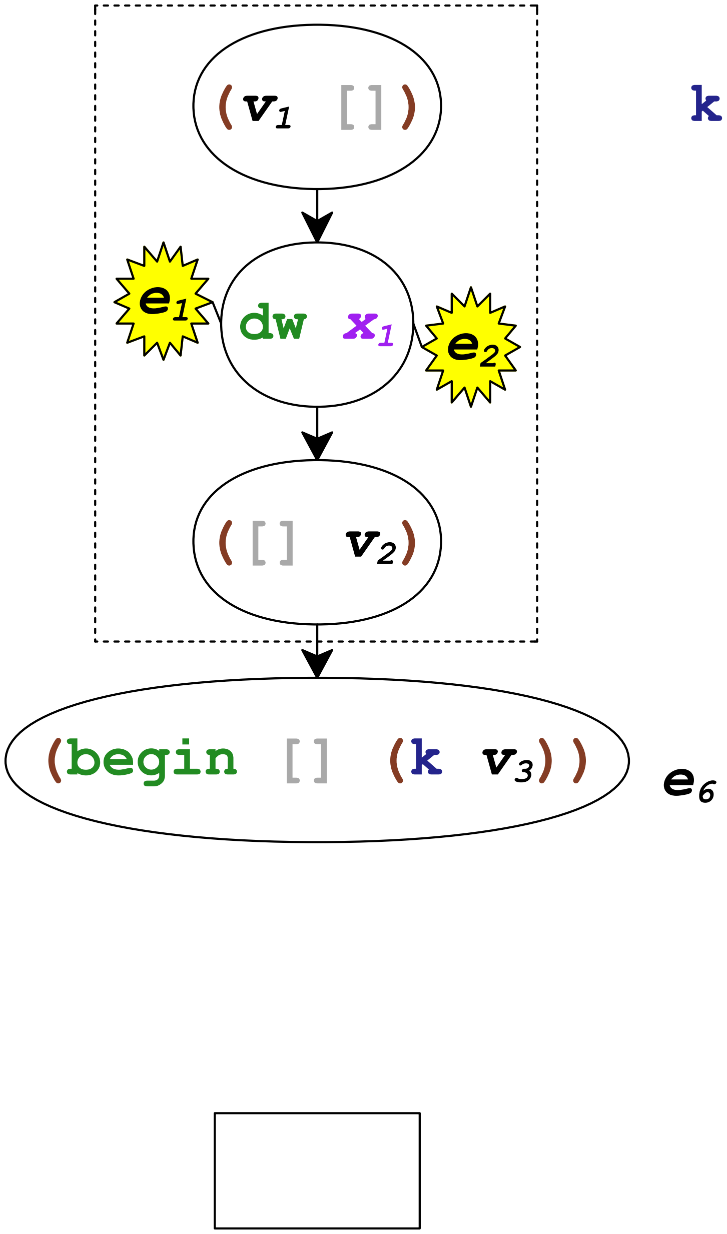


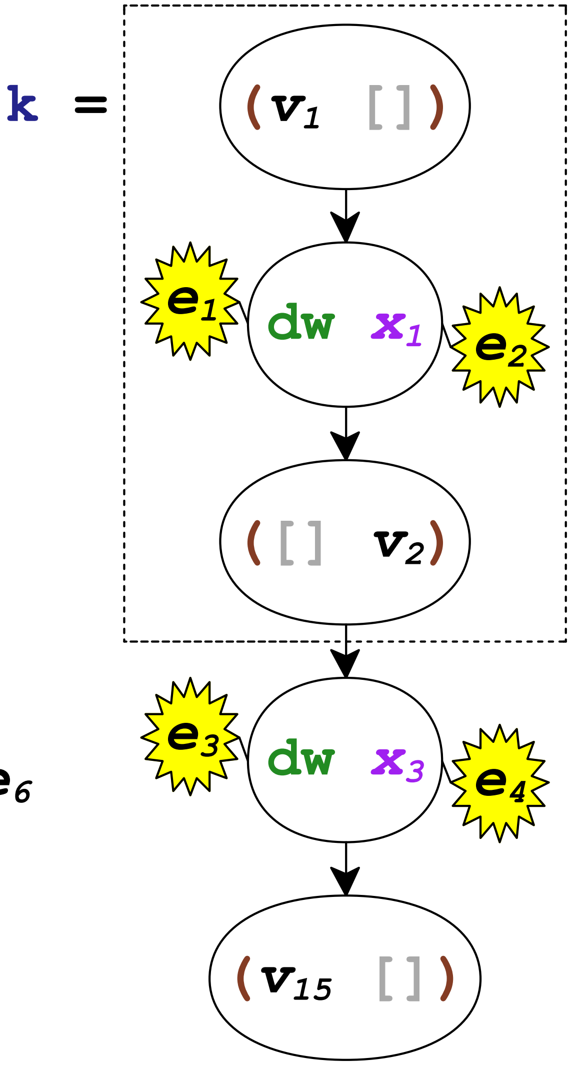




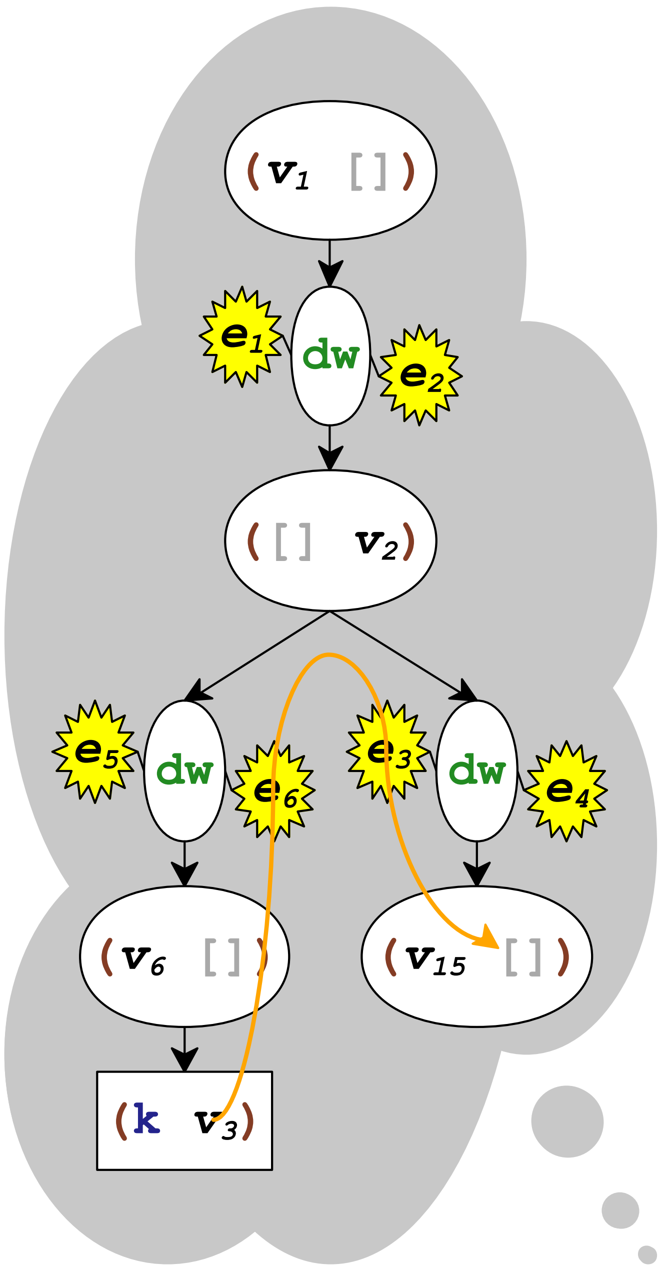
Dynamic Wind and Call/cc

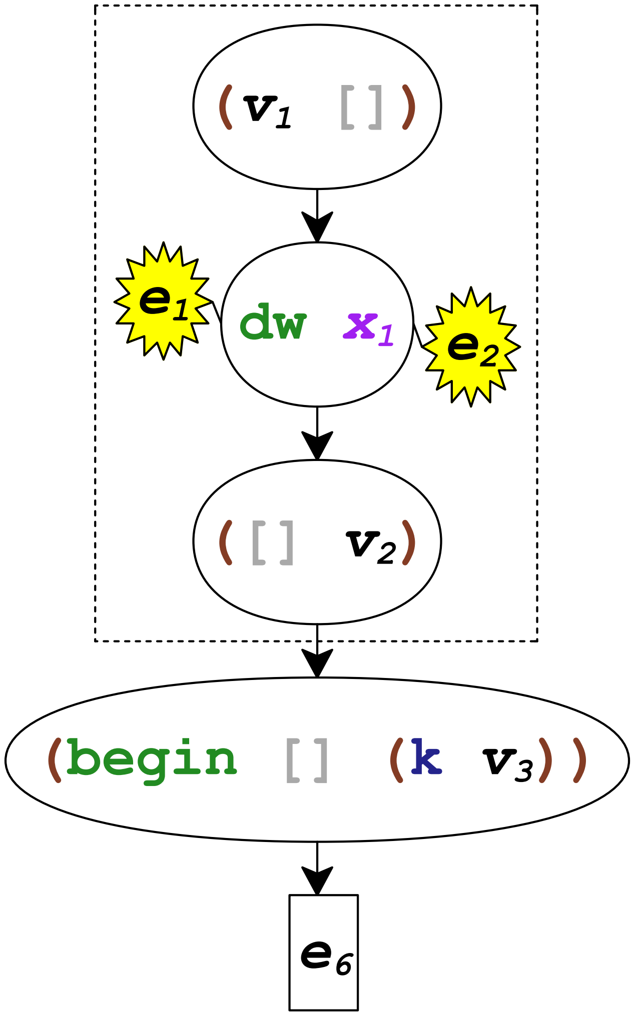


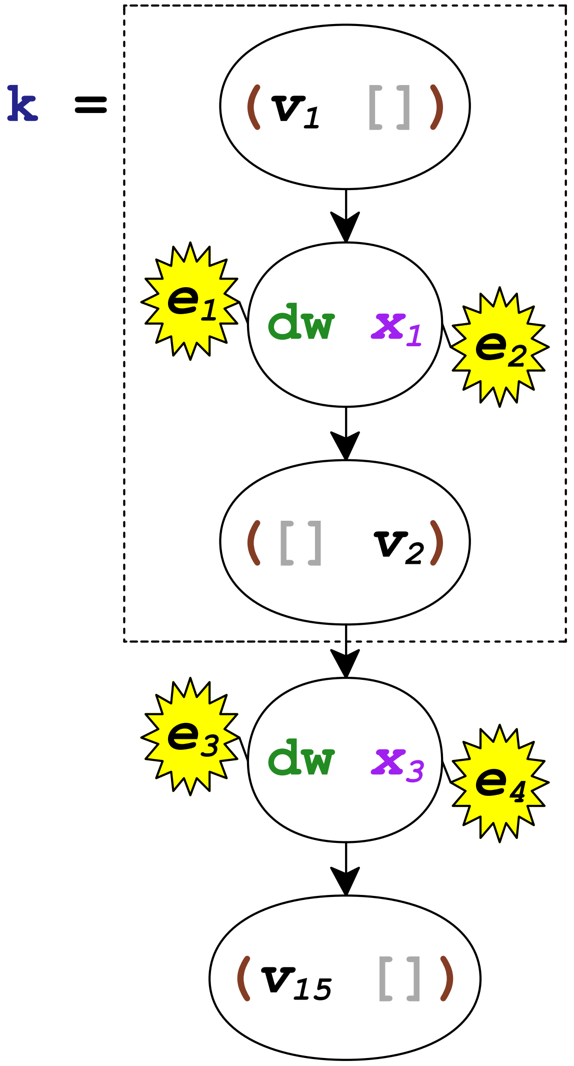




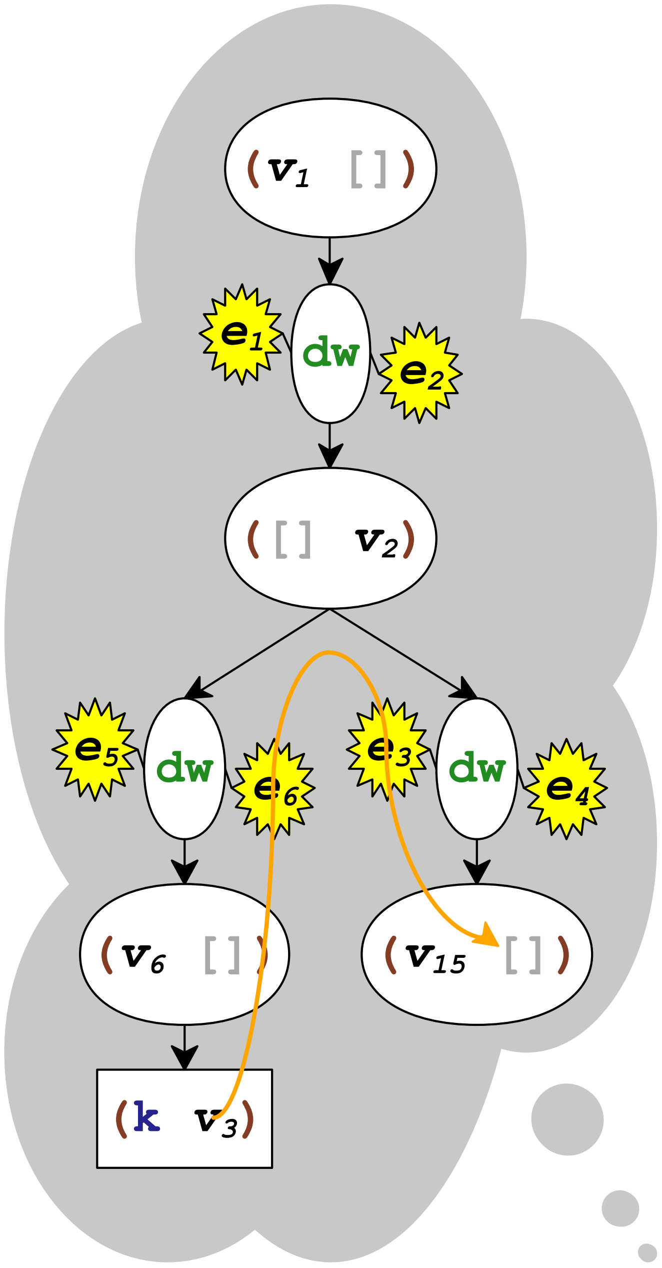
Dynamic Wind and Call/cc

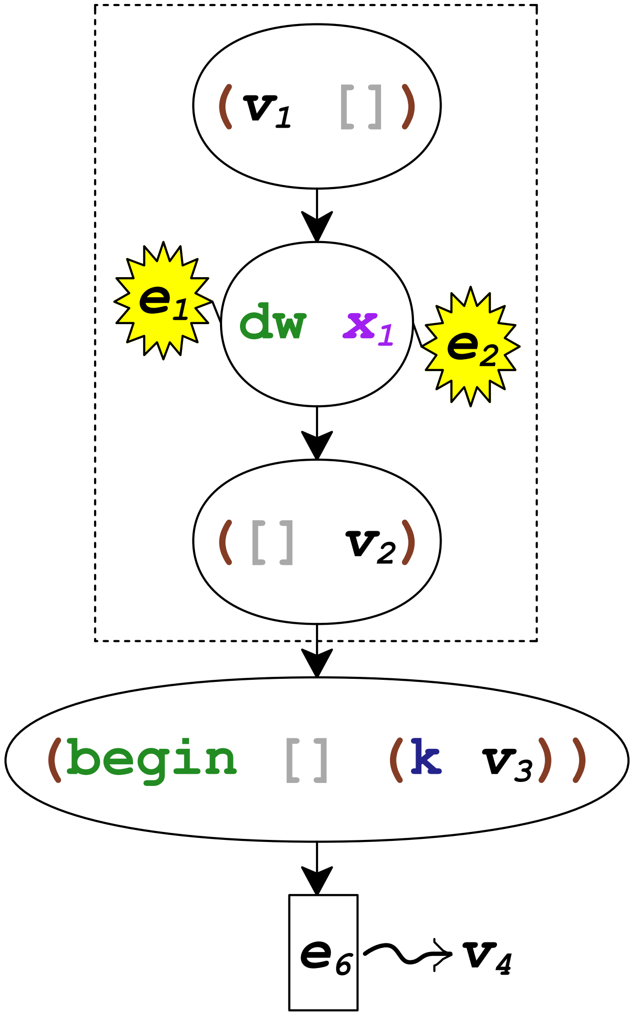


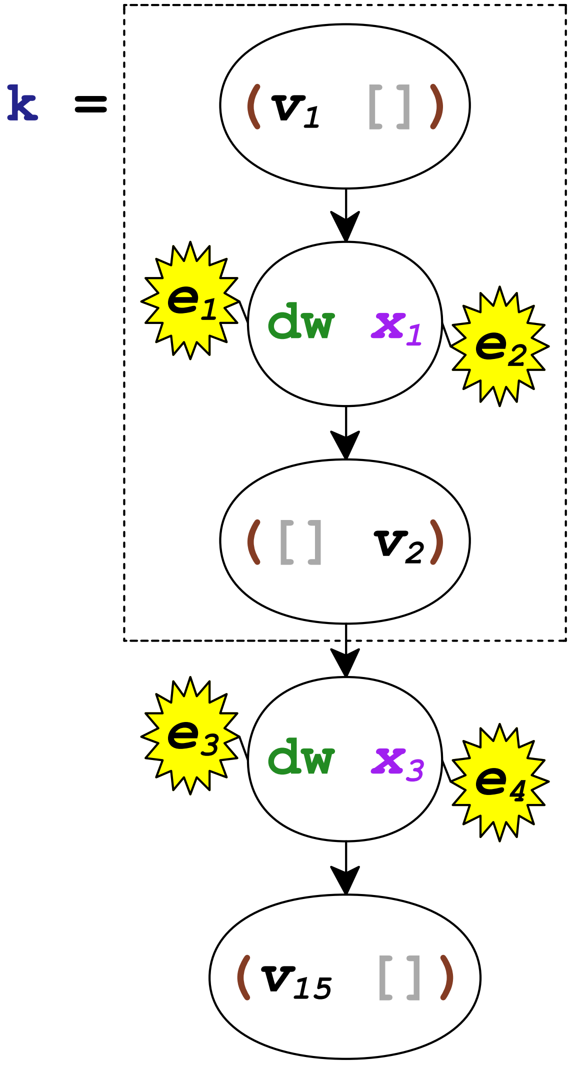




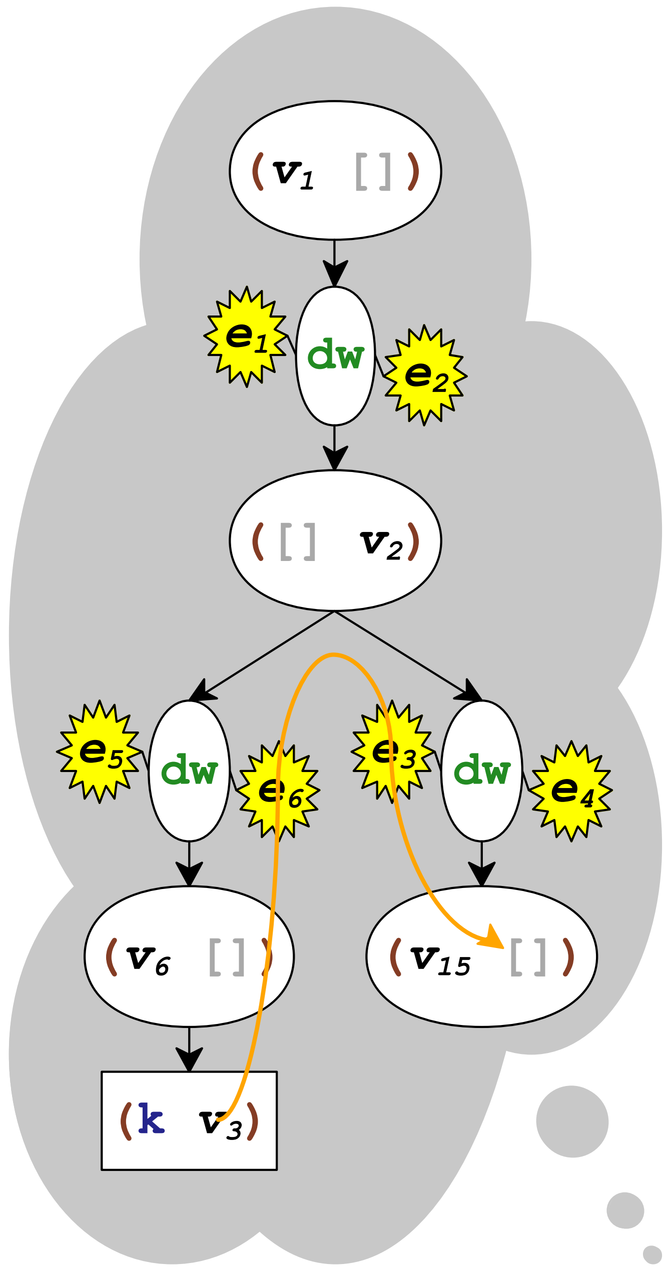
Dynamic Wind and Call/cc

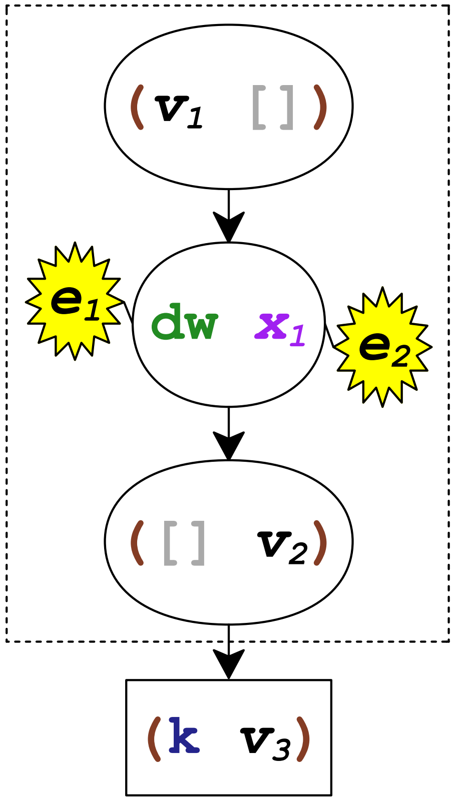


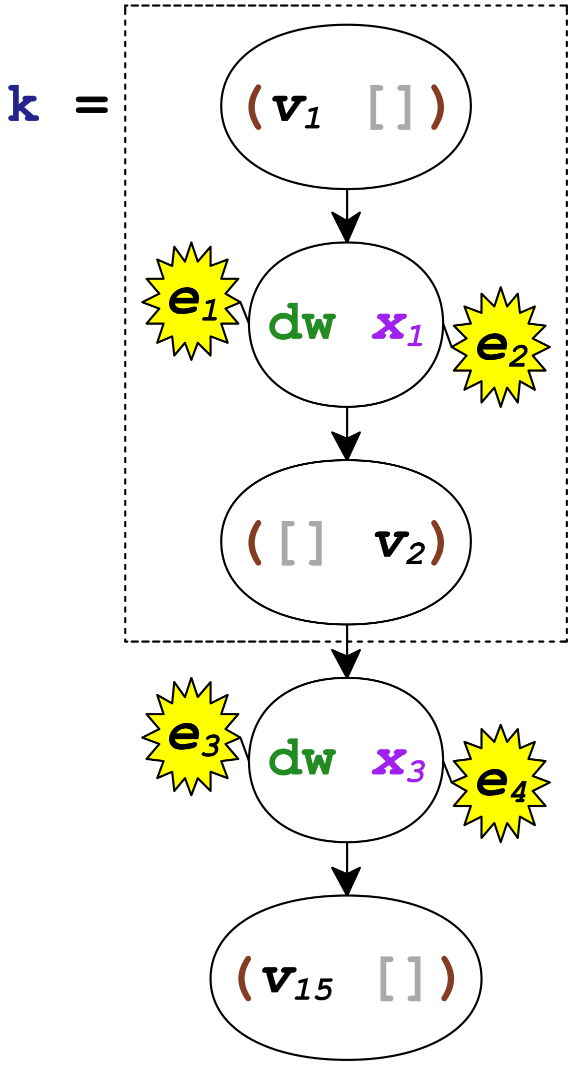




Dynamic Wind and Call/cc







Dynamic Wind Summary

- `dynamic-wind` generates `dw`
- `call/cc` detects sharing in continuation jumps
- capture `dw` thunks in `call/comp` and `call/cc`
- run post thunks in `abort`
- run pre thunks in continuation composition

Model

- Paper:

  ICFP'07 Flatt, Yu, Findler, and Felleisen

- Redex model:

  `http://www.cs.utah.edu/plt/delim-cont/`

Model

$$\begin{aligned}
e &::= m \mid (\mathtt{wcm}\ w\ m)\\
m &::= x \mid v \mid (e\ e\ \ldots) \mid (\mathtt{begin}\ e\ e) \mid (\mathtt{\%}\ e\ e\ e) \mid (\mathtt{dw}\ x\ e\ e\ e)\\
v &::= (\mathtt{list}\ v\ \ldots) \mid (\lambda\ (x\ \ldots)\ e) \mid (\mathtt{cont}\ v\ E) \mid (\mathtt{comp}\ E)\\
&\quad \mid \mathtt{dynamic\text{-}wind} \mid \mathtt{abort} \mid \mathtt{current\text{-}marks}\\
&\quad \mid \mathtt{cons} \mid u\\
u &::= \mathtt{call/cc} \mid \mathtt{call/comp} \mid \mathtt{call/cm}\\
w &::= ((v\ v)\ \ldots)\\
E &::= W \mid W[(\mathtt{dw}\ x\ e\ E\ e)]\\
W &::= M \mid (\mathtt{wcm}\ w\ M)\\
M &::= \mathtt{[\,]} \mid (v\ \ldots\ W\ e\ \ldots) \mid (\mathtt{begin}\ W\ e) \mid (\mathtt{\%}\ W\ e\ e) \mid (\mathtt{\%}\ v\ e\ W) \mid (\mathtt{\%}\ v\ W\ v)\\
D &::= \mathtt{[\,]} \mid E[(\mathtt{dw}\ x\ e\ \mathtt{[\,]}\ e)]
\end{aligned}$$

$\textsc{same}\mathrm{DW}\textsc{s} : E \times E \to \mathit{bool}$

$$\begin{aligned}
&\textsc{same}\mathrm{DW}\textsc{s}(W_1,\ W_2) &&= \text{true}\\
&\textsc{same}\mathrm{DW}\textsc{s}(W_1[(\mathtt{dw}\ x_1\ e_1\ E_1\ e_2)], &&= \textsc{same}\mathrm{DW}\textsc{s}(E_1,\ E_2)\\
&\qquad W_2[(\mathtt{dw}\ x_1\ e_1\ E_2\ e_2)]) &&\\
&\textsc{same}\mathrm{DW}\textsc{s}(\mathit{any}_1,\ \mathit{any}_2) &&= \text{false}
\end{aligned}$$

$\textsc{no}\mathrm{S}\textsc{hared} : E \times E \to \mathit{bool}$

$$\begin{aligned}
&\textsc{no}\mathrm{S}\textsc{hared}(W_1[(\mathtt{dw}\ x_1\ e_1\ E_1\ e_2)], &&= \text{false}\\
&\qquad W_2[(\mathtt{dw}\ x_1\ e_1\ E_2\ e_2)]) &&\\
&\textsc{no}\mathrm{S}\textsc{hared}(\mathit{any}_1,\ \mathit{any}_2) &&= \text{true}
\end{aligned}$$

$[\![\,\bullet\,]\!]_{\mathrm{wcm}} : e \to e$

$$\begin{aligned}
&[\![e_1]\!]_{\mathrm{wcm}} = e_1\\
&\text{where } e_1 \neq (\mathtt{wcm}\ w\ e)\\
&[\![(\mathtt{wcm}\ ()\ e_1)]\!]_{\mathrm{wcm}} = e_1\\
&[\![(\mathtt{wcm}\ ((v_1\ v_2)\ (v_3\ v_4)\ \ldots)\ e_1)]\!]_{\mathrm{wcm}}\\
&= (\mathtt{call/cm}\ v_1\ v_2\ (\lambda\ ()\ [\![(\mathtt{wcm}\ ((v_3\ v_4)\ \ldots)\ e_1)]\!]_{\mathrm{wcm}}))
\end{aligned}$$

$[\![\,\bullet\,,\ \bullet\,,\ \bullet\,]\!]_{\mathrm{marks}} : E \times v \times e \to e$

$$\begin{aligned}
&[\![\mathtt{[\,]},\ v,\ e_2]\!]_{\mathrm{marks}} &&= e_2\\
&[\![(\mathtt{wcm}\ w_1\ E_1),\ v_1,\ e_2]\!]_{\mathrm{marks}} &&= [\![E_1,\ v_1,\ (\mathtt{cons}\ v_3\ e_2)]\!]_{\mathrm{marks}}\\
&\text{where } w_1 = ((v_4\ v_5)\ \ldots\ (v_1\ v_3)\ (v_6\ v_7)\ \ldots) &&\\
&[\![(\mathtt{wcm}\ w_1\ E_1),\ v_1,\ e_2]\!]_{\mathrm{marks}} &&= [\![E_1,\ v_1,\ e_2]\!]_{\mathrm{marks}}\\
&\text{where } v_1 \notin \mathrm{Dom}(w_1) &&\\
&[\![(v\ \ldots\ E_1\ e\ \ldots),\ v_1,\ e_2]\!]_{\mathrm{marks}} &&= [\![E_1,\ v_1,\ e_2]\!]_{\mathrm{marks}}\\
&[\![(\mathtt{begin}\ E_1\ e),\ v_1,\ e_2]\!]_{\mathrm{marks}} &&= [\![E_1,\ v_1,\ e_2]\!]_{\mathrm{marks}}\\
&[\![(\mathtt{\%}\ v_2\ E_1\ v_3),\ v_1,\ e_2]\!]_{\mathrm{marks}} &&= [\![E_1,\ v_1,\ e_2]\!]_{\mathrm{marks}}\\
&[\![(\mathtt{\%}\ v_2\ e_1\ E_1),\ v_1,\ e_2]\!]_{\mathrm{marks}} &&= [\![E_1,\ v_1,\ e_2]\!]_{\mathrm{marks}}\\
&[\![(\mathtt{\%}\ E_1\ e_1\ e_3),\ v_1,\ e_2]\!]_{\mathrm{marks}} &&= [\![E_1,\ v_1,\ e_2]\!]_{\mathrm{marks}}\\
&[\![(\mathtt{dw}\ x\ e_1\ E_1\ e_3),\ v_1,\ e_2]\!]_{\mathrm{marks}} &&= [\![E_1,\ v_1,\ e_2]\!]_{\mathrm{marks}}
\end{aligned}$$

$$\begin{aligned}
&(\mathtt{\%}\ v_1\ v_2\ v_3) \leadsto v_2 && \text{[prompt-v]}\\[4pt]
&(\mathtt{dynamic\text{-}wind}\ (\lambda\ ()\ e_1)\ (\lambda\ ()\ e_2)\ (\lambda\ ()\ e_3)) && \text{[dw]}\\
\leadsto\ &(\mathtt{begin}\ e_1\ (\mathtt{dw}\ x_1\ e_1\ e_2\ e_3))&&\\
&\text{where } x_1 \text{ fresh}&&\\[4pt]
&(\mathtt{dw}\ x\ e_1\ v_1\ e_3) \leadsto (\mathtt{begin}\ e_3\ v_1) && \text{[dw-v]}\\[4pt]
&(\mathtt{\%}\ v_1\ W_2[(\mathtt{abort}\ v_1\ v_2)]\ v_3) \leadsto (v_3\ v_2) && \text{[abort]}\\
&\text{where } W_2 \neq E[(\mathtt{\%}\ v_1\ E\ v)]&&\\[4pt]
&(\mathtt{dw}\ x_1\ e_1\ W_2[(\mathtt{abort}\ v_1\ v_2)]\ e_2) && \text{[abort-post]}\\
\leadsto\ &(\mathtt{begin}\ e_2\ (\mathtt{abort}\ v_1\ v_2))&&\\
&\text{where } W_2 \neq E[(\mathtt{\%}\ v_1\ E\ v)]&&\\[4pt]
&(\mathtt{\%}\ v_2\ E_2[(\mathtt{wcm}\ w_1\ (\mathtt{call/comp}\ v_1\ v_2))]\ v_3) && \text{[call/comp]}\\
\leadsto\ &(\mathtt{\%}\ v_2\ E_2[(\mathtt{wcm}\ w_1\ (v_1\ (\mathtt{comp}\ E_2)))]\ v_3)&&\\
&\text{where } E_2 \neq E[(\mathtt{\%}\ v_2\ E\ v)]&&\\[4pt]
&((\mathtt{comp}\ W_1[(\mathtt{dw}\ x_1\ e_1\ E_2\ e_2)])\ v_1) && \text{[comp-pre]}\\
\leadsto\ &[\![W_1[(\mathtt{begin}\ e_1\ (\mathtt{dw}\ x_1\ e_1\ ((\mathtt{comp}\ E_2)\ v_1)\ e_2))]]\!]_{\mathrm{wcm}}&&\\[4pt]
&((\mathtt{comp}\ W_1)\ v_1) \leadsto [\![W_1[v_1]]\!]_{\mathrm{wcm}} && \text{[comp]}
\end{aligned}$$

$$\begin{aligned}
&(\mathtt{\%}\ v_2\ E_2[(\mathtt{wcm}\ w_1\ (\mathtt{call/cc}\ v_1\ v_2))]\ v_3) && \text{[call/cc]}\\
\leadsto\ &(\mathtt{\%}\ v_2\ E_2[(\mathtt{wcm}\ w_1\ (v_1\ (\mathtt{cont}\ v_2\ E_2)))]\ v_3)&&\\
&\text{where } E_2 \neq E[(\mathtt{\%}\ v_2\ E\ v)]&&\\[4pt]
&(\mathtt{\%}\ v_2\ D_2[E_3[(\mathtt{dw}\ x_1\ e_1\ W_5[((\mathtt{cont}\ v_2\ D_6[E_4])\ v_1)]\ e_2)]]\ v_3) && \text{[cont-post]}\\
\leadsto\ &(\mathtt{\%}\ v_2\ D_2[E_3[(\mathtt{begin}\ e_2\ ((\mathtt{cont}\ v_2\ D_6[E_4])\ v_1))]]\ v_3)&&\\
&\text{where } D_2[E_3] \neq E[(\mathtt{\%}\ v_2\ E\ v)],\ \textsc{same}\mathrm{DW}\textsc{s}(D_2,\ D_6),&&\\
&\qquad W_5 \neq E[(\mathtt{\%}\ v_2\ E\ v)],&&\\
&\qquad \textsc{no}\mathrm{S}\textsc{hared}(E_3[(\mathtt{dw}\ x_1\ e_1\ W_5\ e_2)],\ E_4)&&\\[4pt]
&(\mathtt{\%}\ v_1\ D_2[W_3[((\mathtt{cont}\ v_1\ \mathtt{k})\ v_2)]]\ v_3) && \text{[cont-pre]}\\
\leadsto\ &(\mathtt{\%}\ v_1\ D_6[W_4[(\mathtt{begin}\ e_1\ (\mathtt{dw}\ x_1\ e_1\ ((\mathtt{cont}\ v_1\ \mathtt{k})\ v_2)\ e_2))]]\ v_3)&&\\
&\text{where } D_2[W_3] \neq E[(\mathtt{\%}\ v_1\ E\ v)],\ \textsc{same}\mathrm{DW}\textsc{s}(D_2,\ D_6),&&\\
&\qquad \textsc{no}\mathrm{S}\textsc{hared}(W_3,\ W_4[(\mathtt{dw}\ x_1\ e_1\ E_5\ e_2)]),&&\\
&\qquad \mathtt{k} = D_6[W_4[(\mathtt{dw}\ x_1\ e_1\ E_5\ e_2)]]&&\\[4pt]
&(\mathtt{\%}\ v_1\ D_2[W_3[((\mathtt{cont}\ v_1\ D_6[W_4])\ v_2)]]\ v_3) && \text{[cont]}\\
\leadsto\ &(\mathtt{\%}\ v_1\ D_6[W_4[v_2]]\ v_3)&&\\
&\text{where } D_2[W_3] \neq E[(\mathtt{\%}\ v_1\ E\ v)],\ \textsc{same}\mathrm{DW}\textsc{s}(D_2,\ D_6),&&\\
&\qquad \textsc{no}\mathrm{S}\textsc{hared}(W_3,\ W_4)&&
\end{aligned}$$

$$\begin{aligned}
&(\mathtt{\%}\ v_2\ E_2[(\mathtt{current\text{-}marks}\ v_1\ v_2)]\ v_3) && \text{[marks]}\\
\leadsto\ &(\mathtt{\%}\ v_2\ E_2[[\![E_2,\ v_1,\ (\mathtt{list})]\!]_{\mathrm{marks}}]\ v_3)&&\\
&\text{where } E_2 \neq E[(\mathtt{\%}\ v_2\ E\ v)]&&\\[4pt]
&(\mathtt{wcm}\ w\ v_1) \leadsto v_1 && \text{[wcm-v]}\\[4pt]
&(\mathtt{wcm}\ ((v_1\ v_2)\ \ldots\ (v_3\ v_4)\ (v_5\ v_6)\ \ldots) && \text{[wcm-set]}\\
&\qquad (\mathtt{call/cm}\ v_3\ v_7\ (\lambda\ ()\ e_1)))&&\\
\leadsto\ &(\mathtt{wcm}\ ((v_1\ v_2)\ \ldots\ (v_3\ v_7)\ (v_5\ v_6)\ \ldots)\ e_1)&&\\[4pt]
&(\mathtt{wcm}\ ((v_1\ v_2)\ \ldots)\ (\mathtt{call/cm}\ v_3\ v_4\ (\lambda\ ()\ e_1))) && \text{[wcm-add]}\\
\leadsto\ &(\mathtt{wcm}\ ((v_1\ v_2)\ \ldots\ (v_3\ v_4))\ e_1)&&\\
&\text{where } v_3 \notin (v_1\ \ldots)&&\\[4pt]
&E_1[(u_1\ v_1\ \ldots)] \longrightarrow E_1[(\mathtt{wcm}\ ()\ (u_1\ v_1\ \ldots))] && \text{[wcm-intro]}\\
&\text{where } E_1 \neq E[(\mathtt{wcm}\ w\ \mathtt{[\,]})]&&\\[4pt]
&(\mathtt{begin}\ v\ e_1) \leadsto e_1 && \text{[begin-v]}
\end{aligned}$$

Implementation for Racket

- Thousands of lines of C
- Four stacks: control, value, mark, `dynamic-wind`
- Stacks grouped into meta-continuations

  ... with a trampoline to handle "tail" jumps
- Coarse-grained sharing among continuations
- Caches to make `current-marks` amortized constant time
- Shortcuts for continuation jumps that act as aborts
- Shortcuts for continuations used to implement threads
- Special lightweight continuations for futures
- ...
- Thousands of lines of hand-crafted Racket code in the test suite

Every bit as fragile as it sounds!

Random Testing

```
 e ::= m | (wcm w m)
 m ::= x | v | (e e ...) | (begin e e) | (% e e e) | (dw x e e e)
 v ::= (list v ...) | (λ (x ...) e) | (cont v E) | (comp E)
     | dynamic-wind | abort | current-marks
     | cons | u
 u ::= call/cc | call/comp | call/cm
 w ::= ((v v) ...)
 E ::= W | W[(dw x e E e)]
 W ::= M | (wcm w M)
 M ::= [] | (v ... W e ...) | (begin W e) | (% W e e) | (% v e W) | (% v W v)
 D ::= [] | E[(dw x e [] e)]
```

Random Testing

```
 e ::= m | (wcm w m)
 m ::= x | v | (e e ...) | (begin e e) | (% e e e) | (dw x e e e)
 v ::= (list v ...) | (λ (x ...) e) | (cont v E) | (comp E)
     | dynamic-wind | abort | current-marks
     | cons | u
 u ::= call/cc | call/comp | call/cm
 w ::= ((v v) ...)
 E ::= W | W[(dw x e E e)]
 W ::= M | (wcm w M)
 M ::= [] | (v ... W e ...) | (begin W e) | (% W e e) | (% v e W) | (% v W v)
 D ::= [] | E[(dw x e [] e)]
```

```
(% 5
   (cons
    (call/comp 5 (λ (x) 8))
    (call/cc 5 (λ (x) (x x))))
   (λ (y) (y)))
```

Random Testing

```
 e ::= m | (wcm w m)
 m ::= x | v | (e e ...) | (begin e e) | (% e e e) | (dw x e e e)
 v ::= (list v ...) | (λ (x ...) e) | (cont v E) | (comp E)
     | dynamic-wind | abort | current-marks
     | cons | u
 u ::= call/cc | call/comp | call/cm
 w ::= ((v v) ...)
 E ::= W | W[(dw x e E e)]
 W ::= M | (wcm w M)
 M ::= [] | (v ... W e ...) | (begin W e) | (% W e e) | (% v e W) | (% v W v)
 D ::= [] | E[(dw x e [] e)]
```

```
(% 5
   (cons
    (call/comp 5 (λ (x) 8))
    (call/cc 5 (λ (x) (x x))))
   (λ (y) (y)))
```

Random Testing

$$
\begin{aligned}
e &::= m \mid (\texttt{wcm}\ w\ m) \\
m &::= x \mid v \mid (e\ e\ \ldots) \mid (\texttt{begin}\ e\ e) \mid (\texttt{\%}\ e\ e\ e) \mid (\texttt{dw}\ x\ e\ e\ e) \\
v &::= (\texttt{list}\ v\ \ldots) \mid (\lambda\ (x\ \ldots)\ e) \mid (\texttt{cont}\ v\ E) \mid (\texttt{comp}\ E) \\
&\quad \mid \texttt{dynamic-wind} \mid \texttt{abort} \mid \texttt{current-marks} \\
&\quad \mid \texttt{cons} \mid u \\
u &::= \texttt{call/cc} \mid \texttt{call/comp} \mid \texttt{call/cm} \\
w &::= ((v\ v)\ \ldots) \\
E &::= W \mid W[(\texttt{dw}\ x\ e\ E\ e)] \\
W &::= M \mid (\texttt{wcm}\ w\ M) \\
M &::= [\,] \mid (v\ \ldots\ W\ e\ \ldots) \mid (\texttt{begin}\ W\ e) \mid (\texttt{\%}\ W\ e\ e) \mid (\texttt{\%}\ v\ e\ W) \mid (\texttt{\%}\ v\ W\ v) \\
D &::= [\,] \mid E[(\texttt{dw}\ x\ e\ [\,]\ e)]
\end{aligned}
$$

```
(% 5
   (cons
    (call/comp 5 (λ (x) 8))
    (call/cc 5 (λ (x) (x x))))
   (λ (y) (y)))
```

$(\texttt{\%}\ v_1\ v_2\ v_3) \leadsto v_2$ [prompt-v]

$(\texttt{dynamic-wind}\ (\lambda\ ()\ e_1)\ (\lambda\ ()\ e_2)\ (\lambda\ ()\ e_3))$ [dw]
$\leadsto (\texttt{begin}\ e_1\ (\texttt{dw}\ x_1\ e_1\ e_2\ e_3))$
where x_1 fresh

$(\texttt{dw}\ x\ e_1\ v_1\ e_3) \leadsto (\texttt{begin}\ e_3\ v_1)$ [dw-v]

$(\texttt{\%}\ v_1\ W_2[(\texttt{abort}\ v_1\ v_2)]\ v_3) \leadsto (v_3\ v_2)$ [abort]
where $W_2 \neq E[(\texttt{\%}\ v_1\ E\ v)]$

$(\texttt{dw}\ x_1\ e_1\ W_2[(\texttt{abort}\ v_1\ v_2)]\ e_2)$ [abort-post]
$\leadsto (\texttt{begin}\ e_2\ (\texttt{abort}\ v_1\ v_2))$
where $W_2 \neq E[(\texttt{\%}\ v_1\ E\ v)]$

$(\texttt{\%}\ v_2\ E_2[(\texttt{wcm}\ w_1\ (\texttt{call/comp}\ v_1\ v_2))]\ v_3)$ [call/comp]
$\leadsto (\texttt{\%}\ v_2\ E_2[(\texttt{wcm}\ w_1\ (v_1\ (\texttt{comp}\ E_2)))]\ v_3)$
where $E_2 \neq E[(\texttt{\%}\ v_2\ E\ v)]$

$((\texttt{comp}\ W_1[(\texttt{dw}\ x_1\ e_1\ E_2\ e_2)])\ v_1)$ [comp-pre]
$\leadsto [\![W_1[(\texttt{begin}\ e_1\ (\texttt{dw}\ x_1\ e_1\ ((\texttt{comp}\ E_2)\ v_1)\ e_2))]]\!]_{\mathrm{wcm}}$

$((\texttt{comp}\ W_1)\ v_1) \leadsto [\![W_1[v_1]]\!]_{\mathrm{wcm}}$ [comp]

Random Testing

$$
\begin{aligned}
e &::= m \mid (\texttt{wcm}\ w\ m)\\
m &::= x \mid v \mid (e\ e\ \ldots) \mid (\texttt{begin}\ e\ e) \mid (\texttt{\%}\ e\ e\ e) \mid (\texttt{dw}\ x\ e\ e\ e)\\
v &::= (\texttt{list}\ v\ \ldots) \mid (\lambda\ (x\ \ldots)\ e) \mid (\texttt{cont}\ v\ E) \mid (\texttt{comp}\ E)\\
&\quad \mid \texttt{dynamic-wind} \mid \texttt{abort} \mid \texttt{current-marks}\\
&\quad \mid \texttt{cons} \mid u\\
u &::= \texttt{call/cc} \mid \texttt{call/comp} \mid \texttt{call/cm}\\
w &::= ((v\ v)\ \ldots)\\
E &::= W \mid W[(\texttt{dw}\ x\ e\ E\ e)]\\
W &::= M \mid (\texttt{wcm}\ w\ M)\\
M &::= [\,] \mid (v\ \ldots\ W\ e\ \ldots) \mid (\texttt{begin}\ W\ e) \mid (\texttt{\%}\ W\ e\ e) \mid (\texttt{\%}\ v\ e\ W) \mid (\texttt{\%}\ v\ W\ v)\\
D &::= [\,] \mid E[(\texttt{dw}\ x\ e\ [\,]\ e)]
\end{aligned}
$$

```
(% 5
   (cons
    (call/comp 5 (λ (x) 8))
    (call/cc 5 (λ (x) (x x))))
   (λ (y) (y)))
```

$(\texttt{\%}\ v_1\ v_2\ v_3) \rightsquigarrow v_2$ [prompt-v]

$(\texttt{dynamic-wind}\ (\lambda\ ()\ e_1)\ (\lambda\ ()\ e_2)\ (\lambda\ ()\ e_3))$ [dw]
$\rightsquigarrow (\texttt{begin}\ e_1\ (\texttt{dw}\ x_1\ e_1\ e_2\ e_3))$
where x_1 fresh

$(\texttt{dw}\ x\ e_1\ v_1\ e_3) \rightsquigarrow (\texttt{begin}\ e_3\ v_1)$ [dw-v]

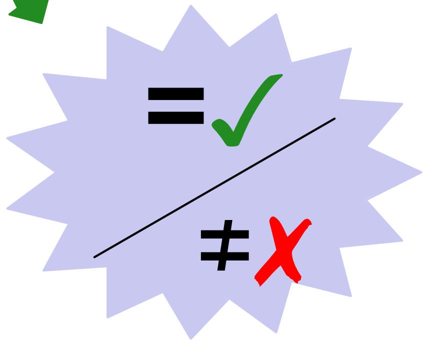
$(\texttt{\%}\ v_1\ W_2[(\texttt{abort}\ v_1\ v_2)]\ v_3) \rightsquigarrow (v_3\ v_2)$ [abort]
where $W_2 \neq E[(\texttt{\%}\ v_1\ E\ v)]$

$(\texttt{dw}\ x_1\ e_1\ W_2[(\texttt{abort}\ v_1\ v_2)]\ e_2)$ [abort-post]
$\rightsquigarrow (\texttt{begin}\ e_2\ (\texttt{abort}\ v_1\ v_2))$
where $W_2 \neq E[(\texttt{\%}\ v_1\ E\ v)]$

$(\texttt{\%}\ v_2\ E_2[(\texttt{wcm}\ w_1\ (\texttt{call/comp}\ v_1\ v_2))]\ v_3)$ [call/comp]
$\rightsquigarrow (\texttt{\%}\ v_2\ E_2[(\texttt{wcm}\ w_1\ (v_1\ (\texttt{comp}\ E_2)))]\ v_3)$
where $E_2 \neq E[(\texttt{\%}\ v_2\ E\ v)]$

$((\texttt{comp}\ W_1[(\texttt{dw}\ x_1\ e_1\ E_2\ e_2)])\ v_1)$ [comp-pre]
$\rightsquigarrow [\![W_1[(\texttt{begin}\ e_1\ (\texttt{dw}\ x_1\ e_1\ ((\texttt{comp}\ E_2)\ v_1)\ e_2))]]\!]_{\text{wcm}}$

$((\texttt{comp}\ W_1)\ v_1) \rightsquigarrow [\![W_1[v_1]]\!]_{\text{wcm}}$ [comp]



Bugs Found

- Inspecting continuation marks from a `dynamic-wind` pre-thunk, where the mark is in a different meta-continuation...

  0b19c6e798b031bc191a3721f351cd4cb4a43ac0

- Meta-continuation frame offsets tracked incorrectly in `dynamic-wind` stack

  e77917db8c401a7d2154df8fe61058188e27d967

- Prompt test for non-composable continuations incorrectly used for composable continuations

  0d3fbb11faa116c21af8b0400d9963d7e1a4b7ef

- Ditto, but in `dynamic-wind` post-thunks

  705b11f2b8e755065c536fc553d045f25fe679d9

- Model: broken substitution and a few missing-case bugs

Practical Delimited and Composable Continuations

`http://www.cs.utah.edu/plt/delim-cont/`

delimited continuations
`prompt call/cc call/comp`

dynamic-wind guards
`dynamic-wind`

exceptions
`prompt abort call/cm`

dynamic binding
`call/cm current-marks`

stack inspection
`current-marks`

`www.racket-lang.org`