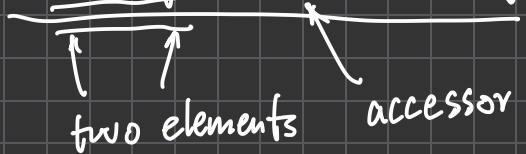
Lecture 13. Encodings of Lists, Structs, Variants

Records

1. Tuples

PAIR : $\lambda x.\ \lambda y.\ \lambda f.\ f\ x\ y$



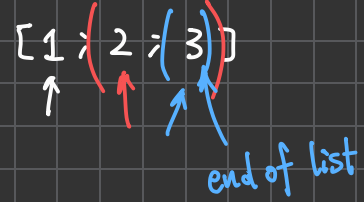
PAIR 1 2 // (1, 2)

$(\lambda x.\ \lambda y.\ \lambda f.\ f\ x\ y)\ 1\ 2$

$\rightarrow_\beta\ (\lambda y.\ \lambda f.\ f\ 1\ y)\ 2$

$\rightarrow_\beta\ (\lambda f.\ f\ 1\ 2)$

Lists, Structs, Variants

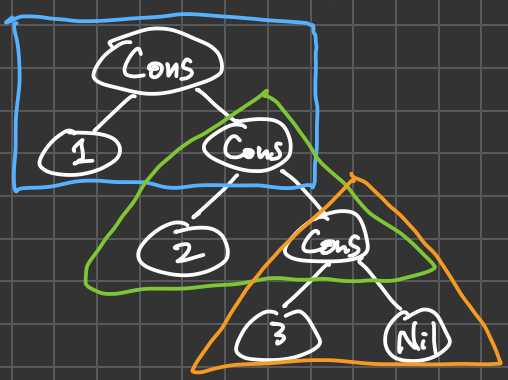



```
type 'a list = Cons of 'a * 'a list
             | Nil
```

(Expr) e ::= ... (Fb syntax) ... | (e_1, e_2)
| Left e | Right e | l

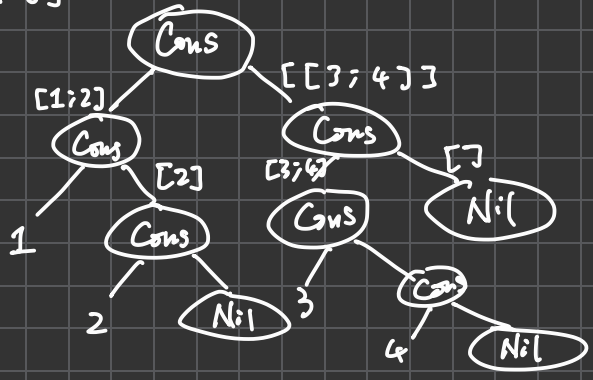
(Label) l ::= 'string // 'Cons, 'Nil, 'p1, 'p2

[1; 2; 3]



('Cons, (1,
('Cons, (2,
('Cons, (3,
Nil
))
))
))

[[1; 2]; [3; 4]]



```
[1;2]:  ('Cons, (1, ('Cons, (2, Nil))))
[3;4]:  ('Cons, (3, ('Cons, (4, Nil))))

('Cons, (
    ('Cons, (1, ('Cons, (2, Nil)))),
    ('Cons,
        ('Cons, (3, ('Cons, (4, Nil)))),
        Nil
    )
))
```

```
Let prepend = Fun x -> Fun list ->      (* to-be-prepended, list *)
    ('Cons, (x, list))
In
...
```

```
Let append = Fun x ->
    Let Rec append_helper list =
        if list = 'Nil
        then ('Cons, (x, Nil))
        else // assume list = ('Cons, (hd, tl))
            Let hd = Left (Right list)) In
            Let tl = Right (Right list)) In
```

('Cons, (hd, (append_helper tl)))

Variant Syntax

$e ::=$ ... Fb syntax ...

$\mid \ \ell\ e$ // 'Cons (hd . tl()

$\mid \ \ell$ // 'Int 1 ← $\ell\ e$

// 'Some 3 ←

// 'None ← ℓ

(Label) $\dfrac{}{\ell \Rightarrow \ell}$

(Variant) $\dfrac{e \Rightarrow v}{\ell\ e \Rightarrow \ell\ v}$

$v ::= i \mid b \mid \text{Fun } x \Rightarrow e \mid (v, v)$

$\mid \ \ell \mid \ell\ v$

```
Match e With
| 'Label1 payload1 -> e1
| 'Label2 pL2      -> e2
| 'Label3 pL3      -> e3
```

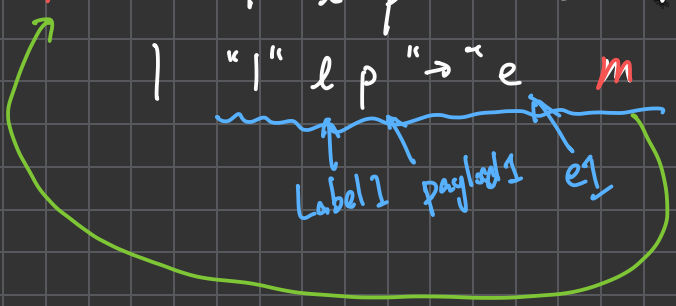
F^bV

$e ::= \ldots$ F^b syntax $\ldots \mid (e_1, e_2) \mid \text{'}l \mid \text{'}l\ e \mid$

$\mid$ Match e With m

$m ::= \text{"|"}\ l\ p\ \text{"->"}\ e$ // Last branch

$\mid \text{"|"}\ l\ p\ \text{"->"}\ e\ m$



$p ::= x \mid (p, p) \mid l\ p$

e.g.

```
Match 'Cons(1, 'Cons(2, 'Nil)) With
| 'Cons(first, 'Cons(second, 'Nil)) ->
```

$$\frac{e \Rightarrow l_i\ v_i \qquad e_i[v_i/x_i] \Rightarrow v_i'}{\text{Match } e \text{ With } \ldots \mid l_i\ x_i \to e_i \Rightarrow v_i'}$$

Structs / Record

F^bR

C / C++ { .field1 = value1, .field2 = value2 }

JavaScript { field1 : value1, field2 : value2 }

Python { key : value }

F^bR

$l ::=$ string

$e ::= \{ l_1 = e_1 ; \ldots ; l_n = e_n \} \mid \ldots$

{ len = 2 ; content = [1 ; 2] }

$\mid e.l$

$v ::= \{ l_1 = v_1 ; \ldots ; l_n = v_n \} \mid \ldots$

JS: object.length

PY: str.upper()

Construct [Structs]

$$\frac{\bigwedge_{i \in [1,n]} e_i \Rightarrow v_i}{\{l_1 = e_1 ; \cdots ; l_n = e_n\} \Rightarrow \{l_1 = v_1 ; \cdots ; l_n = v_n\}}$$

[Projection]

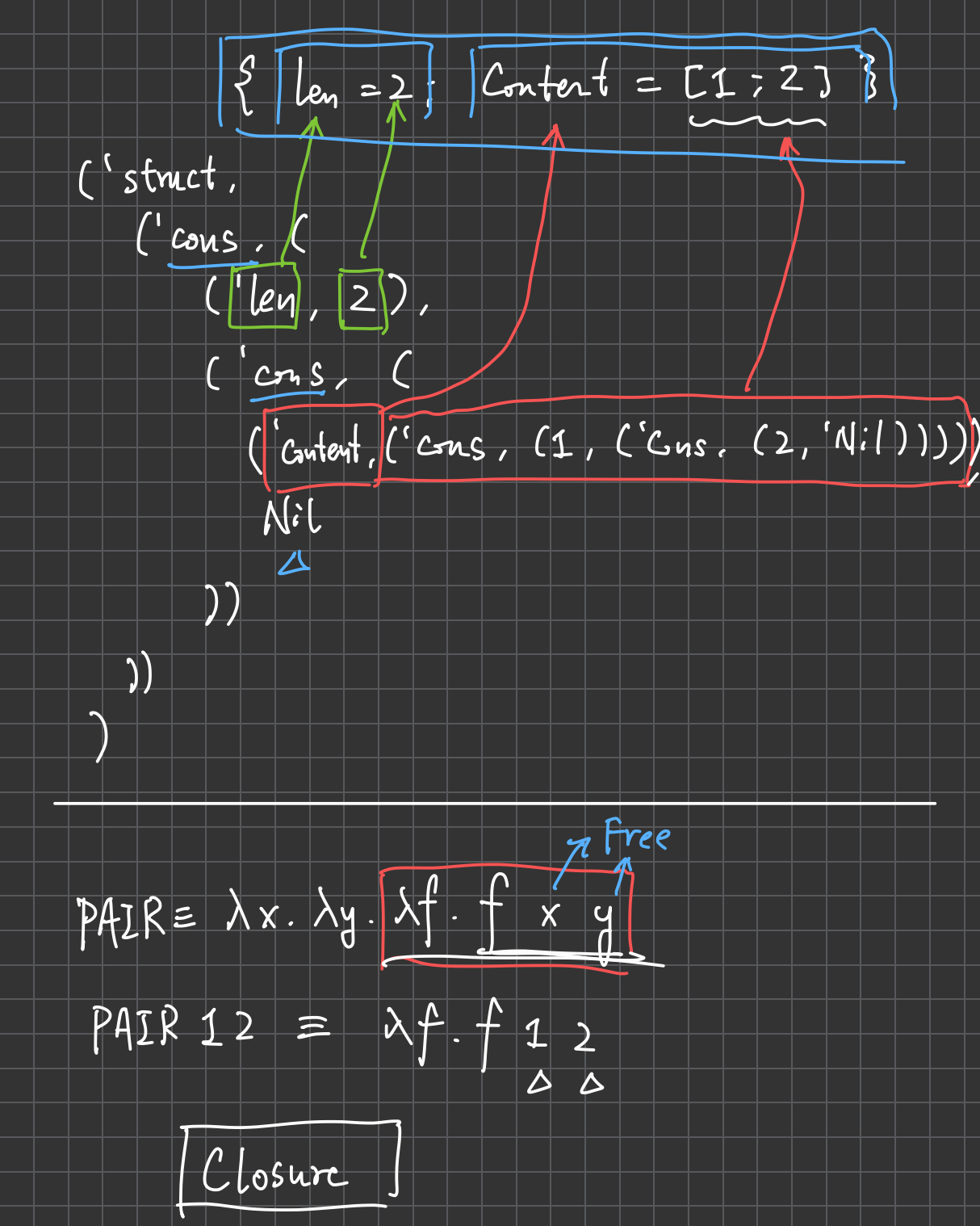
$$\frac{e \Rightarrow \{l_1 = v_1 ; \cdots ; l_n = v_n\}}{e.l_i \Rightarrow v_i}$$

('struct, [(l₁, e₁); (l₂, e₂); ...])

"discriminator"

CPython

- ↳ PyObject
- ↳ PyArray
- ↳ PyTuple
- ↳ PyInt

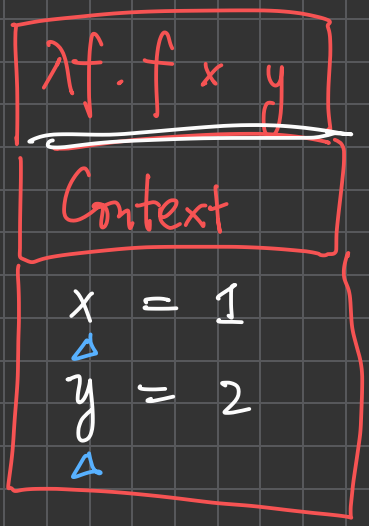


$$PAIR \equiv \lambda x.\ \lambda y.\ \lambda f.\ f\ x\ y$$

(x, y: Free)

$$PAIR\ 1\ 2 \equiv \lambda f.\ f\ 1\ 2$$

Closure


λf. f x y
Context
x = 1
y = 2