

Region Based Shape Analysis with Tracked Locations

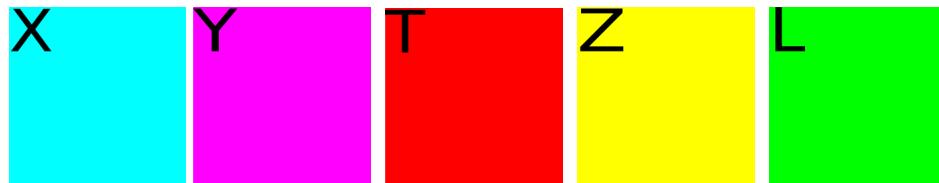
Brian Hackett
Radu Rugina

Computer Science Department
Cornell University

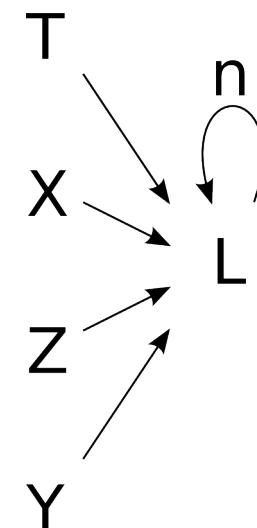
Example Program: Splice

```
1:  typedef struct list {  
2:      struct list *n;  
3:      int data;  
4:  } List;  
5:  
6: List *splice (List *x, List *y) {  
7:     List *t = NULL;  
8:     List *z = y;  
9:     while (x != NULL) {  
10:         t = x;  
11:         x = t->n;  
12:         t->n = y->n;  
13:         y->n = t;  
14:         y = y->n->n;  
15:     }  
16:     return z;  
17: }
```

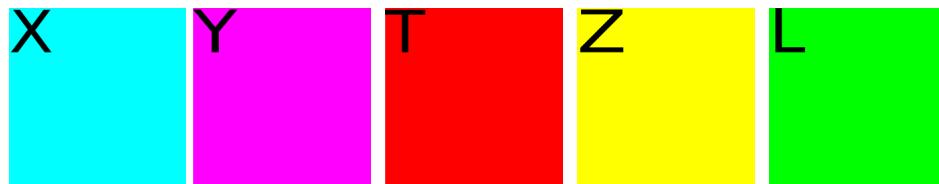
Example Program: Splice



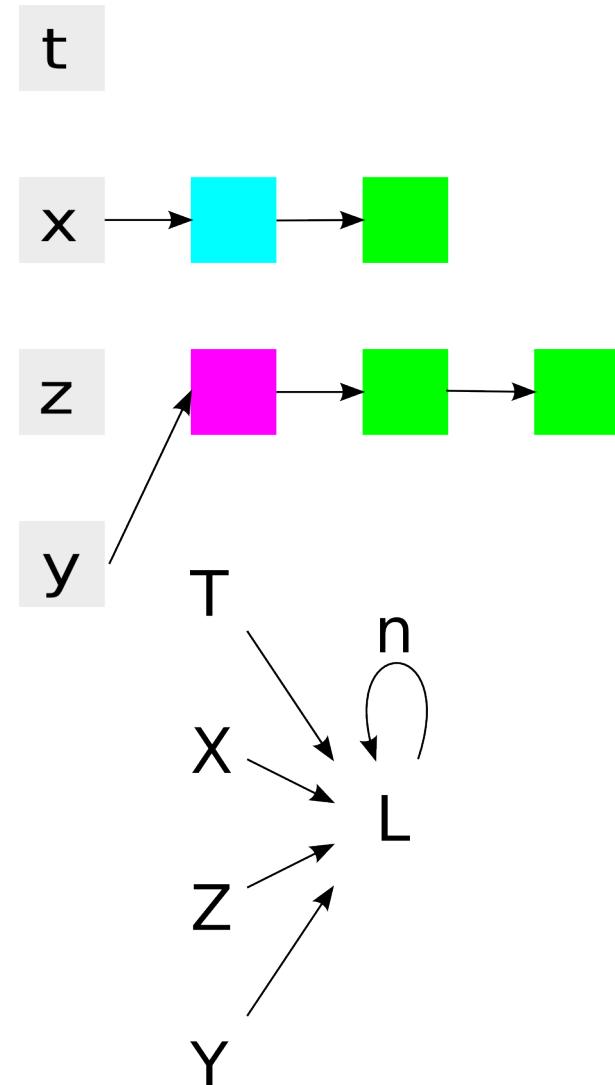
```
6: List *splice (List *x, List *y) {  
7:     List *t = NULL;  
8:     List *z = y;  
9:     while (x != NULL) {  
10:         t = x;  
11:         x = t->n;  
12:         t->n = y->n;  
13:         y->n = t;  
14:         y = y->n->n;  
15:     }  
16:     return z;  
17: }
```



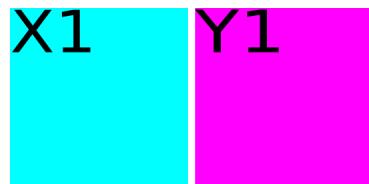
Example Program: Splice



```
6: List *splice (List *x, List *y) {  
7:     List *t = NULL;  
8:     List *z = y;  
9:     while (x != NULL) {  
10:         t = x;  
11:         x = t->n;  
12:         t->n = y->n;  
13:         y->n = t;  
14:         y = y->n->n;  
15:     }  
16:     return z;  
17: }
```



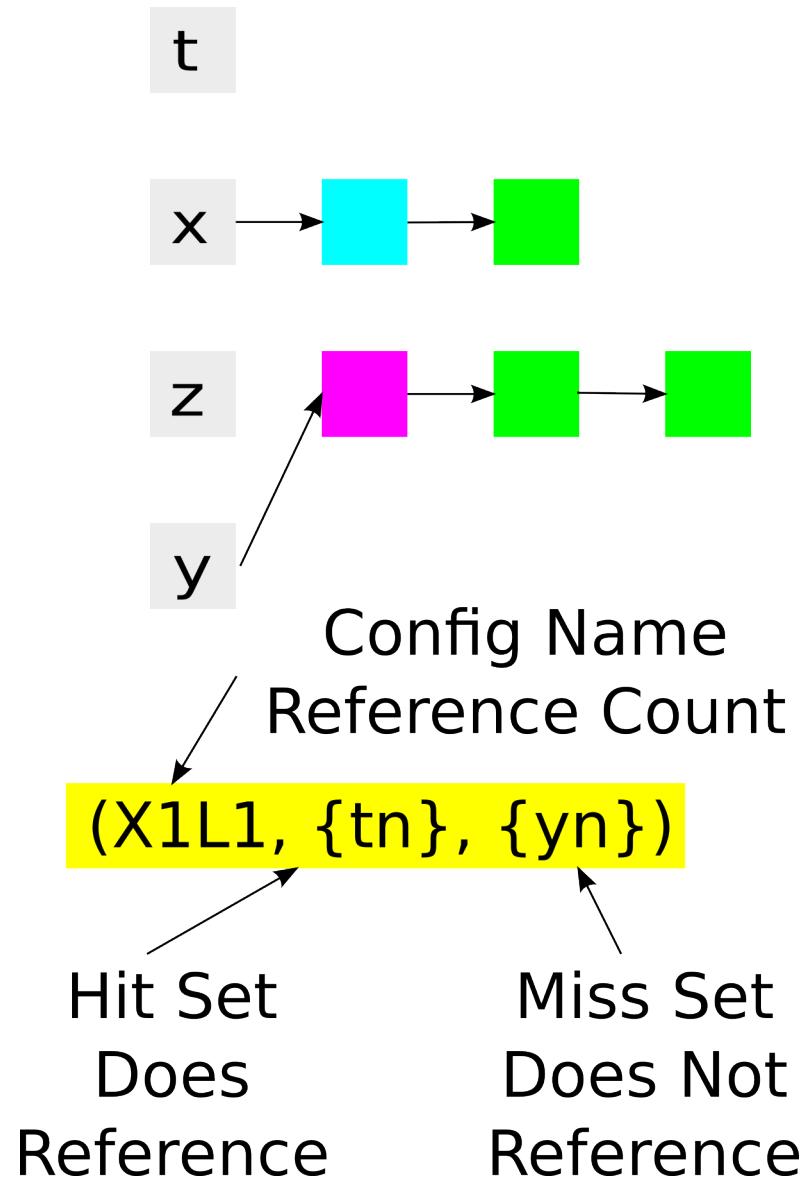
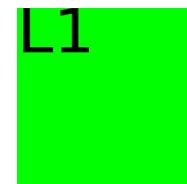
Configurations for heap



$(X1, \{x\}, \{\})$

$(Y1, \{y\}, \{\})$

$(L1, \{\}, \{\})$



Rules for Updating Configurations

Do the expressions in the assignment reference the heap region described by this configuration?

No:

Move to next configuration

Yes:

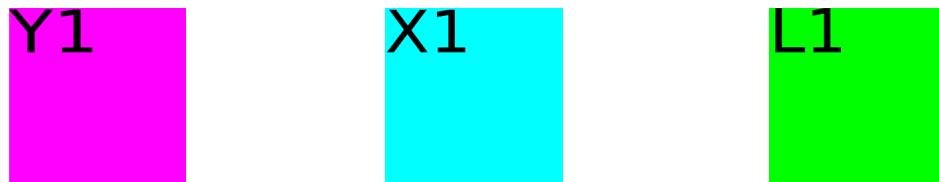
- 1) Recalculate reference counts
- 2) Update hit and miss sets

Maybe:

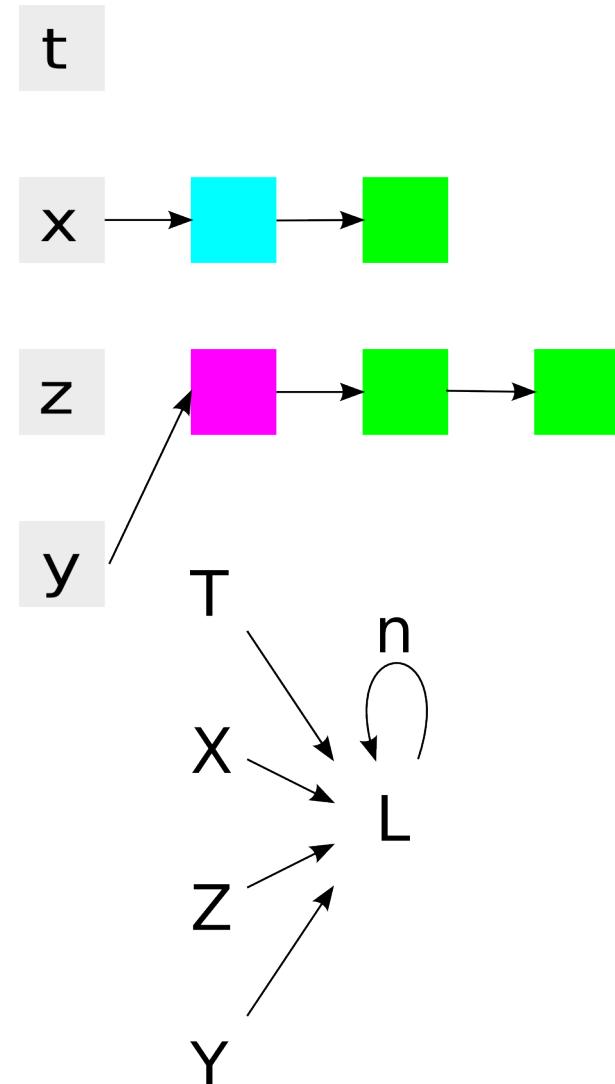
- 1) Split node into 2 node
 - 1 node where expression definitely does reference heap region
 - 1 node where expression definitely does not reference heap region
- 2) recalculate reference counts for each node.
- 3) Update hit and miss sets.

Merge Repeated Configurations

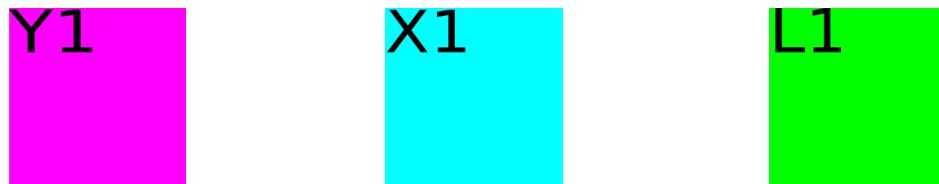
Updating Config



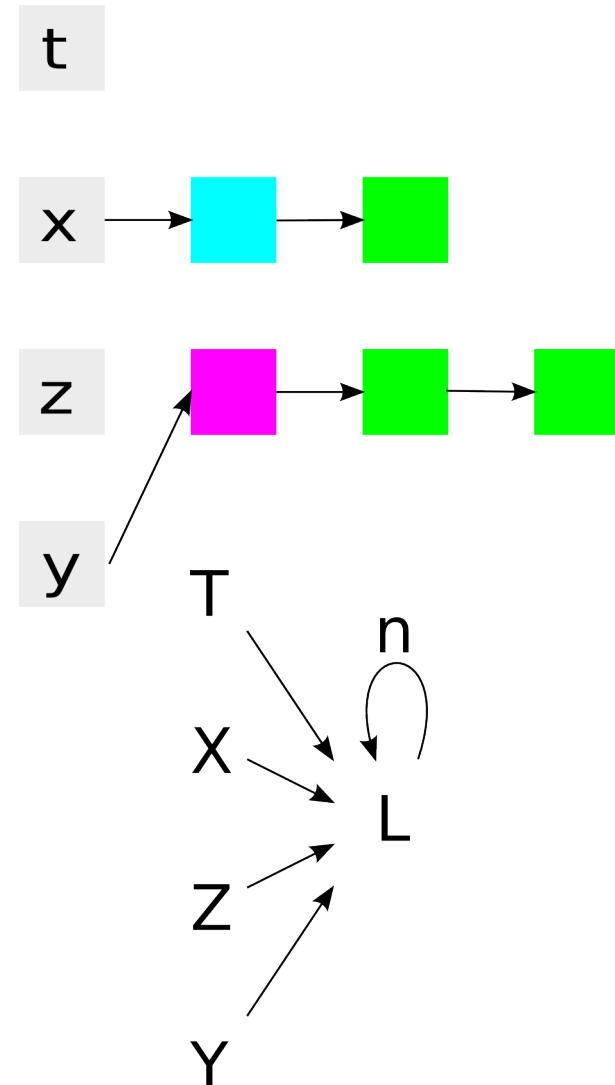
```
6: List *splice (List *x, List *y) {  
7:     List *t = NULL;  
8:     List *z = y;  
9:     while (x != NULL) {  
10:         t = x;  
11:         x = t->n;  
12:         t->n = y->n;  
13:         y->n = t;  
14:         y = y->n->n;  
15:     }  
16:     return z;  
17: }
```



Updating Config



```
6: List *splice (List *x, List *y) {  
7:     List *t = NULL;  
8:     List *z = y; highlighted line  
9:     while (x != NULL) {  
10:         t = x;  
11:         x = t->n;  
12:         t->n = y->n;  
13:         y->n = t;  
14:         y = y->n->n;  
15:     }  
16:     return z;  
17: }
```



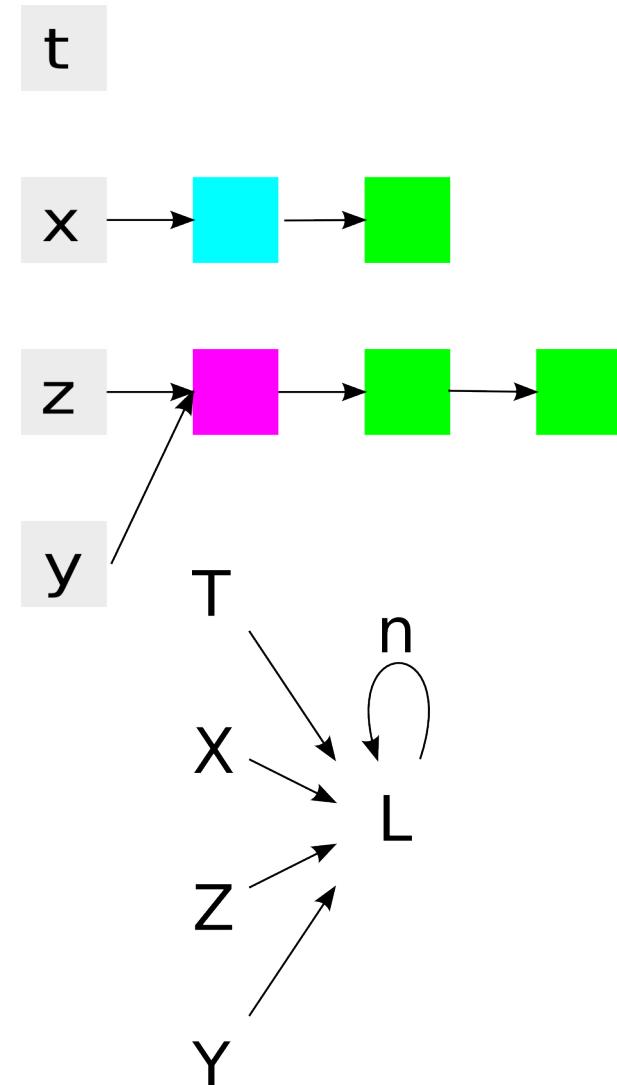
Updating Config

Y1Z1

X1

L1

```
6: List *splice (List *x, List *y) {  
7:     List *t = NULL;  
8:     List *z = y;  
9:     while (x != NULL) {  
10:         t = x;  
11:         x = t->n;  
12:         t->n = y->n;  
13:         y->n = t;  
14:         y = y->n->n;  
15:     }  
16:     return z;  
17: }
```



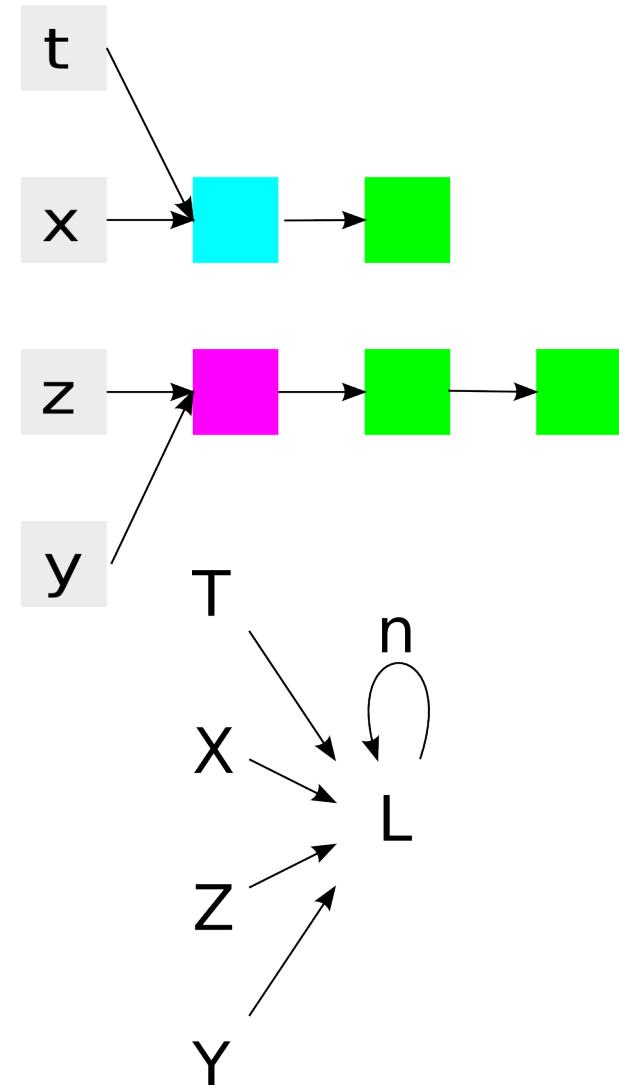
Updating Config

Y1Z1

X1T1

L1

```
6: List *splice (List *x, List *y) {  
7:     List *t = NULL;  
8:     List *z = y;  
9:     while (x != NULL) {  
10:        t = x;  
11:        x = t->n; highlighted line  
12:        t->n = y->n;  
13:        y->n = t;  
14:        y = y->n->n;  
15:    }  
16:    return z;  
17: }
```



Updating Config: No Reference

Y1Z1

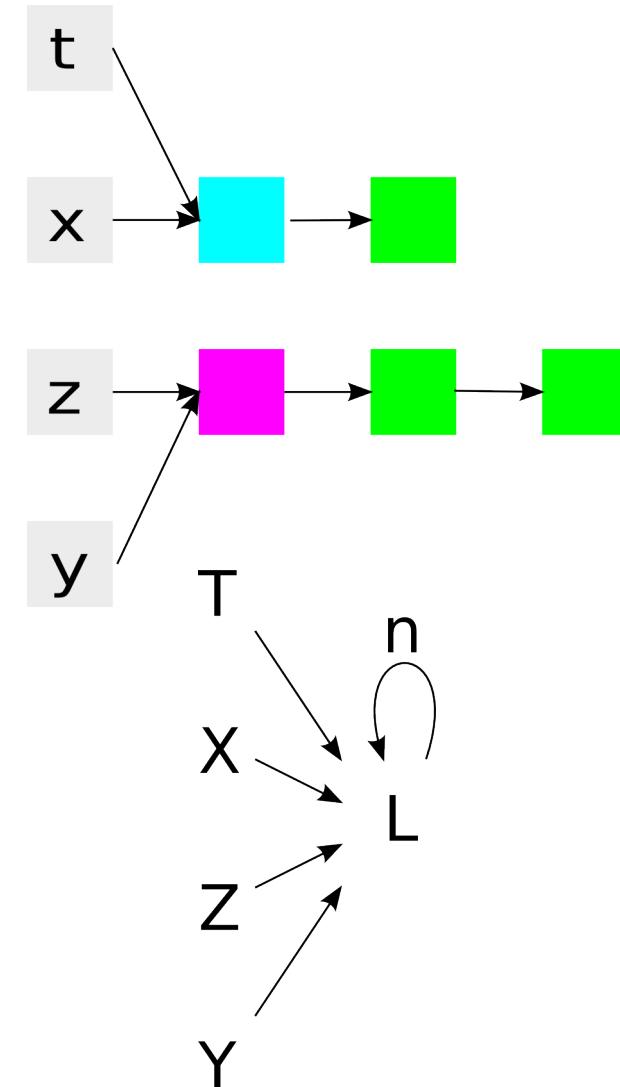
X1T1

L1

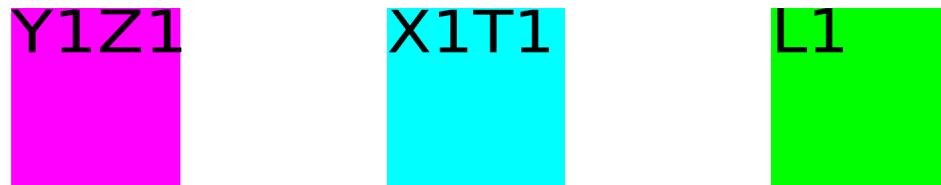
11: $x = t->n$

e0

e1



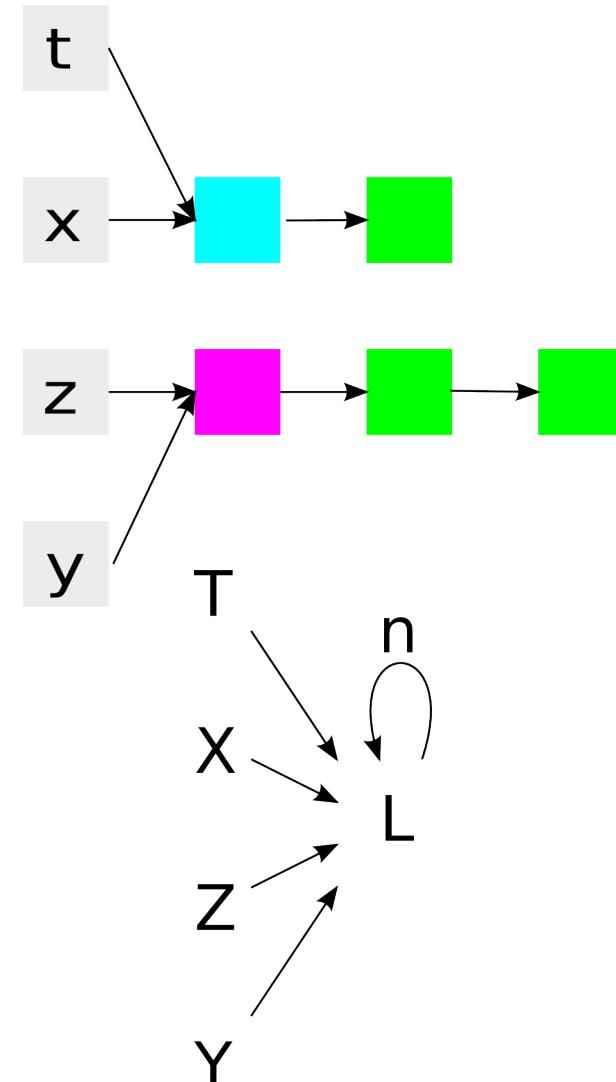
Updating Config: Yes Reference



11: $x = t->n$

e0

e1



Updating Config: Yes Reference

Y1Z1

X1T1

L1

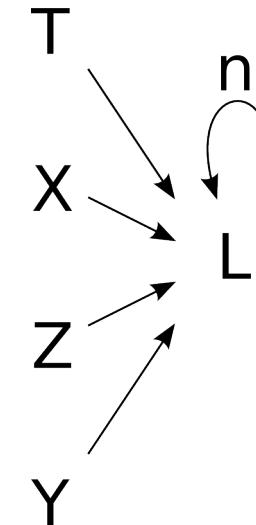
e0 & !e1 = decrement reference count
!e0 & e1 = increment reference count
e0 & e1 = no change
!e0 & !e1 = no change

11: $x = t \rightarrow n$

e0

!e1

Y1Z1



Updating Config: Maybe Reference

Y1Z1

X1T1

L1

e0 & !e1 = decrement reference count
!e0 & e1 = increment reference count
e0 & e1 = no change
!e0 & !e1 = no change

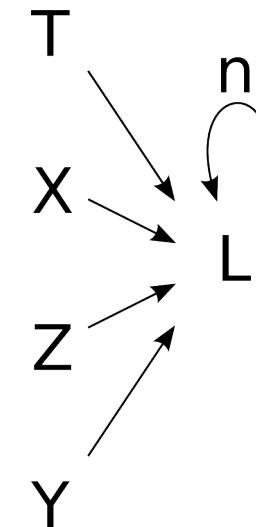
11: $x = t \rightarrow n$

e0

e1

Y1Z1

T1



Updating Config: Maybe Reference

Y1Z1

X1T1

L1

e0 & !e1 = decrement reference count
!e0 & e1 = increment reference count
e0 & e1 = no change
!e0 & !e1 = no change

11: $x = t \rightarrow n$

!e0

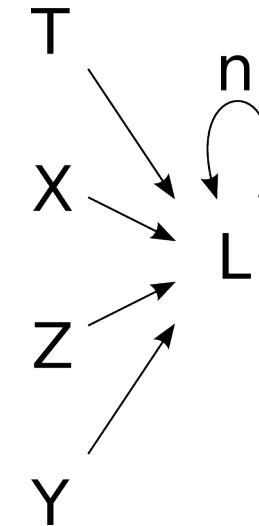
e1

Y1Z1

T1

tn

!tn



Updating Config: Maybe Reference

Y1Z1

X1T1

L1

e0 & !e1 = decrement reference count
!e0 & e1 = increment reference count
e0 & e1 = no change
!e0 & !e1 = no change

11: $x = t \rightarrow n$

!e0

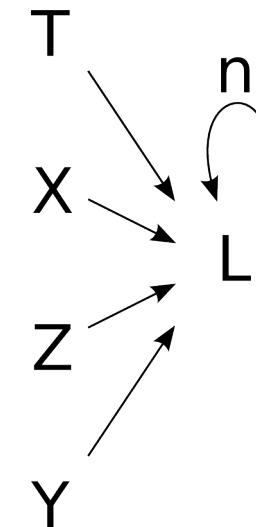
e1

Y1Z1

T1

X1L1
tn

!tn



Updating Config: Maybe Reference

Y1Z1

X1T1

L1

e0 & !e1 = decrement reference count
!e0 & e1 = increment reference count
e0 & e1 = no change
!e0 & !e1 = no change

11: $x = t \rightarrow n$

!e0

!e1

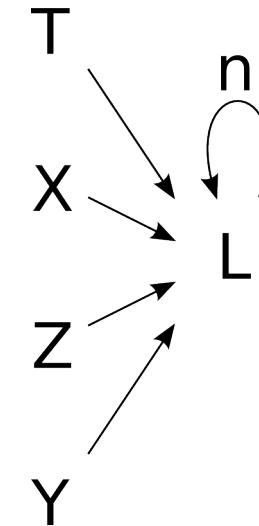
Y1Z1

T1

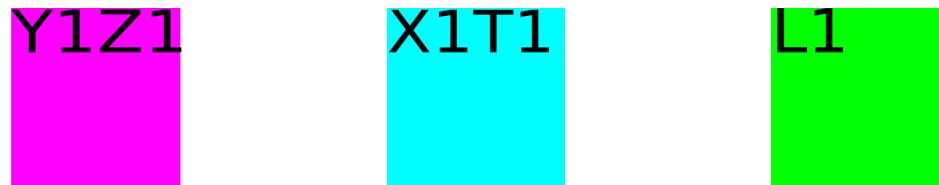
X1L1

tn

!tn

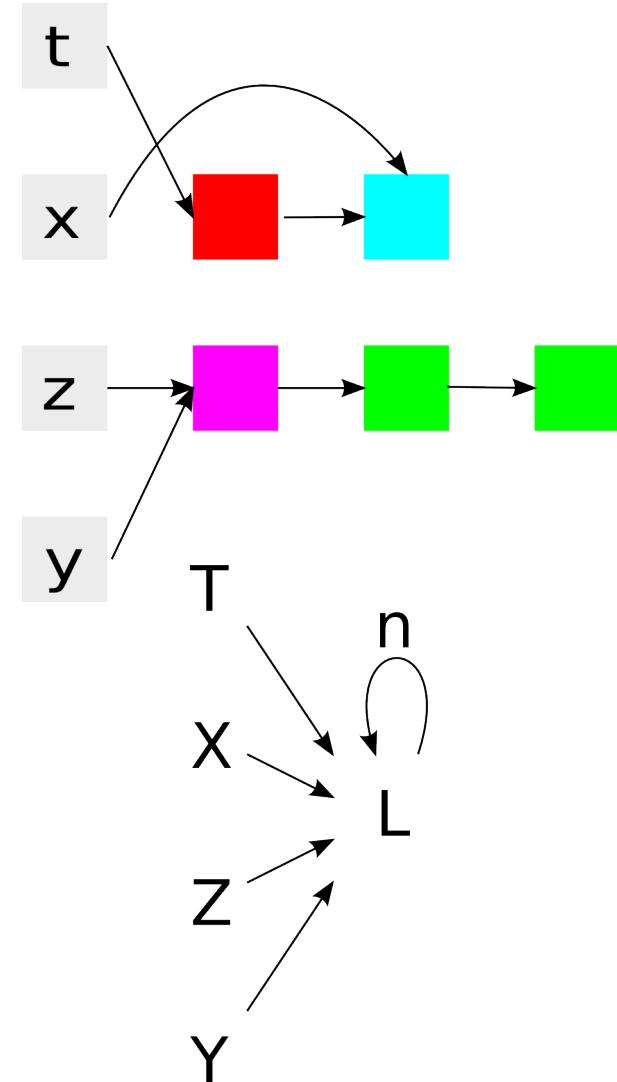
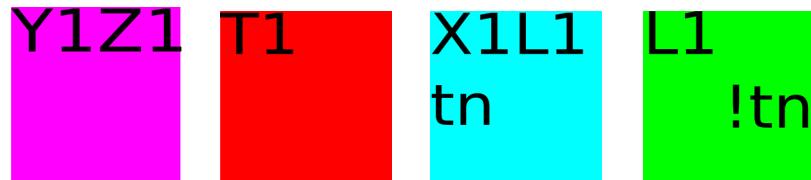


Updating Config: Maybe Reference



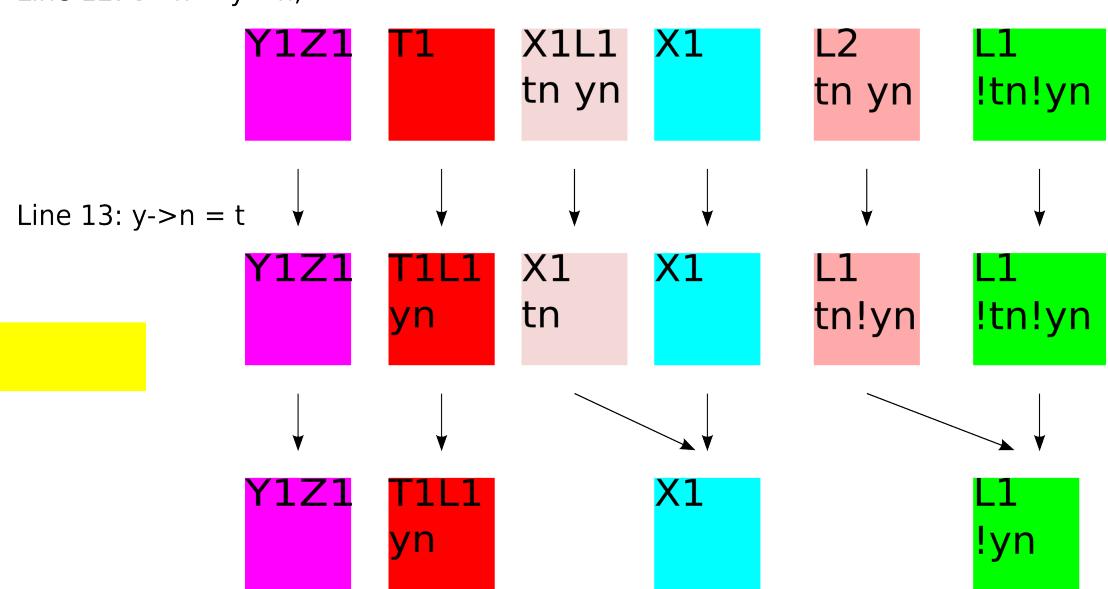
11: $x = t \rightarrow n$

!e0 !e1



Updating Configs: Merging

```
6: List *splice (List *x, List *y) {  
7:     List *t = NULL;           Line 12: t->n = y->n;  
8:     List *z = y;  
9:     while (x != NULL) {  
10:         t = x;  
11:         x = t->n;           Line 13: y->n = t  
12:         t->n = y->n;  
13:         y->n = t;           y->n = t;  
14:         y = y->n->n;  
15:     }  
16:     return z;  
17: }
```



Final Configuration Set

```

8: z = y;

9: while (x != NULL)

10: t = x;

11: x = t->n;

12: t->n = y->n;

13: y->n = t;

14: y = y->n->n;

16: return z;

```

