Backpatching

Lecture 23

Sections 8.4, 8.6

Robb T. Koether

Hampden-Sydney College

Wed, Apr 15, 2009

Outline

The “next” Destination of a Statement

- Every statement will be represented as a `LinkedList` object in the grammar.
- The linked list will be a list of all backpatch labels occurring within the statement that must be resolved to the “next” destination of that statement.
- In the grammar, we must write

```
nonterminal LinkedList stmts, stmt, n;
```

Labels and Jumps in the Grammar

- The productions that will involve jumps are

$$
\begin{aligned}
stmts &\rightarrow stmts\ m\ stmt \\
stmt &\rightarrow \textbf{if}\ (\ cexpr\ )\ m\ stmt \\
stmt &\rightarrow \textbf{if}\ (\ cexpr\ )\ m\ stmt\ n\ \textbf{else}\ m\ stmt \\
func &\rightarrow fbeg\ stmts\ m\ \}
\end{aligned}
$$

- m represents a destination.
- n represents a jump.

Sequences of Statements

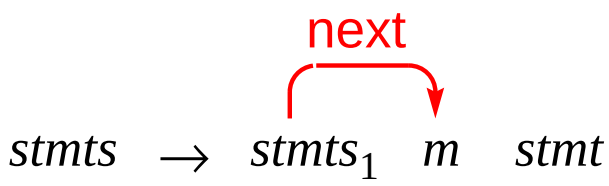
- Every statement has a “next” destination.
- Normally, but not always, this is the next statement.
- This production will insert labels between consecutive statements, even though most of them will not be used.

Sequences of Statements

$$stmts \quad \rightarrow \quad stmts_1 \quad m \quad stmt$$

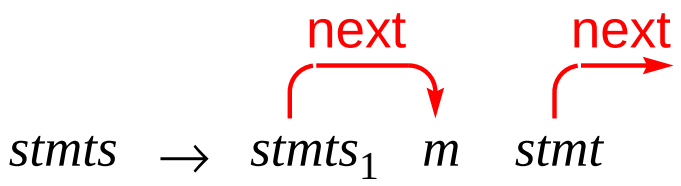
- Consider the production for sequences of statements.

Sequences of Statements



- The label at m serves as the "next" destination of $stmt_1$.

Sequences of Statements



- The label at m serves as the "next" destination of $stmt_1$.
- The "next" destination from $stmt$ becomes the "next" destination from $stmts$ (a synthesized attribute).

Sequences of Statements

- For sequences of statements, we take the following actions.
  - Backpatch $stmt_1$.nextList to m.
  - Return $stmt$.nextList.
- Then $stmt$.nextList becomes nextList of $stmts$, i.e., a synthesized attribute.
- This resolves the "next" destination from $stmt_1$ and leaves the "next" destination from $stmt$ unresolved.

The One-Way `if` Statement

$$stmt \rightarrow \textbf{if} \quad (\; cexpr \;) \quad m \quad stmt_1$$

- Now consider the one-way `if` statement.

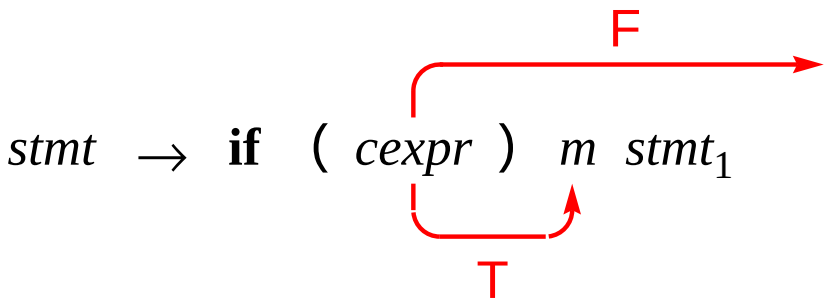
The One-Way `if` Statement

$$stmt \rightarrow \textbf{if} \quad (\; cexpr \;) \quad m \quad stmt_1$$

T

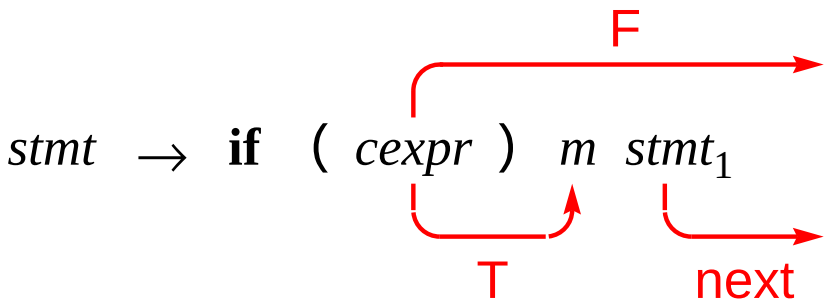
- The "true" list of the backpatch node associated with $cexpr$ will be resolved to m.

The One-Way `if` Statement



- The "true" list of the backpatch node associated with $cexpr$ will be resolved to m.
- The "false" list of $cexpr$ will later be resolved to the label following $stmt_1$.

The One-Way `if` Statement



- The "true" list of the backpatch node associated with $cexpr$ will be resolved to m.
- The "false" list of $cexpr$ will later be resolved to the label following $stmt_1$.
- The "false" list of $cexpr$ and the "next" list of $stmt_1$ have the same destination, which is the "next" destination of $stmt$.

The One-Way `if` Statement

- For the one-way **`if`** statement, we take the following actions.
  - Backpatch $cexpr$.trueList to m.
  - Merge $cexpr$.falseList and $stmt_1$.nextList.
  - Return the merged list.
- The merged list becomes the nextList of $stmt$.

Example

- Consider the following segment of code.

```
if (a)
    b = 900;
a = 200;
```

Example


```
JUMPT INT
   BLABEL blabel=3
   CMPNE INT
      NUM INT value=0
      DEREF INT
         NAME PTR|INT value="a"

JUMP INT
   BLABEL blabel=4

LABEL label=5

ASSIGN INT
   NAME PTR|INT value="b"
   NUM INT value=900
```

```
EQU
   BLABEL blabel=3
   LABEL label=5

LABEL label=6

ASSIGN INT
   NAME PTR|INT value="a"
   NUM INT value=200

EQU
   BLABEL blabel=4
   LABEL label=6
```

Example


```
JUMPT INT
   BLABEL blabel=3
   CMPNE INT
      NUM INT value=0
      DEREF INT
         NAME PTR|INT value="a"

JUMP INT
   BLABEL blabel=4

LABEL label=5

ASSIGN INT
   NAME PTR|INT value="b"
   NUM INT value=900
```

```
EQU
   BLABEL blabel=3
   LABEL label=5

LABEL label=6

ASSIGN INT
   NAME PTR|INT value="a"
   NUM INT value=200

EQU
   BLABEL blabel=4
   LABEL label=6
```

Example


```
JUMPT INT
   BLABEL blabel=3
   CMPNE INT
      NUM INT value=0
      DEREF INT
         NAME PTR|INT value="a"

JUMP INT
   BLABEL blabel=4

LABEL label=5

ASSIGN INT
   NAME PTR|INT value="b"
   NUM INT value=900
```

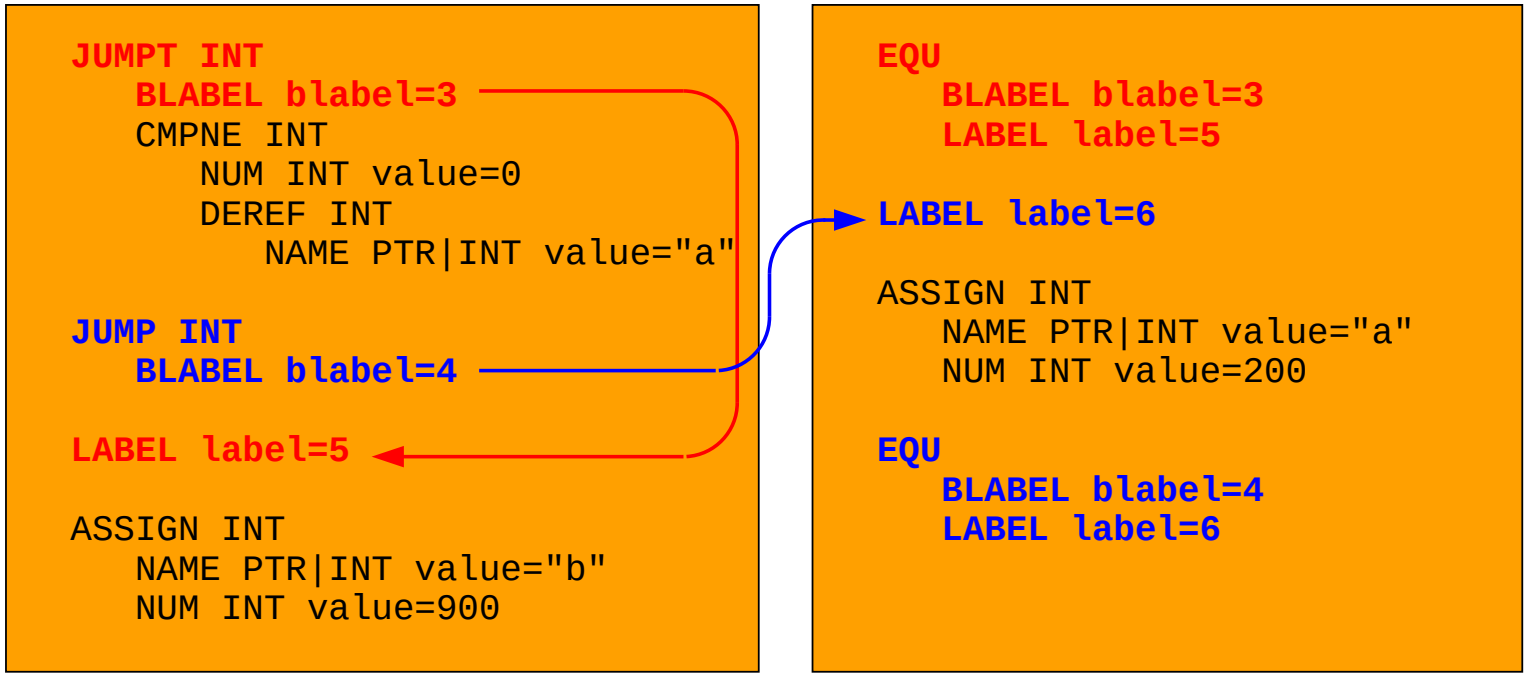
```
EQU
   BLABEL blabel=3
   LABEL label=5

LABEL label=6

ASSIGN INT
   NAME PTR|INT value="a"
   NUM INT value=200

EQU
   BLABEL blabel=4
   LABEL label=6
```

Example


```
JUMPT INT
   BLABEL blabel=3
   CMPNE INT
      NUM INT value=0
      DEREF INT
         NAME PTR|INT value="a"

JUMP INT
   BLABEL blabel=4

LABEL label=5

ASSIGN INT
   NAME PTR|INT value="b"
   NUM INT value=900
```

```
EQU
   BLABEL blabel=3
   LABEL label=5

LABEL label=6

ASSIGN INT
   NAME PTR|INT value="a"
   NUM INT value=200

EQU
   BLABEL blabel=4
   LABEL label=6
```

Example

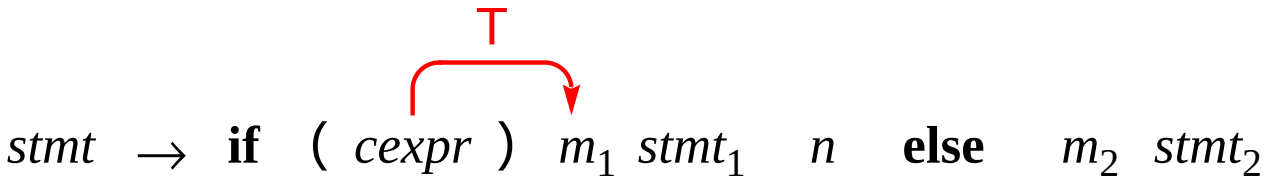

JUMPT INT
BLABEL blabel=3
CMPNE INT
NUM INT value=0
DEREF INT
NAME PTR|INT value="a"
JUMP INT
BLABEL blabel=4
LABEL label=5
ASSIGN INT
NAME PTR|INT value="b"
NUM INT value=900
EQU
BLABEL blabel=3
LABEL label=5
LABEL label=6
ASSIGN INT
NAME PTR|INT value="a"
NUM INT value=200
EQU
BLABEL blabel=4
LABEL label=6

Two-Way `if` Statements

$$stmt \rightarrow \textbf{if} \quad (\; cexpr \;) \quad m_1 \; stmt_1 \quad n \quad \textbf{else} \quad m_2 \; stmt_2$$

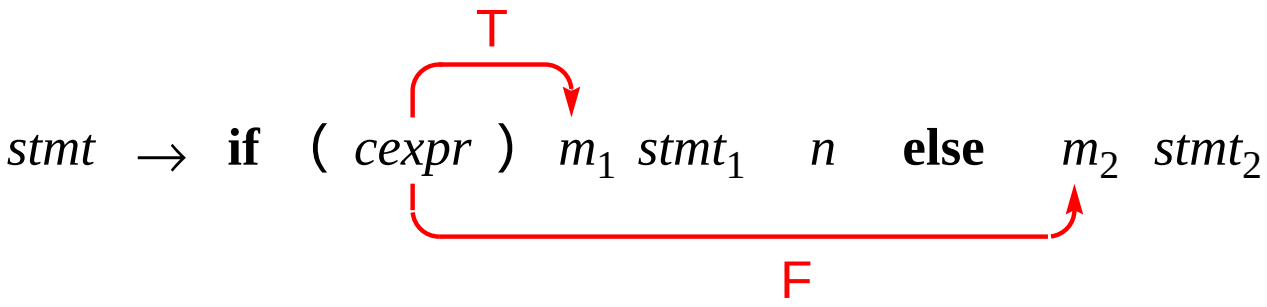
- Consider the two-way `if` statement.

Two-Way `if` Statements



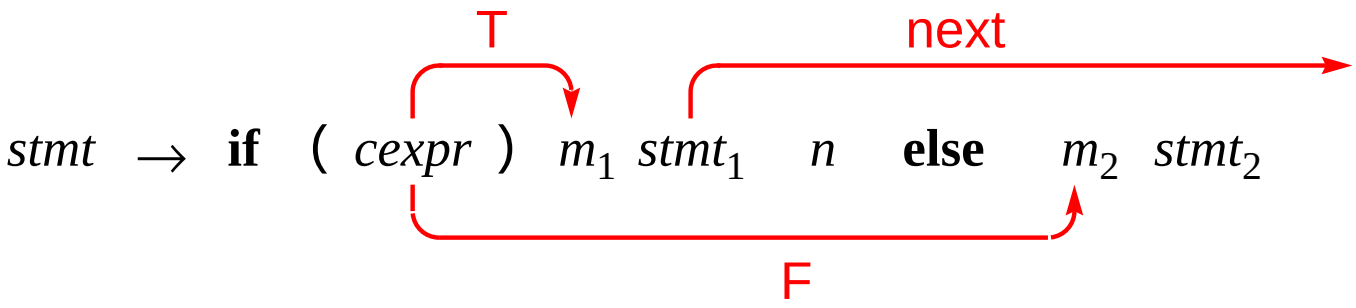
- The "true" destination from $cexpr$ is m_1.

Two-Way `if` Statements



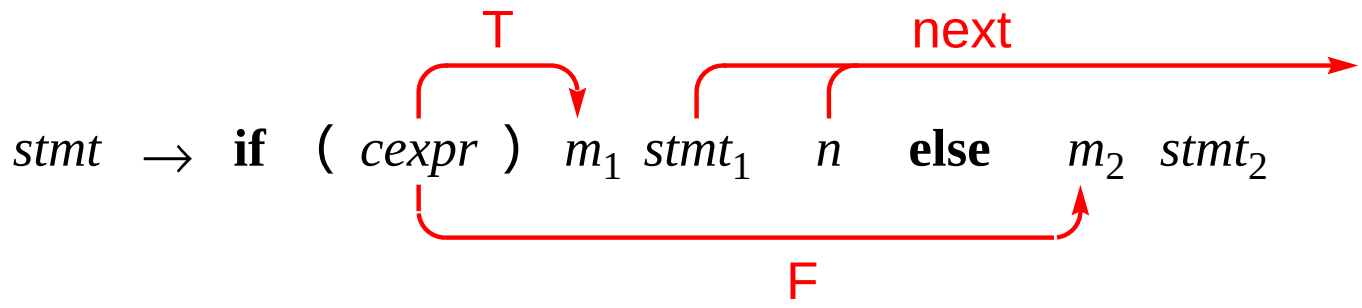
- The "true" destination from $cexpr$ is m_1.
- The "false" destination from $cexpr$ is m_2.

Two-Way `if` Statements



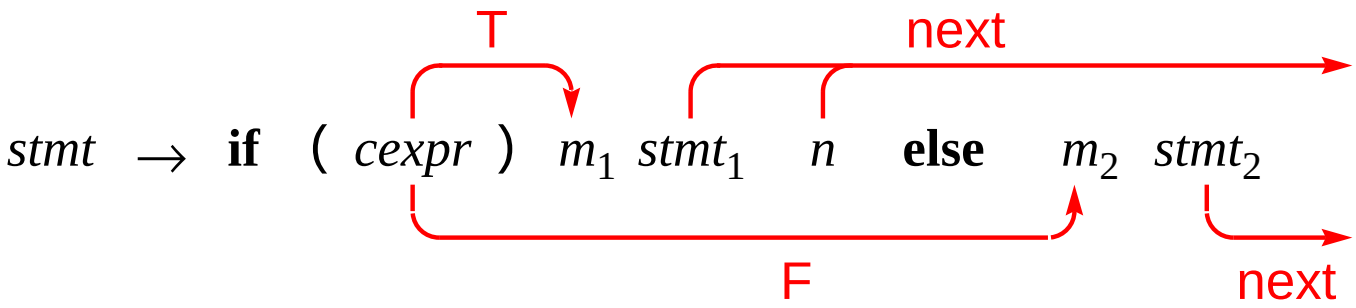
- The "true" destination from $cexpr$ is m_1.
- The "false" destination from $cexpr$ is m_2.
- The "next" destination of $stmt_1$...

Two-Way `if` Statements



- The "true" destination from $cexpr$ is m_1.
- The "false" destination from $cexpr$ is m_2.
- The "next" destination of $stmt_1$ is the same as the "next" destination of n.

Two-Way `if` Statements



- The "true" destination from $cexpr$ is m_1.
- The "false" destination from $cexpr$ is m_2.
- The "next" destination of $stmt_1$ is the same as the "next" destination of n.
- n represents a jump to the "next" destination of $stmt_2$, which becomes the "next" destination of $stmt$.

Two-Way `if` Statements

- For a two-way **`if`** statement, we must take the following actions.
  - Backpatch $cexpr$.trueList to m_1.
  - Backpatch $cexpr$.falseList to m_2.
  - Merge $stmt_1$.nextList, n.nextList, and $stmt_2$.nextList.
- The merged list becomes the nextList of $stmt$.

Example

- Consider the following segment of code.

```
if (a)
    b = 900;
else
    b = 500;
a = 200;
```

Example


```
JUMPT INT
   BLABEL blabel=3
   CMPNE INT
      NUM INT value=0
      DEREF INT
         NAME PTR|INT value="a"

JUMP INT
   BLABEL blabel=4

LABEL label=5

ASSIGN INT
   NAME PTR|INT value="b"
   NUM INT value=900

JUMP INT
   BLABEL blabel=6

LABEL label=7
```

```
ASSIGN INT
   NAME PTR|INT value="b"
   NUM INT value=500

EQU
   BLABEL blabel=3
   LABEL label=5

EQU
   BLABEL blabel=4
   LABEL label=7

LABEL label=8

ASSIGN INT
   NAME PTR|INT value="a"
   NUM INT value=200

EQU
   BLABEL blabel=6
   LABEL label=8
```

Example


```
JUMPT INT
   BLABEL blabel=3
   CMPNE INT
      NUM INT value=0
      DEREF INT
         NAME PTR|INT value="a"

JUMP INT
   BLABEL blabel=4

LABEL label=5

ASSIGN INT
   NAME PTR|INT value="b"
   NUM INT value=900

JUMP INT
   BLABEL blabel=6

LABEL label=7
```

```
ASSIGN INT
   NAME PTR|INT value="b"
   NUM INT value=500

EQU
   BLABEL blabel=3
   LABEL label=5

EQU
   BLABEL blabel=4
   LABEL label=7

LABEL label=8

ASSIGN INT
   NAME PTR|INT value="a"
   NUM INT value=200

EQU
   BLABEL blabel=6
   LABEL label=8
```

Example


```
JUMPT INT
   BLABEL blabel=3
   CMPNE INT
      NUM INT value=0
      DEREF INT
         NAME PTR|INT value="a"

JUMP INT
   BLABEL blabel=4

LABEL label=5

ASSIGN INT
   NAME PTR|INT value="b"
   NUM INT value=900

JUMP INT
   BLABEL blabel=6

LABEL label=7
```

```
ASSIGN INT
   NAME PTR|INT value="b"
   NUM INT value=500

EQU
   BLABEL blabel=3
   LABEL label=5

EQU
   BLABEL blabel=4
   LABEL label=7

LABEL label=8

ASSIGN INT
   NAME PTR|INT value="a"
   NUM INT value=200

EQU
   BLABEL blabel=6
   LABEL label=8
```

Example


```
JUMPT INT
   BLABEL blabel=3
   CMPNE INT
      NUM INT value=0
      DEREF INT
         NAME PTR|INT value="a"

JUMP INT
   BLABEL blabel=4

LABEL label=5

ASSIGN INT
   NAME PTR|INT value="b"
   NUM INT value=900

JUMP INT
   BLABEL blabel=6

LABEL label=7
```

```
ASSIGN INT
   NAME PTR|INT value="b"
   NUM INT value=500

EQU
   BLABEL blabel=3
   LABEL label=5

EQU
   BLABEL blabel=4
   LABEL label=7

LABEL label=8

ASSIGN INT
   NAME PTR|INT value="a"
   NUM INT value=200

EQU
   BLABEL blabel=6
   LABEL label=8
```

Example


```
JUMPT INT
   BLABEL blabel=3
   CMPNE INT
      NUM INT value=0
      DEREF INT
         NAME PTR|INT value="a"

JUMP INT
   BLABEL blabel=4

LABEL label=5

ASSIGN INT
   NAME PTR|INT value="b"
   NUM INT value=900

JUMP INT
   BLABEL blabel=6

LABEL label=7
```

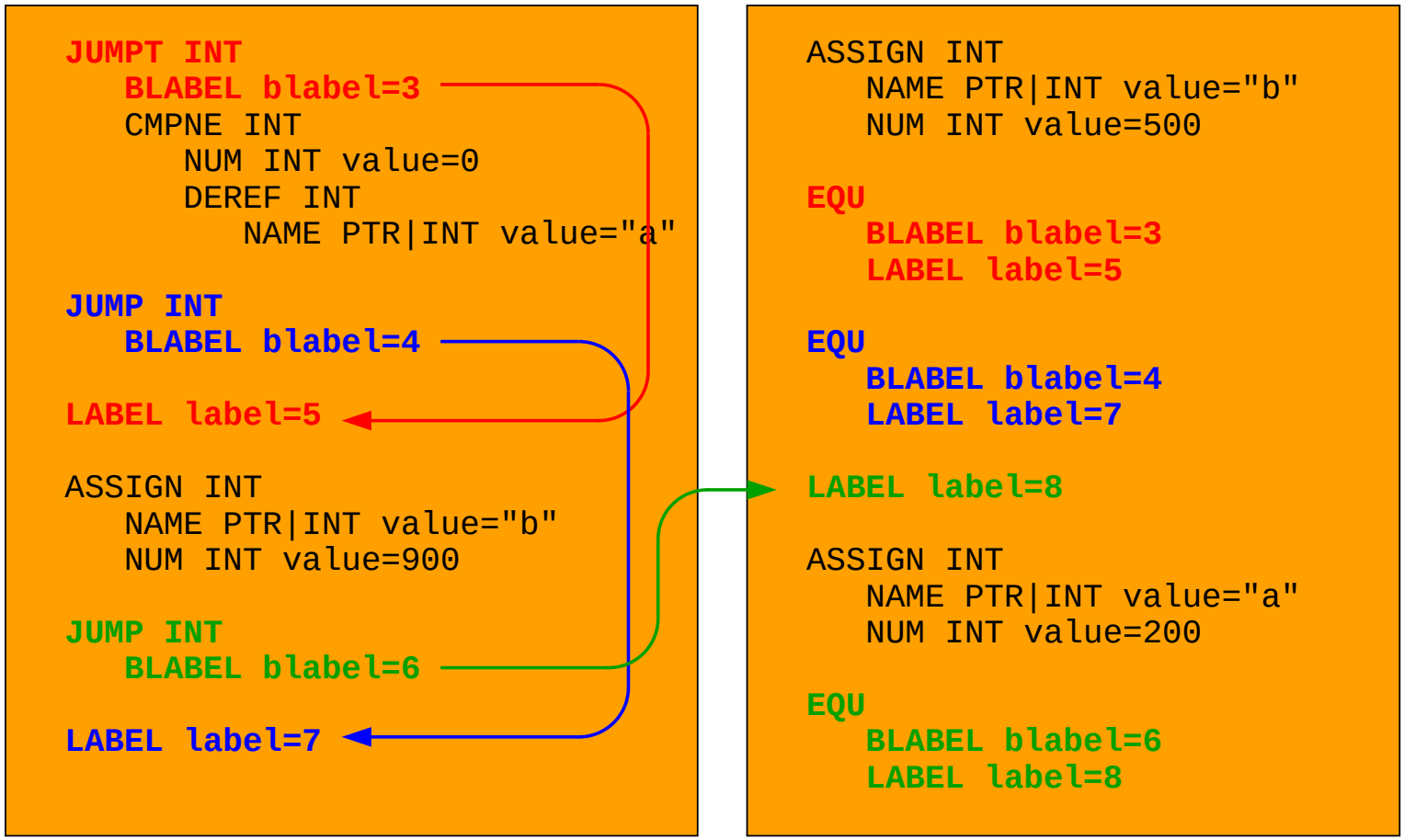
```
ASSIGN INT
   NAME PTR|INT value="b"
   NUM INT value=500

EQU
   BLABEL blabel=3
   LABEL label=5

EQU
   BLABEL blabel=4
   LABEL label=7

LABEL label=8

ASSIGN INT
   NAME PTR|INT value="a"
   NUM INT value=200

EQU
   BLABEL blabel=6
   LABEL label=8
```

Example


```
JUMPT INT
   BLABEL blabel=3
   CMPNE INT
      NUM INT value=0
      DEREF INT
         NAME PTR|INT value="a"

JUMP INT
   BLABEL blabel=4

LABEL label=5

ASSIGN INT
   NAME PTR|INT value="b"
   NUM INT value=900

JUMP INT
   BLABEL blabel=6

LABEL label=7
```

```
ASSIGN INT
   NAME PTR|INT value="b"
   NUM INT value=500

EQU
   BLABEL blabel=3
   LABEL label=5

EQU
   BLABEL blabel=4
   LABEL label=7

LABEL label=8

ASSIGN INT
   NAME PTR|INT value="a"
   NUM INT value=200

EQU
   BLABEL blabel=6
   LABEL label=8
```

Example


JUMPT INT
BLABEL blabel=3
CMPNE INT
NUM INT value=0
DEREF INT
NAME PTR|INT value="a"
JUMP INT
BLABEL blabel=4
LABEL label=5
ASSIGN INT
NAME PTR|INT value="b"
NUM INT value=900
JUMP INT
BLABEL blabel=6
LABEL label=7
ASSIGN INT
NAME PTR|INT value="b"
NUM INT value=500
EQU
BLABEL blabel=3
LABEL label=5
EQU
BLABEL blabel=4
LABEL label=7
LABEL label=8
ASSIGN INT
NAME PTR|INT value="a"
NUM INT value=200
EQU
BLABEL blabel=6
LABEL label=8

Backpatching at a Function End

- Consider the production

$$func \quad \rightarrow \quad fbeg\ stmts\ m\ \texttt{RBRACE}$$

- $stmts$ is a linked list of backpatch labels that must be resolved.
- If we reach the end of the function without yet resolving them, then they must be resolved to the return that occurs at the end of the function.

Backpatching at a Function End

- When the code is generated for this production, it is important that the backpatching occur first, before the return.

Assignment

Homework

- Read Section 8.4, pages 491 - 493.
- Read Section 8.6, pages 504 - 506.