

Compiler 5-3

孔静 2014K8009929022

5.4.4

$S \rightarrow \text{if}(C) \ S_1 \ \text{else} \ S_2$	$L_1 = \text{new}()$ $L_2 = \text{new}()$ $C.\text{true} = L_1$ $C.\text{false} = L_2$ $S_1.\text{next} = S.\text{next}$ $S_2.\text{next} = S.\text{next}$ $S.\text{code} = C.\text{code} \text{label} L_1 S_1.\text{code} \text{goto } S_1.\text{next} \text{label} L_2 S_2.\text{code}$
$S \rightarrow \text{do } S_1 \ \text{while} (C)$	$L_1 = \text{new}()$ $L_2 = \text{new}()$ $C.\text{true} = L_1$ $C.\text{false} = S.\text{next}$ $S_1.\text{next} = L_2$ $S.\text{code} = \text{label} L_1 S_1.\text{code} \text{label} L_2 C.\text{code}$
$S \rightarrow \{'L'\}$ $L \rightarrow L_1 S$ $L \rightarrow \varepsilon$	$L.\text{next} = S.\text{next}$ $S.\text{code} = L.\text{code}$ $L_2 = \text{new}()$ $L_1.\text{next} = L_2$ $S.\text{next} = L.\text{next}$ $L.\text{code} = L_1.\text{code} \text{label} L_2 S.\text{code}$ $L.\text{code} = \text{" "}$

5.5.4

-
- Diagram illustrating the state transitions and stack updates for the Synthesize action, showing the flow from the initial state through various actions and stack manipulations.
- Initial State:**
- top
 - S
 - next=x
- Synthesize S.code**
- code=?
 - data
- top**
- if**
- Actions:**
- Action**
 - snext=x
 - L₁=?
 - L₂=?
 - C**
 - false=?
 - true=?
 - Synthesize C.code**
 - code=?
 - S₁**
 - next=?
 - Synthesize S₁.code**
 - code=?
 - else**
 - S₂**
 - next=?
 - Synthesize S₂.code**
 - code=?
 - l2=?
 - l1=?
 - Ccode=?
 - S₁code=?
 - Synthesize S.code**
 - code=?
 - data
 - actions
- Stack Updates:**
- stack[top-8].Ccode=code;
 - stack[top-3].S₁ code=code;
 - stack[top-4].S₁next=next;
 - stack[top-1].code = Ccode || "label" || l1 || S₁ code || goto "l2" || "label" || l2 || code;
- Code Block:**
- ```

L1 = new ();
L2 = new ();
stack[top - 1].true = L1;
stack[top - 1].false = L2;
stack[top - 8].l1 = L1;
stack[top - 8].l2 = L2;
stack[top - 4].next = snext;
stack[top - 7].next = snext;

```

- 
- The diagram illustrates the Synthesize algorithm for a while loop. It shows a sequence of actions and their corresponding code synthesis steps.
- Initial State:**
- top** points to a box containing **S** and **next=x**.
  - Synthesize S.code** box contains **code=?** and **data**.
- Loop Header:**
- do** (action) points to **do** (action).
  - do** (action) box contains **snext=x**.
  - S<sub>1</sub>** (action) box contains **next=?**.
  - Synthesize S<sub>1</sub>.code** box contains **code=?**.
  - while** (action) box contains **code=?**.
  - (** (action) box contains **code=?**.
- Loop Body:**
- C** (action) box contains **true=?** and **false=?**.
  - Synthesize C.code** box contains **code=?** and **S<sub>1</sub>code=?**.
  - l1=?** and **l2=?** (actions) boxes are shown below the Synthesize C.code box.
  - )** (action) box contains **code=?**.
  - Synthesize S.code** box contains **code=?** and **data**.
- Code Synthesis:**
- The **do** (action) box is linked to a large box containing the following code:
 

```

L1 = new ();
L2 = new ();
stack[top - 1].next = L2;
stack[top - 5].true = L1;
stack[top - 5].false = snext;
stack[top - 6].l1 = L1;
stack[top - 6].l2 = L2;

```
  - The **Synthesize S<sub>1</sub>.code** box is linked to a box containing the code:
 

```

stack[top-5].S1code=code;

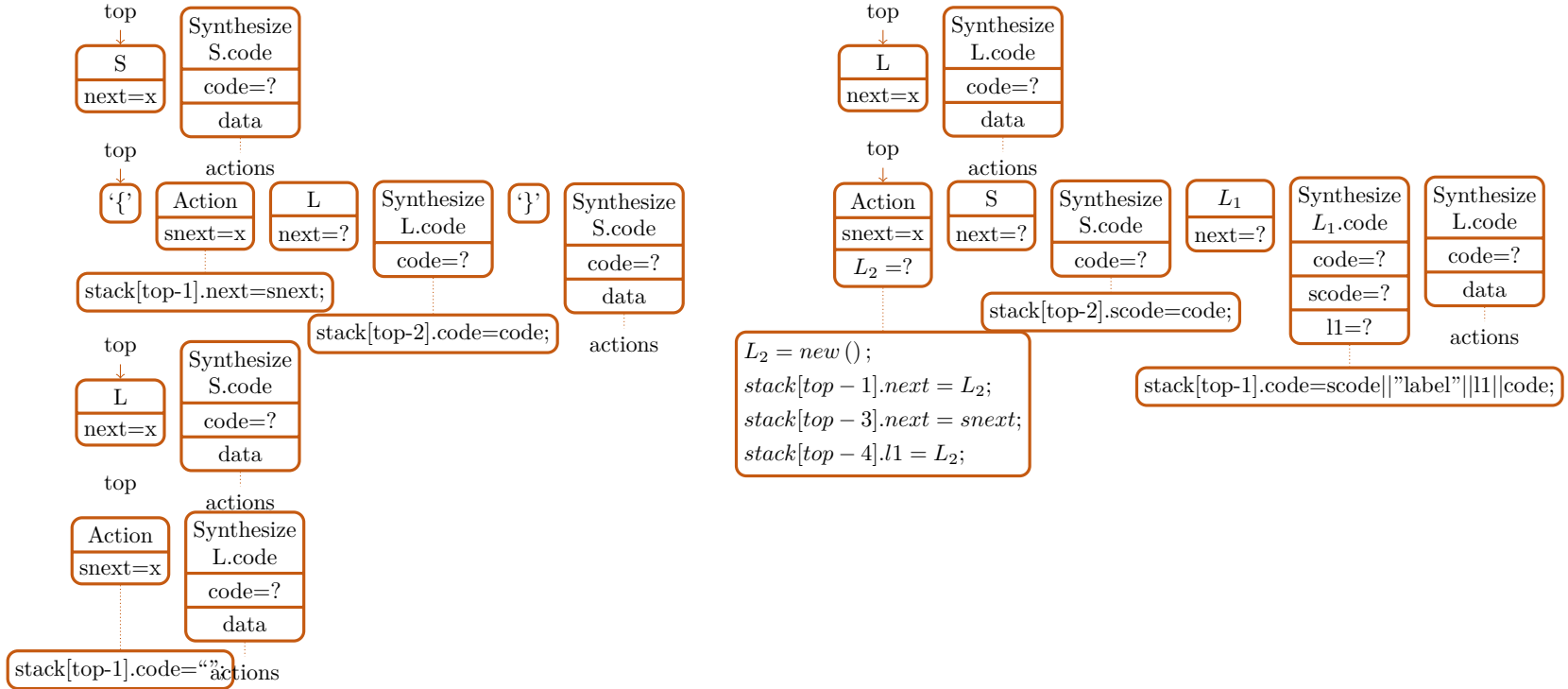
```
  - The **Synthesize C.code** box is linked to a box containing the code:
 

```

stack[top - 2].code = "label" ||
l1 || S1code || "label" || l2 || code;

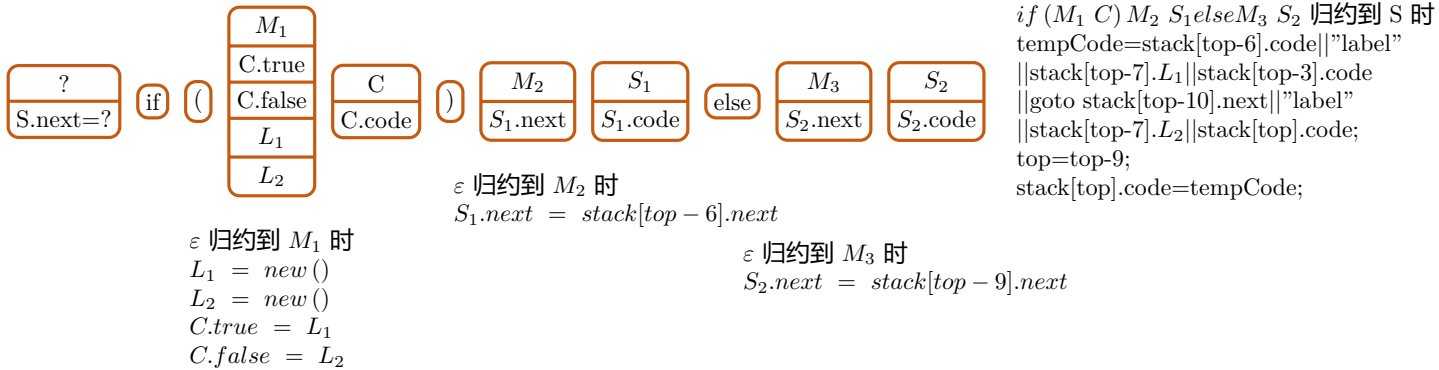
```

• 3

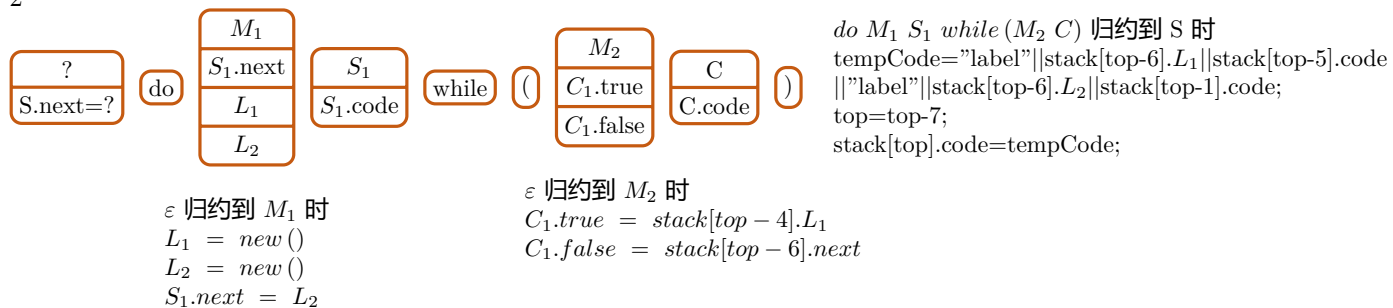


### 5.5.5

• 1



• 2



• 3

