习题课

2024/12

苏程浩

Outline (Part I)

1. 词法分析
2. 语法分析
3. 语法制导翻译
4. 中间代码生成

词法分析：回顾

正则表达式

- 设计与描述
- 构造NFA

NFA

- NFA的表示
- 子集构造法

DFA

- 最小化

注意：从正则表达式到状态转换图不一定要按照NFA到DFA这个流程走

词法分析作业讲解

E1 问题3: 根据正则表达式给出的状态转换图

我们有这些思路：

1. 直接看出描述的语言，然后构造一个状态转换图
   - 例： $((\epsilon|a)b^*)^* = (a \mid b)^*$
   - 优点：方便
   - 缺点：依赖直观，容易出错
     - 代数定律不是一个非常好的工具
     - 务必要小心确保描述的语言不变
2. 从正则表达式到NFA再到DFA

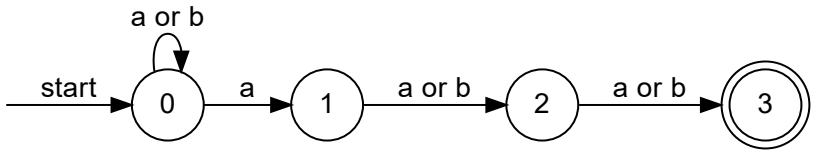
定律	描述
$r\|s = s\|r$	|是可以交换的
$r\|(s\|t) = (r\|s)\|t$	|是可结合的
$r(st) = (rs)t$	连接是可结合的
$r(s\|t) = rs\|rt;\ (s\|t)r = sr\|tr$	连接对|是可分配的
$\epsilon r = r\epsilon = r$	ϵ是连接的单位元
$r^* = (r\|\epsilon)^*$	闭包中一定包含ϵ
$r^{**} = r^*$	*具有幂等性

图 3-7　正则表达式的代数定律

词法分析作业讲解

E1 问题3: 根据正则表达式给出的状态转换图

2. 从正则表达式到NFA再到DFA
   - 例：$(a \mid b)^* a(a \mid b)(a \mid b)$
     - 注：该状态转换图是NFA
   - 然后依次应用算法
     1. 构造NFA
        1. 在纸上找一个**足够大**的空间
        2. 由内而外，从左到右
     2. 构造DFA
        1. 子集构造
     3. 最小化
        1. 区分接受状态和普通状态，一步步迭代

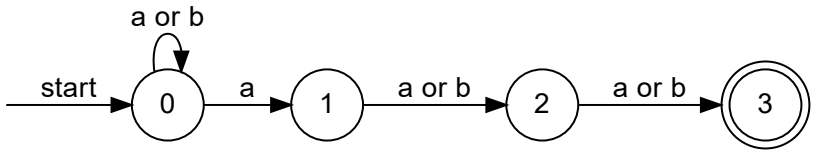


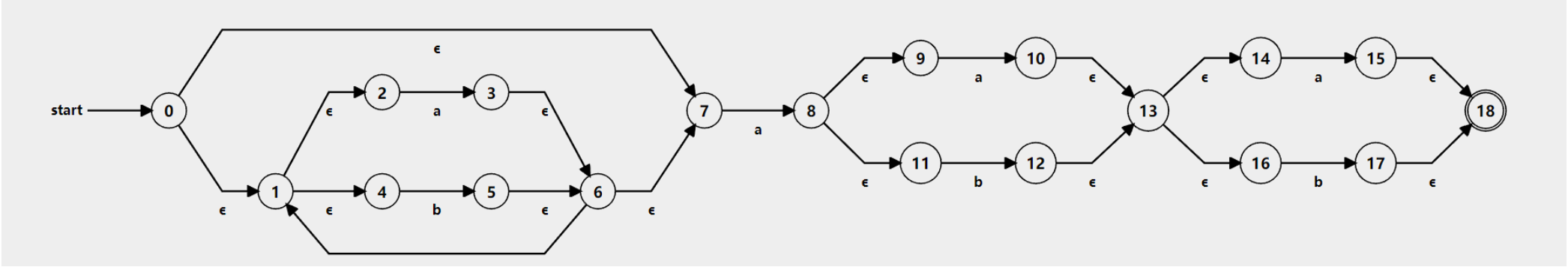
词法分析作业讲解

E1 问题3: 根据正则表达式给出的状态转换图

2. 从正则表达式到NFA再到DFA
   - 例：$(a \mid b)^* a(a \mid b)(a \mid b)$

NFA



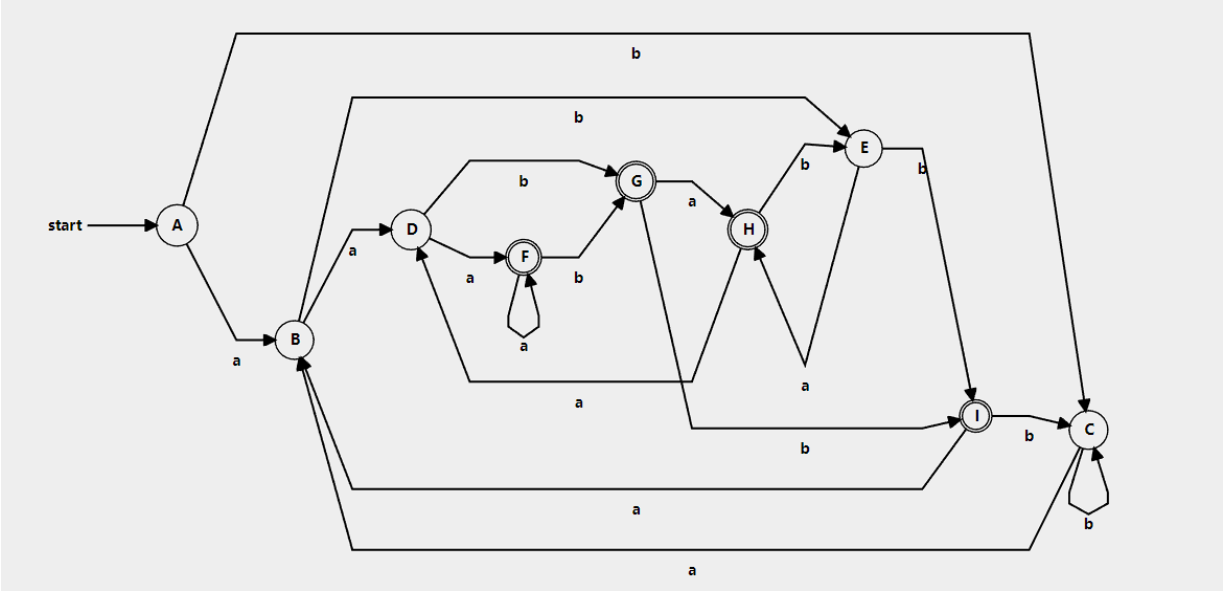


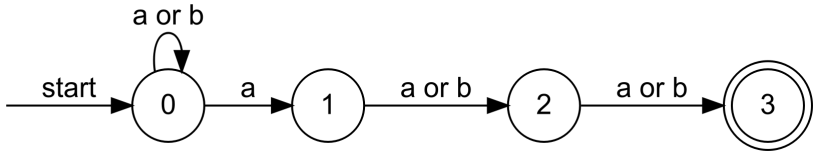
词法分析作业讲解

E1 问题3: 根据正则表达式给出的状态转换图

2. 从正则表达式到NFA再到DFA
   - 例：$(a \mid b)^* a(a \mid b)(a \mid b)$

DFA



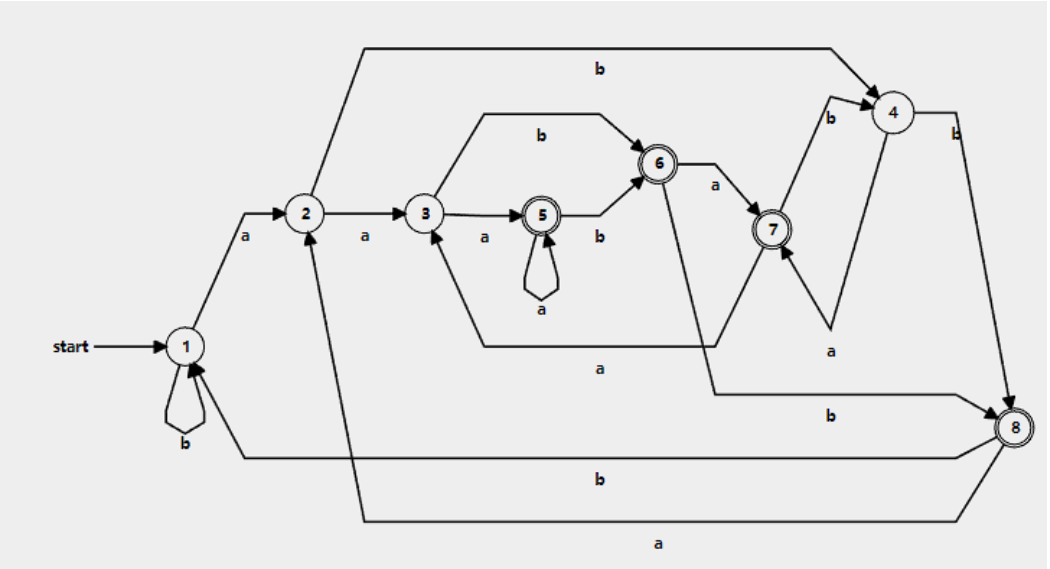


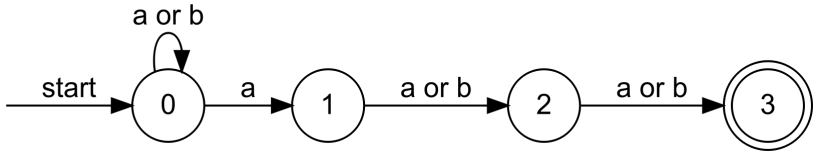
词法分析作业讲解

E1 问题3: 根据正则表达式给出的状态转换图

2. 从正则表达式到NFA再到DFA
   - 例：$(a \mid b)^* a(a \mid b)(a \mid b)$

最小化





词法分析作业讲解

E1问题2：正则表达式设计

- 思路：
  1. 不含子序列aba
     1. a*
     2. b*
     3. b*a*
     4. a*b*
     5. b*a*b*
  2. 求并，得b*a*b*

问题 2. (原书3.3.5，薄书3.2.5，非原题)若语言$\mathcal{L}$由满足下列条件的句子构成:

1. 由字母a和b构成，可空
2. 不含子序列aba

给出一个符合条件的$\mathcal{L}$正则定义。注意s的子序列是从s中删除零个或多个符号后得到的串。

语法分析：回顾

文法

- 文法设计
- First & Follow

LL

- 消除左递归
- 提取左公因式
- LL(1)预测分析表

LR

- LR(0)自动机构造
- 从自动机到分析表
- LR(1)与LALR项集族

语法分析：问题

语法分析的题目计算比较繁琐，容易出错。

1. First/Follow ✓
2. 消除左递归算法 ✓
    1. 注意消除时机
3. LL分析表构造 ✓
4. LR自动机与LR分析表构造 ✗

语法分析作业题讲解

E2 问题4

问题 4. (原书4.4.1，薄书4.4.1)为下面的文法给出预测分析表，你可能需要先消除左递归

$$S \rightarrow (L) \mid a$$
$$L \rightarrow L, S \mid S$$

1. 消除左递归：
2. 计算First和Follow集
    1. 注意Follow的规则2
3. 给出预测分析表

语法分析作业题讲解

E2 问题4

问题 4. (原书4.4.1，薄书4.4.1)为下面的文法给出预测分析表，你可能需要先消除左递归

$$S \rightarrow (L) \mid a$$
$$L \rightarrow L, S \mid S$$

1. 消除左递归：
   1. 直接消除L的立即左递归：

$$S \rightarrow (L) \mid a$$
$$L \rightarrow SL'$$
$$L' \rightarrow , SL'$$
$$L' \rightarrow \epsilon$$

语法分析作业题讲解

E2 问题4

问题 4. (原书4.4.1，薄书4.4.1)为下面的文法给出预测分析表，你可能需要先消除左递归

$$S \rightarrow (L) \mid a$$
$$L \rightarrow L, S \mid S$$

1. 消除左递归：
   2. 使用消除左递归算法全体消除（尽管没有意义）

$$S \rightarrow (L) | a$$
$$L \rightarrow (L)L' | aL'$$
$$L' \rightarrow , SL'$$
$$L' \rightarrow \epsilon$$

语法分析作业题讲解

E2 问题4

问题 4. (原书4.4.1，薄书4.4.1)为下面的文法给出预测分析表，你可能需要先消除左递归

$$S \rightarrow (L) \mid a$$
$$L \rightarrow L, S \mid S$$

2. 计算First与Follow：

$$S \rightarrow (L)|a$$
$$L \rightarrow SL'$$
$$L' \rightarrow, SL'$$
$$L' \rightarrow \epsilon$$

符号	First	Follow
S	(a	$,
L	(a	)
L’	, epsilon	)

语法分析作业题讲解

E2 问题4

问题 4. (原书4.4.1，薄书4.4.1)为下面的文法给出预测分析表，你可能需要先消除左递归

$$S \rightarrow (L) \mid a$$
$$L \rightarrow L, S \mid S$$

3. 预测分析表：

$S \rightarrow (L)|a$
$L \rightarrow SL'$
$L' \rightarrow ,SL'$
$L' \rightarrow \epsilon$

符号	First	Follow
S	(a	$,
L	(a	)
L’	, epsilon	)

非终结符号	输入符号				
	(	)	a	,	$
S	S→(L)		S→a		
L	L→SL’		L→SL’		
L’		L’→ε		L’→,SL’	

附加题

归纳证明

$$S \longrightarrow S\ S\ + \mid S\ S\ * \mid a$$

文法的归纳含义：

1. a在S中
2. 如果u,v在S中，那么uv+也在S中
3. 如果u,v在S中，那么uv*也在S中
4. 最小（没有其它元素）

5.（附加题）注意到每棵语法树都唯一地对应一个最右推导。对生成的串的长度进行归纳，依次证明：

(a) 对于 S 生成的任何两个串 u 与 v，如果 u 是 v 的后缀，则 $u = v$

(b) 此文法不具有二义性

语法分析作业题讲解

E2 问题12

$$S \longrightarrow S\ S\ + \mid S\ S\ * \mid a$$

5. (原书 4.7.1，薄书 4.7.1) 为该文法构造规范 LR 项集族和 LALR 项集族。

1. 主要问题在于规范LR项集族的构造，容易漏算

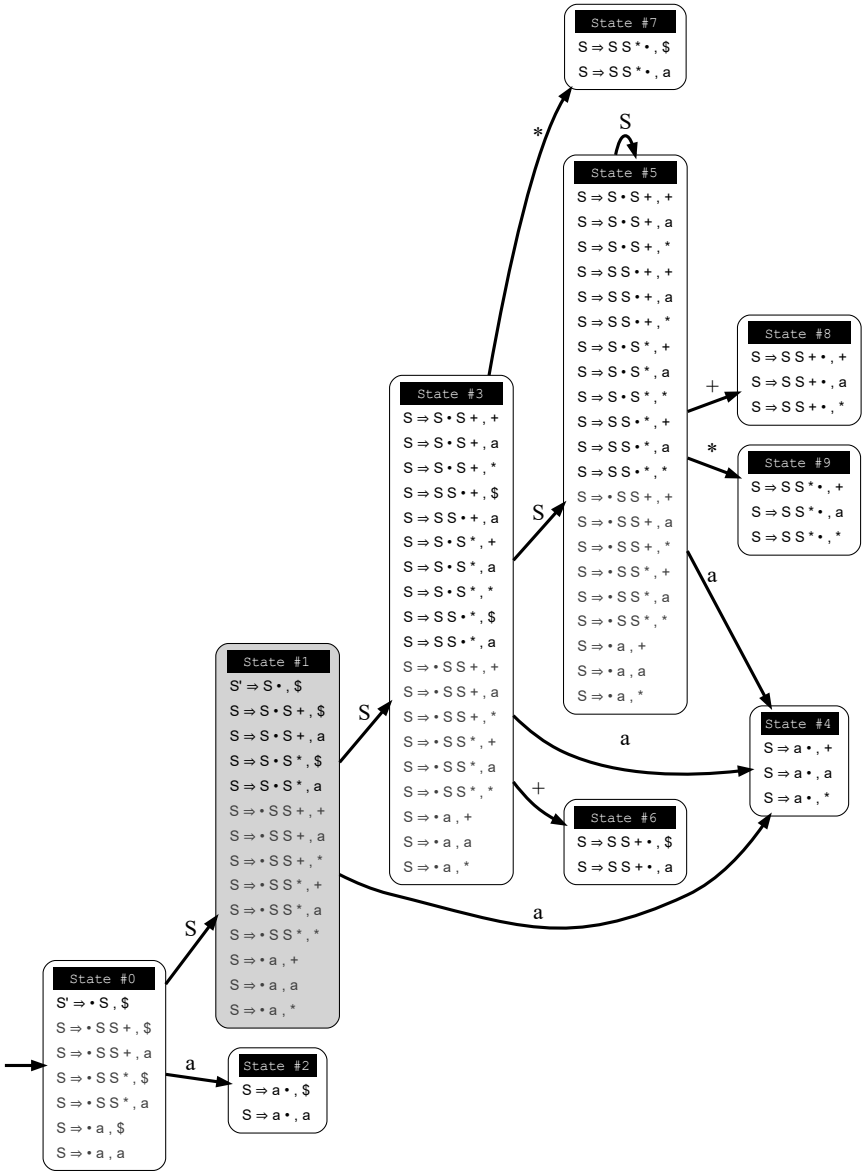
语法分析作业题讲解

E2 问题12

$$S \longrightarrow S\ S\ +\ |\ S\ S\ *\ |\ a$$

5. (原书 4.7.1，薄书 4.7.1) 为该文法构造规范 LR 项集族和 LALR 项集族。

- First(S) = {a}
- Follow(S) = {a,+,*,$}



语法分析作业题讲解

E2 问题12

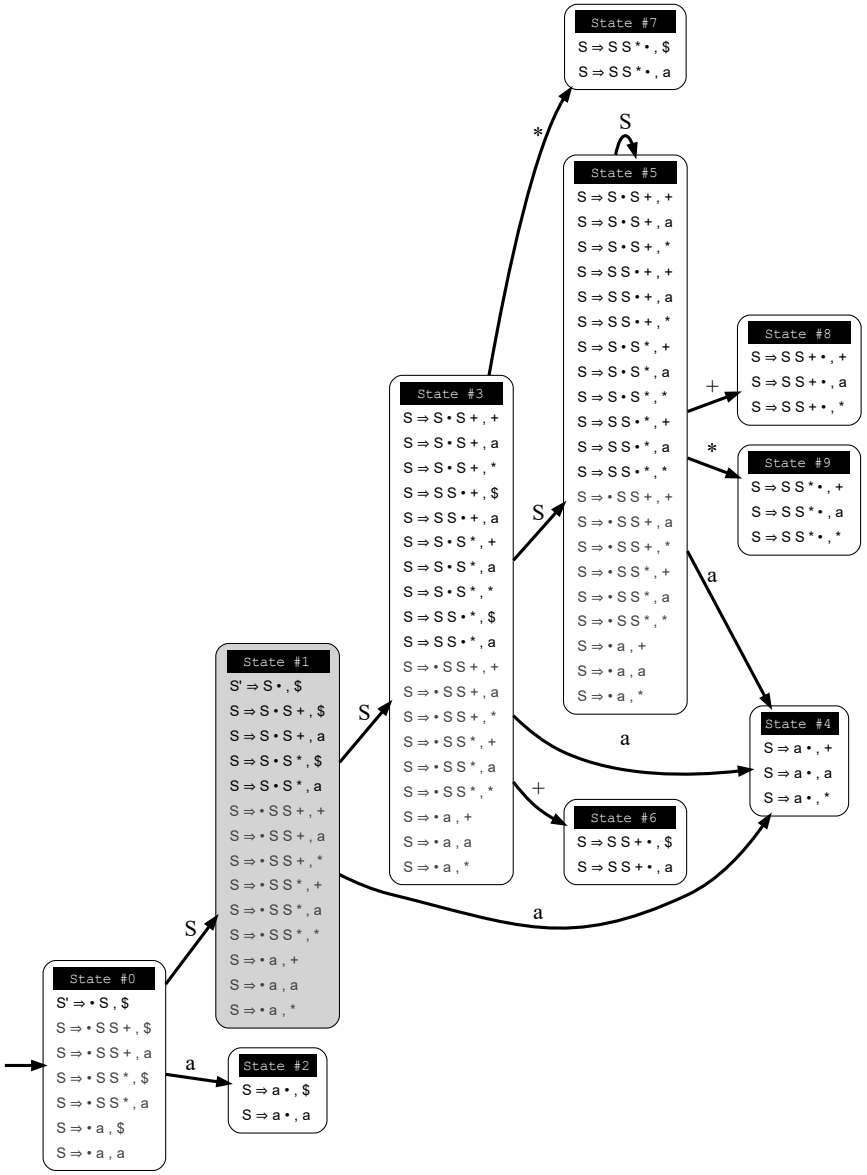
- First(S) = {a}
- Follow(S) = {a,+,*,$}

- 状态2:一次计算

S => S., $
S => S.S+, $/a
S => S.S*, $/a
S => .S S +, +/*
S => .S S *, +/*
S => .a, + /*

- 状态2:计算完毕

S => S., $
S => S.S+, $/a
S => S.S*, $/a
S => .S S +, +/*/a
S => .S S *, +/*/a
S => .a, + /*/a



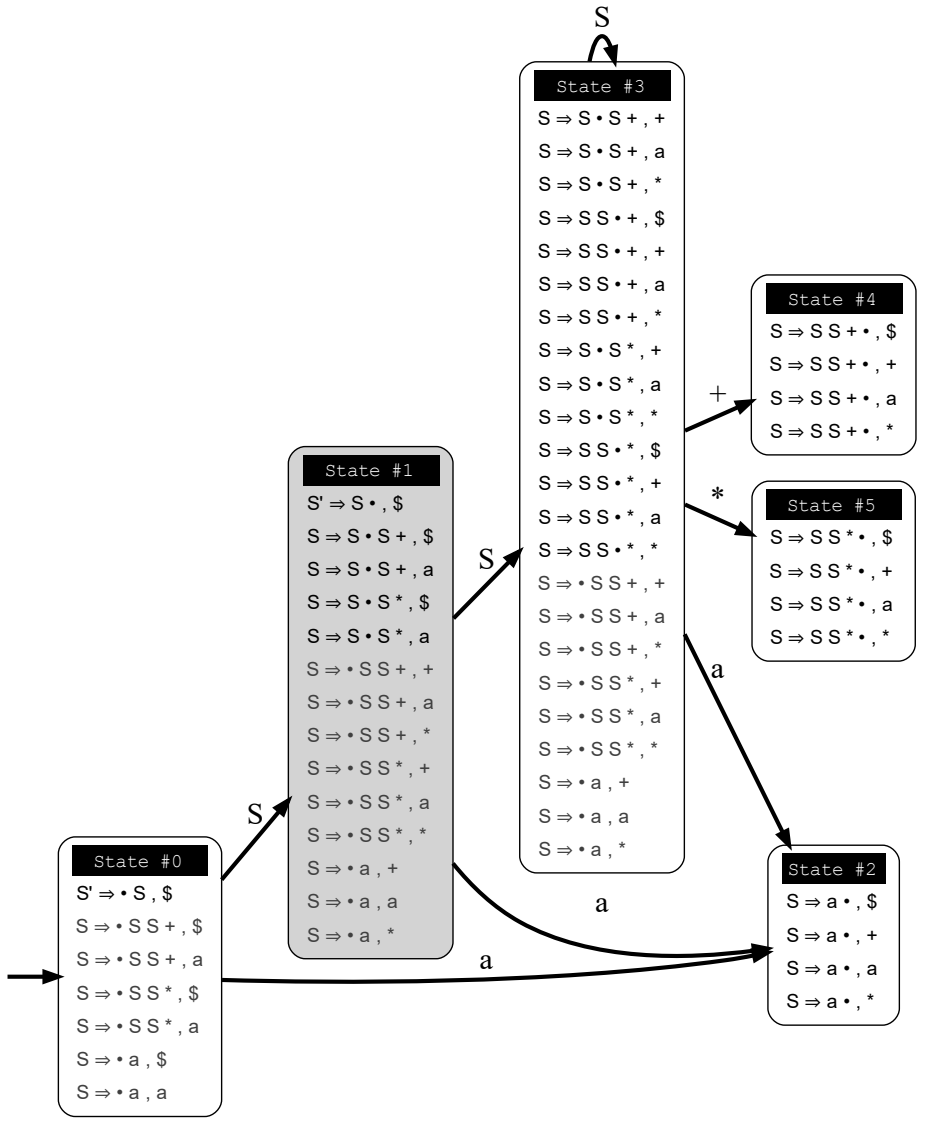
语法分析作业题讲解

E2 问题12

问题 2. (原书4.7.1，薄书4.7.1)为文法$S \to SS+ \mid SS* \mid a$构造：

1. 规范LR项集族
2. LALR项集族

- First(S) = {a}
- Follow(S) = {a,+,*,$}



语法分析随堂练习

大家算算First和Follow集

$$
\begin{aligned}
E &\rightarrow \mathrm{id}\, X \\
X &\rightarrow \varepsilon \,|\, (A) \,|\, (E) \\
A &\rightarrow EY \\
Y &\rightarrow \epsilon \,|\, ; A
\end{aligned}
$$

符号	First	Follow
E	id	$) ;
X	ϵ (	$) ;
A	id	)
Y	ϵ ;	)

语法制导翻译：回顾

SDD

- 综合属性与继承属性
- 注释语法分析树
- 依赖图
- S属性与L属性的SDD

SDT

- S属性SDD与后缀SDT
- L属性的SDD转换成SDT

语法制导翻译：作业题讲解

E4 问题4

Exercise 4:

(原书 5.4.4，薄书 5.4.4) 仿照书中例 5.19，为下面的产生式写出一个 **L 属性**的 SDD，然后转换为 SDT

$$S \rightarrow \texttt{do}\ S_1\ \texttt{while}\ (C)$$

SDD：

$S \rightarrow \textbf{do}\ S_1\ \textbf{while}\ (C)$	$L_1 = \text{new}();\ L_2 = \text{new}()$
	$S_1.\text{next} = L_2$
	$C.\text{true} = L_1$
	$C.\text{false} = S.\text{next}$
	$S.\text{code} = \text{label}\|\|L_1\|\|S_1.\text{code}\|\|\text{label}\|\|L_2\|\|C.\text{code}$

SDT:

$S \rightarrow \textbf{do}$	$\{L_1 = \text{new}();\ L_2 = \text{new}();\ S_1.\text{next} = L_2\}$
$S_1\ \textbf{while}\ ($	$\{C.\text{true} = L_1;\ C.\text{false} = S.\text{next}\}$
$C)$	$\{S.\text{code} = \text{label}\|\|L_1\|\|S_1.\text{code}\|\|\text{label}\|\|L_2\|\|C.\text{code}\}$

语法制导翻译随堂练习

为语言构造while-then-else生成SDD与SDT。

$$S \rightarrow \texttt{while}\, C : S_1\, \texttt{then}\, S_2\, \texttt{else}\, S_3$$

语义：

首先执行条件C，如果成立，执行循环体S2，S2结束后再执行更新语句S1，然后**再执行一次本句**的语义；如果C不成立，执行语句S3。

习题答案

$$S \rightarrow \texttt{while}\, C : S_1\, \texttt{then}\, S_2\, \texttt{else}\, S_3$$

$$L_1 = \text{new}(), L_2 = \text{new}(), L_3 = \text{new}(), L_4 = \text{new}($$

$$\begin{aligned}
C.\text{true} &= L_1 \\
C.\text{false} &= L_2 \\
S_1.\text{next} &= L_3 \\
S_2.\text{next} &= L_4 \\
S_3.\text{next} &= S.\text{next} \\
S.\text{code} &= \texttt{label}\, L_3 \quad C.\text{code} \quad \texttt{label}\, L_1 \quad S_2.\text{code} \\
&\quad\ \texttt{label}\, L_4 \quad S_1.\text{code} \quad \texttt{label}\, L_2 \quad S_3.\text{code}
\end{aligned}$$

$$\begin{aligned}
S \rightarrow\ & \{L_1 = \text{new}() \ , L_2 = \text{new}(), L_3 = \text{new}(), L_4 = \text{new}(\} \\
& \{C.\text{true} = L_1, C.\text{false} = L_2\ \} \\
\texttt{while}\, C\ & \{S_1.\text{next} = L_3\} \\
: S_1\ & \{S_2.\text{next} = L_4\} \\
\texttt{then}\, S_2\ & \{S_3.\text{next} = S.\text{next}\} \\
\texttt{else}\, S_3\ & \{S.\text{code} = \texttt{label}\, L_3 \quad C.\text{code} \quad \texttt{label}\, L_1 \quad S_2.\text{code} \\
& \qquad\qquad\ \texttt{label}\, L_4 \quad S_1.\text{code} \quad \texttt{label}\, L_2 \quad S_3.\text{code}\}
\end{aligned}$$

中间代码生成：回顾

中间代码	类型	表达式的翻译	控制流

• DAG
• 三地址代码
  • 四元式
  • 三元式
	• 类型表达式
	• 类型的宽度
		• 运算
		• 数组元素的寻址
		• 数组的引用
			• 布尔表达式
			• 回填
			• 控制转移语句

中间代码生成：问题

主要的问题是

1. 翻译数组寻址和引用时没考虑宽度
   1. 指明8位宽
2. 赋值时没有计算地址并写入到地址
   1. 没有规定时，大家自己发明的符号也算对
   2. 考试时会写明

中间代码生成：作业讲解

E3 问题5

Exercise 6:

(原书 6.2.2, 薄书 6.2.2) 考虑下列赋值语句

1. $a = b[i] + c[j]$
2. $a[i] = b * c - b * d$

假定每个数组元素占八个存储单元，将赋值语句翻译成：

1. 四元式序列
2. 三元式序列

中间代码生成：作业讲解

E3 问题5

1. $a = b[i] + c[j]$

	op	arg1	arg2	result
0	*	i	8	t_1
1	=[]	b	t_1	t_2
2	*	j	8	t_3
3	=[]	c	t_3	t_4
4	+	t_2	t_4	t_5
5	=	t_5		a
	…			

	op	arg1	arg2
0	*	i	8
1	=[]	b	(0)
2	*	j	8
3	=[]	c	(2)
4	+	(1)	(3)
5	=	a	(4)
	…		

中间代码生成：作业讲解

E3 问题5

2. $a[i] = b * c - b * d$

	op	arg1	arg2	result
0	*	b	c	t_1
1	*	b	d	t_2
2	-	t_1	t_2	t_3
3	*	i	8	t_4
4	[]=	t_4	t_3	a
	…			

	op	arg1	arg2
0	*	b	c
1	*	b	d
2	-	t_1	t_2
3	*	i	8
4	+	a	(3)
5	*=	(4)	(2)
	…		

Outline (Part II)

1. 代码生成
2. 机器无关优化

代码生成：回顾

目标语言

- 一个简单的目标机模型
- 目标代码中的地址
  - 静态内存
  - 栈式内存

基本块与流图

- 基本块的划分
- 基本块形成流图
  - 循环
- 基本块的优化

寄存器分配与指派

- 寄存器与地址描述符
- getReg(I):寄存器选择
- 代码生成算法

代码生成作业讲解

生成机器码

$$
\begin{aligned}
y &= *q \\
q &= q+4 \\
*p &= y \\
p &= p+4
\end{aligned}
$$

$$
\begin{aligned}
y &= *q \\
&\quad \mathrm{LD}\ R_1, q \\
&\quad \mathrm{LD}\ R_2, 0(R_1) \\
&\quad \mathrm{ST}\ y, R_2 \\
q &= q+4 \\
&\quad \mathrm{ADD}\ R_1, R_1, 4 \\
&\quad \mathrm{ST}\ q, R_1 \\
*p &= y \\
&\quad \mathrm{LD}\ p, R_1 \\
&\quad \mathrm{ST}\ 0(R_1), R_2 \\
p &= p+4 \\
&\quad \mathrm{ADD}\ R_1, R_1, 4 \\
&\quad \mathrm{ST}\ p, R_1
\end{aligned}
$$

代码生成作业讲解

循环

问题本身不难，但是很多作业循环找错了

代码就变得非常重要。很多代码转换依赖于对流图中“循环”的识别。如果下列条件成立，我们就说流图中的一个结点集合 L 是一个循环。

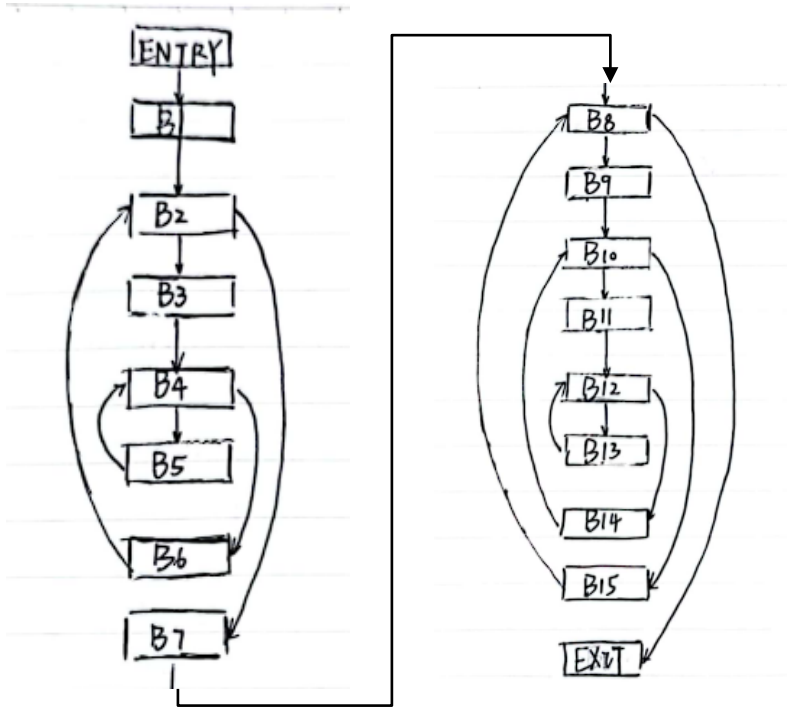
1）在 L 中有一个被称为循环入口（loop entry）的结点，它是唯一的其前驱可能在 L 之外的结点。也就是说，从整个流图的入口结点开始到 L 中的任何结点的路径都必然经过循环入口结点，并且这个循环入口结点不是整个流图的入口结点本身。

2）L 中的每个结点都有一个到达 L 的入口结点的非空路径，并且该路径全部在 L 中。

代码生成作业讲解

循环

1. L存在唯一的循环入口，除了循环入口之外，L中其它节点的**所有前驱**都在L中
2. L中的每个节点都有一个到达循环入口的非空路径，且路径中**所有节点**都在L中

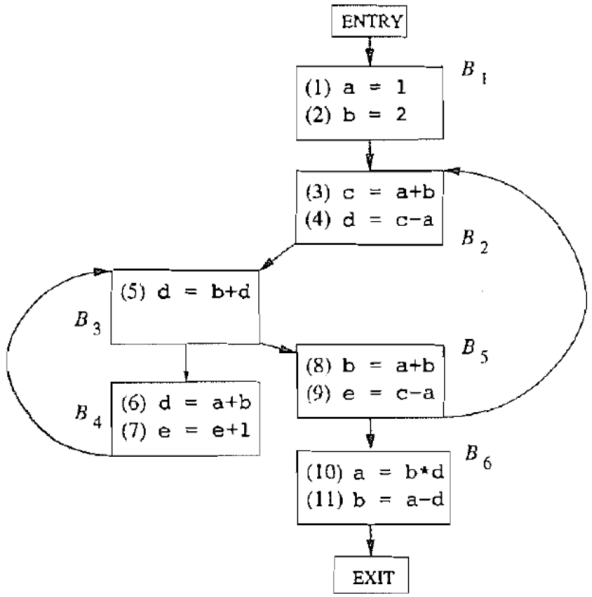


1. {4,5}
2. ~~{2,3,4,5}~~
3. ~~{2,3,4,6}~~
4. {2,3,4,5,6}
5. {12,13}
6. ~~{10,11,12,14,15}~~
7. {10,11,12,13,14}
8. {8,9,10,11,12,13,14,15}

代码生成作业讲解

循环

1. L存在唯一的循环入口，除了循环入口之外，L中其它节点的**所有前驱**都在L中
2. L中的每个节点都有一个到达循环入口的非空路径，且路径中**所有节点**都在L中



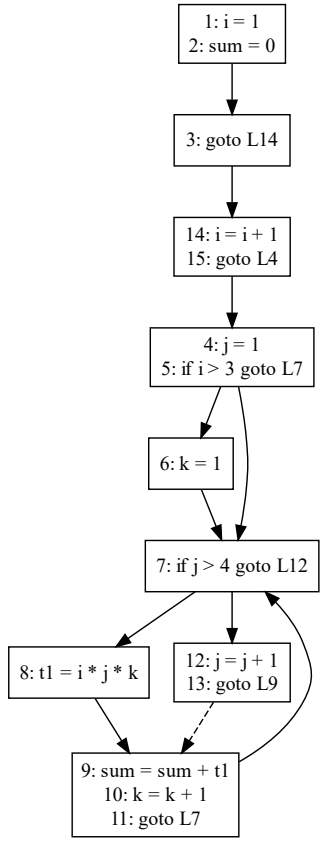
1. {3,4}
2. ~~{2,3,5}~~
3. {2,3,4,5}

随堂练习

```
1: i = 1
2: sum = 0
3: goto L14
4: j = 1
5: if i > 3 goto L7
6: k = 1
7: if j > 4 goto L12
8: t1 = i * j * k
9: sum = sum + t1
10: k = k + 1
11: goto L7
12: j = j + 1
13: goto L9
14: i = i + 1
15: goto L4
```

将其分解为基本块，绘制流图并找到所有循环。

随堂练习答案


1: i = 1
2: sum = 0
3: goto L14
14: i = i + 1
15: goto L4
4: j = 1
5: if i > 3 goto L7
6: k = 1
7: if j > 4 goto L12
8: t1 = i * j * k
12: j = j + 1
13: goto L9
9: sum = sum + t1
10: k = k + 1
11: goto L7

代码生成随堂练习

(8.6.1,8.6.4)将下列C语言语句翻译为三地址码，并使用简单代码生成算法翻译为机器码。假设有两个可用的寄存器，写出每一步的地址描述符与寄存器描述符。

$$x = a[i] + 1$$

$$\begin{aligned} t &= a[i] \\ t &= t+1 \\ x &= t \end{aligned}$$

$$\begin{aligned} t &= a[i] \\ &\quad \mathrm{LD}\ R1, a \\ &\quad \mathrm{LD}\ R2, i \\ &\quad \mathrm{LD}\ R1, R1\,(i) \\ t &= t+1 \\ &\quad \mathrm{ADD}\ R1\ R11 \\ x &= t \\ &\quad \mathrm{ST}\ x, R1 \end{aligned}$$

R1	R2	x	t	a	i
		x	t	a	i

t	i	x	R1	a	i R2

t	i	x	R1	a	i R2

t,x	i	x R1	R1	a	i R2

更新大致过程：

1. 更新寄存器描述符
2. 更新地址描述符
3. 值发生更改再做讨论

机器无关优化：回顾

局部代码优化

- 全局公共子表达式
- 复制传播
- 死代码消除
- 代码移动
- 归纳变量和强度消减

数据流分析

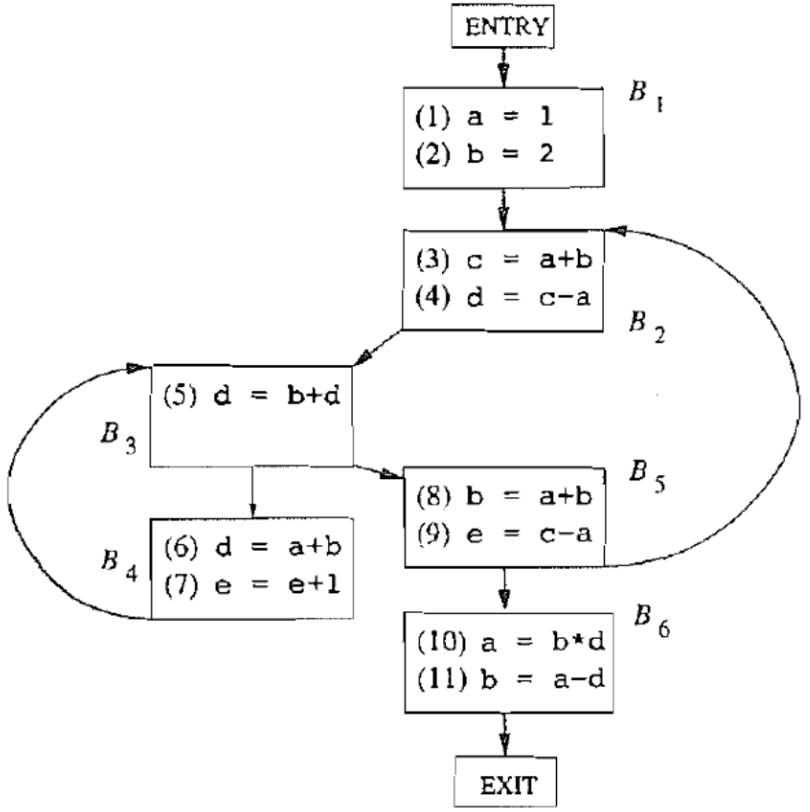
- 到达定值分析
- 活跃变量分析
- 可用表达式分析

循环

- 支配节点
- 寻找支配节点
- 深度优先生成树

机器无关优化作业讲解

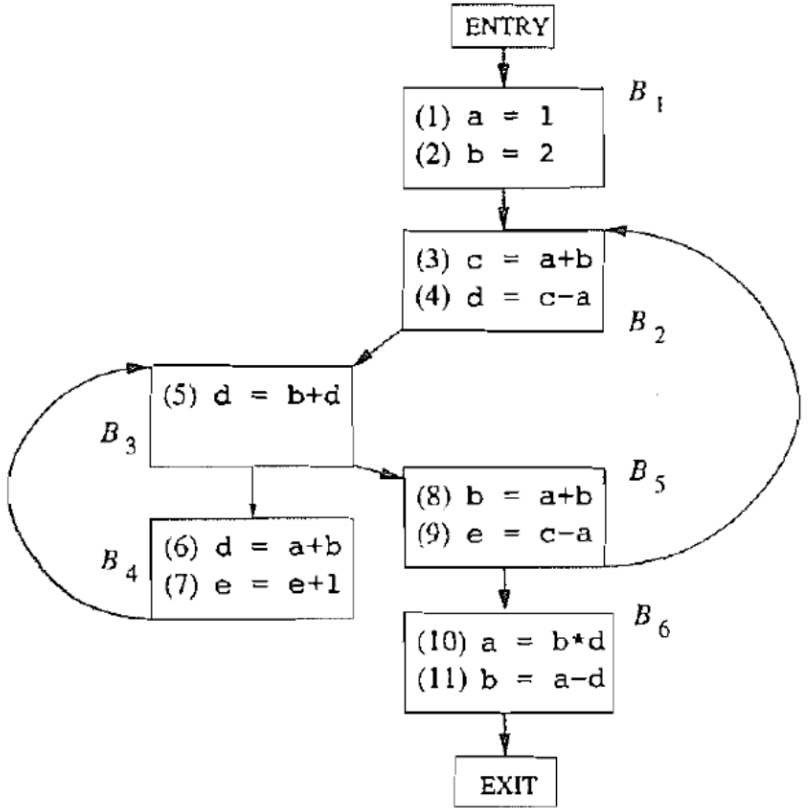
1. 找出所有循环
2. 常量替换
3. 对每个循环，找出所有的全局公共子表达式
4. 对每个循环，寻找归纳变量
5. 对每个循环，寻找全部的循环不变计算



机器无关优化作业讲解

问题1

1. {3,4}，{2,3,4,5}
2. (3) (4) (6) (8) (9)$a \Rightarrow 1$
3. {2,3,4,5}: $a+b, c-a$
4. {2,3,4,5}:b,c,~~e~~
   {3,4}:e
5. {3,4}:a+b

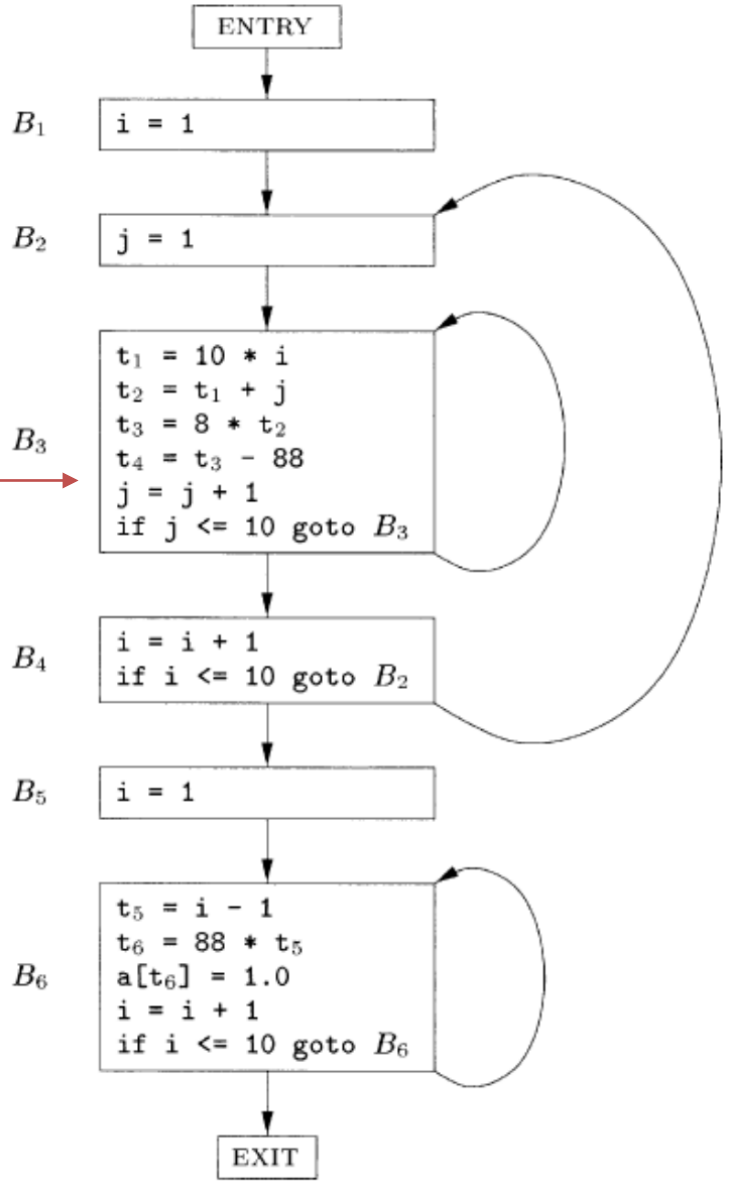


机器无关优化随堂练习

(9.1.2)对图8-9应用本节(9.1节)介绍的优化技术

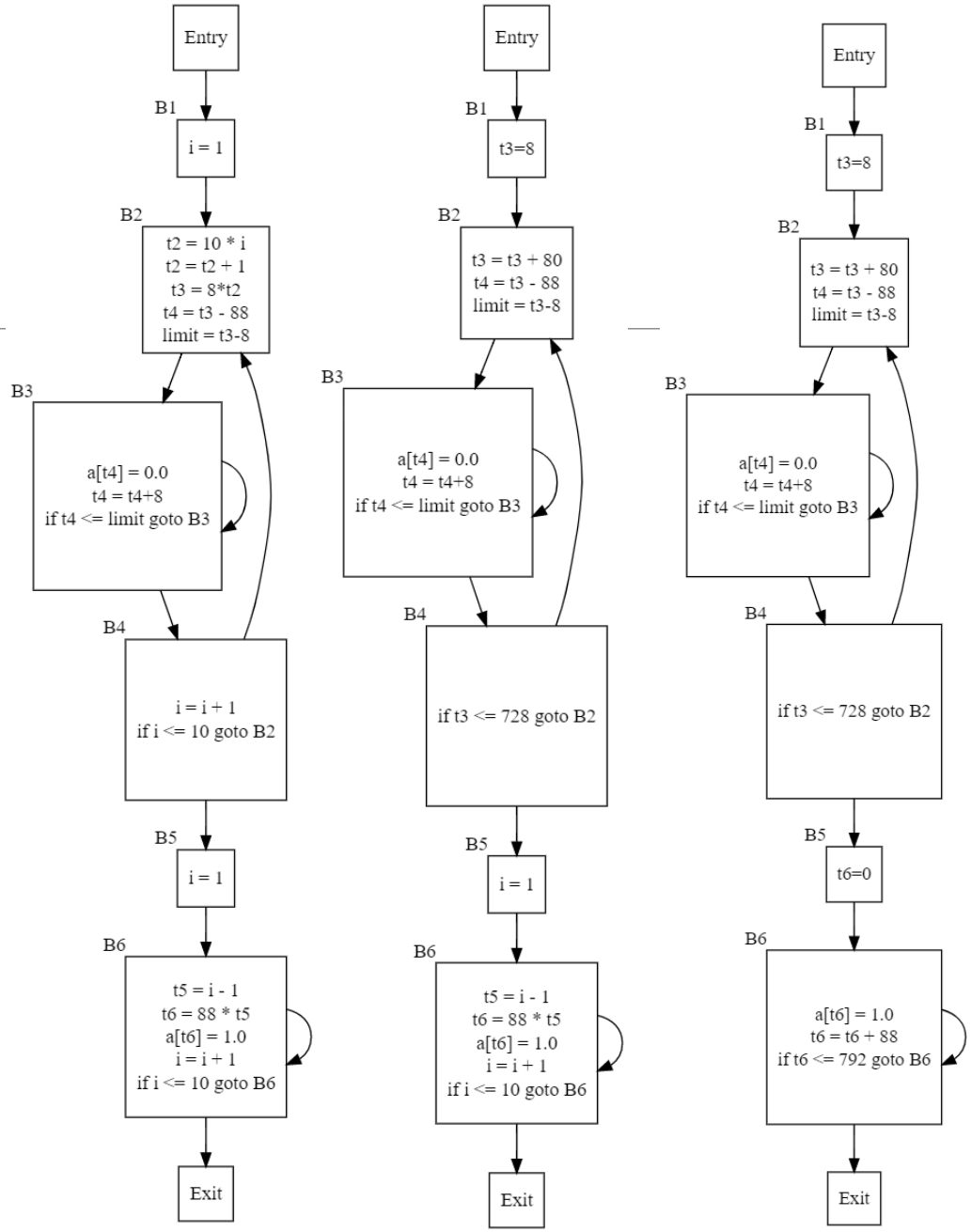
- 全局公共子表达式
- 复制传播
- 死代码消除
- 代码移动
- 归纳变量和强度消减
- ……

$a[t_4] = 0.0$



机器无关优化随堂练习

1. 优化循环B3：
2. 优化循环{B2,B3,B4}
3. 优化循环B6
    1. 注意到，如果使用代数恒等式，则还可以进一步化简



数据流分析

易错点：

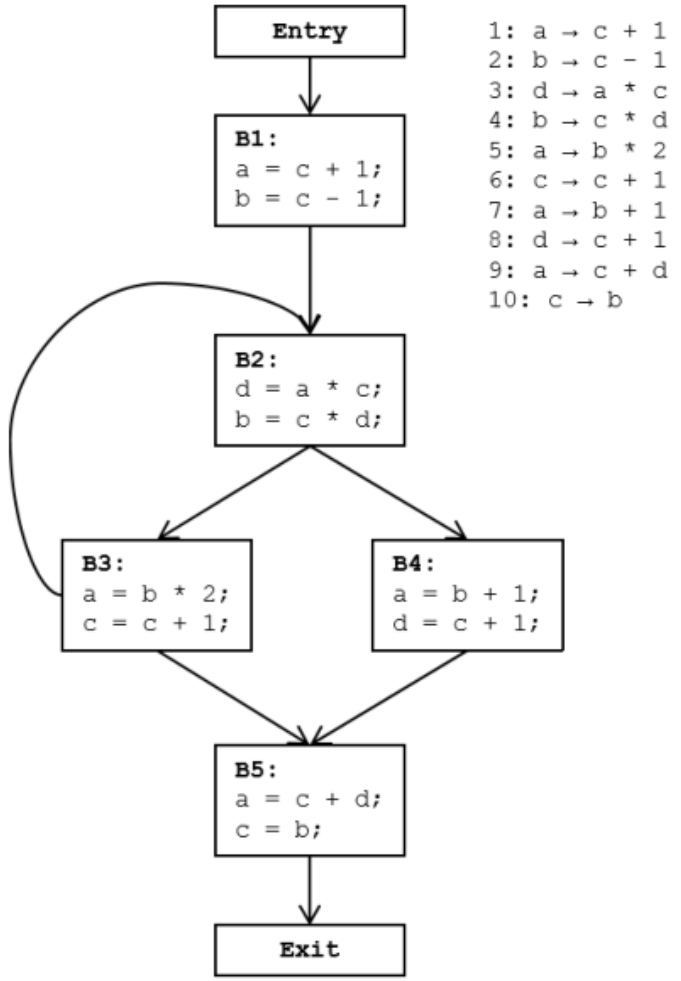
1. 基本块传递函数容易写错
2. 不动点算法计算也容易出错

到达定值

$$f_B(x) = gen_B \cup (x - kill_B)$$

$$kill_B = kill_1 \cup kill_2 \cup \cdots \cup kill_n$$

$$gen_B = gen_n \cup (gen_{n-1} - kill_n) \cup (gen_{n-2} - kill_{n-1} - kill_n) \cup \cdots \cup (gen_1 - kill_2 - kill_3 - \cdots - kill_n)$$

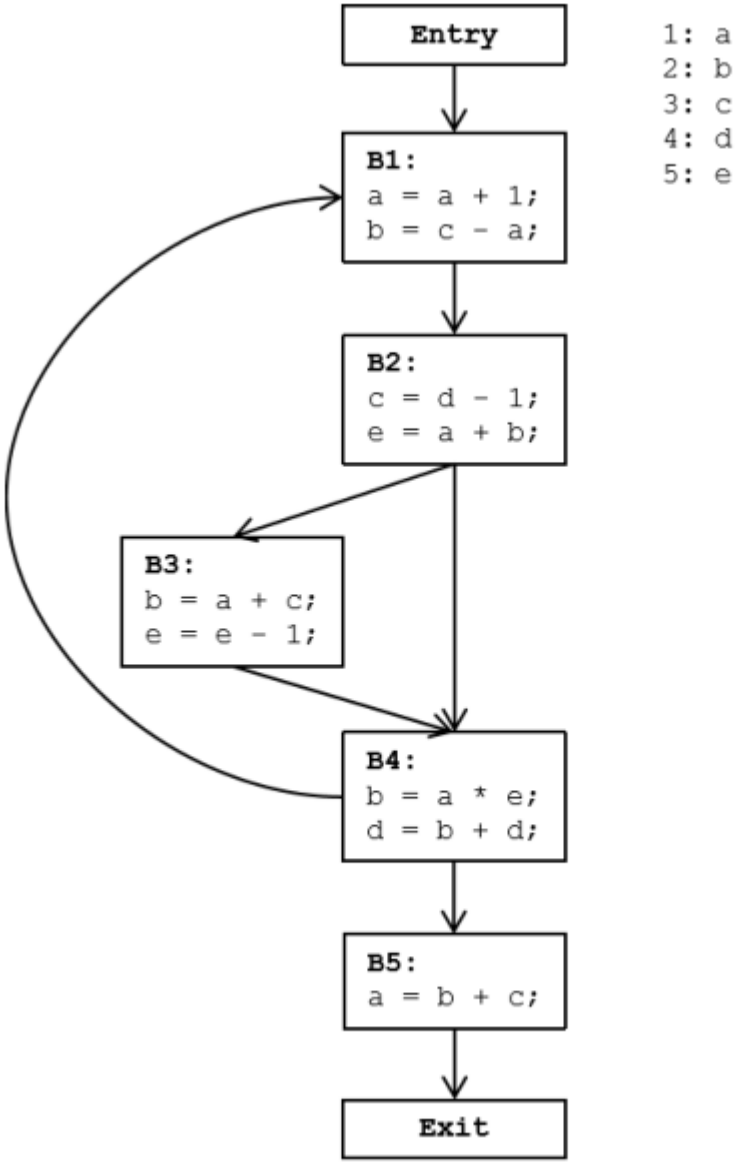


活跃变量

语句的传递函数

- s：x=y+z
- $use_s = \{y, z\}$
- $def_s=\{x\}-\{y, z\}$，

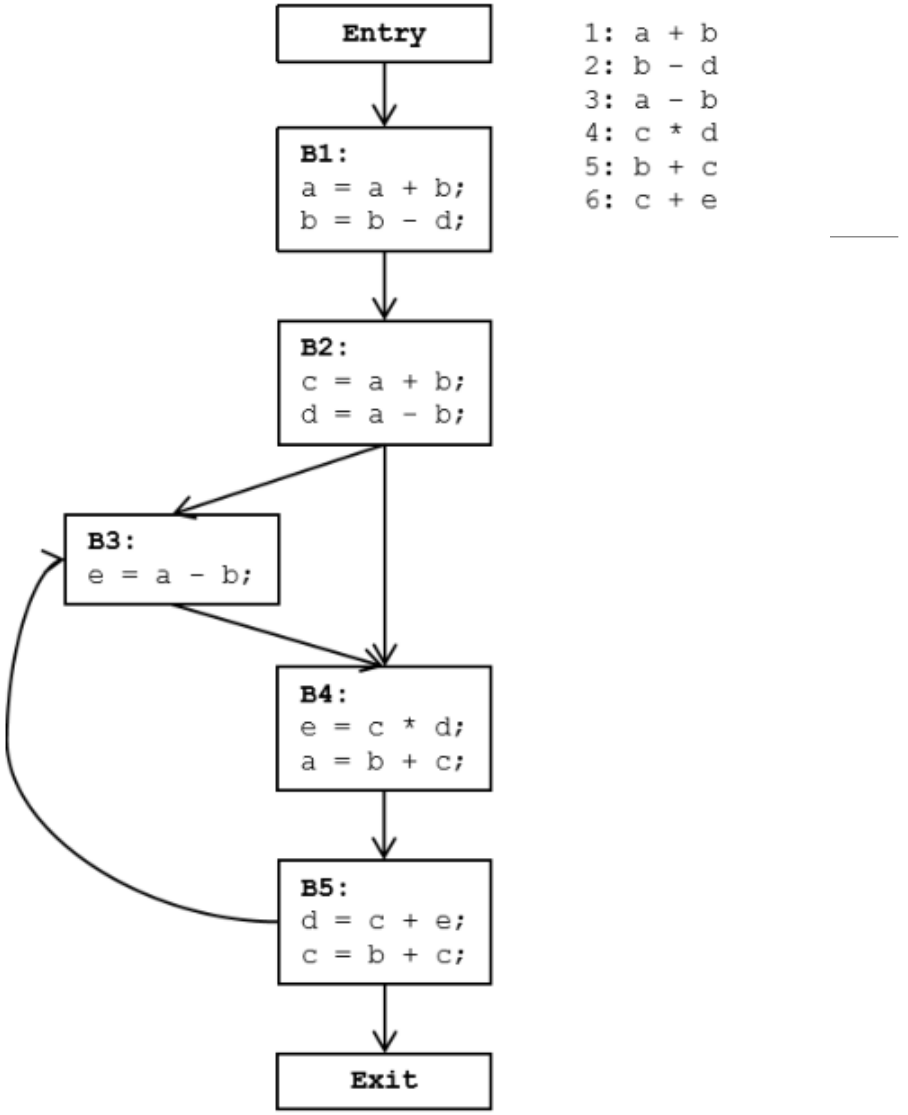
$$use_B = use_1 \cup (use_2-def_1) \cup (use_3-def_1-($$
$$\cup (use_n-def_1-def_2\cdots-def_{n-1})$$
$$def_B=def_1 \cup def_2 \cup \cdots \cup def_n$$



可用表达式

- 初始化S={ }
- 从头到尾逐个处理基本块中的指令x=y+z
  - 把y+z添加到S中；
  - 从S中删除任何涉及变量x的表达式
- 遍历结束时得到基本块生成的表达式集合；

- 杀死的表达式集合
  - 表达式的某个分量在基本块中定值，且没有被再次生成



完