中国科学院大学

University of Chinese Academy of Sciences

一种面向RISC-V的编译原理教学方案探索

中国科学院大学

中国科学院软件研究所

张洪滨

北京工业大学

中国科学院软件研究所

韩柳彤

目录

CONTENTS

1 教学现状

教学问题

目前高校编译原理课程的实践环节覆盖范围也大多仅限于编译器前端部分，导致学生对编译理论的认识不全面，也不能深入了解编译器在计算机系统分层结构中的位置以及作用。

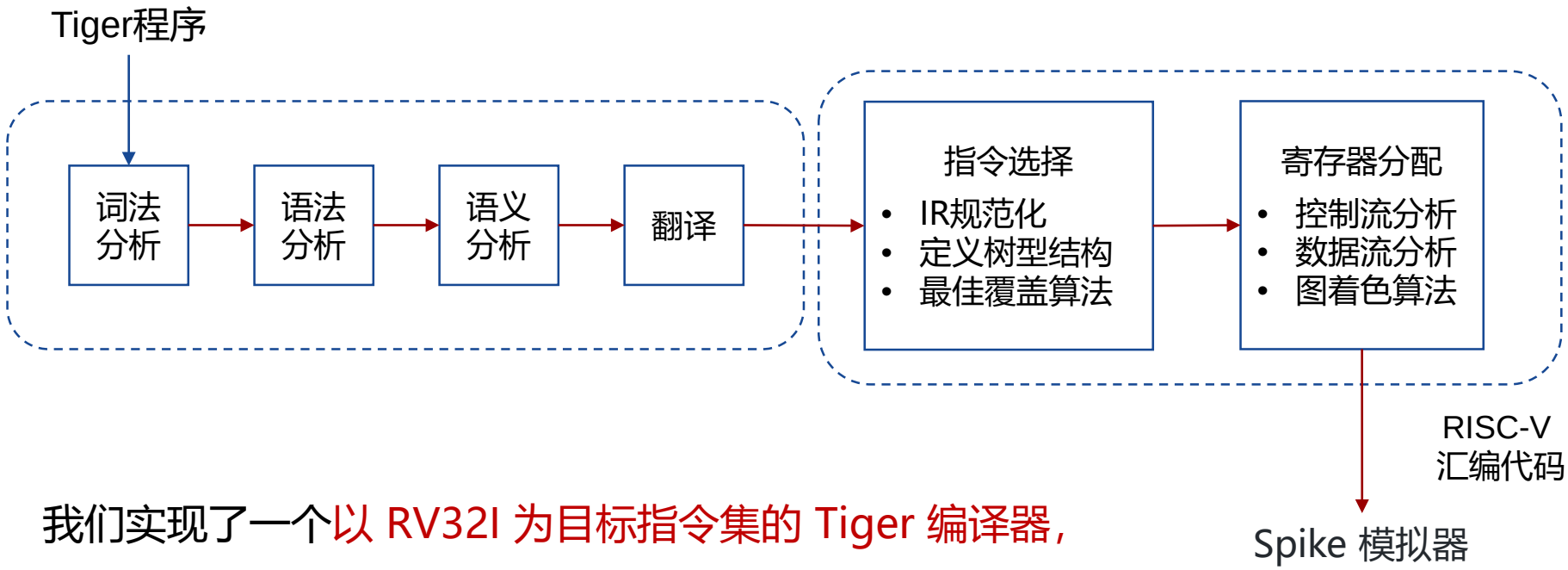
问题的原因

我们认为，其主要原因是编译器后端面向特定的指令集架构，而传统的指令集架构在教学中显得过于庞大，学习成本过高。

RISC-V为何能够解决该问题？

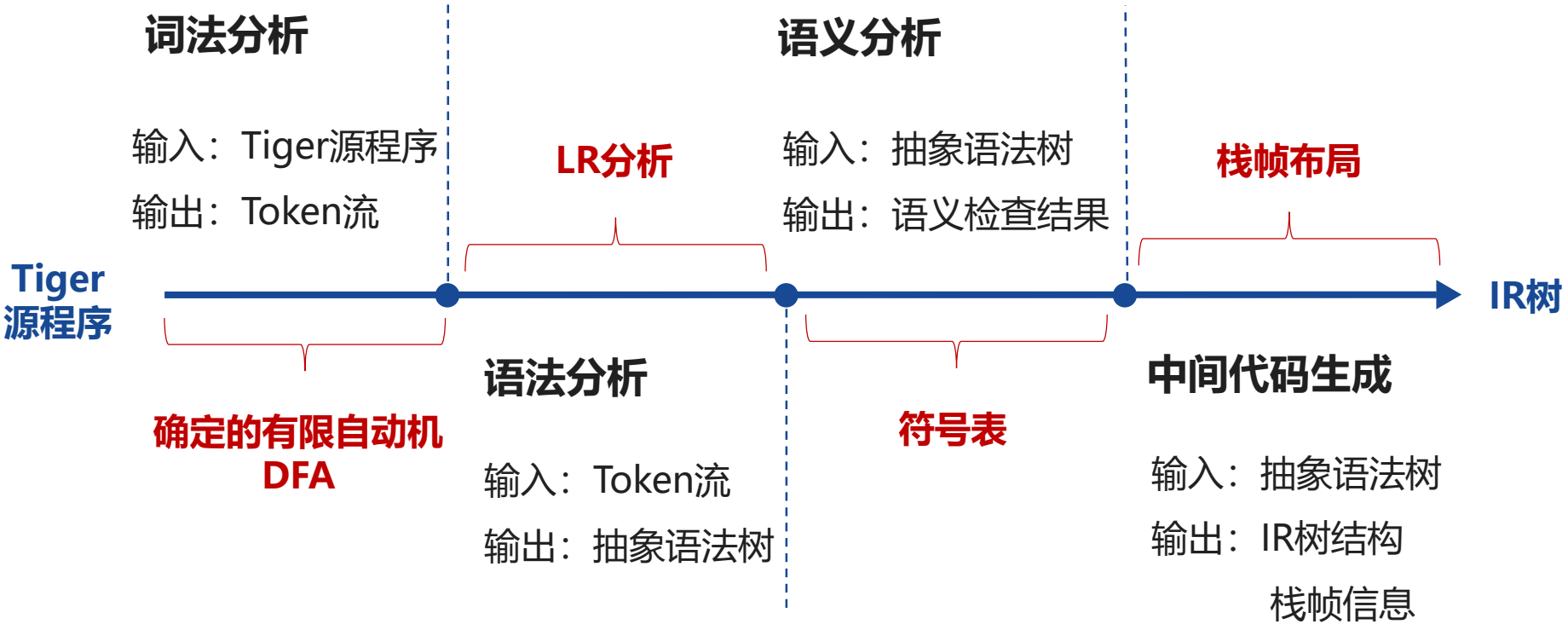
- 具有模块化的优势，且没有向前兼容的包袱，文档页数和指令条数少，降低学习成本。
- 寻址模式和寄存器种类简单，因此降低了编译器后端指令选择和寄存器分配模块的复杂度。

2 教学方案



我们实现了一个以 RV32I 为目标指令集的 Tiger 编译器，编译器的源代码可以面向高校教师和教学组织定向公开，以便在此基础上向学生提供代码框架，匹配不同难度和课时的实践环节。

3 编译器实现 | 前端


词法分析
输入：Tiger源程序
输出：Token流
LR分析
语义分析
输入：抽象语法树
输出：语义检查结果
栈帧布局
Tiger
源程序
IR树
确定的有限自动机
DFA
语法分析
输入：Token流
输出：抽象语法树
符号表
中间代码生成
输入：抽象语法树
输出：IR树结构
栈帧信息

词法分析

使用确定的有限自动机（DFA），通过识别最长的匹配，

将源程序的字符序列切分成一个个Token，从而形成Token流。

每一个Token将会作为一个终结符用于后续的语法分析中。

Tiger 源程序

```
/* correct if */
if (10 > 20) then 30 else 40
```

使用**LEX**
作为分析工具

Token 流

```
    IF    2
LPAREN    5
   INT    6 10
    GT    9
   INT   11 20
RPAREN   13
  THEN   15
   INT   20 30
  ELSE   23
   INT   28 40
```

语法分析

使用产生式表示文法规则，将输入的Token流进行自底向上的LR分析的同时，构造抽象语法树。

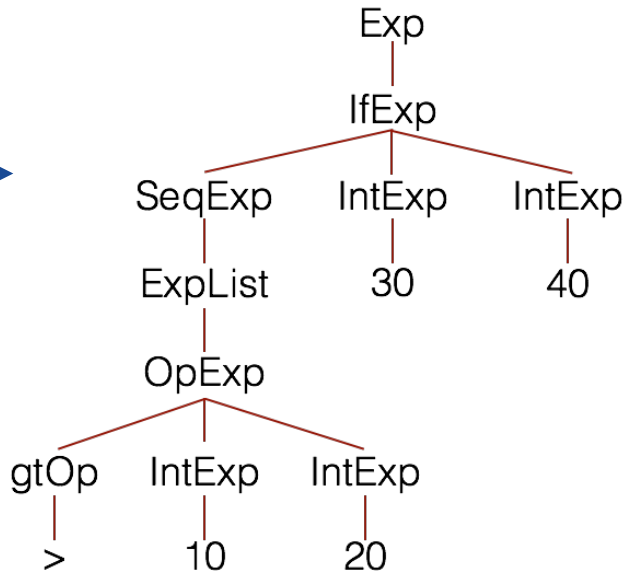
抽象语法树，传递源程序的短语结构，消除了Token流中冗余部分，为后续分析提供清晰接口。

Token 流

```
    IF    2
LPAREN    5
   INT    6 10
    GT    9
   INT   11 20
RPAREN   13
  THEN   15
   INT   20 30
  ELSE   23
   INT   28 40
```

使用**BISON**
作为分析工具

抽象语法树

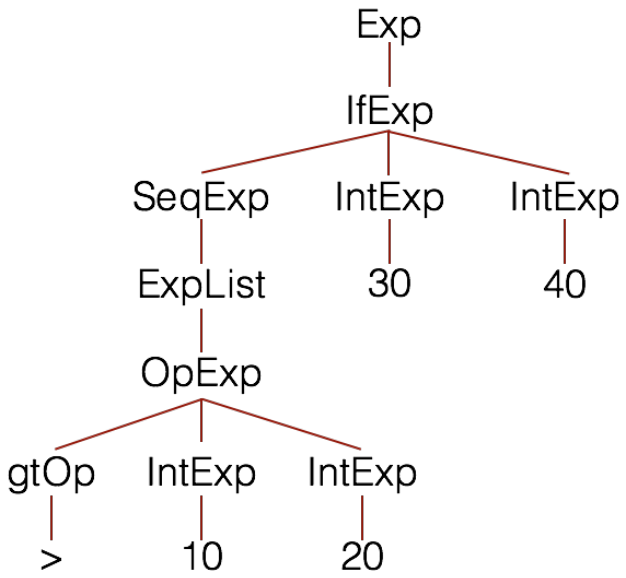


语义分析

遍历抽象语法树，同时进行符号表的管理，将标识符映射到它们的类型和存储位置。

将变量的定义与它们的各个使用联系起来，检查每一个表达式是否有正确的类型。

抽象语法树

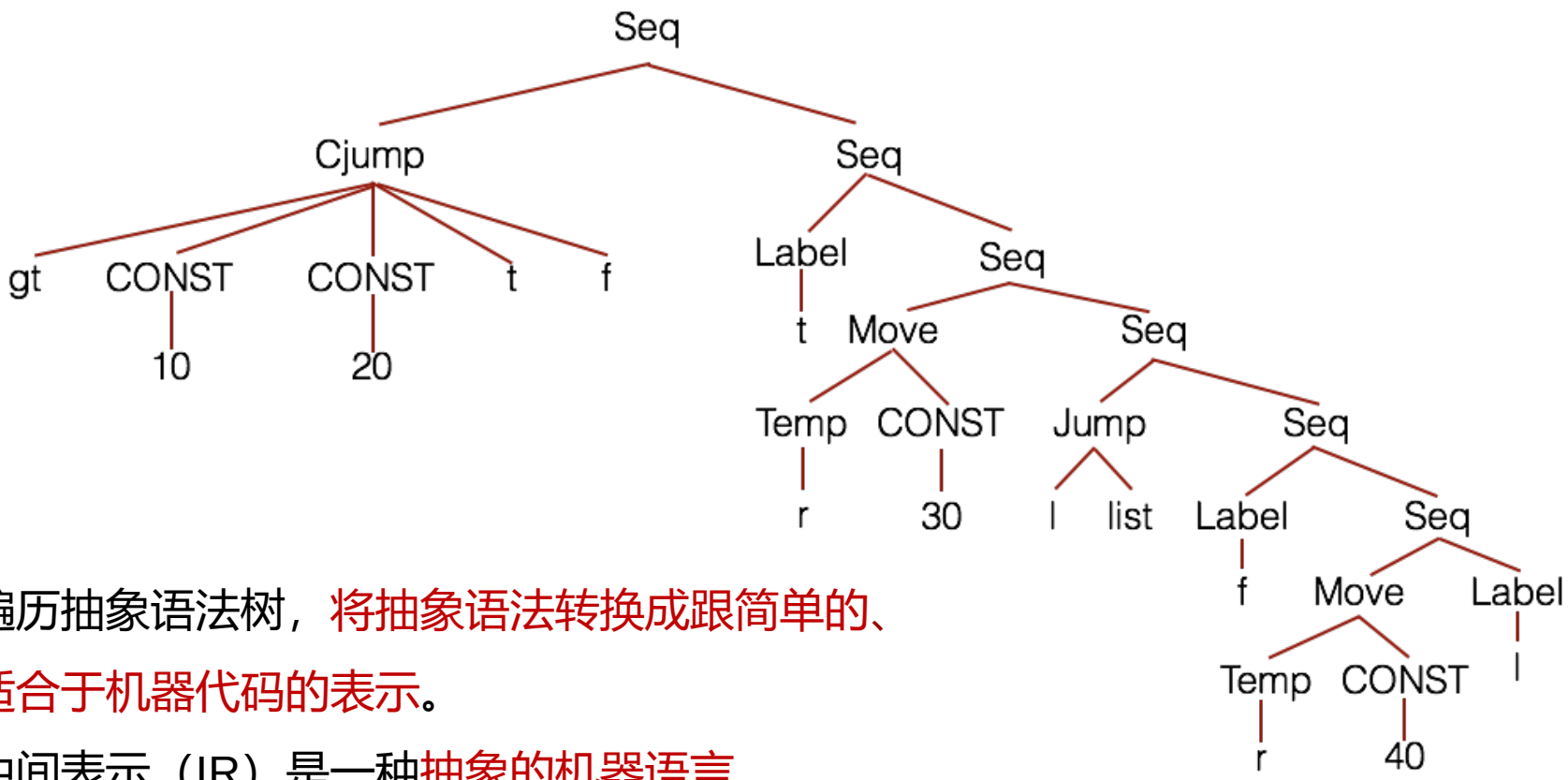


遍历抽象语法树
进行**静态语义检查**

- 声明的类型检查
- 表达式的类型检查

翻译成中间代码

IR树

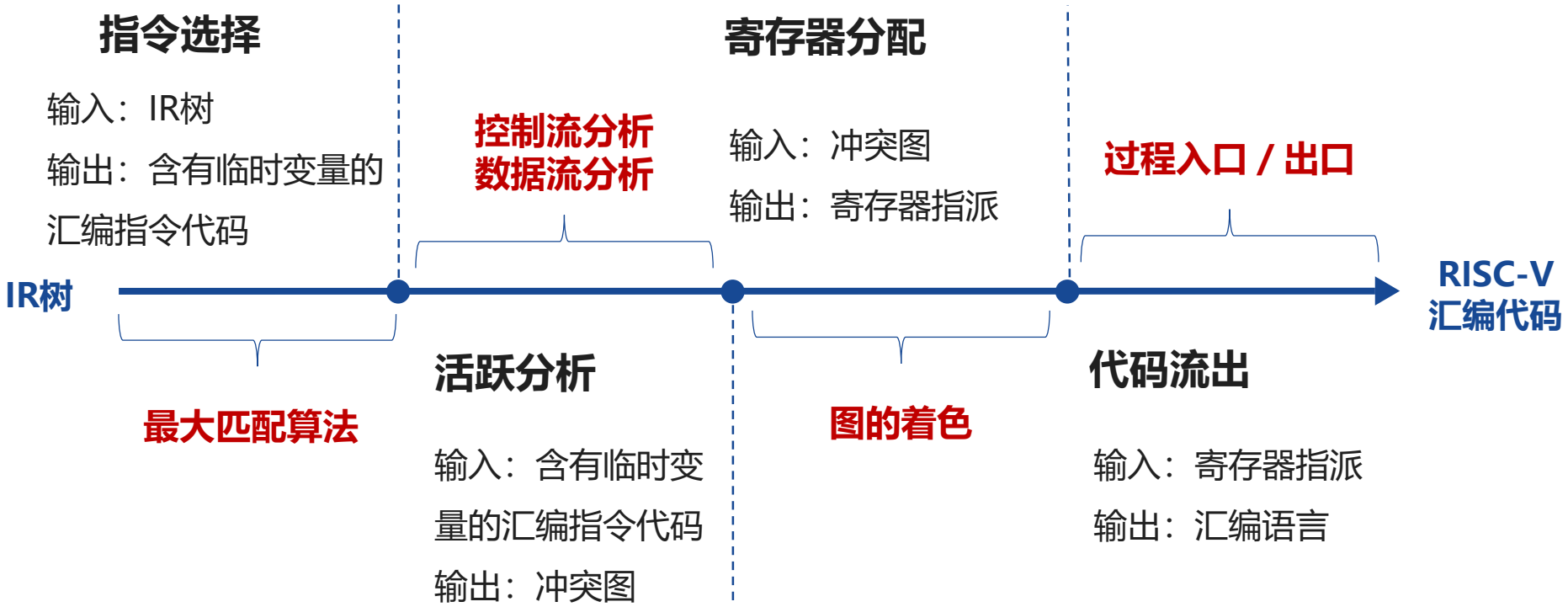


遍历抽象语法树，将抽象语法转换成跟简单的、适合于机器代码的表示。

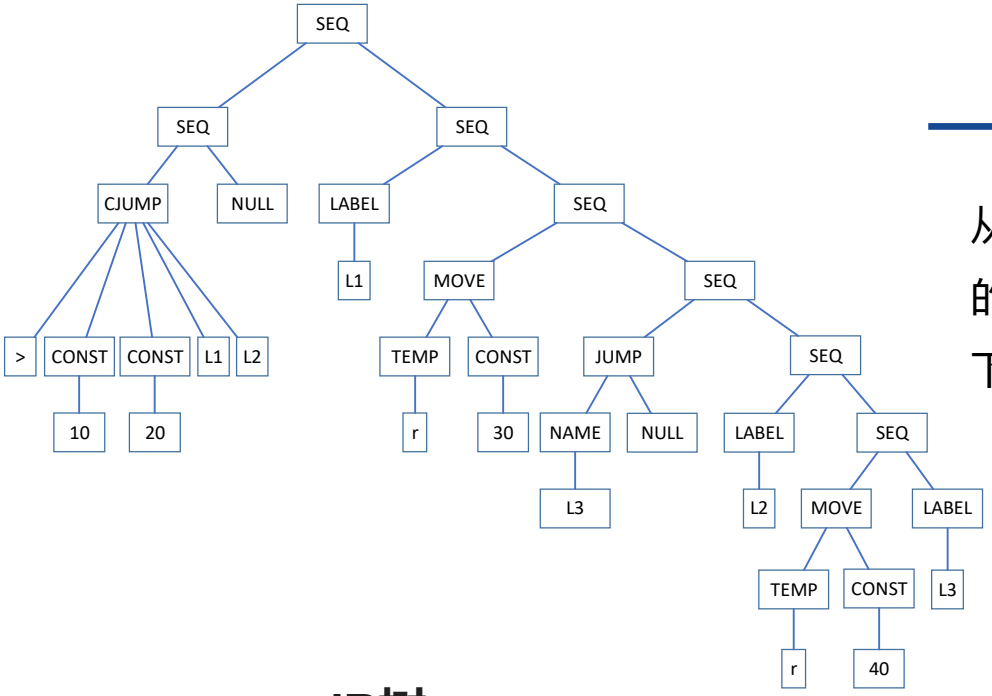
中间表示（IR）是一种抽象的机器语言，

它可以表示目标机的操作而不需要太多地涉及及其相关的细节。

3 编译器实现 | 后端


指令选择
输入：IR树
输出：含有临时变量的汇编指令代码
IR树
最大匹配算法
控制流分析
数据流分析
活跃分析
输入：含有临时变量的汇编指令代码
输出：冲突图
寄存器分配
输入：冲突图
输出：寄存器指派
图的着色
过程入口 / 出口
代码流出
输入：寄存器指派
输出：汇编语言
RISC-V
汇编代码

指令选择



IR树

最大匹配算法

从树根节点开始，选择适合的最大树形覆盖节点，对余下子树重复执行该算法。

```
li t101,10
li t102,20
bgt t101,t102,L1
L2:
li t100,40
L3:
j L4
L1:
li t100,30
j L3
L4:
```

含有临时变量的汇编指令代码

活跃分析

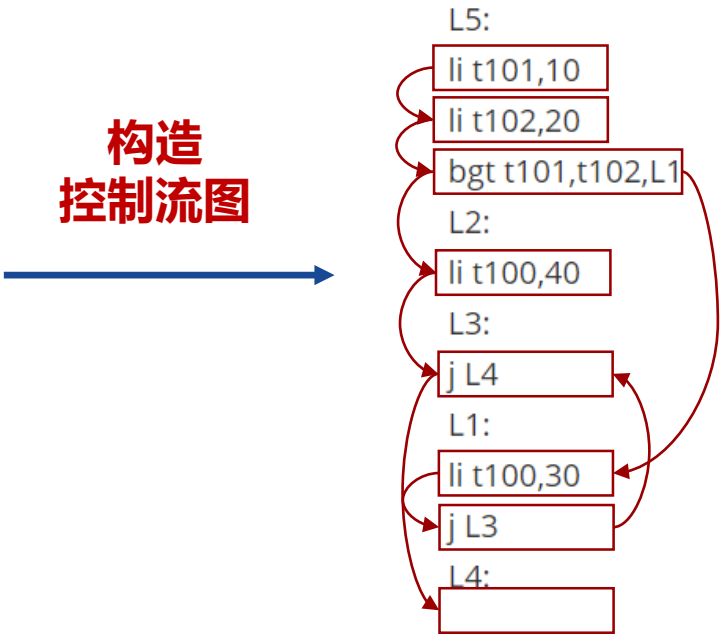
图中的每个节点代表一条指令，如果指令n的执行可以跟随在指令m之后，则图中会有一条边（m，n）。

构造程序的控制流图，用于进行数据流分析，得到活跃区间。

```
L5:
li t101,10
li t102,20
bgt t101,t102,L1
L2:
li t100,40
L3:
j L4
L1:
li t100,30
j L3
L4:
```

含有临时变量的汇编指令代码

构造控制流图



控制流图

活跃分析

(0): 1
(1): 2
(2): 5 3
(3): 4
(4): 7
(5): 6
(6): 4
(7):

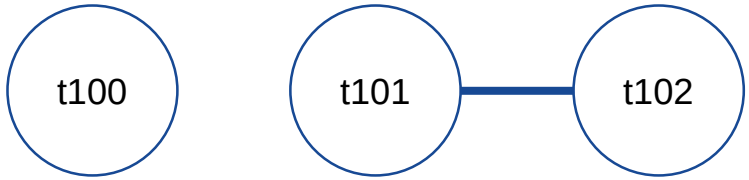
控制流图
(以邻接表形式存储)

$$in[n] = use[n] \cup (out[n] - def[n])$$

$$out[n] = \bigcup_{s \in succ} in\,[s]$$

解数据流方程

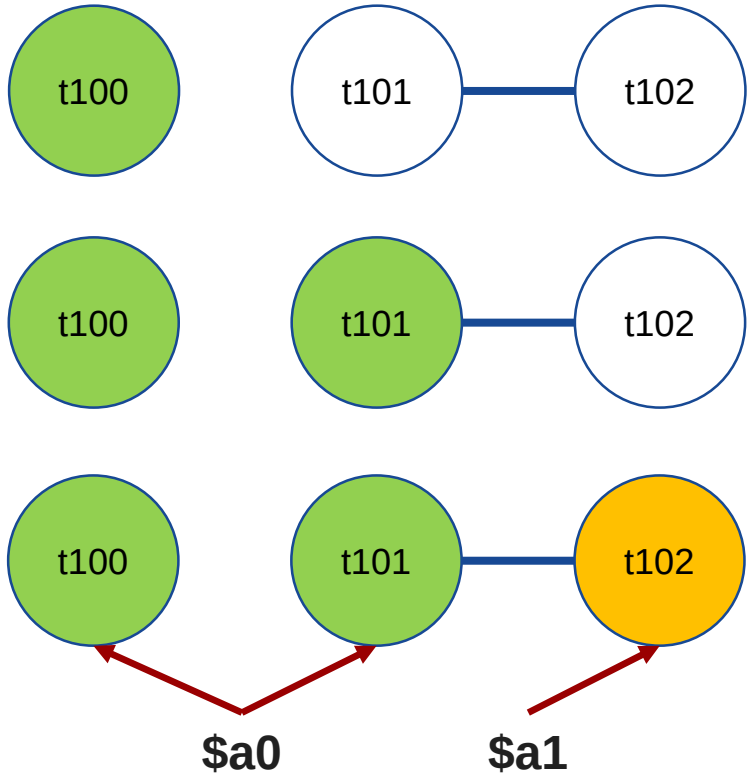
	0		1		2		3		4		5		6	
	in	out	in	out	in	out	in	out	in	out	in	out	in	out
0	\	\	\	\	\	\	\	\	\	t101	\	t101	\	t101
1	\	\	\	\	\	t101 t102	t101	t101 t102	t101	t101 t102	t101	t101 t102	t101	t101 t102
2	\	\	t101 t102	\	t101 t102	\	t101 t102	\	t101 t102	\	t101 t102	\	t101 t102	\
3	\	\	\	\	\	\	\	\	\	t100	\	t100	\	t100
4	\	\	\	\	\	t100	t100	t100	t100	t100	t100	t100	t100	t100
5	\	\	\	\	\	\	\	\	\	\	\	t100	\	t100
6	\	\	\	\	\	\	\	t100	t100	t100	t100	t100	t100	t100
7	\	\	t100	\	t100	\	t100	\	t100	\	t100	\	t100	\



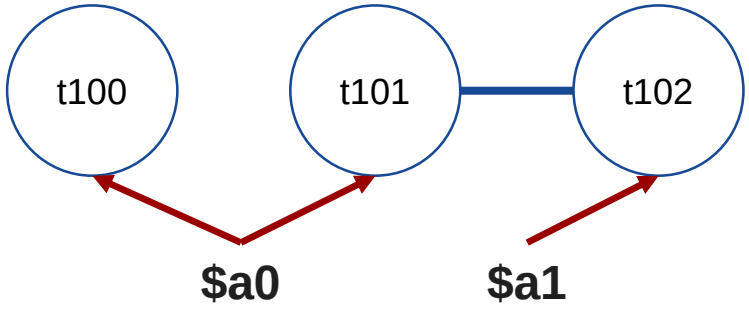
冲突图

寄存器分配

使用图的着色算法进行寄存器分配，给冲突图进行着色。假设目标机器有k个寄存器，只要冲突图是k可着色的，就可以将寄存器合理的分配给不同的临时变量，否则，需要溢出图中某个变量到内存里，直到新的冲突图是k可着色的。



代码流出




```
L5:
li t101,10
li t102,20
bgt t101,t102,L1
L2:
li t100,40
L3:
j L4
L1:
li t100,30
j L3
L4:
```

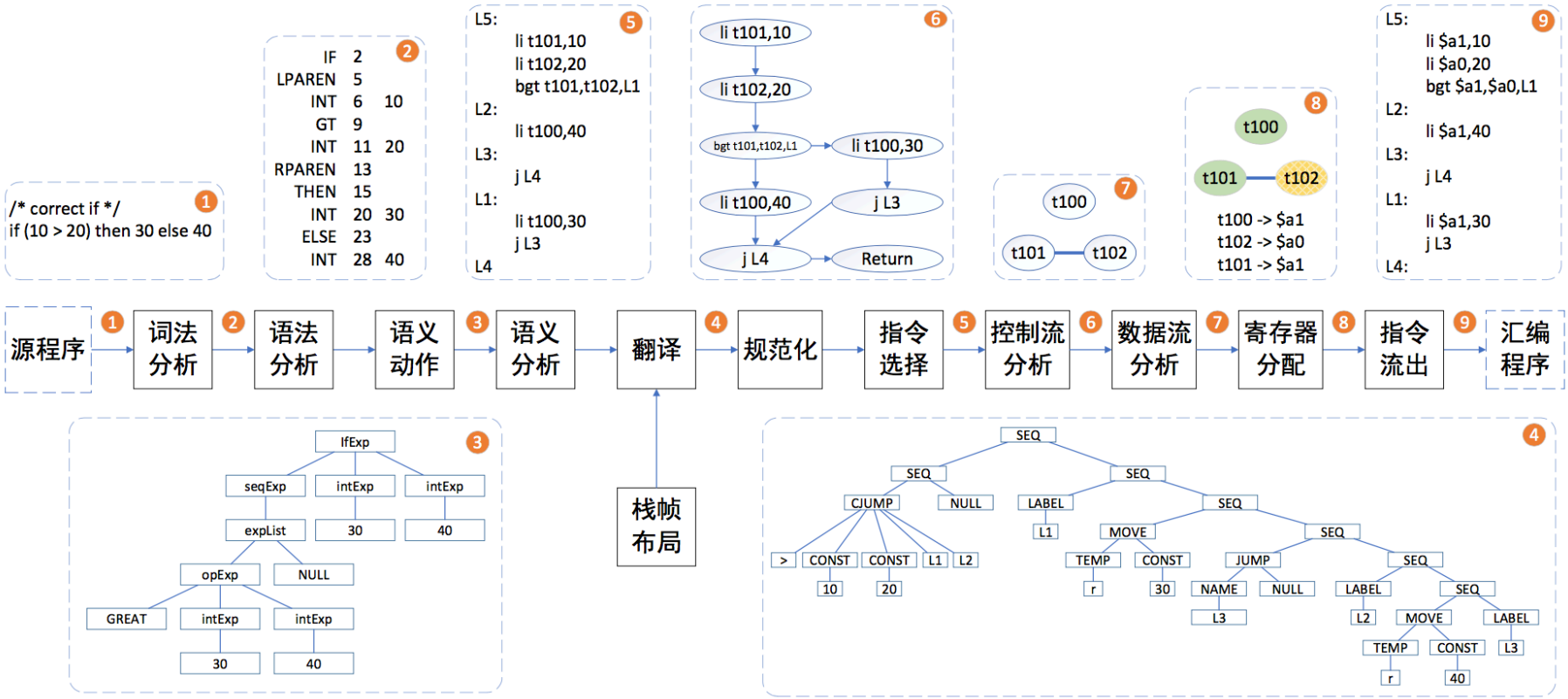
+

```
t100 -> $a1
t102 -> $a0
t101 -> $a1
```

=

```
L5:
li $a1,10
li $a0,20
bgt $a1,$a0,L1
L2:
li $a1,40
L3:
j L4
L1:
li $a1,30
j L3
L4:
```

3 编译器各阶段及示例


/* correct if */
if (10 > 20) then 30 else 40
IF 2
LPAREN 5
INT 6 10
GT 9
INT 11 20
RPAREN 13
THEN 15
INT 20 30
ELSE 23
INT 28 40
L5:
li t101,10
li t102,20
bgt t101,t102,L1
L2:
li t100,40
L3:
j L4
L1:
li t100,30
j L3
L4
li t101,10
li t102,20
bgt t101,t102,L1
li t100,30
li t100,40
j L3
j L4
Return
t100
t101
t102
t100 -> $a1
t102 -> $a0
t101 -> $a1
L5:
li $a1,10
li $a0,20
bgt $a1,$a0,L1
L2:
li $a1,40
L3:
j L4
L1:
li $a1,30
j L3
L4:
源程序
词法分析
语法分析
语义动作
语义分析
翻译
规范化
指令选择
控制流分析
数据流分析
寄存器分配
指令流出
汇编程序
栈帧布局
IfExp
seqExp
intExp
intExp
expList
30
40
opExp
NULL
GREAT
intExp
intExp
30
40
SEQ
SEQ
SEQ
CJUMP
NULL
LABEL
SEQ
>
CONST
CONST
L1
L2
L1
MOVE
SEQ
10
20
TEMP
CONST
JUMP
SEQ
r
30
NAME
NULL
LABEL
SEQ
L3
L2
MOVE
LABEL
TEMP
CONST
L3
r
40

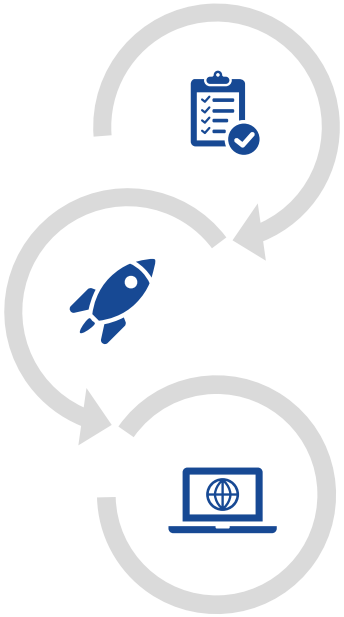
4 方案评估

体系结构难度对比	8086	MIPS32	RISC-V (RV32I)
文档页数	53	321	15
指令条数	116	126	40
寻址方式	7	1	1

方案对比	现有方案	面向 RISC-V 的 Tiger 编译器方案
基础部分	基础的编译器前端 (80%)	包括符号表、作用域的完整前端 支持静态语义检查 (50%)
进阶部分	中间代码、解释器等 (20%)	生成 RISC-V 汇编代码的编译器后端 (40%)
高级部分	无	RISC-V 扩展指令集编译高级特性等 (10%)

5 未来工作

中国科学院大学
University of Chinese Academy of Sciences



完善方案基础部分

编写开发文档、指导手册。

实现编译器的高级功能

实现垃圾回收等高级功能，作为进阶内容为分层教学提供支持。

开发在线实训平台

提供代码框架、运行环境和自动代码测试、赋分反馈功能，帮助提升实践课程效果。

Q & A