



机器无关代码优化

Part5: 基本块内的优化

李 诚

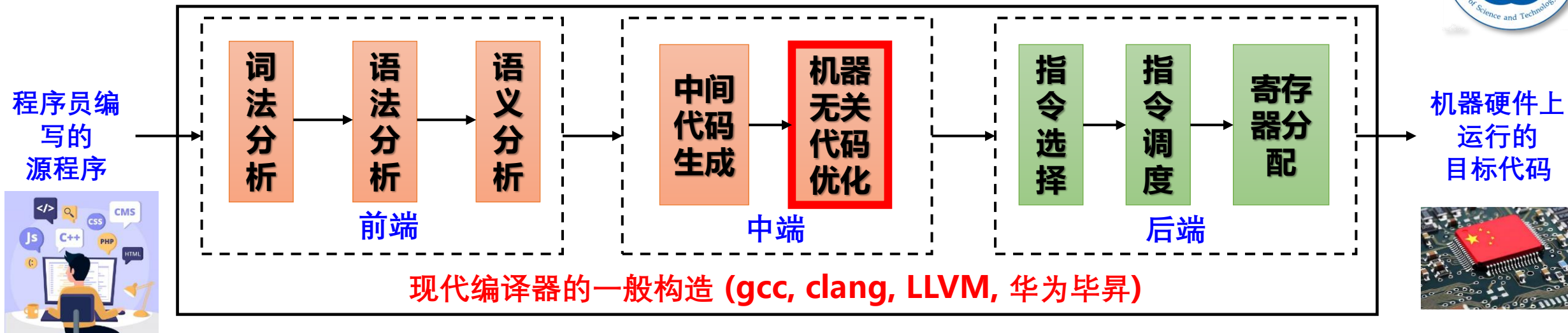
国家高性能计算中心(合肥)、信息与计算机国家级实验教学示范中心

计算机科学与技术学院

2023年11月20日



本节提纲



• 基本块的DAG表示

• 优化的基本方法

- 删除局部公共子表达式、删除死代码
- 代数恒等式的使用、数组引用的表示

• 从DAG到基本块的重组



优化的实现方式



- **局部视角-基本块的优化**
 - DAG表示
- **全局视角-跨基本块的优化**
 - 数据流分析



• 基本块DAG的构造方式

- 每个变量有一个对应的DAG结点表示其初值
- 每条语句s都对应一个内部结点N
 - 结点N的标号是s中的运算符
 - 有一组变量被关联到N，表示s是在此基本块中最晚对这些变量定值的语句
 - N的子结点是基本块中在s之前，最后一个对s所使用的某个运算分量进行定值的语句对应的结点。如果某个运算分量在基本块中在s之前没有被定值，则这个分量对应的子结点就是其初始值对应的结点，用下标0区分
- 某些结点是输出结点，在出口处活跃



基本块的DAG(有向无环图)表示



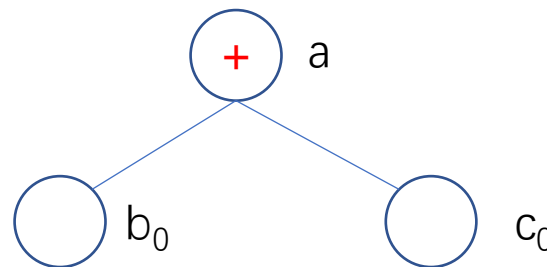
考虑如下基本块：

$a = b + c$

$b = a - d$

$c = b + c$

$d = a - d$



- 每个变量有一个对应的DAG结点表示其初值
- 每条语句s都对应一个内部结点N
 - 结点N的标号是s中的运算符
 - 有一组变量被关联到N，表示s是在此基本块中最晚对这些变量定值的语句
 - N的子结点是基本块中在s之前，最后一个对s所使用的某个运算分量进行定值的语句对应的结点。如果某个运算分量在基本块中在s之前没有被定值，则这个分量对应的子结点就是其初始值对应的结点，用下标0区分



基本块的DAG(有向无环图)表示



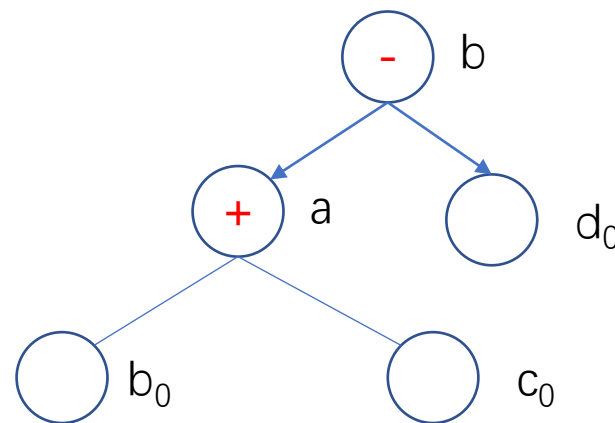
考虑如下基本块：

$a = b + c$

$b = a - d$

$c = b + c$

$d = a - d$



- 每个变量有一个对应的DAG结点表示其初值
- 每条语句s都对应一个内部结点N
 - 结点N的标号是s中的运算符
 - 有一组变量被关联到N，表示s是在此基本块中最晚对这些变量定值的语句
 - N的子结点是基本块中在s之前，最后一个对s所使用的某个运算分量进行定值的语句对应的结点。如果某个运算分量在基本块中在s之前没有被定值，则这个分量对应的子结点就是其初始值对应的结点，用下标0区分



基本块的DAG(有向无环图)表示



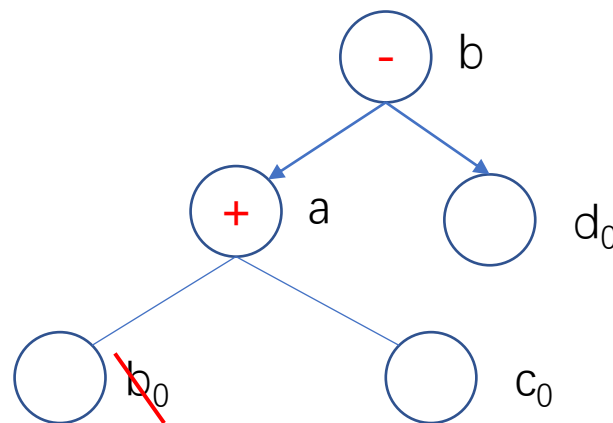
考虑如下基本块:

$a = b + c$

$b = a - d$

$c = b + c$

$d = a - d$



- 每个变量有一个对应的DAG结点表示其初值

- 每条语句s都对应一个内部结点N

- 结点N的标号是s中的运算符

- 有一组变量被关联

注1: 在为语句 $x = y + z$ 构造结点N的时候, 如果x已经被关联

- N的子结点是基本块中在s之前没有被定值的语句对应的结点。如果某个运算分量在基本块中在s之前没有被定值, 则这个分量对应的子结点就是其初始值对应的结点, 用下标0区分

到某结点M上, 那么需要从M的关联变量中删除变量x



基本块的DAG(有向无环图)表示



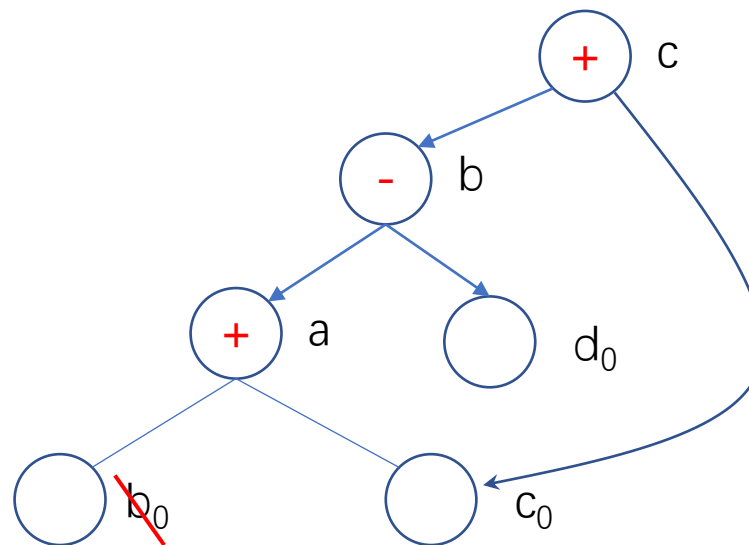
考虑如下基本块：

$a = b + c$

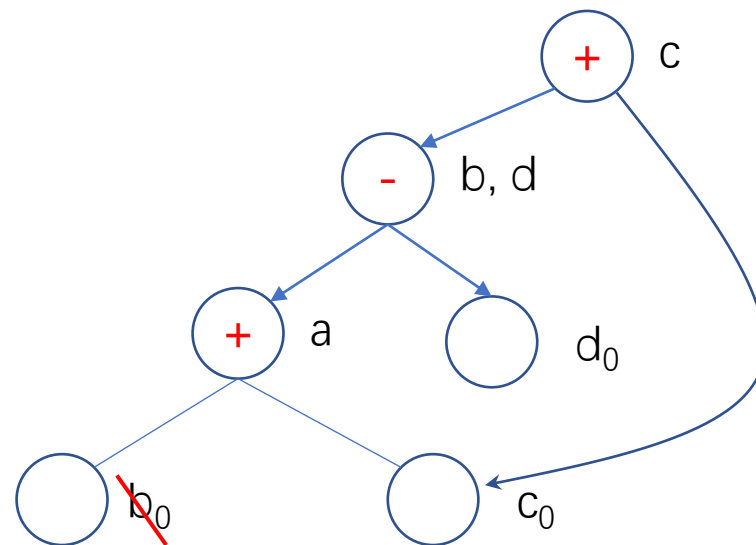
$b = a - d$

$c = b + c$

$d = a - d$



- 每个变量有一个对应的DAG结点表示其初值
- 每条语句s都对应一个内部结点N
 - 结点N的标号是s中的运算符
 - 有一组变量被关联到N，表示s是在此基本块中最晚对这些变量定值的语句
 - N的子结点是基本块中在s之前，最后一个对s所使用的某个运算分量进行定值的语句对应的结点。如果某个运算分量在基本块中在s之前没有被定值，则这个分量对应的子结点就是其初始值对应的结点，用下标0区分


$$\mathbf{a} = \mathbf{b} + \mathbf{c}$$
$$\mathbf{b} = \mathbf{a} - \mathbf{d}$$
$$\mathbf{c} = \mathbf{b} + \mathbf{c}$$
$$\mathbf{d} = \mathbf{a} - \mathbf{d}$$


- 每个变量有一个对应的DAG结点表示其初值
- 每条语句s都对应一个内部结点N

- 结点N的**标号**是s中的运算符

- 有一组变量被

- N的子结点是

注2: 对于 $x=y+z$ 语句, 可对 $y+z$ 进行值编码, 并在该基本快DAG中查询是否已经存在一个结点表示 $y+z$, 如果有, 就不需要在DAG中加入新的结点, 而是在已存在的结点的关联变量中增加 x

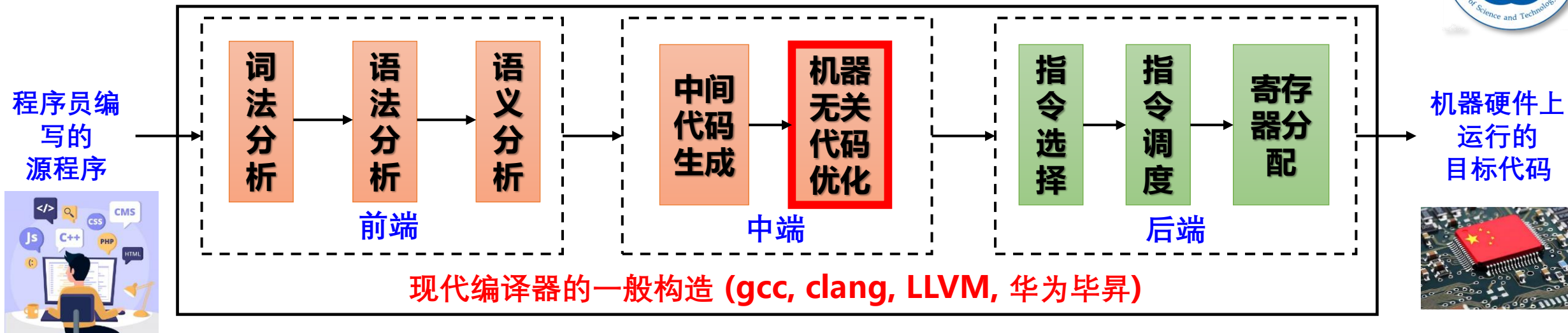
语句

先行定值的语

应的子结点就是其初始值对应的结点，用下标0区分



本节提纲



- 基本块的DAG表示
- 优化的基本方法
 - 删除局部公共子表达式、删除死代码
 - 代数恒等式的使用、数组引用的表示
- 从DAG到基本块的重组



删除局部公共子表达式



考虑如下基本块：

$$a = b + c$$

$$b = a - d$$

$$c = b + c$$

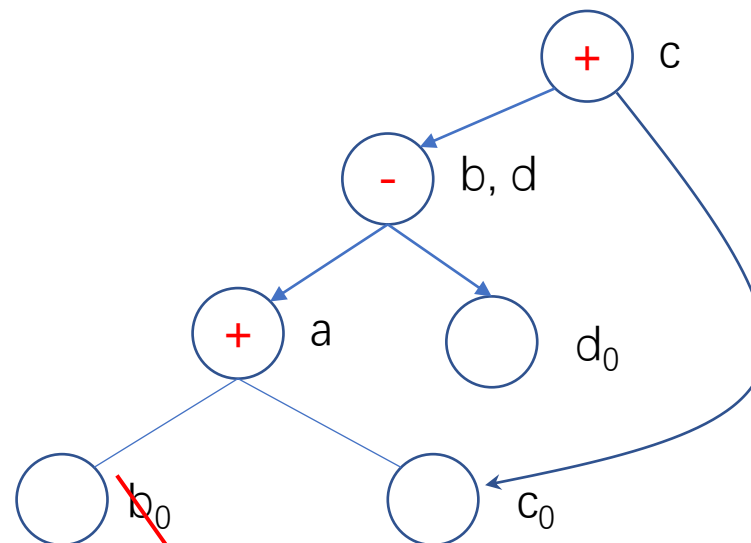
$$d = a - d$$

假设 b 在基本块出口处不活跃

$$a = b + c$$

$$d = a - d$$

$$c = d + c$$





删除局部公共子表达式



- 当完成基本块优化后，就可以根据优化得到的DAG生成新的等价的三地址代码
- 如果结点有多个关联的活跃变量，就必须引入复制语句，为每个变量赋予正确的值



删除局部公共子表达式



考虑如下基本块：

$$a = b + c$$

$$b = a - d$$

$$c = b + c$$

$$d = a - d$$

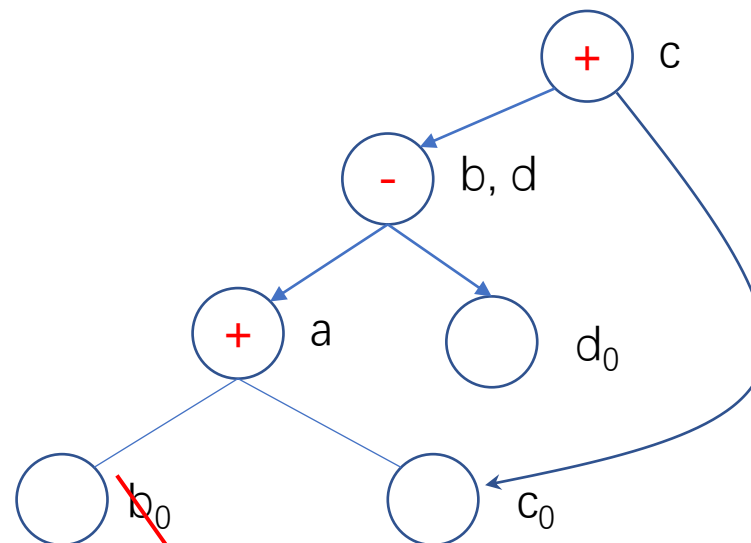
假设b和d在基本块出口处都活跃

$$a = b + c$$

$$d = a - d$$

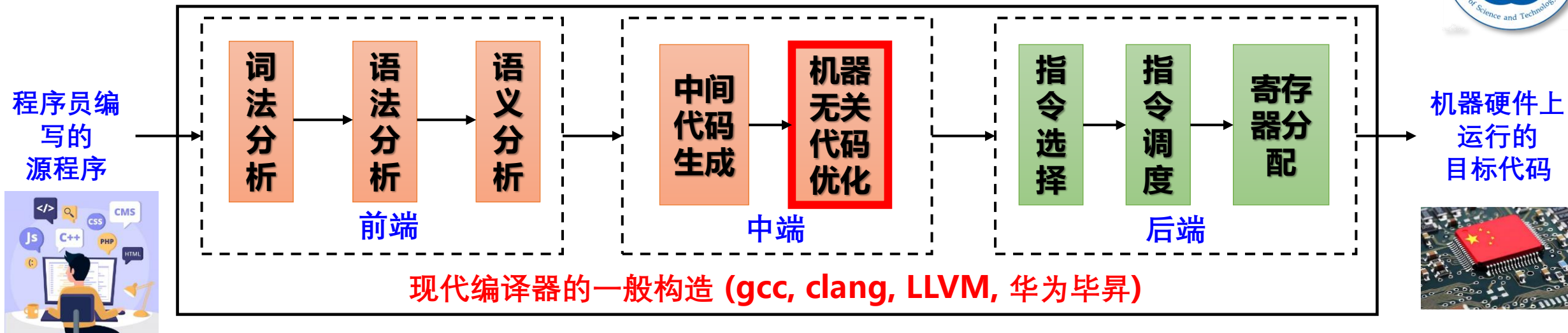
$$b = d$$

$$c = d + c$$





本节提纲



- 基本块的DAG表示
- 优化的基本方法
 - 删除局部公共子表达式、删除死代码
 - 代数恒等式的使用、数组引用的表示
- 从DAG到基本块的重组



- 活跃变量是指其值可能会在以后被使用的变量
- 在DAG上删除死代码，可进行如下操作
 - 1. 删除所有没有关联活跃变量的根结点
 - 2. 重复1操作

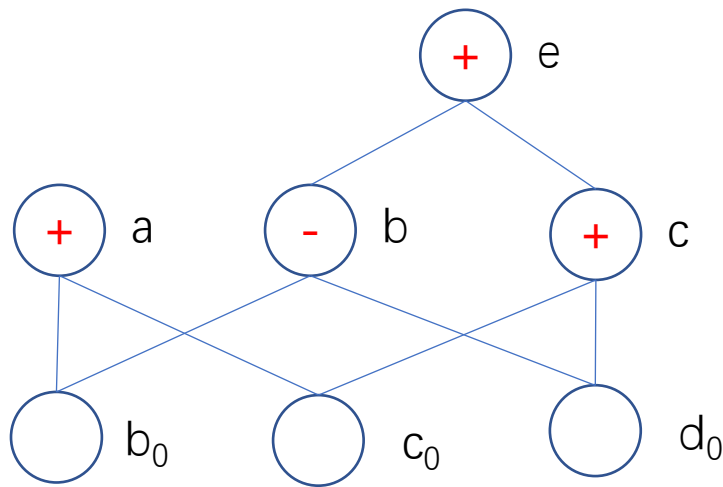
考虑如下基本块：

$a = b + c$

$b = b - d$

$c = c + d$

$e = b + c$



假设a和b是活跃变量，但c和e不是



- 活跃变量是指其值可能会在以后被使用的变量
- 在DAG上删除死代码，可进行如下操作
 - 1. 删除所有没有关联活跃变量的根结点
 - 2. 重复1操作

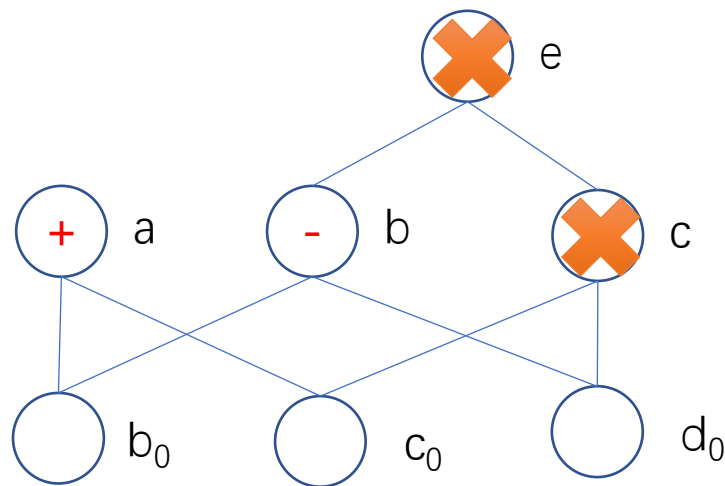
考虑如下基本块：

$a = b + c$

$b = b - d$

$c = c + d$

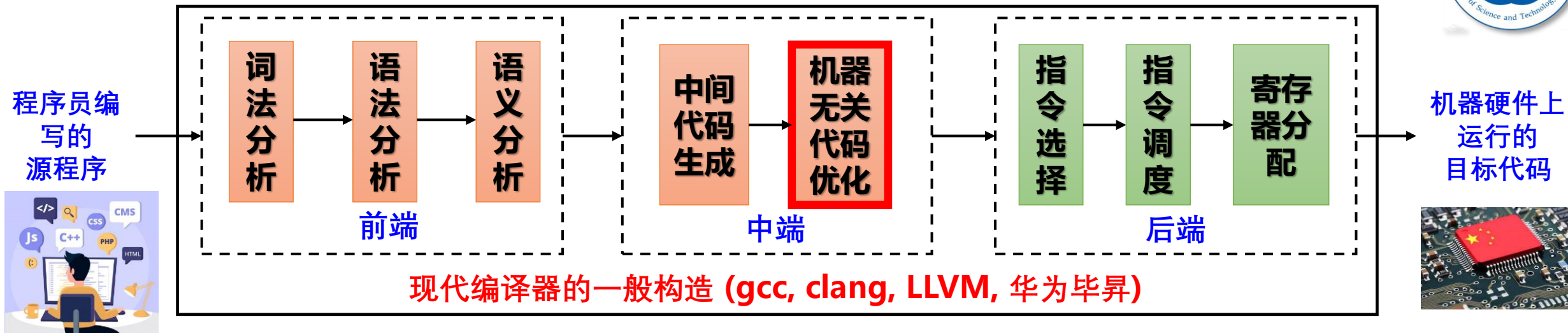
$e = b + c$



假设a和b是活跃变量，但c和e不是



本节提纲



- 基本块的DAG表示
- 优化的基本方法
 - 删除局部公共子表达式、删除死代码
 - 代数恒等式的使用、数组引用的表示
- 从DAG到基本块的重组



代数恒等式的使用



- $\mathbf{x} + \mathbf{0} = \mathbf{0} + \mathbf{x} = \mathbf{x}$
- $\mathbf{x} - \mathbf{0} = \mathbf{x}$
- $\mathbf{x} * \mathbf{1} = \mathbf{1} * \mathbf{x} = \mathbf{x}$
- $\mathbf{x} / \mathbf{1} = \mathbf{x}$
- 相关的计算可以消除

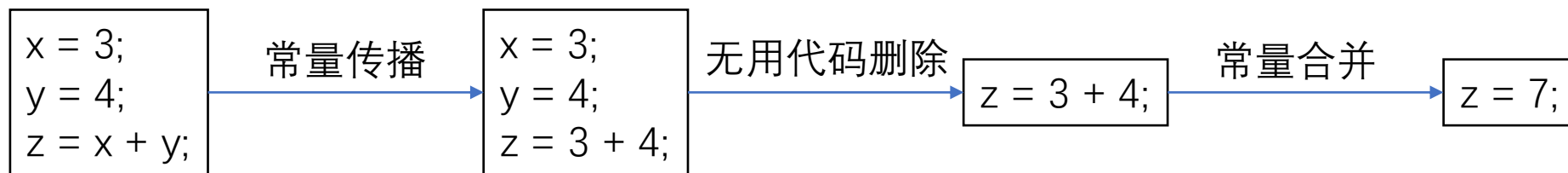


• 局部强度消减

- $x^2 = x * x$
- $2 * x = x + x$
- $x / 2 = x * 0.5$



• 常量合并 (constant folding) : 计算常量表达式的值



• 常量来源

- 程序员编写的, 比如magic number
- 宏定义展开后带来的, 这种情况在大型工程文件中非常普遍
- 在程序优化过程中由其它的优化技术带来的常数



代数恒等式的使用



• 交换律

- $x * y = y * x$

- 识别可用表达式时，这两者等价，需要考虑两种情况

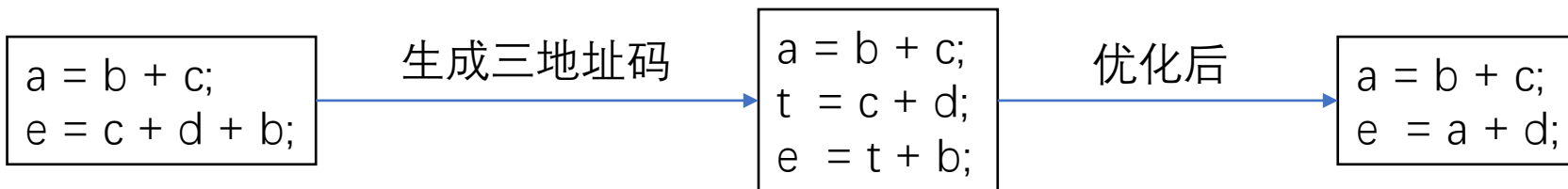


• 交换律

- $x * y = y * x$

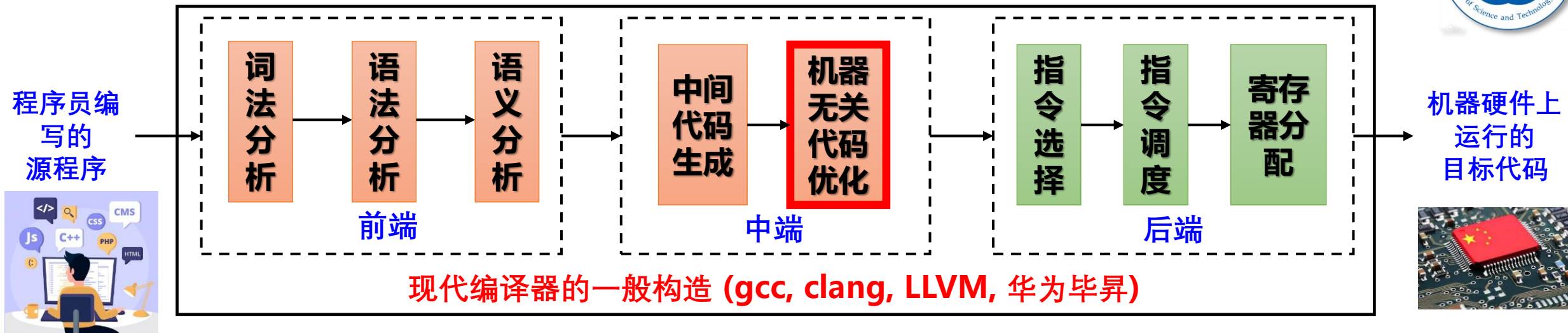
- 识别可用表达式时，这两者等价，需要考虑两种情况

• 结合律





本节提纲



- 基本块的DAG表示
- 删除局部公共子表达式
- 删除死代码
- 数组引用的表示
- 从DAG到基本块的重组



数组引用的表示



考虑如下基本块：

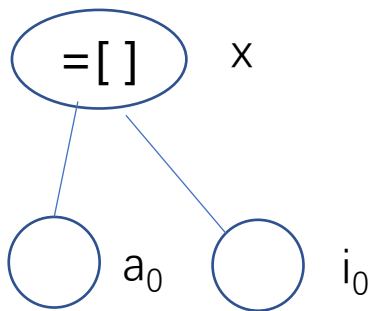
$x = a[i]$

$a[j] = y$

$z = a[i]$

在构造DAG时，
如何避免将 $a[i]$
误判为公共子
表达式

- 对于形如 $x=a[i]$ 的三地址指令创建一个运算符为 $=[]$ 的结点
- 该结点的子结点为 a 和 i
- 该结点的关联变量是 x





数组引用的表示



考虑如下基本块：

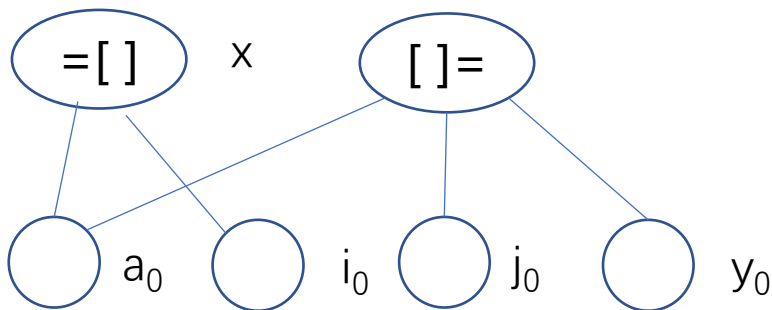
$x = a[i]$

$a[j] = y$

$z = a[i]$

在构造DAG时，
如何避免将 $a[i]$
误判为公共子
表达式

- 对于形如 $a[j]=y$ 的三地址指令创建一个运算符为 $[]=$ 的结点
- 该结点的子结点为 a, j 和 y
- 该结点没有关联变量





数组引用的表示



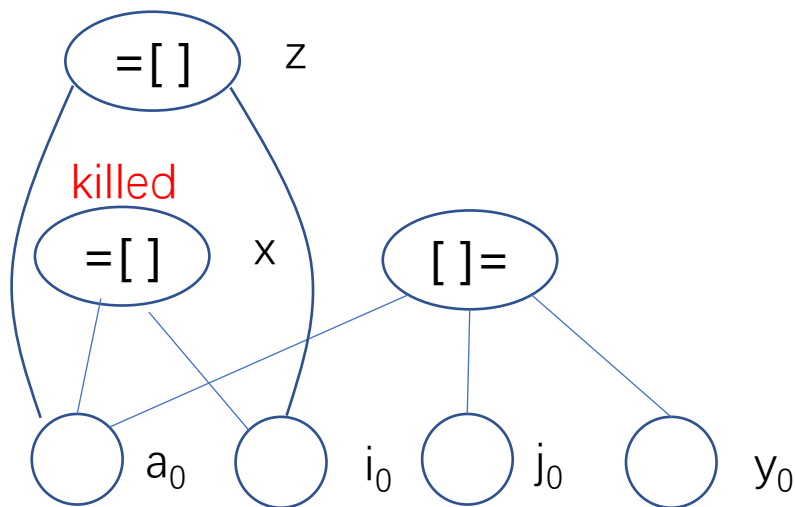
考虑如下基本块：

$x = a[i]$

$a[j] = y$

$z = a[i]$

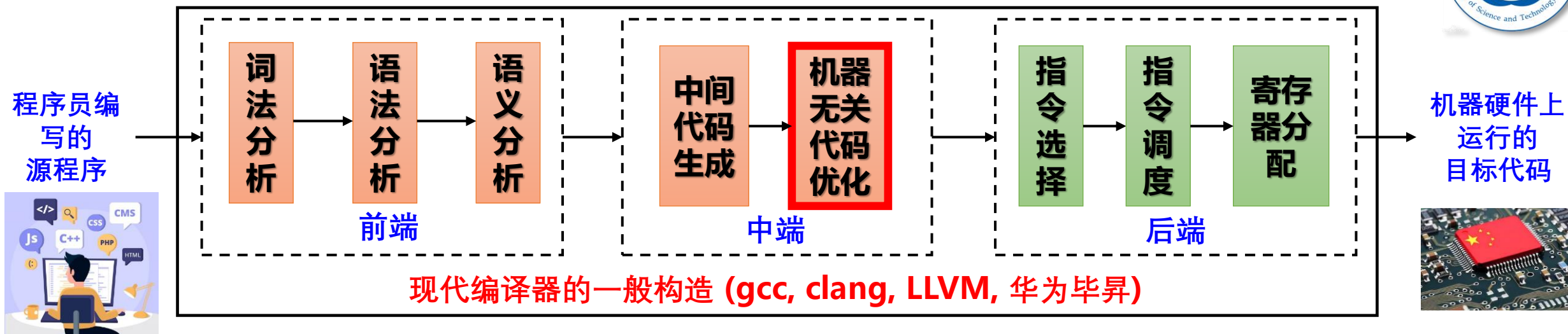
在构造DAG时，
如何避免将 $a[i]$
误判为公共子
表达式



- 对于形如 $a[j]=y$ 的三地址指令创建一个运算符为 $[]=$ 的结点
- 该结点的子结点为 a , j 和 y
- 该结点没有关联变量
- 该结点将杀死所有已经建立的、其值依赖于 a_0 的结点
- 被杀死的结点不能再关联定值变量，也就不能成为公共子表达式



本节提纲



- 基本块的DAG表示
- 删除局部公共子表达式
- 删除死代码
- 数组引用的表示
- 从DAG到基本块的重组



- 当完成基本块优化后，就可以根据优化得到的DAG生成新的等价的三地址代码
- 对每个具有若干关联定值变量的结点，构造一个三地址指令来计算其中某个变量的值
 - 倾向于把计算得到的结果赋给一个在基本块出口处活跃的变量(如果没有全局活跃变量的信息作为依据，就要假设所有变量在基本块出口处活跃，不包含编译器为处理表达式而生成的临时变量)
 - 如果结点有多个附加活跃变量，必须用复制语句



从DAG到基本块的重组—举例



考虑如下基本块：

$$b = 3$$

$$d = a + c$$

$$e = a * c$$

$$f = e + d$$

$$g = b * f$$

$$h = a + c$$

$$i = a * c$$

$$j = h + i$$

$$k = b * 5$$

$$l = k + j$$

$$m = l$$



从DAG到基本块的重组—举例



考虑如下基本块：

$b = 3$

$d = a + c$

$e = a * c$

$f = e + d$

$g = b * f$

$h = a + c$

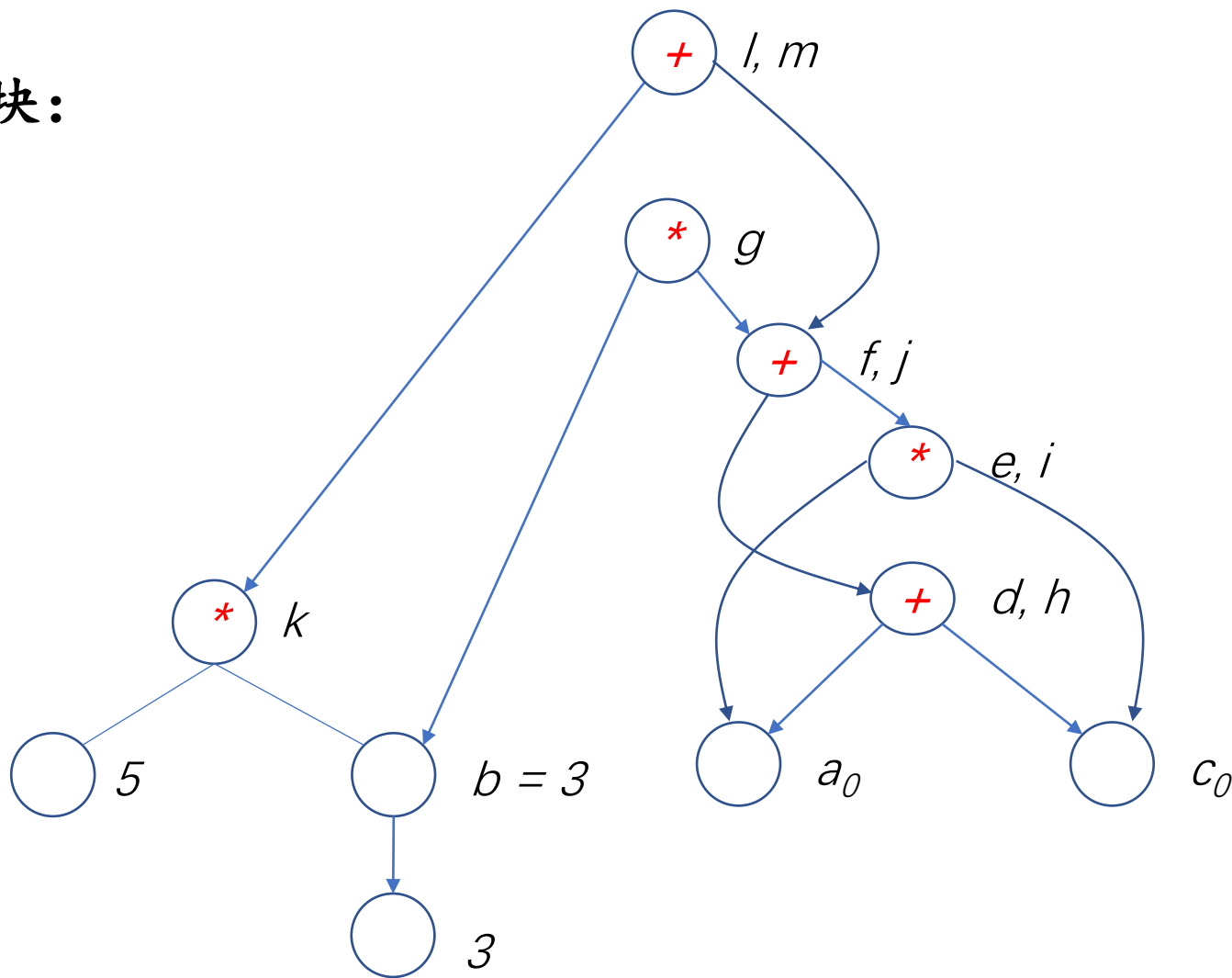
$i = a * c$

$j = h + i$

$k = b * 5$

$l = k + j$

$m = l$





从DAG到基本块的重组—举例



考虑如下基本块：

$b = 3$

$d = a + c$

$e = a * c$

$f = e + d$

$g = b * f$

$h = a + c$

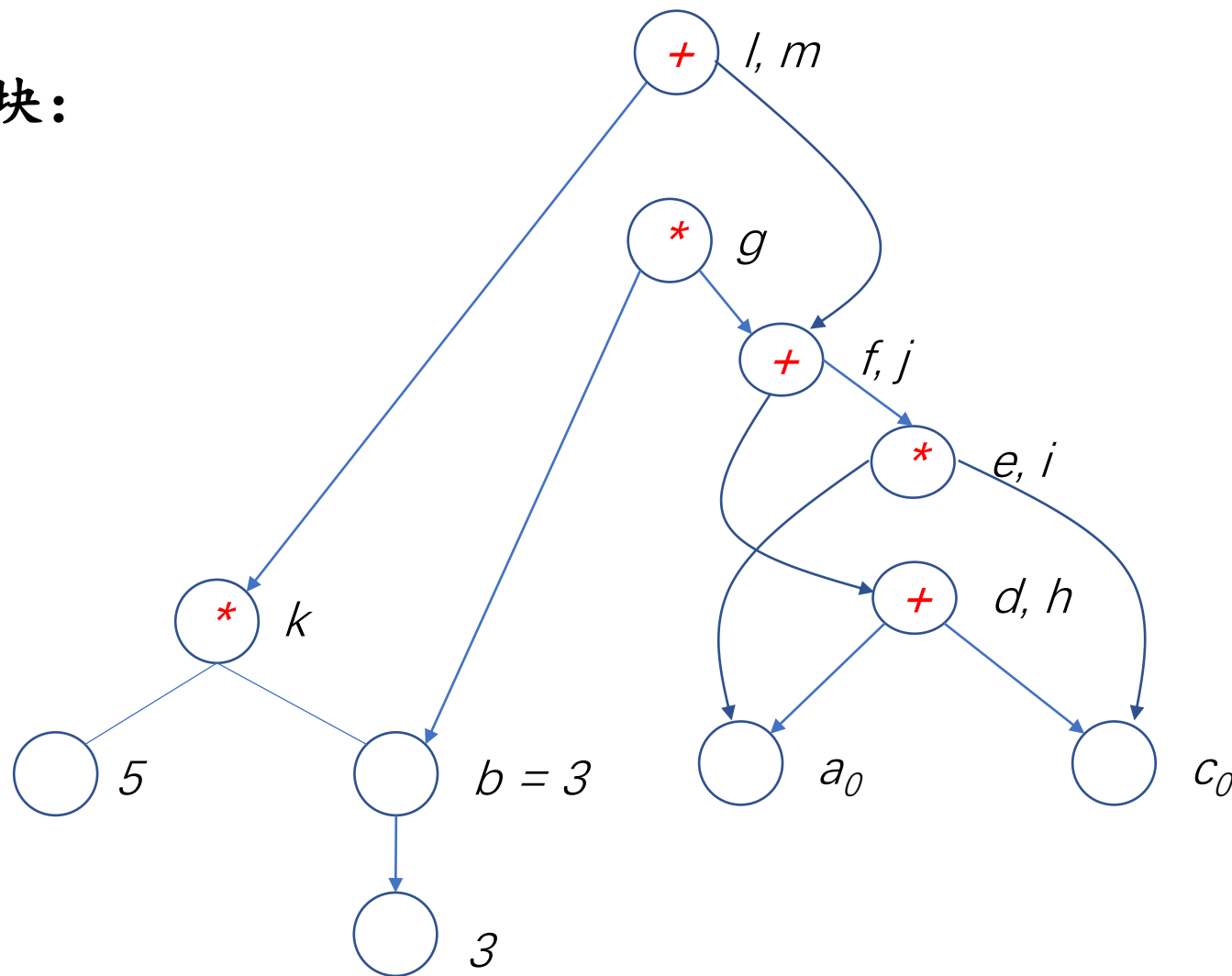
$i = a * c$

$j = h + i$

$k = b * 5$

$l = k + j$

$m = l$



假设基本块结束时仅有 l 是活跃的



从DAG到基本块的重组—举例



考虑如下基本块：

$b = 3$

$d = a + c$

$e = a * c$

$f = e + d$

$g = b * f$

$h = a + c$

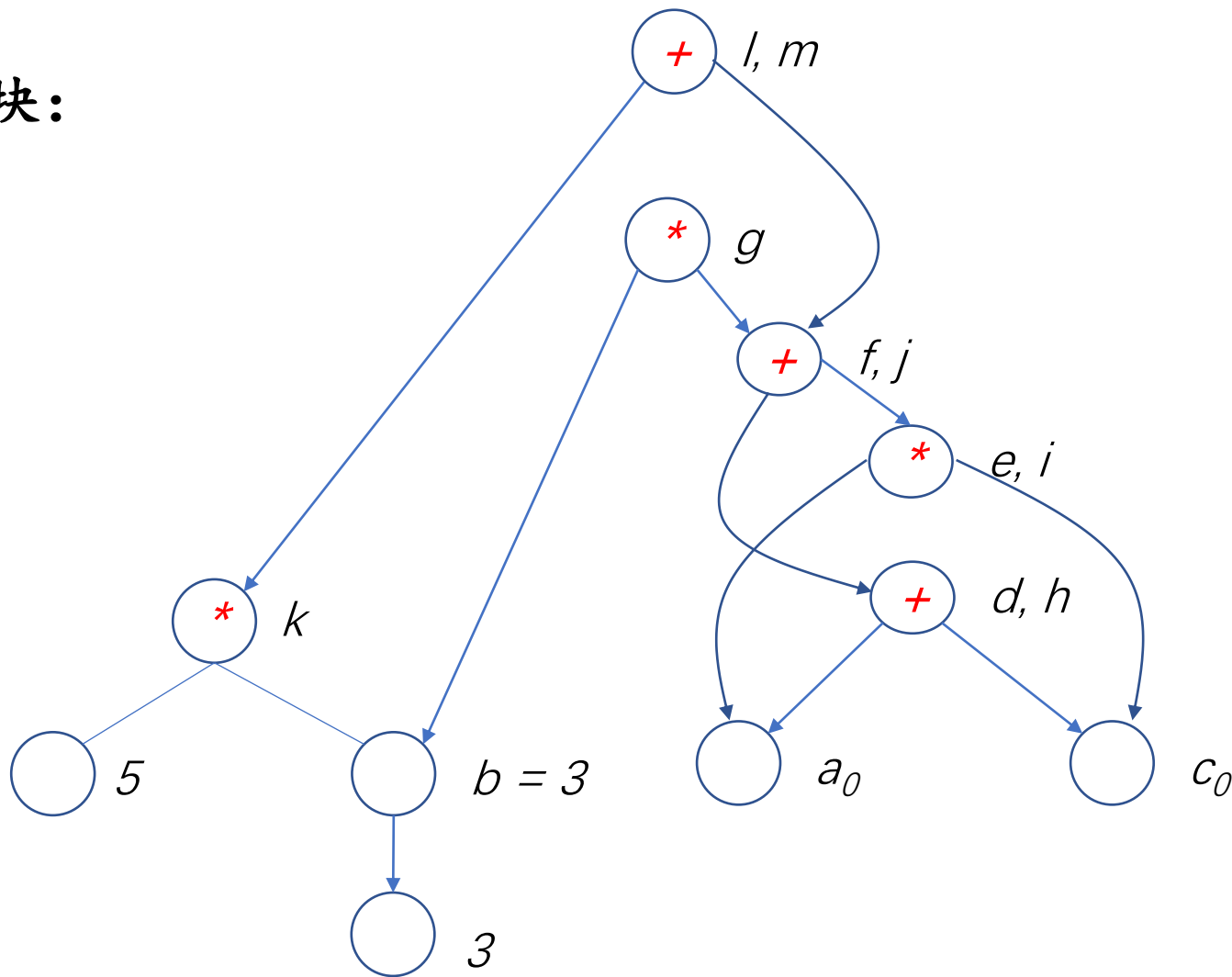
$i = a * c$

$j = h + i$

$k = b * 5$

$l = k + j$

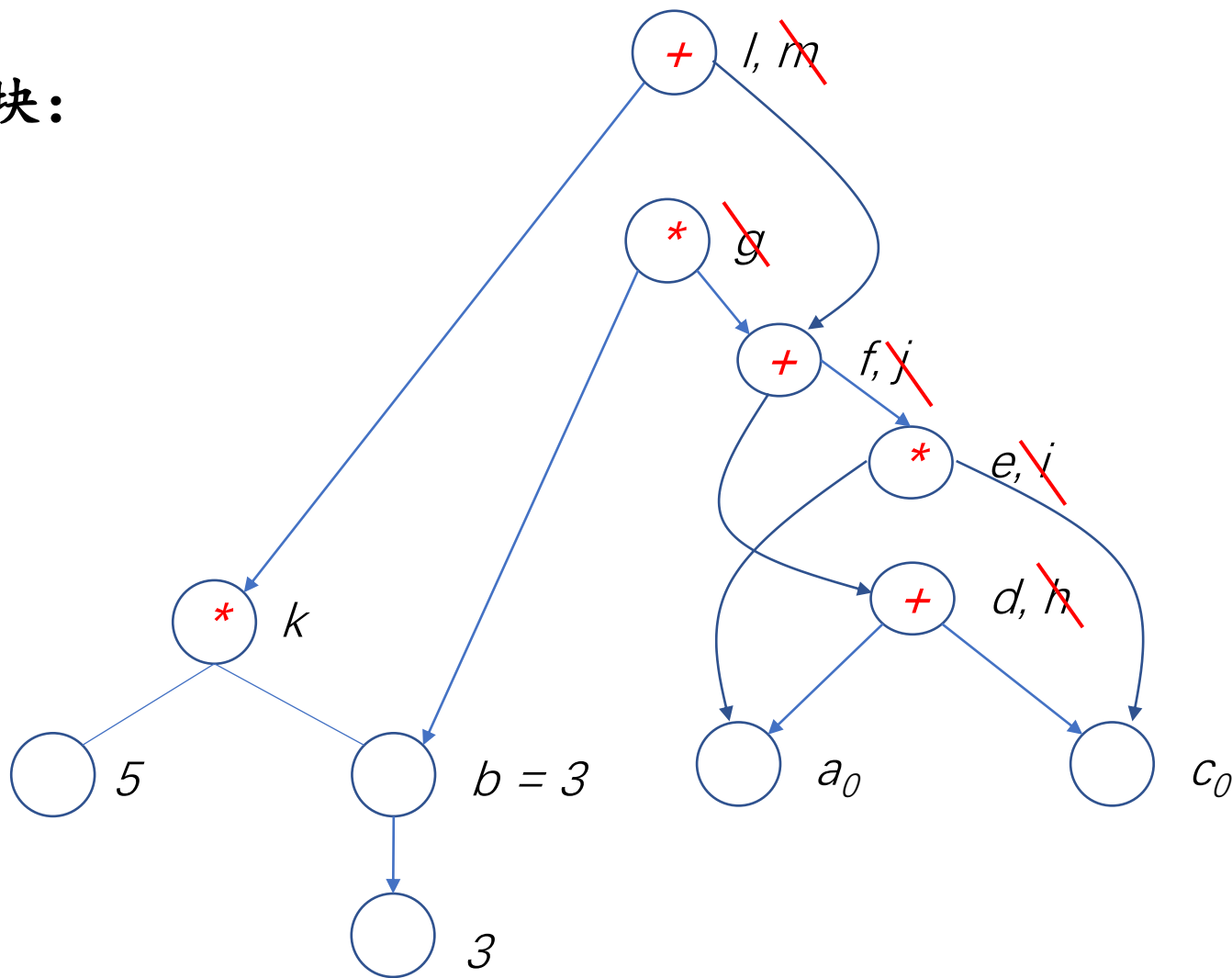
$m = l$



假设基本块结束时仅有 l 是活跃的


$$\mathbf{d} = \mathbf{a} + \mathbf{c}$$
$$\mathbf{e} = \mathbf{a} * \mathbf{c}$$
$$\mathbf{f} = \mathbf{e} + \mathbf{d}$$
$$\mathbf{g} = \mathbf{b} * \mathbf{f}$$

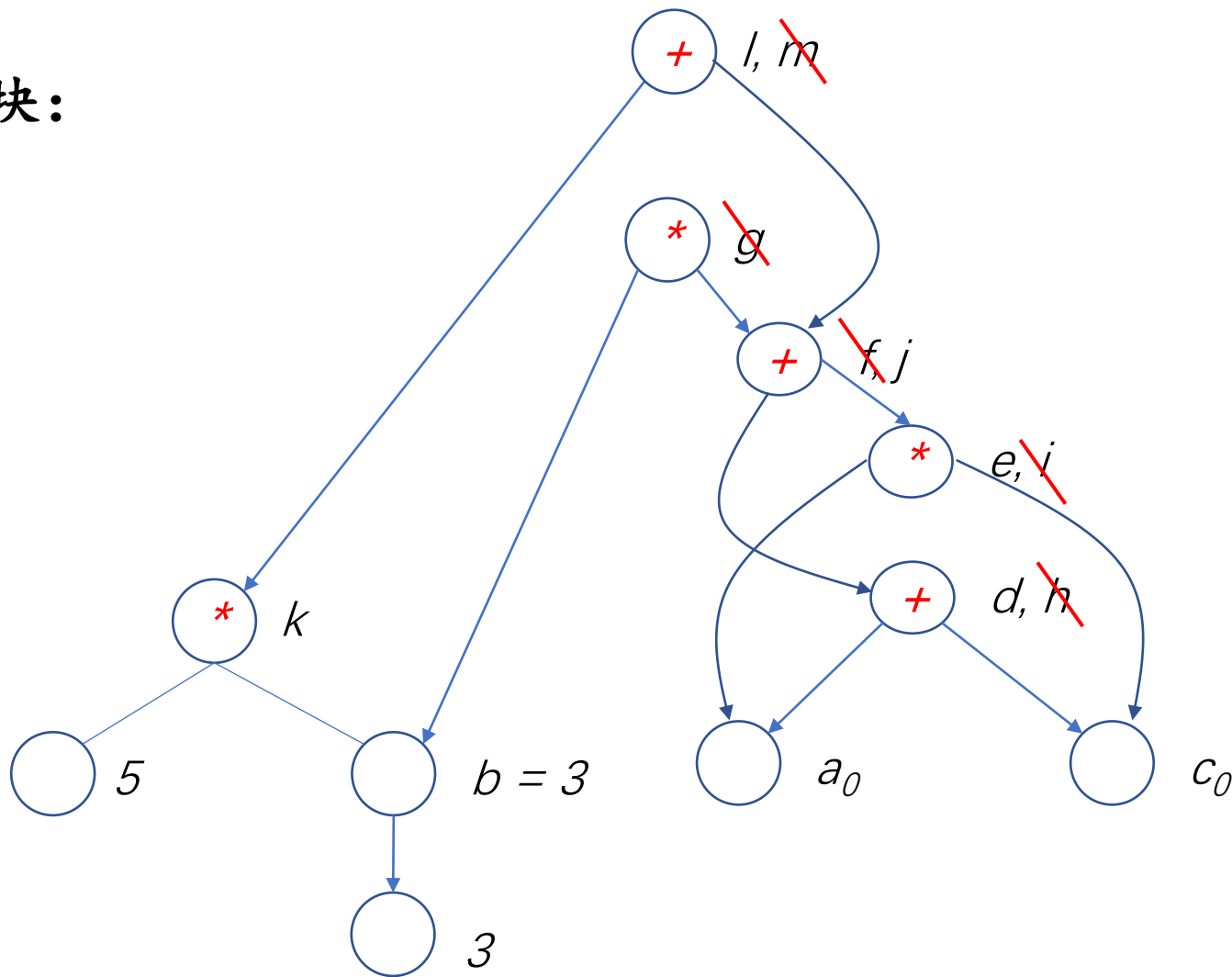
h = a + c

$$\mathbf{i} = \mathbf{a} * \mathbf{c}$$
$$\mathbf{j} = \mathbf{h} + \mathbf{i}$$
$$\mathbf{k} = \mathbf{b} * 5$$
$$\mathbf{l} = \mathbf{k} + \mathbf{j}$$
$$\mathbf{m} = \mathbf{l}$$


假设基本块结束时仅有 I 是活跃的


$$\mathbf{d} = \mathbf{a} + \mathbf{c}$$
$$\mathbf{e} = \mathbf{a} * \mathbf{c}$$
$$\mathbf{f} = \mathbf{e} + \mathbf{d}$$
$$\mathbf{g} = \mathbf{b} * \mathbf{f}$$

h = a + c

$$\mathbf{i} = \mathbf{a} * \mathbf{c}$$
$$\mathbf{j} = \mathbf{h} + \mathbf{i}$$
$$\mathbf{k} = \mathbf{b} * 5$$
$$\mathbf{l} = \mathbf{k} + \mathbf{j}$$
$$\mathbf{m} = \mathbf{l}$$


优化后的序列

$$\mathbf{d} = \mathbf{a} + \mathbf{c}$$
$$\mathbf{e} = \mathbf{a} * \mathbf{c}$$
$$\mathbf{j} = \mathbf{e} + \mathbf{d}$$
$$\mathbf{l} = 15 + \mathbf{j}$$

假设基本块结束时仅有 i 和 j 是活跃的



一起努力 打造国产基础软硬件体系！

李 诚

国家高性能计算中心(合肥)、信息与计算机国家级实验教学示范中心

计算机科学与技术学院

2023年11月20日