冗余指令消除

- 多余的LD/ST指令
  - LD R_0, a
  - ST a, R_0　　// 非跳转目标；可删除

- 级联跳转代码
  - if debug == 1 goto L1; goto L2; L1: …; L2: …;
    - => if debug != 1 goto L2; L1: …; L2: …;
  - 如果已知debug一定是0，那么替换成为goto L2

控制流优化

- goto L1; … …; L1: goto L2
  - => goto L2; … …; L1: goto L2

- if a < b goto L1; … … ; L1: goto L2
  - => if a < b goto L2; … … ; L1: goto L2

代数化简/强度消减和机器特有指令

- 应用代数恒等式进行优化
  - 消除$x = x + 0, x = x * 1, \ldots$
  - 用$x * x$替换x^2
- 使用机器特有指令
  - INC, DEC, …

树重写实现指令选择

- 在某些机器上，同一个三地址指令可以使用多种机器指令实现，有时多个三地址指令可以使用一个机器指令实现

- **指令选择**
  - 为实现中间表示形式中出现的运算符选择适当的机器指令
  - 用树来表示中间代码，按照特定的规则不断覆盖这棵树并生成机器指令

树翻译方案生成目标指令示例

- 如何完成树匹配?
  - 把树重写规则替换成相应的上下文无关文法的产生式
  - 产生式的右部是其指令模板的前缀表示
- 如果在某个时刻有多个模板可以匹配
  - 匹配到大树优先

1)	R_i	$\rightarrow$ $\mathbf{c}_a$	{ LD Ri, #a }
2)	R_i	$\rightarrow$ M_x	{ LD Ri, x }
3)	M	$\rightarrow$ $= M_x\ R_i$	{ ST x, Ri }
4)	M	$\rightarrow$ $=$ **ind** $R_i\ R_j$	{ ST *Ri, Rj }
5)	R_i	$\rightarrow$ **ind** $+$ $\mathbf{c}_a\ R_j$	{ LD Ri, a(Rj) }
6)	R_i	$\rightarrow$ $+\ R_i$ **ind** $+$ $\mathbf{c}_a\ R_j$	{ ADD Ri, Ri, a(Rj) }
7)	R_i	$\rightarrow$ $+\ R_i\ R_j$	{ ADD Ri, Ri, Rj }
8)	R_i	$\rightarrow$ $+\ R_i\ \mathbf{c}_1$	{ INC Ri }
9)	R	$\rightarrow$ **sp**	
10)	M	$\rightarrow$ **m**	

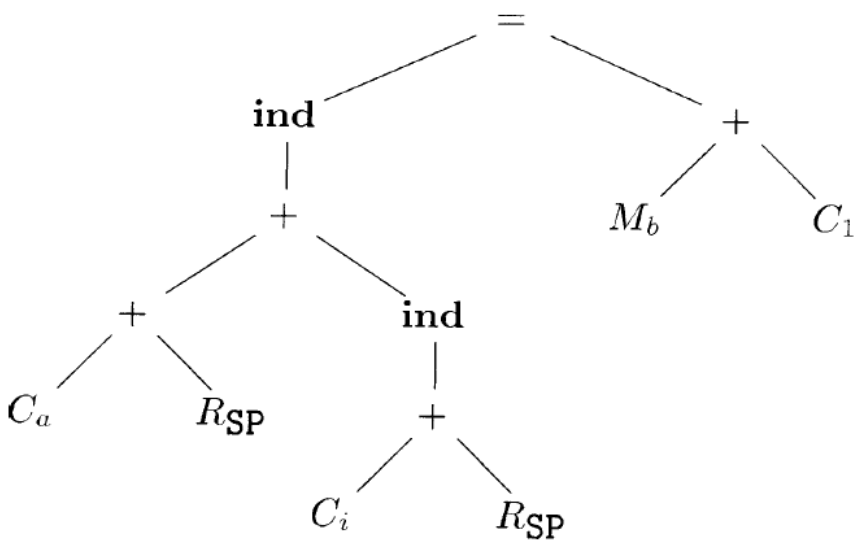

图 8-21　由图 8-20 构造得到的语法制导翻译方案