

10. 压缩条件: (1) 立即数或偏移量的值较小.

(2) 其中一个寄存器为零寄存器 $x0$, 返回地址寄存器 $x1$, 或栈指针 $x2$.

(3) 目标寄存器与第一个源寄存器相同.

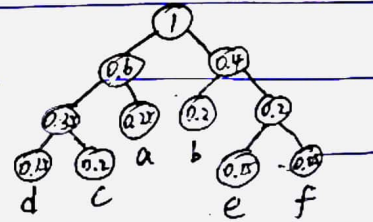
(4) 只使用 8 个寄存器.

满足一个条件即可.

不能. 基于寄存器的访存指令, 分支指令, 部分运算指令只能访问 $x8 \sim x15$.

18.

a_i	p_i	l_i	$c(a_i)$
a	0.25	2	01
b	0.2	2	10
c	0.2	3	001
d	0.15	3	000
e	0.15	3	110
f	0.05	3	111



9. (1) $\pm 1 \text{ MB}$

(2) $\pm 4 \text{ KB}$

(3) 可以. 对 32 位地址 $a[31:0]$, 使用:

`lui r1, a[31:12]` // 加载地址的高 20 位.

`jalr r1, r2, a[11:0]` // 联合高 20 位和低 12 位.

实现跳转.

`lui` 可将 20 位立即数存入寄存器高位. `jalr` 可引用一寄存器的值, 再加上 12 位偏移量, 刚好实现 32 位地址跳转.

$$\text{平均长度 } \bar{l} = \sum_{i=1}^6 p_i l_i = 2.55$$

$$\text{信息冗余量 } R = 1 - \frac{H}{\bar{l}} = 0.033$$

19. (1) 每次调用函数均需在堆栈中开辟新区域, 保存调用信息. 而堆栈空间有限. 若开辟次数过多则将占据全部空间并溢出.

(2) 在函数正常运作中调用次数过多时, 可修改系统设置中堆栈的大小, 一定程度上增加允许调用的次数. 也可在编程中优化算法, 尽量减少堆栈的使用.

若函数错误会导致程序死循环, 则应增加判



断环节,防止陷入死循环.

若程序编写错误导致试运行中陷入死循环,
则应尽快强制中断运行,并排查错误.

20. ra(F1)

so(F1)

ao(F1)

to(F1)

ra(F2)

so(F2)

ao(F2)

al(F2)

to(F2)

tl(F2)

