

9. 1) jal 可跳转空间范围为 $2^{20}-1 \sim -2^{20}$

2) 空间范围: $2^{12}-1 \sim -2^{12}$

3) 可以, 用 lui 存入一个寄存器的高 31~12 位, 然后用 jalr 进行低 12 位的跳转

10. 能被压缩的条件

① 立即数或偏移量很小

② 其中一个寄存器是 x_0 或 x_1 或 x_2

③. 目标寄存器和源寄存器一样

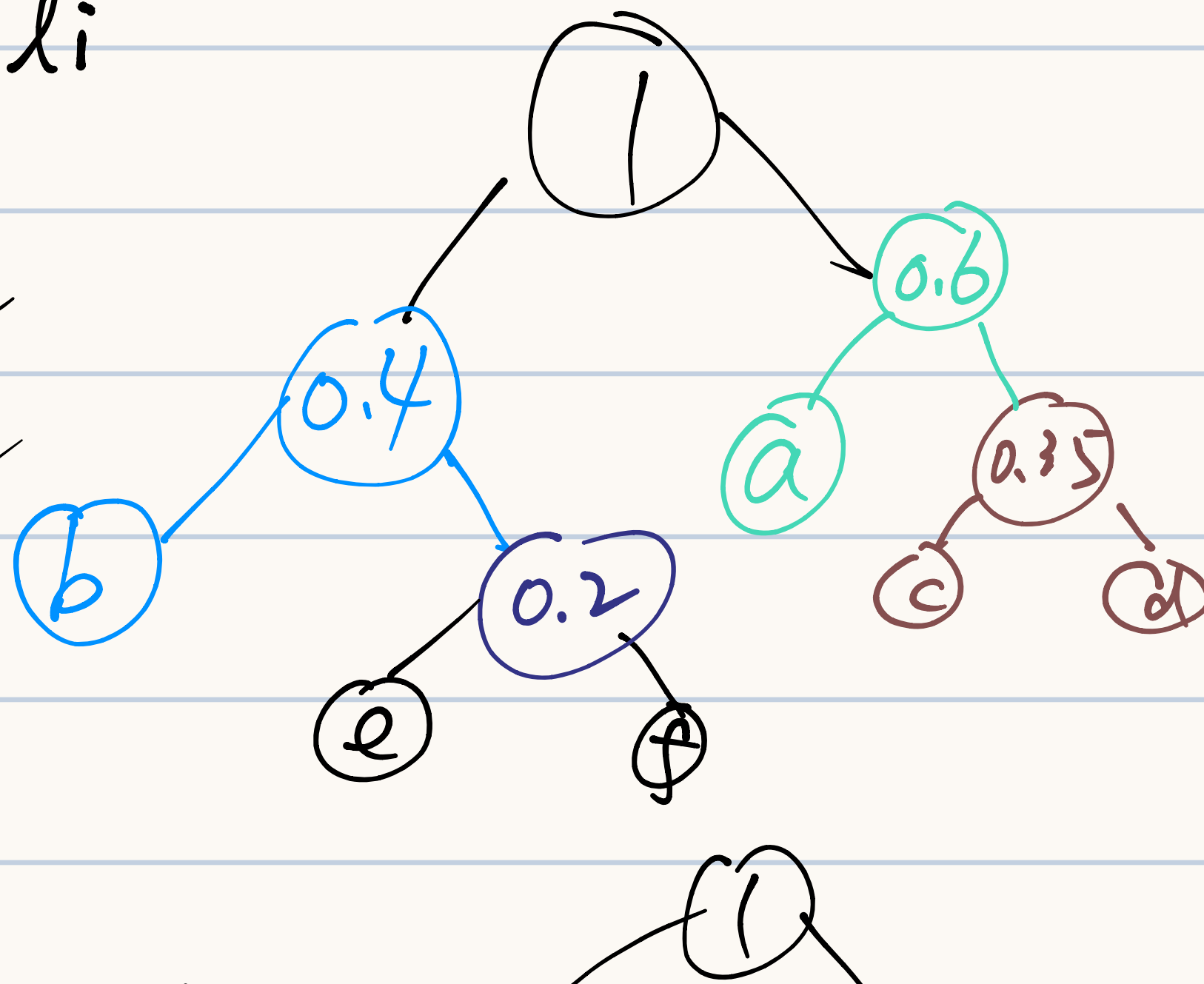
④ 使用的寄存器是最常用的 8 个之一

不是所有 C 指令都能使用全部 32 个寄存器:

CR, CI, CSS 可用 32 个

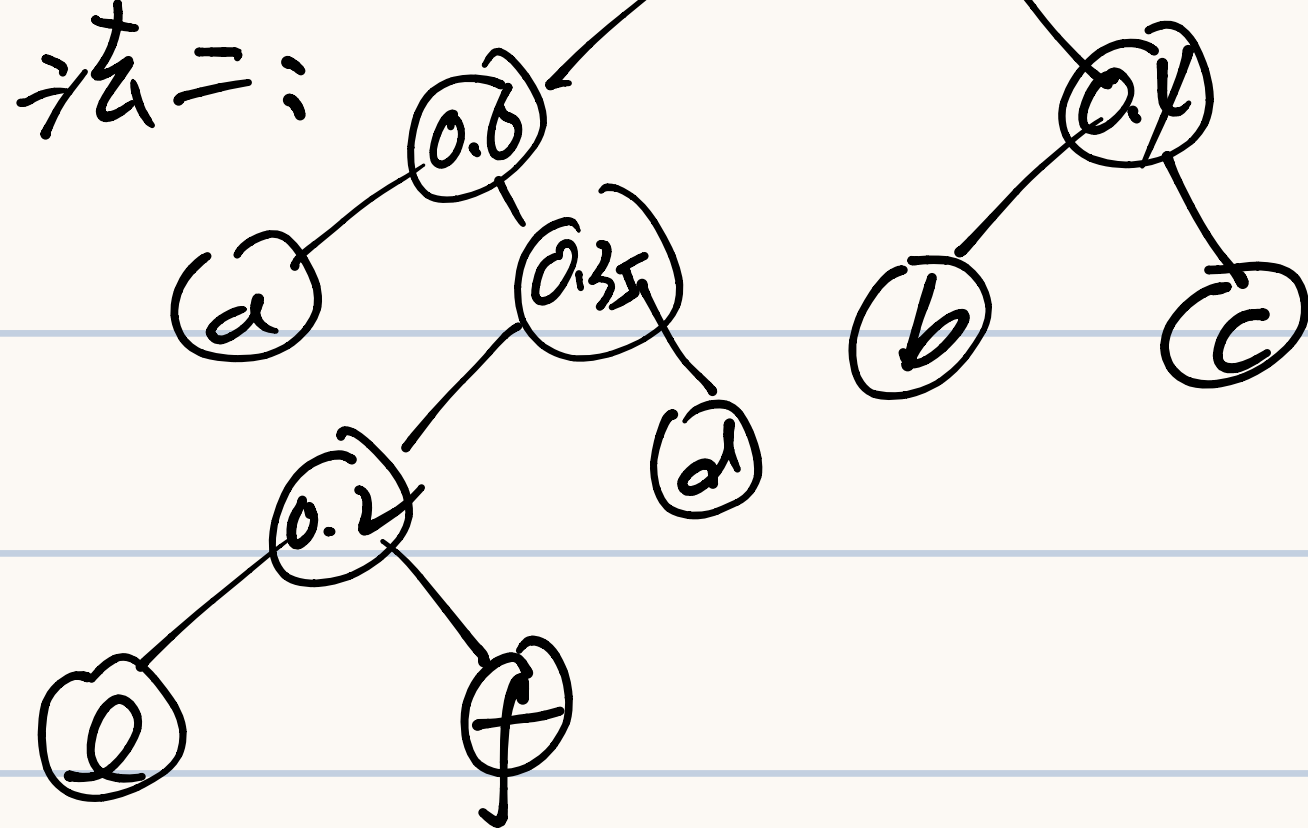
CIW, CL, CS, CA, CB 只能用 8 个

18.	a_i	p_i	法= l_i	法= l_i
	a	0.25	2	2
	b	0.20	2	2
✓	c	0.20	3	2
✓	d	0.15	3	3
✓	e	0.15	3	4
✓	f	0.05	3	4



$$0.5 + 0.4 + 0.4 + 0.45 + 0.6 + 0.2$$

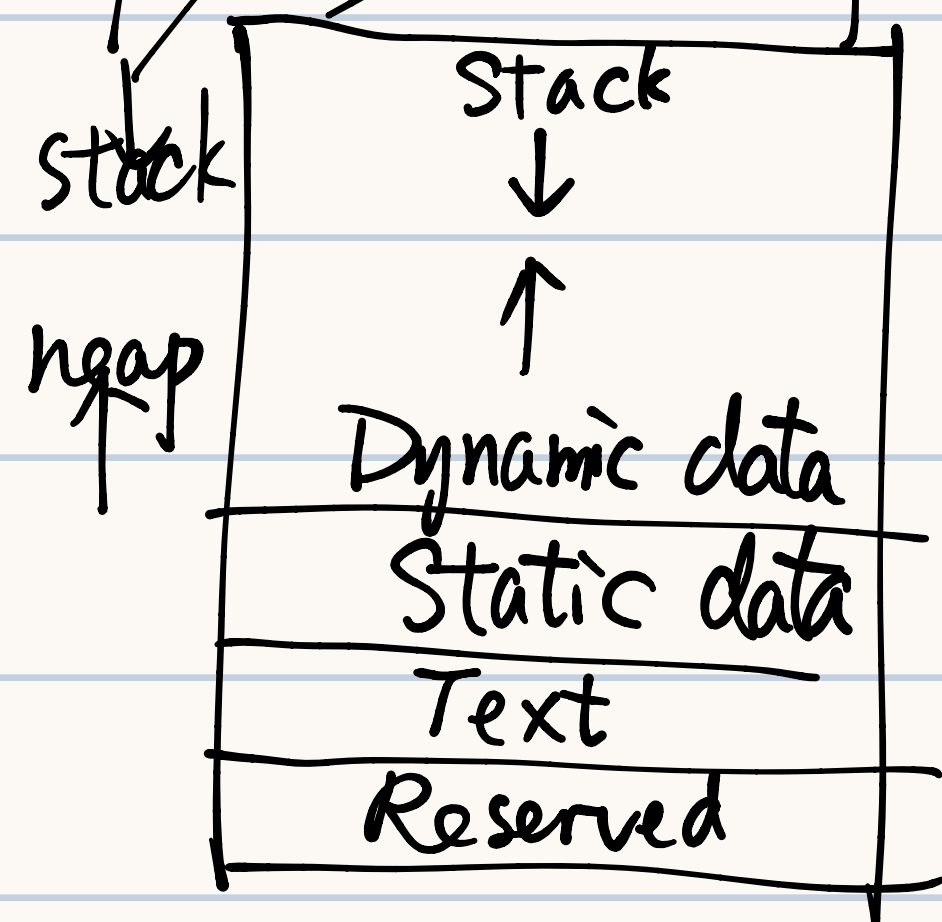
平均长度: $\sum_{i=1}^6 p_i l_i =$ 法一: 2.55
法二: 2.55



信息冗余度:

$$R = 1 - \frac{H}{2.55} = 0.033$$

19. 1) 由于计算机给程序分配的空间如下:



由于嵌套调用层数过多, stack 不断向下生长覆盖到其它需要的数据, 即栈溢出

2) 方法: 1° 减少函数调用层次, 慎用递归函数

2° 多使用指针, 从 heap 空间分配内存, 减少反复定义占用内存较多的 auto 变量

3° 在函数传参中使用指针而非直接传递大型结构或对象

20. ra(F1)

t0(F1)

s0(F1)

ra(F1)

F1: t0, s0

F2: t0-tl, s0-sl

→ 传参保护

$ra(F_2)$

$a_3(F_2)$

$ra(F_2)$

$t_0(F_2)$

$t_1(F_2)$

$s_0(F_2)$

$s_1(F_2)$

$ra(F_3)$

