1. 假设我们希望升级一个提供 Web 服务的处理器。新处理器执行 Web 服务应用程序的计算速度是原处理器的 10 倍。假设原处理器有 40%的时间忙于计算，60%的时间等待 I/O，进行这一升级后，所得到的总加速比为多少？

解答：

根据 Amdahl 定律：

$$总加速比=\frac{1}{(1-升级比例)+\frac{升级比例}{升级加速比}}$$

升级加速比为 10，升级比例为 40%，$总加速比=\frac{1}{0.6+\frac{0.4}{10}}=\frac{1}{0.64}\approx 1.56$。

2. 公司刚刚购买了一个新的 Intel Core i5 双核处理器,你接到针对这一处理器来优化软件的任务。你将在这个双核处理器上运行两个应用程序,但它们的资源需求并不一样。第一个程序需要 80%的资源,另一个仅需要 20%的资源。假定对该程序的一部分进行并行化时，该部分的加速比为 2。

a. 假定第一个应用程序的 40%可以并行化，那么在隔离运行时，通过这个应用程序可以实现多大的加速比?

b. 假定第二个应用程序的 99%可以并行化,那么在隔离运行时，这个应用程序可以达到多大的加速比?

c. 假定第一个应用程序的 40%可以并行化，如果对其实现并行化，系统总加速比为多少?

d. 假定第二个应用程序的 99%可以并行化，如果对其实现并行化，系统总加速比为多少?

解答：

a. $总加速比=\frac{1}{(1-升级比例)+\frac{升级比例}{升级加速比}}$，升级加速比为 2，升级比例为 40%，

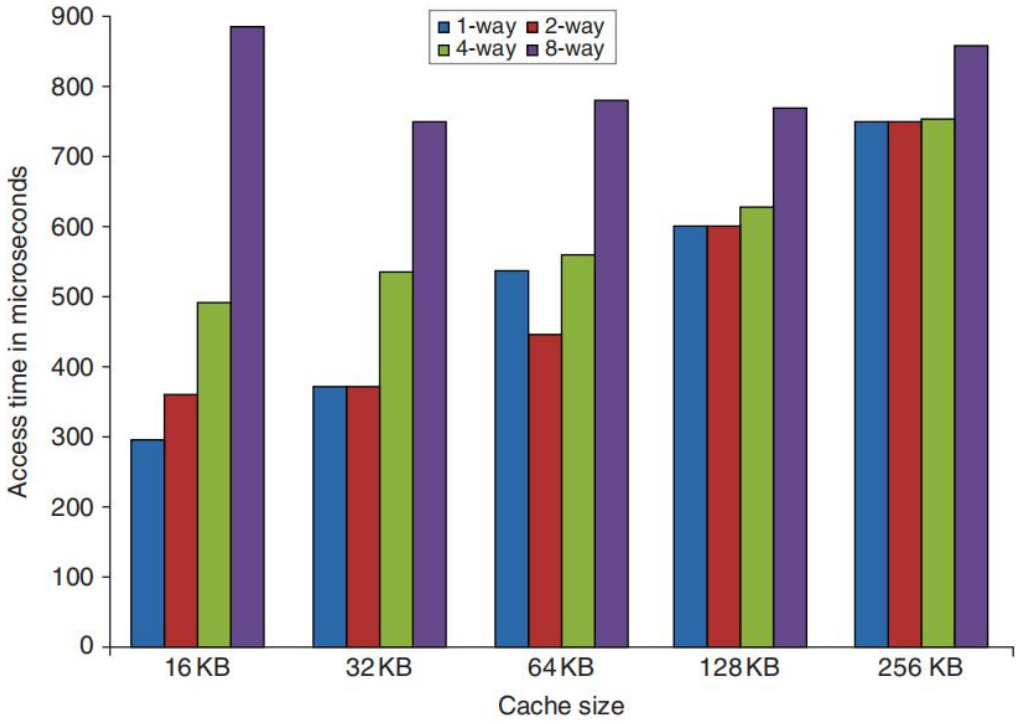
$总加速比=\frac{1}{0.6+\frac{0.4}{2}}=1.25$。

b. 升级加速比为 2，升级比例为 99%，$总加速比=\frac{1}{0.01+\frac{0.99}{2}}=1.98$。

c. $总加速比=\frac{1}{0.2+0.8\times 0.6+\frac{0.8\times 0.4}{2}}=1.19$

d. $总加速比=\frac{1}{0.8+0.2\times 0.01+\frac{0.2\times 0.99}{2}}=1.11$

3. 利用如图所示的数据，判断 32 KB 四路组相联 L1 缓存的存储器访问时间是否快于 32 KB 两路组相联 L1 缓存器。假定 L2 的缺失代价是快速 L1 缓存访问时间的 15 倍。忽略超出 L2 之外的缺失。哪种缓存的存储器平均缓存时间较短?



32	一路	0.042	0.0001	0.2%	0.037	89%	0.005	11%
32	两路	0.038	0.0001	0.2%	0.037	99%	0.000	0%
32	四路	0.037	0.0001	0.2%	0.037	100%	0.000	0%
32	八路	0.037	0.0001	0.2%	0.037	100%	0.000	0%

解答：

设两路组相联缓存的访问时间为 1。则对于两种缓存:

存储器平均访问时间（两路）=命中时间+缺失率 x 缺失代价=1+0.038x 15=1.57。

对于四路缓存,访问速度是它的 1.4 倍。缺失代价占用的时间为 15/1.4=10.1。

存储器平均访问时间（四路）=命中时间（两路）x 1.4+缺失率 x 缺失代价=1.4+0.037x 10=1.77。

所以，四路组相联平均缓存时间较短。

4. 在直写 L1 缓存与写回 L2 缓存之间设计一个写缓冲区。L2 缓存写数据总线的宽度为 16B，可以每 4 个处理器周期向一个独立缓存地址执行一次写操作。

a. 每个写缓冲区项目应当为多少字节?

b. 如果所有其他指令可以与存储指令并行发射，块存在于 L2 缓存中，在通过执行 64 位存储指令将存储器置零时，使用一个合并写缓冲区来代替非合并缓冲区，在稳定状态下可以得到什么样的加速比?

c. 对于采用阻塞缓存与非阻塞缓存的系统，可能出现的 L1 缺失对于所需写缓冲区项目的个数有什么样的影响?

解答：a.16B 字节，为了匹配 L2 缓存写路径。

b.假设合并写缓冲区条目宽度为 16B，由于每个存储可以写 8B，合并写缓冲区项目将需要 2 个周期。L2 缓存将花费 4 个周期来写入每个项目。非合并写入缓冲区需要 4 个周期来写入每个存储的 8B 结果。这意味着合并缓冲区的处理速度是非合并的 2 倍。

c.对于阻塞高速缓存，未命中的存在有效地延缓了机器的进度，因此是否存在未命中不会改变所需的写缓冲区项目的数量。使用非阻塞高速缓存，可以在未命中期间从写缓冲区中处理写操作，所以它的项目数量可以少一点。

5.在具有 4K 项的（0,2）分支预测中有多少位？在具有同样位数的（2,2）预测器中有多少项？

解答：

一个(m,n)预测器的位数为：

$$2^m \times n \times \text{由分支地址选中的预测项数目}$$

所以具有 4K 的预测器拥有：$2^0 \times 2 \times 4K = 8K$，

在具有同样位数的（2,2）预测器中：$2^2 \times 2 \times \text{由分支地址选中的预测项数目} = 8K$，

$\text{由分支地址选中的预测项数目} = 1K$。

6. 如果你不清楚寄存器重命名程序必须做哪些工作，可以回过头来看看正在执行的汇编代码，问问自己，必须具备哪些条件才能获得正确结果。例如，考虑一个 3 路超标量机器，同时对下面这 3 条指令进行重命名:

ADDI RI, R1, R1;

ADDI R1, R1, R1;

ADDI R1, R1, R1;

如果 R1 的值在初始为 5,那么执行这一序列之后，它的值应当是多少?

解答：

ADDI RI, R1, R1; 5+5->10

ADDI R1, R1, R1; 10+10->20

ADDI R1, R1, R1; 20+20->40

最后结果为 40。

7. 如果在先前指令执行完毕之前，不会开始执行新的执行，那表中代码序列的基准性能如何(用每次循环迭代的时钟周期表示) ？忽略前端提取与译码过程。假定执行进程没有因为缺少下一条指令而停顿，但每个周期只能发射一条指令。假定该分支被选中，而且存在一个时钟周期的分支延迟槽。

				超过一个时钟周期的延迟
Loop:	LD	F2,0(RX)	存储器LD	+4
I0:	DIVD	F8,F2,F0	存储器SD	+1
I1:	MULTD	F2,F6,F2	整数ADD, SUB	+0
I2:	LD	F4,0(Ry)	分支	+1
I3:	ADDD	F4,F0,F4	ADDD	+1
I4:	ADDD	F10,F8,F2	MULTD	+5
I5:	ADDI	Rx,Rx,#8	DIVD	+12
I6:	ADDI	Ry,Ry,#8		
I7:	SD	F4,0(Ry)		
I8:	SUB	R20,R4,Rx		
I9:	BNZ	R20,Loop		

解答：

Loop: LD F2,0(Rx) 1 + 4

DIVD F8,F2,F0 1 + 12

MULTD F2,F6,F2 1 + 5

LD F4,0(Ry) 1 + 4

ADDD F4,F0,F4 1 + 1

ADDD F10,F8,F2　1 + 1
ADDI Rx,Rx,#8　1
ADDI Ry,Ry,#8　1
SD F4,0(Ry)　1 + 1
SUB R20,R4,Rx　1
BNZ R20,Loop　1 + 1
共计 40 循环迭代的时钟周期。

8. 假设希望用 100 个处理器获得 80 倍的加速比。原计算中串行部分可以占多大比例？

解答：

Amdahal 定律：

$$\text{Speedup} = \frac{1}{\frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}} + (1 - \text{Fraction}_{\text{enhanced}})}$$

假设此程序仅以两种模式运行，一种为并行，所有处理器都可以充分工作；另一种模式为串行，仅有一个处理器工作，则此时增强模式下的加速比就是处理器的数目。
代入公式得：

$$80 = \frac{1}{\frac{Fraction_{parallel}}{100} + (1 - Fraction_{parallel})}$$

$Fraction_{parallel} = 0.9975$ 。

9. 假定有一个应用程序运行在包含 32 个处理器的多处理器上，它在引用远程存储器时需要的时间为 200 ns。对于这一应用程序，假定除涉及通信的引用之外，其他所有引用都会在本地存储器层次结构中命中。处理器会在远程请求时停顿，处理器时钟频率为 3.3 GHz。如果基础 CPI (假定所有引用都在缓存中命中)为 0.5,请对比在没有通信、0.2%的指令涉及远程通信引用这两种情况下，多处理器会快多少?

解答：

$$\text{CPI} = \text{Base CPI} + \text{Remote request rate} \times \text{Remote request cost}$$

多处理器实际的 $\text{CPI} = 0.5 + 0.2\% \times \text{Remote request cost}$

$$\text{Remote request cost} = \frac{\text{Remote access cost}}{\text{Cycle time}} = \frac{200}{0.3} = 666 cycles$$

CPI=1.5+0.2=1.7，当所有引用都是本地引用时，多处理器要快出 1.7/0.5=3.4 倍。

10. 考虑下面这样一个循环：

for (i=0; i<100; =+1) {

```
A[i+1] = A[i] + C[i]:    /*S1 */
   B[i+1] = B[i] + A[i+1]; /* 52 */}
```

假定 A、B 和 C 是没有重叠的不同数组。(在实践中，这些数组有时可能相同，或可能重叠。因为这些数组可能是作为参数传递给包含这一循环的过程，为了判断数组是否重叠或相同，通常需要对程序进行复杂的过程间分析。)在这个循环中，语句 S1 和 S2 之间的数据相关如何？

解答：

共有以下两种不同相关。

（1）S1 使用一个在先前迭代中由 S1 计算的值，这是因为迭代 i 计算 A[i+1], 然后在迭代 i+1 中读取它。对 B[i]和 B[i+1]来说，S2 也是如此。S1 依赖于 S1 的先前迭代,所以这种相关是循环间相关。这种相关迫使这个循环的连续选代必须按顺序执行。

（2）S2 使用由同一迭代中 S1 计算的值 A[1+1]。这种相关(S2 对 S1 的依赖)位于是同一层次上的迭代，不是循环间相关。因此，如果它是仅有的相关，那这个循环的多个迭代就能并行执行，只要一个迭代中的每个语句保持相对顺序即可。