

【程序性能优化】

1. 有如下的定义:

```
// 以下都是局部变量
int i, j, temp, ians;
int *p, *q, *r;
double dans;
// 以下都是全局变量
int iMat[100][100];
double dMat[100][100];
// 以下都是函数
int foo(int x);
```

判断编译器是否会自动将下列左侧代码优化为右侧代码:

(1)

```
ians = 0;
for (j = 0; j < 100; j++)
    for (i = 0; i < 100; i++)
        ians += iMat[i][j];
```

```
ians = 0;
for (i = 0; i < 100; i++)
    for (j = 0; j < 100; j++)
        ians += iMat[i][j];
```

【答】会

(2)

```
dans = 0;
for (j = 0; j < 100; j++)
    for (i = 0; i < 100; i++)
        dans += dMat[i][j];
```

```
dans = 0;
for (i = 0; i < 100; i++)
    for (j = 0; j < 100; j++)
        dans += dMat[i][j];
```

【答】不会, 因为浮点数不能结合

(3)

```
for (i = 0; i < foo(100); i++)
    ians += iMat[0][i];
```

```
temp = foo(100);
for (i = 0; i < temp; i++)
    ians += iMat[0][i];
```

【答】不会, 因为 foo 可能有副作用

(4)

```
*p += *q;
*p += *r
```

```
temp = *q + *r
*p += temp;
```

【答】不会, 如果 pqr 指向同一个元素那么两个运算不等价

2. 阅读下列 C 代码以及它编译生成的汇编语言

```
long func() {
    long ans = 1;
    long i;
    for (i = 0; i < 1000; i += 2) {
        ans = ans [?] (A[i] [?] A[i+1]);
    }
    return ans;
}
```

```

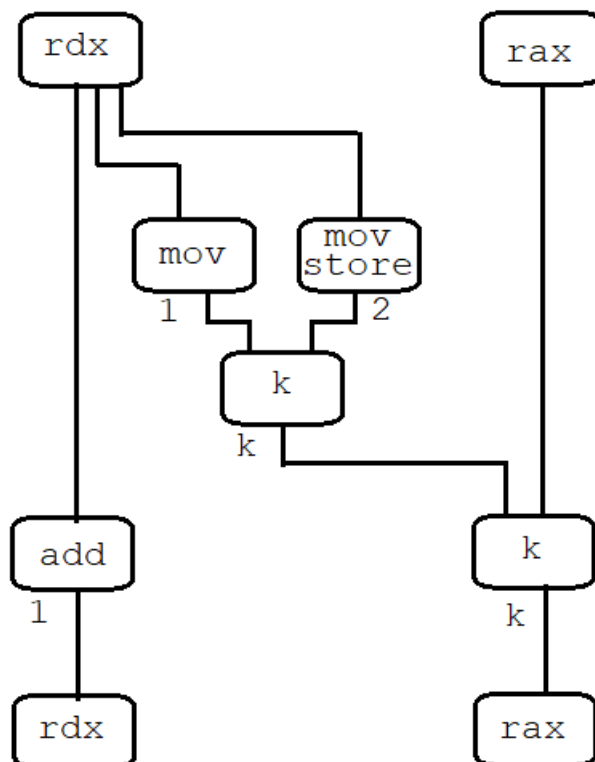
func:
    movl    $0, %edx
    movl    $1, %eax
    leaq    A(%rip), %rsi
    jmp .L2
.L3:
    movq    8(%rsi,%rdx,8), %rcx           // 2 cycles
    ??      (%rsi,%rdx,8), %rcx           // k + 1 cycles
    ??      %rcx, %rax                     // k cycles
    addq    $2, %rdx                       // 1 cycles
.L2:
    cmpq    $999, %rdx                     // 1 cycles
    jle .L3
    rep ret

```

该程序每轮循环处理两个元素。在理想的机器上（执行单元足够多），每条指令消耗的时间周期如右边所示。

- (1) 当问号处为乘法时， $k = 8$ 。此时这段程序的 CPE 为 4。
- (2) 当问号处为加法时， $k = 1$ 。此时这段程序的 CPE 为 0.5。

【答】数据相关图如下



【存储技术】

3. 对于下列描述, 是 SRAM 更符合还是 DRAM 更符合, 还是均符合?

		S	D
(1)	访问速度更快	S	
(2)	每比特需要的晶体管数目少		D
(3)	单位容量造价更便宜		D
(4)	常用作主存		D
(5)	需要定期刷新		D
(6)	断电后失去存储的信息	S	D
(7)	支持随机访问	S	D

4. 勾出所有的易失性存储介质

() DRAM () SRAM () ROM () 软盘 () SSD () U 盘

【答】DRAM 与 SRAM

5. 已知一个双面磁盘有 2 个盘片、10000 个柱面, 每条磁道有 400 个扇区, 每个扇区容量为 512 字节, 则它的存储容量是 8.192 GB。

【答】 $2 \times 2 \times 10000 \times 400 \times 512 = 8,192,000,000 \text{ Byte} = 8.192\text{GB}$

6. 已知一个磁盘的平均寻道时间为 6ms, 旋转速度为 7500RPM, 那么它的平均访问时间大约为()ms

A. 6 B. 8 C. 10 D. 14

【答】 $6\text{ms} + 0.5 \times (60/7500 \times 1000)\text{ms} = 10\text{ms}$

7. 已知一个磁盘每条磁道平均有 400 个扇区, 旋转速度为 6000RPM, 那么它的平均访问时间大约为()ms

A. 0.020 B. 0.025 C. 0.040 D. 0.050

【答】 $60 / 6000$ (转一圈的时间) $\times 1 / 400$ (转过一个磁道的时间) $\times 1000 = 0.025\text{ms}$

8. 考虑如下程序

```
for (int i = 0; i < n; i++) {  
    B[i] = 0;  
    for (int j = 0; j < m; j++)  
        B[i] += A[i][j];  
}
```

判断下列说法的正确性

(1) () 对于数组 A 的访问体现了时间局部性。

- (2) () 对于数组 A 的访问体现了空间局部性。
- (3) () 对于数组 B 的访问体现了时间局部性。
- (4) () 对于数组 B 的访问体现了空间局部性。

【答】 错误， 正确， 正确， 正确

【高速缓存】

9. 一个容量为 8K 的直接映射高速缓存，每一行的容量为 32B，那么它有 256 组，每组有 1 行。

10. 一个容量为 8K 的全相联高速缓存，每一行的容量为 32B，那么它有 1 组，每组有 256 行。

11. 一个容量为 16K 的 4 路组相联告诉缓存，每一行的容量为 64B，那么一个 16 位地址 0xCAFE 应映射在第 43 组内。

【答】计算 $16K / (64 * 4) = 64$ 组，因此组号长度为 6，块内偏移长度为 6 字节，因此 0xCAFE 的组号为 43。

12. 考虑下列的程序

```
// A 有定义 int A[MAXN];  
for (int i = 0; i < 25; i++) {  
    int x = A[i];  
    int y = A[i+1];  
    int z = A[i+2];  
    A[i+3] = x + y + z;  
}
```

假设编译成汇编语言的时候没有任何优化，变量 x、y、z 均放在寄存器中，运行之前 cache 所有行都是无效的。A 的起始地址为 0。

(1) 假设 cache 的容量为 8 字节，每一行的容量为 4 字节，替换策略为 LRU，组策略为直接映射高速缓存。在这个 cache 上运行上述代码，得到的 cache 命中率是 24 %

(2) 假设 cache 的容量为 8 字节，每一行的容量为 4 字节，替换策略为 LRU，组策略为全相联高速缓存。在这个 cache 上运行上述代码，得到的 cache 命中率是 0 %

(3) 假设 cache 的容量为 32 字节，每一行的容量为 8 字节，替换策略为 LRU，组策略为 2 路组相联。画出程序运行结束时 cache 的情况(用 M[0-7] 表示第 0 到第 7 字节的地址)

	有效位	内容	有效位	内容
组 0	1	M[96 - 103]	1	M[80 - 87]
组 1	1	M[104 - 111]	1	M[88 - 95]

Cache 命中率是 86 %

(4) 假设 cache 的每一行的容量为 4 字节，运行该程序，得到的 cache 命中率的可能最大值为 72 %

13. 判断下列说法的正确性

- (1) () 保持块大小与路数不变，增大组数，命中率一定不会降低。
- (2) () 保持总容量与块大小不变，增大路数，命中率一定不会降低。

- (3) () 保持总容量与路数不变，增大块大小，命中率一定不会降低。
- (4) () 使用随机替换代替 LRU，期望命中率可能会提高。

【答】1 正确；2 错误；3 错误；4 正确。