**Performance Measurement**

姓名

Date: 2019-9-22

**Chapter 1: Induction：**

测量了3种不同的算法(普通算法，二分迭代算法，二分回归算法)计算X的N次方的速度。

**Chapter 2：Algorithm Specification：**

普通算法：

Begin(算法开始);

Input X and N;

result: = 1;

Repeat:

result: = result \* x; n = n – 1;

until n <= 0

Output result;

End;

二分迭代算法：

Begin;

Input X and N;

result: = 1;

If: N == 0

Output: result = 1;

Else:

Repeat:

If: (N & 1) == 1 (位操作)

result: = result \* x;

x: = x\*x;

n(二进制)右移1位;

Until: N: != 0;

Output result;

End;

二分递归算法：

Begin;

Input X and N

result: = 1;

If: N == 1

Output: X;

End;

Else

If:N%2 == 0

Result = Algorithm(X, N/2) \* Algorithm(X, N/2);

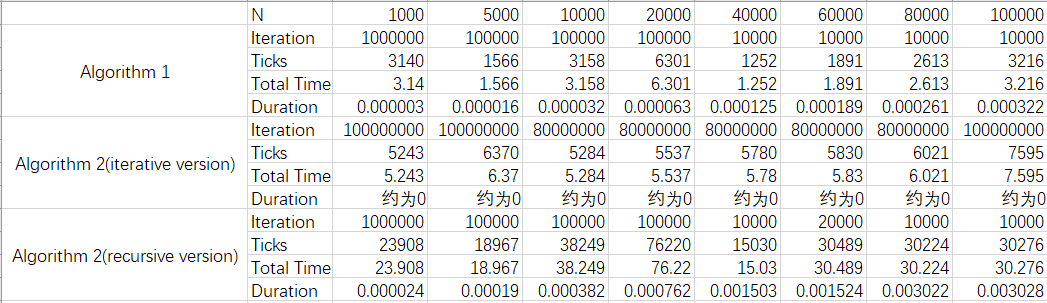
Else:

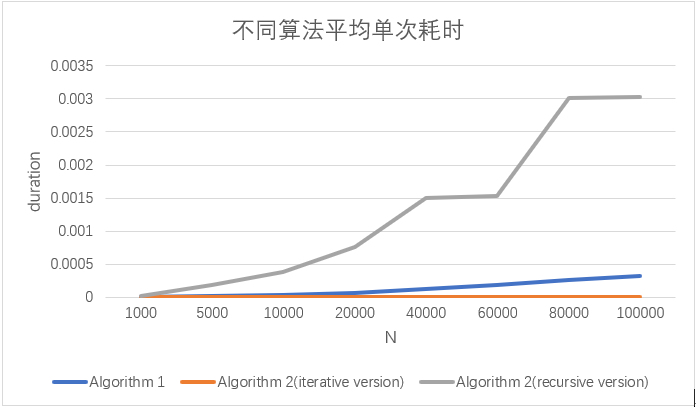
Result = Algorithm(X, (N-1)/2) \* Algorithm(X, (N-1)/2);

Output: result;

End;

**Chapter 3：Testing Results:**





由图表可以看出算法1运行时间大致随N线性增长；算法二(迭代)运行时间最短，几乎为0；算法二(递归)所需时间增长趋势逐渐变大

**Chapter 4: Analysis and Comments**

算法1

O(n) = n; T(n) = 2n +3;

算法2 (迭代)

O(n) = logn;

算法2(递归)

O(n) = n^2; T(n) = n^2 + 2;

改进：选择是否进行下一步运算是是可调用键盘回调函数，相较于输入减少了操作。

**Appenddix: Source Code(in C):**

#include <time.h>

#include <stdio.h>

#include <stdlib.h>

clock\_t start, stop; //clock\_t是处理器时间(计时)的内置类型

double duration; //记录函数的运行时间(秒)

double Algorithm\_1(double x, int n)//算法1

{

double result = 1;

for (int i = 0; i < n; i++)

{

result \*= x;

};

return result;

};

double Algorithm\_2\_r(double x, int n)//算法2(递归)

{

double result = 1;

if (n == 1)//递归跳出点

return x;

else//递归主体

{

if ((int)n % 2 == 0)

return Algorithm\_2\_r(x, n / 2) \* Algorithm\_2\_r(x, n / 2);

else

return Algorithm\_2\_r(x, (n - 1) / 2) \* Algorithm\_2\_r(x, (n - 1) / 2) \* x;

};

};

double Algorithm\_2\_i(double x, int n)//算法2(迭代)

{

double result = 1;

if (n == 0)

return result;

while (n != 0)

{

if ((n & 1) == 1)//n的二进制表达该位是否为1

result \*= x;

x \*= x;

n >>= 1;//位运算：将n的二进制表达右移1位

}

return result;

};

void run(int ch)

{

double x, n, num = 0, time, UnitTime, ticks;

printf("Input X(变量), N(指数), Time(运行次数)\n");

scanf("%lf %lf %lf", &x, &n, &time);

//选择运行3个算法并输出

if (ch == 1)

{

//run Algorithm 1

start = clock();//计时器1

for (int i = 0; i < time; i++)//运行Time次

{

num = Algorithm\_1(x, n);

};

stop = clock();

duration = ((double)(stop - start)) / CLK\_TCK;

UnitTime = duration / time;

printf("算法一答案为： %f\n", num);

printf(" 总运行时间： %lf\n", duration);

printf(" 平均单次运行时间： %lf\n\n", UnitTime);

}

else if (ch == 2)

{

//run Algorithm 2(迭代)

start = clock();//计时器1

for (int i = 0; i < time; i++)//运行Time次

{

num = Algorithm\_2\_i(x, n);

};

stop = clock();

duration = ((double)(stop - start)) / CLK\_TCK;

UnitTime = duration / time;

printf("算法二(迭代)答案为： %f\n", num);

printf(" 总运行时间： %lf\n", duration);

printf(" 平均单次运行时间： %lf\n\n", UnitTime);

}

else

{

//run Algorithm 2(递归)

start = clock();//计时器2

for (int i = 0; i < time; i++)//运行Time次

{

num = Algorithm\_2\_r(x, n);

};

stop = clock();

duration = ((double)(stop - start)) / CLK\_TCK;

UnitTime = duration / time;

printf("算法二(递归)答案为： %f\n", num);

printf(" 总运行时间： %lf\n", duration);

printf(" 平均单次运行时间： %lf\n\n", UnitTime);

};

printf("Ticks = %f\n", duration \* CLK\_TCK);

};

int main()

{

int next = 0;

while (next == 0)

{

int ch;

printf("------1: 算法一 2:算法二(迭代) 3:算法二(递归)------\n");

scanf("%d", &ch);

run(ch);

printf("0: Run Again 1: Exit\n");

scanf("%d", &next);

};

};

**Declaration：**

We hereby declare that all the work done in this project titled "XXX" is of our independent ef ort as a group。

**Duty Assignments:**

**Programmer:**

**Tester:**

**Report Writer:**