UNIVERSIDADE FEDERAL DE MINAS GERAIS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

NICOLAS VON DOLINGER MOREIRA ROCHA

ESTRUTURA DE DADOS

RELATÓRIO PRÁTICA 2

BELO HORIZONTE

2023

Introdução

Neste relatório, estão presentes os resultados de uma série de experimentos realizados para avaliar o desempenho de diferentes implementações dos cálculos do fatorial e da sequência de Fibonacci utilizando a linguagem C. Os objetivos principais são medir o tempo de execução, comparar com a ordem de complexidade teórica e analisar o impacto dos recursos do sistema em algoritmos recursivos.

Implementações

Foram desenvolvidas implementações para os cálculos de fatorial e Fibonacci com base em um parâmetro de linha de comando que determina qual tarefa executar.

- Implementação iterativa
- Implementação recursiva
- Implementação recursiva com carga computacional adicional

Plano de Experimentos

Faixa de Valores

Para avaliar o desempenho, realizamos experimentos em um intervalo de valores de 1 a 20. Nesse sentido, os valores escolhidos foram 7, 9, 10, 11 e 20, que são adequados para cada tarefa e não permitem que saia do limite permitido pelo tipo "**long long int**", durante o cálculo.

Medição do Tempo de Relógio

Para medir o tempo de relógio, foi utilizada a função "**clock()**" da biblioteca "**time.h**". Cada implementação foi executada dez vezes para calcular a média do tempo de execução. A complexidade dos algoritmos é comparada com os tempos de relógio medidos para verificar a correlação teórica.

Medição dos Tempos de Utilização de Recursos

Para medir os tempos de utilização de recursos (usuário e sistema), utilizamos a ferramenta time do sistema operacional. Isso nos permitiu comparar esses tempos com os tempos de relógio e entender o uso efetivo dos recursos.

Análise de Algoritmos Recursivos

Para as implementações recursivas, realizamos análises adicionais:

- Medimos o tempo de execução para cada chamada recursiva usando "**clock()**" e o "**gprof**" para comparar as diferenças.
- Foi introduzida uma função que consumia recursos computacionais (cálculo do seno repetidamente) nas chamadas recursivas e as medições foram repetidas para observar o impacto na performance.

Resultados

Valor de entrada: 7

```
  Valor sendo calculado: 7

- FATORIAL -

  Valor: 5040
  Tempo de relógio gasto para calcular recursivamente: 0 secs / 2934 nano-secs
  Tempo de usuário gasto para calcular recursivamente: 0 secs / 1 nano-secs
  Tempo de relógio gasto para calcular iterativamente: 0 secs / 1886 nano-secs
  Tempo de usuário gasto para calcular iterativamente: 0 secs / 1 nano-secs

- FIBONACCI -

  Valor: 13
  Tempo de relógio gasto para calcular recursivamente: 0 secs / 2375 nano-secs
  Tempo de usuário gasto para calcular recursivamente: 0 secs / 1 nano-secs
  Tempo de relógio gasto para calcular iterativamente: 0 secs / 2305 nano-secs
  Tempo de usuário gasto para calcular iterativamente: 0 secs / 1 nano-secs
```

Valor de entrada: 9

```
  Valor sendo calculado: 9

- FATORIAL -

  Valor: 362880
  Tempo de relógio gasto para calcular recursivamente: 0 secs / 4330 nano-secs
  Tempo de usuário gasto para calcular recursivamente: 0 secs / 1 nano-secs
  Tempo de relógio gasto para calcular iterativamente: 0 secs / 3841 nano-secs
  Tempo de usuário gasto para calcular iterativamente: 0 secs / 1 nano-secs

- FIBONACCI -

  Valor: 34
  Tempo de relógio gasto para calcular recursivamente: 0 secs / 3912 nano-secs
  Tempo de usuário gasto para calcular recursivamente: 0 secs / 2 nano-secs
  Tempo de relógio gasto para calcular iterativamente: 0 secs / 4260 nano-secs
  Tempo de usuário gasto para calcular iterativamente: 0 secs / 1 nano-secs
```

Valor de entrada: 10

```
  Valor sendo calculado: 10

- FATORIAL -

  Valor: 3628800
  Tempo de relógio gasto para calcular recursivamente: 0 secs / 5308 nano-secs
  Tempo de usuário gasto para calcular recursivamente: 0 secs / 0 nano-secs
  Tempo de relógio gasto para calcular iterativamente: 0 secs / 3911 nano-secs
  Tempo de usuário gasto para calcular iterativamente: 0 secs / 0 nano-secs

- FIBONACCI -

  Valor: 55
  Tempo de relógio gasto para calcular recursivamente: 0 secs / 4400 nano-secs
  Tempo de usuário gasto para calcular recursivamente: 0 secs / 0 nano-secs
  Tempo de relógio gasto para calcular iterativamente: 0 secs / 2863 nano-secs
  Tempo de usuário gasto para calcular iterativamente: 0 secs / 0 nano-secs
```

Valor de entrada: 11

```
  Valor sendo calculado: 11

- FATORIAL -

  Valor: 39916800
  Tempo de relógio gasto para calcular recursivamente: 0 secs / 3772 nano-secs
  Tempo de usuário gasto para calcular recursivamente: 0 secs / 0 nano-secs
  Tempo de relógio gasto para calcular iterativamente: 0 secs / 2864 nano-secs
  Tempo de usuário gasto para calcular iterativamente: 0 secs / 0 nano-secs

- FIBONACCI -

  Valor: 89
  Tempo de relógio gasto para calcular recursivamente: 0 secs / 5587 nano-secs
  Tempo de usuário gasto para calcular recursivamente: 0 secs / 0 nano-secs
  Tempo de relógio gasto para calcular iterativamente: 0 secs / 3352 nano-secs
  Tempo de usuário gasto para calcular iterativamente: 0 secs / 0 nano-secs
```

Valor de Entrada: 20

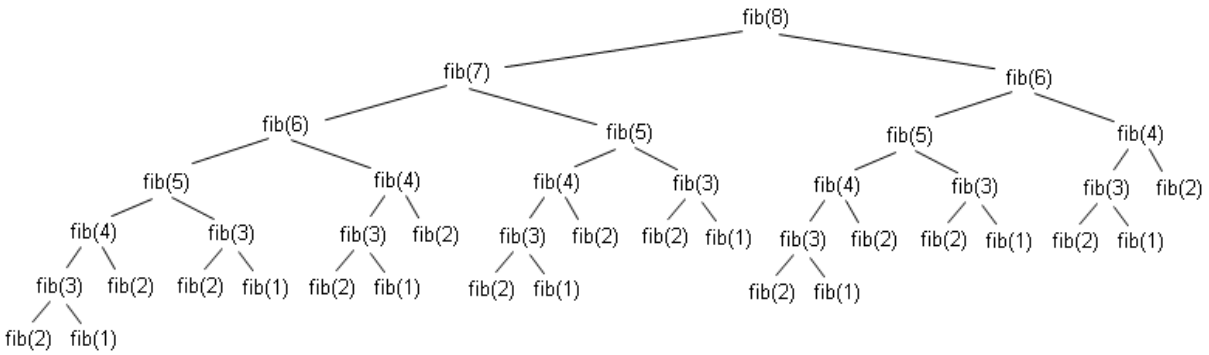
```
  Valor sendo calculado: 20

- FATORIAL -

  Valor: 2432902008176640000
  Tempo de relógio gasto para calcular recursivamente: 0 secs / 4190 nano-secs
  Tempo de usuário gasto para calcular recursivamente: 0 secs / 0 nano-secs
  Tempo de relógio gasto para calcular iterativamente: 0 secs / 2863 nano-secs
  Tempo de usuário gasto para calcular iterativamente: 0 secs / 0 nano-secs

- FIBONACCI -

  Valor: 6765
  Tempo de relógio gasto para calcular recursivamente: 0 secs / 215741 nano-secs
  Tempo de usuário gasto para calcular recursivamente: 0 secs / 0 nano-secs
  Tempo de relógio gasto para calcular iterativamente: 0 secs / 2933 nano-secs
  Tempo de usuário gasto para calcular iterativamente: 0 secs / 0 nano-secs
```

Discussão dos Resultados

Com os resultados obtidos, é possível perceber que o tempo levado para calcular tanto o fatorial quanto o Fibonacci aumentam consideravelmente com o parâmetro que é passado. Nesse sentido, a análise assintótica sugere que ambas as implementações têm uma complexidade de tempo de O(n).

Também é possível notar que o tempo utilizado para calcular os valores iterativamente é menor, devido a maneira como a recursividade funciona. Pois, para calcular, por exemplo, o valor do n-esimo número da sequência de Fibonacci, de maneira recursiva, é necessário calcular os valores dos n-1 e n-2 termos, e assim

por diante. Abaixo, segue o exemplo dessa operação, que mostra a sequência para calcular o oitavo termo:



Além disso, foi utilizada a ferramenta **gprof** para medir o número de chamadas de cada função e o tempo que cada uma levava para ser executada.

```
Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  ms/call  ms/call  name
 66.67      0.04     0.04                             _init
 33.33      0.06     0.02        2    10.00    10.00  calcSin
  0.00      0.06     0.00        6     0.00     0.00  resultOp
  0.00      0.06     0.00        2     0.00     0.00  calcularFatorialRecursivo
  0.00      0.06     0.00        2     0.00     0.00  calcularFibonacciRecursivo
  0.00      0.06     0.00        2     0.00    10.00  result
  0.00      0.06     0.00        1     0.00     0.00  calcularFatorialIterativo
  0.00      0.06     0.00        1     0.00     0.00  calcularFibonacciIterativo
```

Entretanto, os algoritmos de Fibonacci e de cálculo de fatorial apresentavam um tempo inexpressivo, utilizando essa ferramenta. Nesse sentido, foi desenvolvida a função “**calcSin()**”, uma função iterativa que calcula, em cada uma de suas chamadas, o seno de um número um milhão de vezes. Para efeito de comparação, ela foi introduzida antes das chamadas recursivas das funções de cálculo de fatorial e de Fibonacci.

```
granularity: each sample hit covers 4 byte(s) for 16.67% of 0.06 seconds

index % time    self  children    called     name
                                                 <spontaneous>
[1]     66.7    0.04    0.00                 _init [1]
-----------------------------------------------
                0.02    0.00       2/2           result [3]
[2]     33.3    0.02    0.00       2         calcSin [2]
-----------------------------------------------
                0.00    0.02       2/2           main [4]
[3]     33.3    0.00    0.02       2         result [3]
                0.02    0.00       2/2           calcSin [2]
                0.00    0.00       4/6           resultOp [5]
-----------------------------------------------
                                                 <spontaneous>
[4]     33.3    0.00    0.02                 main [4]
                0.00    0.02       2/2           result [3]
                0.00    0.00       2/6           resultOp [5]
-----------------------------------------------
                0.00    0.00       2/6           main [4]
                0.00    0.00       4/6           result [3]
[5]      0.0    0.00    0.00       6         resultOp [5]
                0.00    0.00       2/2           calcularFibonacciRecursivo [7]
                0.00    0.00       2/2           calcularFatorialRecursivo [6]
                0.00    0.00       1/1           calcularFibonacciIterativo [9]
                0.00    0.00       1/1           calcularFatorialIterativo [8]
-----------------------------------------------
                                  38             calcularFatorialRecursivo [6]
                0.00    0.00       2/2           resultOp [5]
[6]      0.0    0.00    0.00       2+38      calcularFatorialRecursivo [6]
                                  38             calcularFatorialRecursivo [6]
-----------------------------------------------
                                27056             calcularFibonacciRecursivo [7]
                0.00    0.00       2/2           resultOp [5]
[7]      0.0    0.00    0.00       2+27056   calcularFibonacciRecursivo [7]
                                27056             calcularFibonacciRecursivo [7]
-----------------------------------------------
                0.00    0.00       1/1           resultOp [5]
[8]      0.0    0.00    0.00       1         calcularFatorialIterativo [8]
-----------------------------------------------
                0.00    0.00       1/1           resultOp [5]
[9]      0.0    0.00    0.00       1         calcularFibonacciIterativo [9]
-----------------------------------------------
```

Como é possível notar, a função “**calcSin()**“ apresentou o maior custo para o processamento do código(33.3%), devido a quantidade de operações.

Também é possível comparar o número de chamadas que cada uma das funções de cálculo de fatorial e Fibonacci tiveram em suas versões recursivas. Inclusive, com a função “**calcSin()**“ sendo chamada antes de cada uma delas, o que ocasionou no consumo elevado de recursos computacionais.

Nesse sentido, podemos concluir que, com o aumento do valor dos parâmetros passados, as funções recursivas, no geral, requerem mais processamento e tempo de execução. Entretanto, o tempo de execução das funções iterativas não foi alterado.

Conclusão

Com base nos resultados dos experimentos, podemos concluir que as implementações para o cálculo do fatorial e da sequência de Fibonacci, tanto iterativa quanto recursiva, são eficientes para valores dentro da faixa testada. As implementações mostraram tempos de execução na ordem de nanossegundos, mesmo para valores relativamente grandes de n. Isso está de acordo com a análise assintótica, que sugere uma complexidade de tempo próxima de O(n) para ambas as tarefas.

No entanto, é importante ressaltar que esses resultados são específicos para a faixa de valores testados. Para valores muito maiores de n, pode ser necessário considerar otimizações adicionais, a fim de manter um desempenho aceitável.

Em resumo, para valores de n dentro da faixa testada, as implementações apresentam um desempenho satisfatório e estão alinhadas com a análise assintótica esperada.

Fontes

https://sites.google.com/a/liesenberg.biz/cjogos/_/rsrc/1288878942364/home/materiais-de-apoio/topicos-relativos-a-c/recursao/serie-de-fibonacci/fibonacciRecursivo.gif