Edson Yudi Toma → 9791305

Pseudocódigo – Binário para Decimal Iterativo

Passa-se um binário a ser convertido:

Laço que para quando o 'binário' for menor do que 1 {
Uma variável auxiliar acumula a soma do MOD do binário a ser convertido multiplicado por uma potência de 2 (assim, caso 0, soma-se 0 e caso 1, soma-se a potência de 2);
(Necessita-se do uso de outra variável auxiliar para realizar as potências de 2 por meio de shifts (menos custosos que as multiplicações), incrementando-a no final de cada laço.)
(Também é necessário dividir o 'binário' por 10 e redefinir seu valor como o resultado dessa operação, assim move-se para a análise do próximo bit do binário.)

}
Por fim, retorna-se a variável que acumulou as somas;

Código – Binário para Decimal Iterativo

```
int bintodec_iterative (int bin) {
    int dec = 0;

    for (int counter = 0; bin >= 1; counter++, bin/=10)
        dec += (bin%10) * (1<<counter);

    return dec;
}
```

Análise de complexidade – Binário para Decimal Iterativo

Seguindo o código acima temos:

Declaração e atribuição de uma variável (1 vez);
Um laço onde é declarada e atribuída uma variável (1 vez), além da operação de checagem (n+1 vezes), incremento (n vezes) e divisão/atribuição (n vezes);
Dentro do laço há uma soma/atribuição, uma operação MOD, uma multiplicação e uma operação de shift. (todas acontecem n vezes);

Importante notar que n seria a quantidade de bits do número binário em questão.

Assim, somando-se as operações, obtém-se 3 + 3*n + 4*n, portanto 7*n + 3 operações, podendo-se dizer que o algoritmo é O(n).

Pseudocódigo – Binário para Decimal Recursivo

Passa-se um binário a ser convertido e um auxiliar:

A recursão termina quando o 'binário' for menor do que 1;

Enquanto a recursão não terminar deve-se realizar uma operação de MOD com o binário multiplicando o valor de uma potência de 2 (assim como no caso iterativo, quando o bit do 'binário' for 0, soma-se 0, e quando for 1, soma-se a potência de 2). No caso da recursão, deve-se retornar na função esse valor somado com a chamada da própria função passando-se como argumentos o próprio 'binário' (mas ele estará dividido por 10 para que o próximo bit seja analisado) e um contador auxiliar incrementado (para realizar as potências de 2, ele deve ser incrementado para que a potência de 2 também seja);

(Do mesmo modo que na função iterativa o contador auxiliar é utilizado para a realização das potências de 2 por meio de 'shifts', por esses serem menos custosos que as multiplicações.)

Código – Binário para Decimal Recursivo

```
int bintodec_recursive (int bin, int counter) {

    return (bin < 1) ? 0 : (bin%10) * (1<<counter) +
                    bintodec_recursive (bin/10, ++counter);
}
```

Análise de complexidade – Binário para Decimal Recursivo

Seguindo o código acima temos:

Uma checagem de término da recursão;

Uma operação MOD, uma multiplicação, uma operação de shift, uma soma, uma chamada de função com divisão e incremento;

Importante notar que n seria a quantidade de bits do número binário e que a recursão em questão se comporta como um laço, portanto todas as operações realizadas vão ocorrer n vezes (com exceção da checagem, que ocorrerá n+1 vezes).

Assim, somando-se as operações e multiplicando-as por n (não se esquecendo de somar 1 devido à última checagem), obtém-se 8*n + 1 operações, podendo-se dizer que o algoritmo é O(n);