Algoritmos e estruturas de Dados A07

Recursividade

Prof. Mauri Ferrandin

Recursividade

Frase sobre recursividade:

- Para entender recursividade é preciso primeiro entender a recursividade;

Recursividade

- Definição
  - É um princípio poderoso que permite que um problema seja definido em termos de instâncias menores e menores do próprio problema.
- Computacão
  - Na computação resolvemos problemas recursivos usando funções recursivas que são funções que invocam (chamam) a si próprias.

Recursividade

- Exemplo: cálculo do fatorial.
  - O fatorial de n, definido por n!, é o produto de todos os números de n até 1 (decrementando de 1 em 1). Por exemplo, 4! = 4 * 3 * 2 * 1.
  - Pensando em uma solução iterativa para o problema podemos denotá-lo como:

    n! = (n) * (n - 1) * (n - 2) ... (1)
  - Solução iterativa em C

```
int fatorial(int n){
  int i, result = 1;
  for (i = n; i > 1; i--){
    result = result * i;
  }
  return result;
}
```

Recursividade

- Exemplo: cálculo do fatorial.
  - Outra forma de pensarmos o problema do fatorial é definirmos:

    $n! = n * (n - 1)!$

  - Desta forma, a solução para (n -1)! é a mesma usada para n!, apenas um pouco menor. Generalizando, se assumirmos que: (n -1)! = (n – 1) * (n – 2)! e assim por diante até n = 1, encontraremos uma solução decompondo o problema em partes menores. Esta é uma solução recursiva para o problema, denotada por:

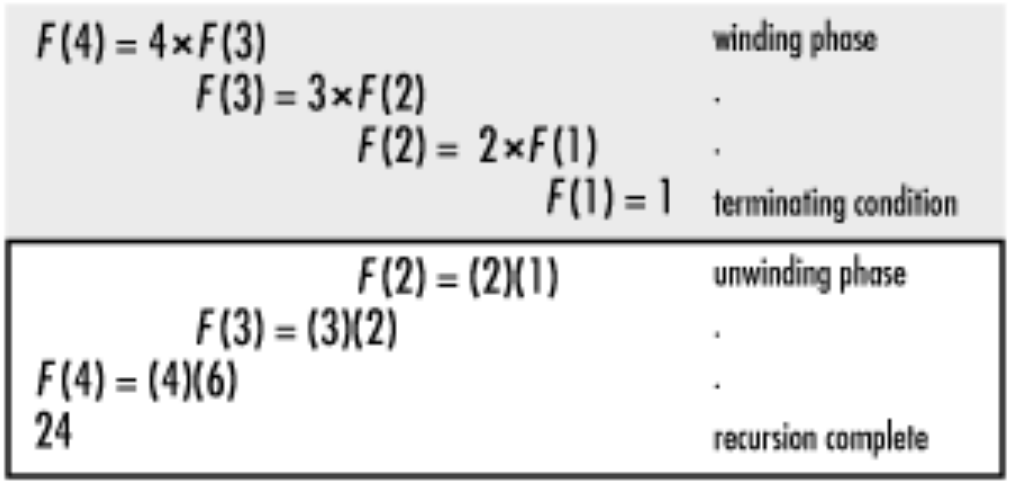
$$F(n) = \begin{cases} 1 & \text{if } n = 0,\ n = 1 \\ nF(n-1) & \text{if } n > 1 \end{cases}$$

- Solução recursiva em C

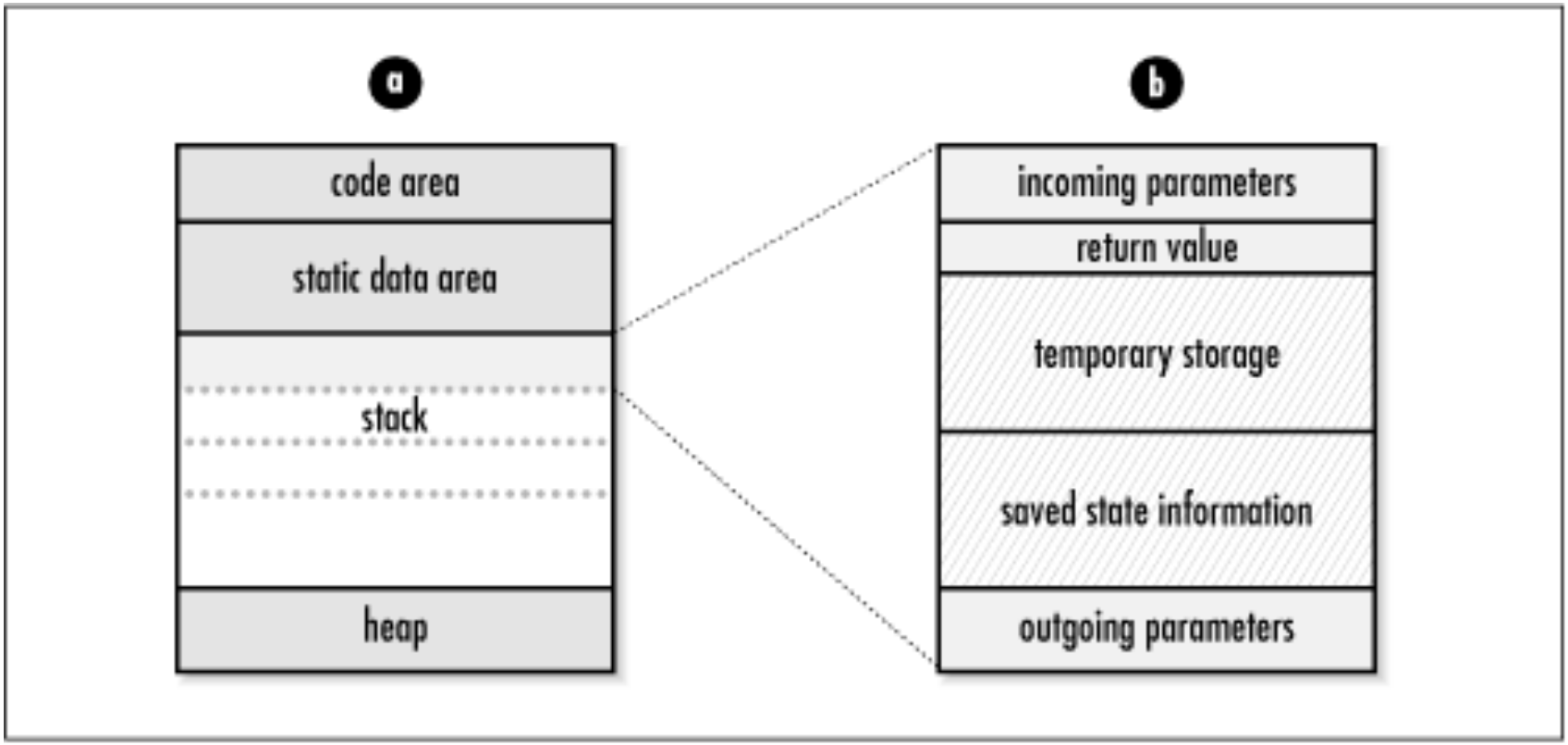
```
int fatorial(int n){
   if (n < 0)
         return 0;
   if (n == 1)
         return 1;
   return n * fatorial(n -1);
}
```

Recursividade: fatorial(4) = ?

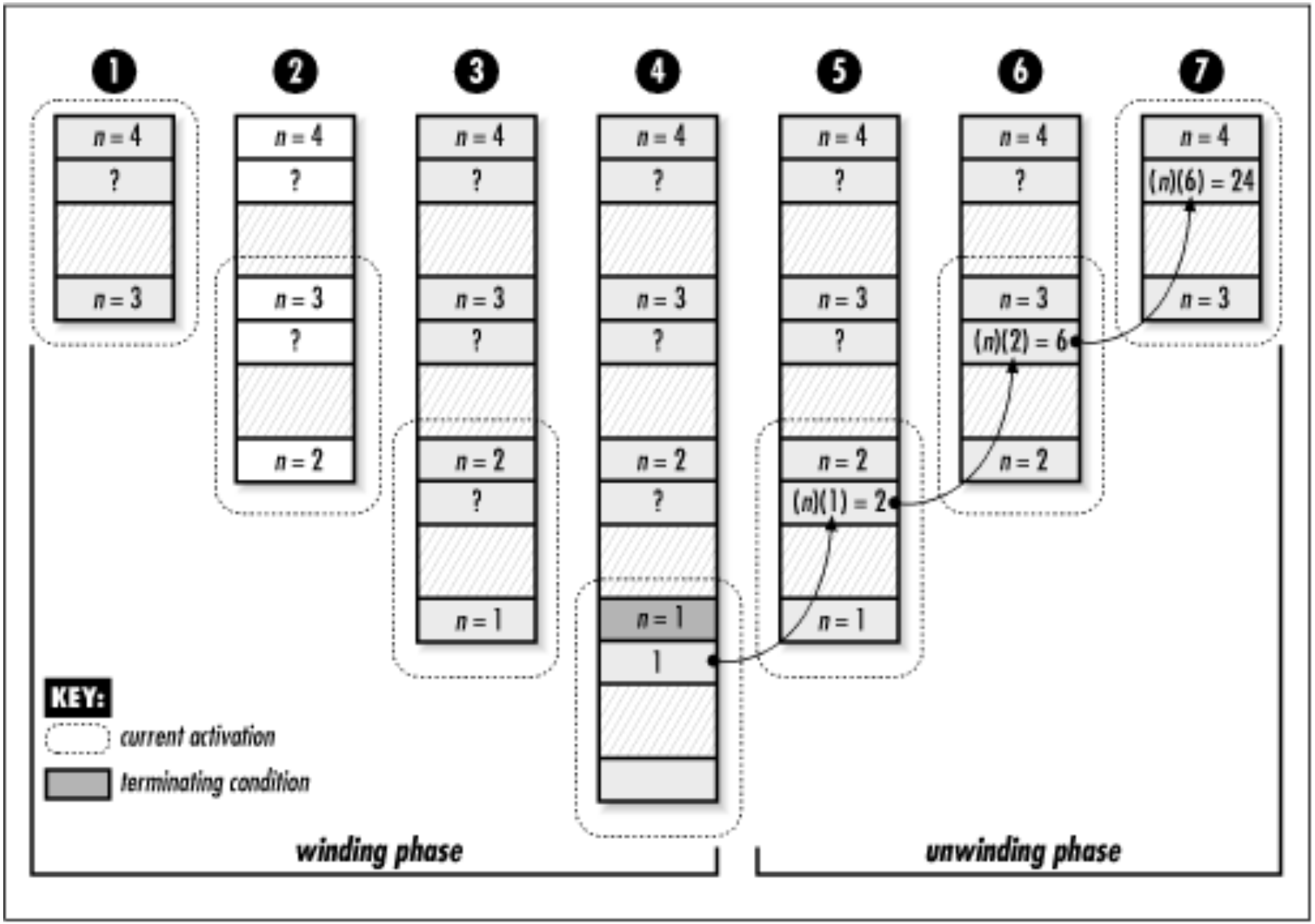
```
int fatorial(int n){
      if (n < 0)
            return 0;
      if (n == 0 || n == 1)
            return 1;
      return n * fatorial(n -1);

}
```



Estrutura de uma chamada de função (memória)



Memória: fatorial(4)



Estudo avançado (opcional)

Recursividade de cauda (*tail recursion*).

Referências

Figuras retiradas do livro:
Loudon, Kyle. Mastering Algorithms with C. O'Reilly Media, Inc. 2009.