

Data Structures and Algorithms

Exercise 3

Markus Innerebner, Romans Kasperovics
dsa(a)inf.unibz.it

Free University of Bozen - Bolzano, Italy

Thursday, March 12, 2009

Naming Conventions

- For each assignment create a folder *nn_yourlogin*
 - *nn* – is a number of the assignment (01, 02, ..., 10)
 - *yourlogin* – is your login to the university network
- Name your files as *task#.c* or *task#.pdf*, where '#' is the number of the corresponding task in the assignment, for example:

```
/ 01_minnerebner
   task12.pdf
   task3.c
   task4.c
```

- Put your name and student ID in all files
- Send the solution folder via e-mail archived as ***.zip** (standard windows) or ***.tar.gz** (standard linux) file, for example:

```
01_minnerebner.tar.gz
```

Assignment 03

Task 1

Implement in C the tromino tiling algorithm from the lecture notes. Model the $2^n \times 2^n$ board with a two dimensional array of integers. Model the empty cells with '0' and the initial hole with '1'. Fill it in with numbers >1 so that same numbers belong to the same tile. Print the array after filling.

Array before tiling

0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Array after tiling

4	4	6	6	14	14	16	16
4	3	6	1	14	13	13	16
5	3	3	7	15	15	13	17
5	5	7	7	2	15	17	17
9	9	11	2	2	19	21	21
9	8	11	11	19	19	18	21
10	8	8	12	20	18	18	22
10	10	12	12	20	20	22	22

For the tiling algorithm from Task 1

- Show the exact running time $T(n)$ as a recurrence
- Guess a tight asymptotic upper bound by expanding the recurrence by substitution and noticing the pattern
- Verify the solution by the mathematical induction

Consider the following recurrence:

$$T(n) = \begin{cases} 1 & , \text{ if } n = 1 \\ T(n/2) + T(n/4) + T(n/8) + n & , \text{ if } n > 1 \end{cases}$$

- Draw a recursion tree to guess the tight upper bound.
- Use the substitution method to prove that your guess is correct.

- $\log_a(b \cdot c) = \log_a b + \log_a c$ – logarithm of a multiplication
- $\log_a \frac{b}{c} = \log_a b - \log_a c$ – logarithm of a fraction
- $\log_a b^c = c \cdot \log_a b$ – logarithm of a power function
- $\log_a b = \frac{\log_c b}{\log_c a}$ – base change
- $(\log_a b)' = \frac{1}{b \cdot \ln a}$ – derivative of a logarithm

- $1 + 2 + \dots + n = \frac{n(n+1)}{2}$ (arithmetic series)
- $1 + r + r^2 + \dots + r^n = \frac{1 - r^{n+1}}{1 - r}$ (geometric series)
- for $-1 < r < 1$, $\sum_{k=0}^{\infty} r^k = \frac{1}{1 - r}$
- $1^2 + 2^2 + 3^2 + \dots + n^2 = \frac{n(n+1)(2n+1)}{6}$ (sum of squares)