Institut for Matematik og Datalogi
Syddansk Universitet

November 18, 2009

Peter Schneider-Kamp

Introduction to Computer Science
E09 – Week 10

Lecture: Monday, November 16

Jœrgen Bang-Jensen started to lecture on the theory of computation based on Chapter 12.1 to 12.5.

Lecture: Wednesday, November 18

Jœrgen Bang-Jensen finished lecturing on computability and continued with NP-complete problems and combinatorial optimization. The later is based on the notes "Algorithmisk Kombinatorik - et datalogisk emne i matematik" by Bjarne Toft (IMADA). These notes are available from Blackboard.

Lecture: Monday, November 23, 12-14 (U140)

Jœrgen Bang-Jensen will continue to lecture on combinatorial optimization based on the notes of Bjarne Toft.

Lecture: Monday, November 30, 12-14

Marco Chiarandini will start to lecture on artificial intelligence based on Chapter 11.

Lecture, Wednesday, December 2, 14-16

Marco Chiarandini will continue lecture on artificial intelligence based on Chapter 11.

Discussion section: November 24, 10:15-12 (U37)

- Course book, Pages 610-611, Problems 38 and 41.
- Exercises in Chapter 1 of notes by Bjarne Toft: 1.4, 1.5, 1.6, 1.10, 1.13, and 1.15.
- Exercises in Chapter 1 of notes by Bjarne Toft: 1.25 (only the version which I called Kruskal's algorithm in the lecture, that is, keep adding the cheapest edge that can be added without introducing a cycle in the set of chosen edges) and 1.26.
- Exercises in Chapter 2 of notes by Bjarne Toft: 2.2, 2.4 ("valens af punkt v" means the number of edges with one end in v) and 2.9.

Lab: November 27, 10:15-12 (terminal room above U49)

- **Experiments with (possibly) halting computations:**
  Consider the following computation of a sequence of integers: $f_0 := n$ (the input value) and for $t \geq 1$ set $f_t = f_{t-1}/2$ if f_{t-1} is even, set $f_t = 3 * f_{t-1} + 1$ if f_{t-1} is odd and greater than 1, and finally stop if $f_t = 1$ holds. Denote this value of t by $t(n)$. We may also view the process above as defining the function $t(n)$.
  - Write a program in Java, Maple, Python or your favorite programming language which given the input value n prints the values $f(t)$ until it becomes 1 and outputs $t(n)$.
  - Experiment with different input values for n to see how large $t(n)$ can become.
  - Consider how you could make a table with values $n, t(n)$, say for $n = 1$ to $n = 1000$, with less work than simply looping through all values of n and calculating $t(n)$ for each of these without using any knowledge obtained so far (e.g. what is $t(n)$ if n is even?).
- **Graph algorithms in Maple:**
  - Start maple in worksheet mode.
  - Consult the manual for the GraphTheory package (find this via the Help menu under 'manuals').

- To build a graph with edge weights do the following: Load the GraphTheory and RandomGraphs packages by typing 'with(GraphTheory):' and then 'with(RandomGraphs):'. Here and below it is understood that you always type return after each Maple command.
- Build a (in this case complete) graph on 5 vertices and 10 edges by typing 'G:=Graph(weighted, $\{\{1,2\},\{1,3\},\{1,4\},\{1,5\},\{2,3\},\{2,4\},\{2,5\},\{3,4\},\{3,5\},\{4,5\}\}$)'.
- Assign random integer weights between 1 and 10 to the edges of G by typing 'AssignEdgeWeights(G, 1 .. 10)'.
- Draw the graph G by typing 'DrawGraph(G)'.
- Find a minimum spanning tree T in G and draw it by typing 'T := MinimalSpanningTree(G)' followed by 'DrawGraph(T)'.
- Find an optimal Traveling salesman tour in G by typing 'TravelingSalesman(G)'.
- Compare the cost of the optimal Traveling salesman tour with the cost of the optimal spanning tree.
- Consult the manual page for 'TravelingSalesman' and try to draw an optimal traveliong salesman tour.
- Repeat the experiment above with other graphs that you type in yourselves.

- Build a bipartite graph B on 6 vertices (3 in each part) using the method above.
- Consult the Maple package 'BipartiteMatching'.
- Find a maximum matching in B using the procedure 'BipartiteMatching'.
- Repeat this with a couple of other examples or try out other graph packages in Maple.