Advanced practical **Programming** for Scientists

Thorsten Koch

Zuse Institute Berlin

TU Berlin

7. July 2017

Exercise 6

Using your self written shortest path algorithm, implement the Steiner Tree heuristic by

H. Takahashi, A. Matsuyama

An Approximate Solution for the Steiner Problem in Graphs

Math. Jap. 24 (1980), 573-577

As examples use the graphs from the previous exercise.

Terminals are all nodes which number is prime.

Node Numbers start at 1.

Heaps and Priority Queues

Jon Bentley, Programming Pearls, Addision Wesley, 1989

Kevin Wayne, Princeton, 2002

http://www.cs.princeton.edu/~wayne/cs423/lectures/heaps-4up.pdf

Shape property: Full binary tree (No holes in the tree), all terminal nodes on maximal two levels, with those on the bottom as far left as possible.

Heap property: For all $2 \leq i \leq N$: $x[i \text{ div } 2] \leq x[i]$

Move hole, Sentinels, move key, decrease min or not/store location

Parallel computing

Shared memory

Distributed memory

Processes/Threads

Communication/Synchronization/Deadlocks

Hyperthreading

Instruction level parallelization

Vectorization

Parallel Processing

Python:

https://wiki.python.org/moin/ParallelProcessing

http://sebastianraschka.com/Articles/2014_multiprocessing.html

http://scipy-cookbook.readthedocs.io/items/ParallelProgramming.html

Java:

http://winterbe.com/posts/2015/04/07/java8-concurrency-tutorial-thread-executor-examples/

http://www.vogella.com/tutorials/JavaConcurrency/article.html

C/C++/Fortran

http://www.openmp.org/

http://bisqwit.iki.fi/story/howto/openmp/

https://computing.llnl.gov/tutorials/openMP/

Auto vectorization and parallelization

Sample Code Fragment

```
for (I=0;i<=MAX;i++)
    c[i]=a[i]+b[i];
```

Where does the vectorization speedup come from?
Consider the following sample code fragment, where a, b and c are integer arrays:
If vectorization is not enabled (that is, you compile using O1 or [Q]vec- options), for each iteration, the compiler processes the code such that there is a lot of unused space in the SIMD registers, even though each of the registers could hold three additional integers.
If vectorization is enabled (compiled using O2 or higher options), the compiler may use the additional registers to perform four additions in a single instruction.
The compiler looks for vectorization opportunities whenever you compile at default optimization (O2) or higher.

Auto vectorization and parallelization

https://software.intel.com/en-us/articles/a-guide-to-auto-vectorization-with-intel-c-compilers

https://gcc.gnu.org/wiki/AutoParInGCC

https://msdn.microsoft.com/de-de/library/hh872235.aspx

https://software.intel.com/en-us/articles/automatic-parallelization-with-intel-compilers

Final Exercise 6b (#9 on github)

Implement the **improved version** of the Steiner Tree heuristic.

Make it run in parallel for given number of starting nodes.

Ex6 filename.gph start_node

Print the edges (as node pairs) of the shortest Steiner tree you find.

Also print the length of the tree.

(Your code should check that what you return is actually a tree and a feasible solution.)