**Introduction**

Algorithms that process huge data sets need to access external memory (EM) that has access cost 1 000 000 times higher than the cost of a main memory access. In the EM algorithms the main performance measure is the number of performed input and output (I/O) operations. There exist EM software libraries (e.g. LEDA-SM,TPIE) which implement some I/O efficient algorithms. They are good proofs of concept but have drawbacks that make them of limited practical use. Currently we are developing the new EM library **STXXL** which aims for high performance and ease of use. Our objective is to be able to process very large data sets fast.

**STXXL**

The core of STXXL is an implementation of the C++ standard template library STL. Our library can process huge volumes of data that only fit on hard disks. Here is a selection of high performance features of STXXL:

- transparent support of parallel disks
- overlapping of I/O and computation
- prevention of OS file buffering overhead

**Parallel Disk Sorting**

The core routine of almost any EM algorithm is I/O efficient sorting. We have developed a new parallel disk sorting algorithm with:

- I/O cost close to the lower bound
- guaranteed overlapping between I/O and computation
- very efficient implementation, 2–10 times faster than previous libraries

**Pipelining**

A new interface that allows an implementation to save many I/Os:

- similar to the technology used in data bases
- record by record ⇒ lazy evaluation
- unique feature of STXXL

**Projects based on STXXL**

- A fast MST algorithm, processing very large graphs with billions of nodes (D. Schultes)
- Spanning trees and connected components (D. Schultes)
- Suffix array construction (J. Mehnert)
- External breadth first search (D. Ajwani)
- Parallel disk search trees (T. Nowak)