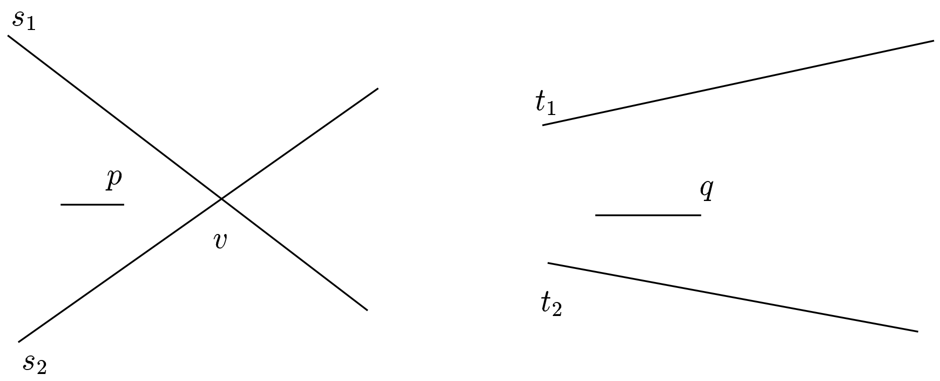
Effective Computational Geometry
The Theory and Practice of Implementing Geometric Algorithms
Exercise 1

Lutz Kettner Kurt Mehlhorn Elmar Schömer

October 23, 2001

Exercise 1: Read Section 10.7.3 of the LEDAbook. You are expected to have read the section up to the paragraph called "An Optimization" by Thursday. I will discuss the section with you in the classes on Thursday and Tuesday. By Tuesday you should be familiar with the entire section.

Exercise 2: The LEDA implementation for line segment intersection uses a dictionary *inter_dic* in order to avoid the recomputation of intersections between segments. The dictionary avoids the recomputation of the intersection of s_1 and s_2 when the right endpoint p is swept. It does not avoid that t_1 and t_2 are checked for intersection when q is swept.



1. What is known when there is no entry in *inter_dic* for a pair (s_1, s_2) of segments?

2. Change the semantics of *inter_dic* in the following way. If there is no entry for a pair (s_1, s_2) then the pair has never been checked for intersection. If there is an entry, the entry indicates the number of intersections between the segment that lie ahead of the sweep line. If the number of intersections is one, the entry also points to the corresponding entry of the X-structure. Which changes are required in the algorithm?

3. (optional; this is a project which will occupy you two to three weeks) Make the changes. What effect has the change on the running time on the generators provided

by LEDA? Try to find examples, where the effect is big. What do you think why KM and SN did not use the modified semantics of *inter_dic*?

This exercise will be discussed in the second week.

Exercise 3: The purpose of this exercise is to give you a practical introduction to LEDA. We will discuss this exercise with you this week and you are supposed to run the programs next week.

⟨*⟩≡

```
#include <LEDA/d_array.h>
#include <LEDA/string.h>
int main(){
d_array<string,int> D(0);
string s;
while ( cin >> s ) D[s]++;
forall_defined(s,D) cout << "\n" << s << " " << D[s];
return 0;
}
```

1. Use the LEDAbook to find out what this program does? You can find all chapters of the LEDAbook in `www.mpi-sb.mpg.de/~mehlhorn/ftp/LEDAbook`.
   The chapters "Introduction" and "Advanced Datatypes: Sparse Arrays" discuss the example avove.
2. Compile the program and run it on the input `Mehl Mehl Kurt control-D`.

   The following command compiles the program on KM's machine. The directory `/LEDA/INSTALL/solaris/g++/incl` contains the LEDA-include files and the directory `/LEDA/INSTALL/solaris/g++` contains the libraries.

   ```
   g++ -O -DLEDA_CHECKING_OFF -I/LEDA/INSTALL/solaris/g++/incl
   -L/LEDA/INSTALL/solaris/g++  -R/LEDA/INSTALL/solaris/g++  -c  dic_array.c
   g++ -O -DLEDA_CHECKING_OFF -I/LEDA/INSTALL/solaris/g++/incl
   -L/LEDA/INSTALL/solaris/g++  -R/LEDA/INSTALL/solaris/g++ -o  dic_array
   dic_array.o -lGeoW -lW -lD3 -lP -lG -lL -lX11 -lm -lsocket -lthread
   ```

3. Download the file `Exercise1.lw` from the webpage of this course and inspect it.
4. Run `lw2dvi Exercise1` and then `xdvi Exercise1`. The command `lw2dvi` is a command of the LEDA system. In order to use it you must add the LEDA-command directory `/LEDA/SRC/Manual/lw2dvi` to your PATH.
5. Have a look at the chapter Documentation of the LEDAbook in order to find out what has happened.

6. Run `notangle -L Exercise1.lw > dic_array.c` On my machine the command `notangle` is in directory `/KM/local/web/noweb/solaris_bin/`.

7. Same as item 5.