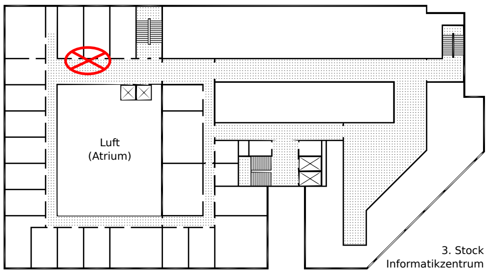
Algorithms Group
Departement of Computer Science - IBR
TU Braunschweig

Michael Hemmer

Summer '14

Geometric Algorithms
Exercise 1
April 29, 2014

This sheet is comprised of several exercise. Most of them are theoretical but some are also applied, i.e., coding tasks. The solutions should be handed in by **May 6.**. This can be done either at the beginning of the lecture or by placing them in the appropriate box of the exercise locker, see floor plan on the right. Applied exercises should be handed in via email to `hemmer@ibr.cs.tu-bs.de`. Please mark those exercises that you would like to present during the tutorial.



In order to achieve the *"Studienleistung"*, you must have presented at least three exercises until the end of the term. Moreover, you must pass the midterm exam at the beginning of June.

Exercise T1 (Smallest Enclosing Disk):

The expected run time of the randomized incremental algorithm presented in the lecture is $O(n)$.

a) What is the worst case complexity of this algorithm ?
b) Design a set of points and insertion order that realizes this worst case complexity.

Exercise T2 (Expected Value):

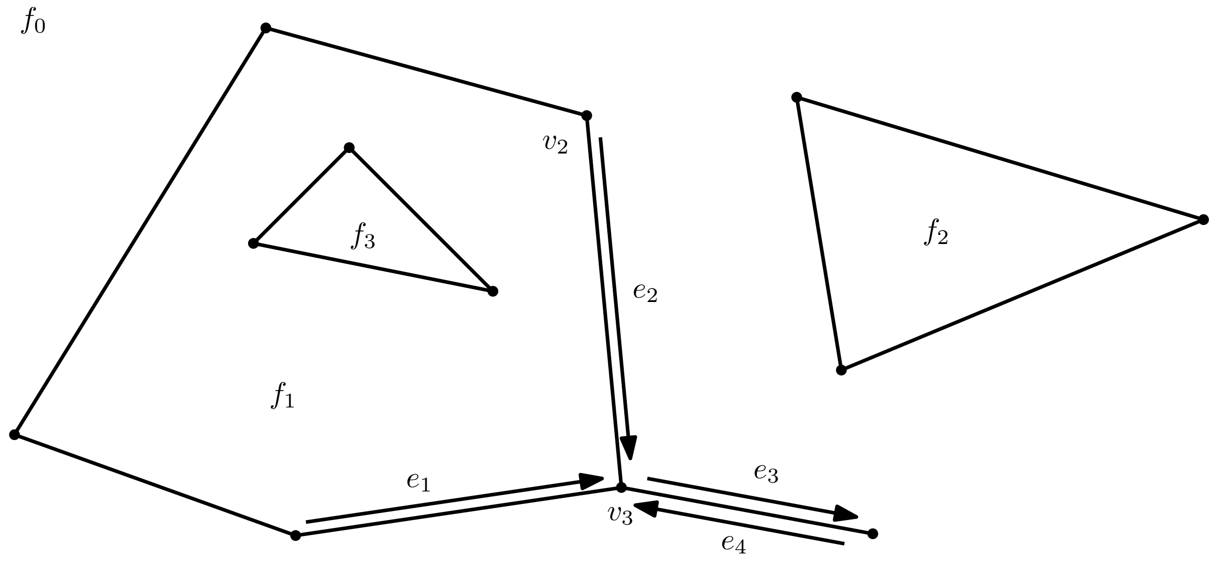
Let X and Y be two independent random variables. Show that

$$E[X \cdot Y] = E[X] \cdot E[Y].$$

Exercise T3 (Randomized Incremental Construction - Backward Analysis):

Consider an algorithm that constructs a binary search tree T via randomized incremental construction. That is, the algorithm selects a random insertion order in the beginning and then constructs the tree by inserting the elements without any balancing operations. Given a set of elements S of cardinality n and another query point p. Show that the expected query time to located p in T is $O(\log n)$.
Hint: Consider the path in T to p already during construction. What is the probability that the length of the path increases due to an insertion of an element ?

Exercise T4 (Doubly Connected Edge List):



a) Consider the picture above, given a call sequence to reach:
- v_3 from e_1
- e_4 from e_1
- f_1 from e_4
- f_1 from f_3
- f_0 from e_1

For instance, we can reach v_3 from e_2 by:

$$v_3 = e_2.next().source()$$

b) Given the set of all edges. How could one easily identify e_3 and e_4?
c) Give a general function to iterated over all faces starting from the infinite face f_0.

For an advanced implementation of the interface you may want to have a look at:
http://doc.cgal.org/latest/Arrangement_on_surface_2/classCGAL_1_1Arrangement__2.html

Exercise P1 (Intersection Points):
Implement the algorithm *NaiveIntersect* and *SimpleIntersect* from the lecture to compute the set of all intersection points for a given set of segments.