General Computer Science II (320102) Spring 2015
Assignment 12: Problem Formulation and Search
– Given May. 7., Due May. 14. –

Problem 12.1 (The wolf, the goat and the lettuce) 20pt

A classical children problem is the following:
A man needs to cross a river, together with his wolf, goat and lettuce. However, only himself and one of his three possessions can fit in the boat, so he needs to carry them one by one. The problem is that in the absence of the man the goat eats the lettuce and the wolf eats the goat (if left together).
Formally express this problem as a search problem (by clearly defining the 4 parameters S, O, I, G) and then give a solution.

Problem 12.2 (Water Jug Problem in SML) 20pt

The water pouring problem is a famous old problem. You are given two buckets of different sizes and you are asked to measure a certain quantity of water with them. For example, given a bucket that can hold $4l$ of water and a bucket that can hold $9l$ of water, measure exactly $6l$. You are only allowed to use the following steps:

- Choose an empty bucket and fill it until it's full
- Pour from one bucket to the other until either the first bucket is empty or the second one is full
- Empty a bucket completely

Write a function *pour* in SML that takes three positive integers as arguments(the first two are the sizes of the buckets and the last one is the quantity we need to measure) and returns a list of pairs of how many liters each bucket has at step i. Please use BFS to find the shortest possible number of moves necessary to measure the correct amount of liters.
An example run would be:
$pour(3, 4, 2) = [(0, 0), (3, 0), (0, 3), (3, 3), (2, 4)]$ You can assume that the input will always be valid. There is no need to take care of exceptions(no one will stop you if you want to do that though).

Problem 12.3 (A^* search on Jacobs campus) 30pt

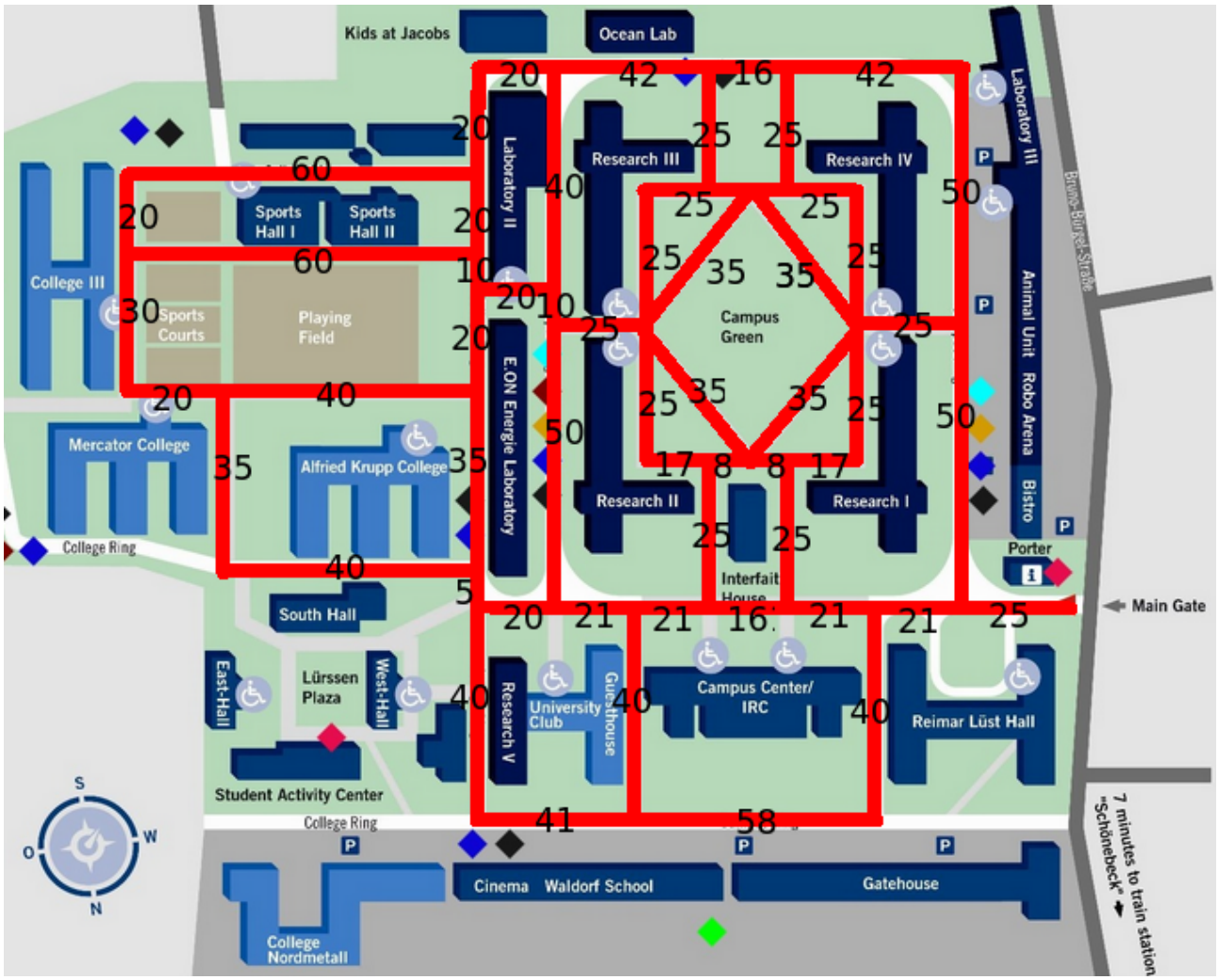
Implement the A^* search algorithm in SML and test it on the problem of walking from the main gate to the main entrance of College 3 C-Block with linear distance as heuristic. The length of line segments are annotated in the map below.

Function signature:

```
val astar = fn
  : 'a
    -> ('a -> ('b * int * 'a) list)
       -> ('a -> int) -> ('a -> bool) -> 'b list * int * 'a
```

Since it's quite confusing, here is how it is called:

```
astar initial_state next heuristic goal_predicate;
```



Problem 12.4 (Search comparison)

Give a concrete example of a state space in which iterative deepening search performs much worse than depth-first search (for example $O(n2)$ vs. $O(n)$). State exactly why this is the case (i.e briefly show that the time complexity is better for depth-first search in this example). In general, can you think of a certain type of problems where iterative deeping search will almost always perform worse than Depth first search. 20pt