

Sommersemester 2010

## Übungen zu Informatik II - Blatt 11

Abgabe in der Übung am 29. 06 / 30. 06. 2010

Bitte beachten Sie, dass die Programmieraufgaben von Ihnen in der Übung vorgeführt und erklärt werden müssen. Zusätzlich senden Sie die Lösung, unter Angabe ihres Namens, an **dm@tu-clausthal.de**.

### Aufgabe 1 (Backtracking in Python, 6+2 Punkte)

Sie sitzen in der Mitte eines Labyrinthes und möchten einen Weg (Zusatzaufgabe: alle Wege) in die Freiheit finden. Eine gute Möglichkeit, diesen Weg zu finden ist die Verwendung von Backtracking. Laden Sie sich hierfür von der Webpage das Archiv `labyrinth.zip`. Dies enthält das Framework `labyrinthFramework.py` und ein paar Labyrinth. Diese Labyrinth dienen zum Testen ihrer Lösung! Sie können dem Framework mittels Aufruf:

```
python labyrinthFramework.py TESTLABYRINTH
```

eine Labyrinth-Datei übergeben. Ein Aufruf von:

```
python labyrinthFramework.py -h
```

erklärt den Aufbau der Labyrinth-Datei, falls Sie sich ein eigenes Labyrinth erstellen möchten. Auch ein parameterfreier Aufruf ist möglich, in diesem Fall wird das eingebaute Labyrinth verwendet. Ziel ist in diesem Fall nicht das Finden des kürzesten Weges, sondern die korrekte Anwendung von Backtracking. Bitte kommentieren Sie ihren Code und geben Sie am Ende ihren gefunden Weg in folgender Form aus.

Weg:

```
(1, 1) (2, 1) (3, 1) (3, 2) (3, 3) (3, 4) (3, 5) (2, 5) (1, 5) (1, 4)
```

oder

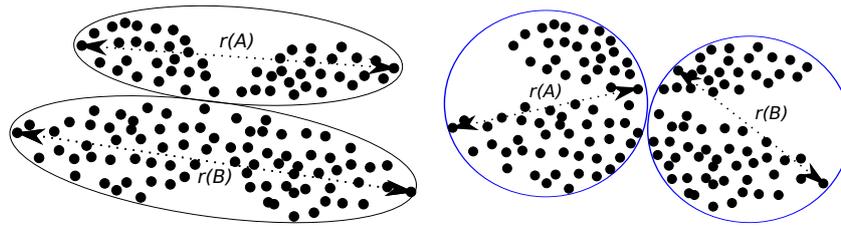
```
# # # # # # # # # # # #  
# x # Z x x # # #  
# x # x # # #  
# x x x x x #  
# # # # # # # # # # #
```

- Implementieren Sie ihren Algorithmus in das Framework `labyrinthFrame.py`
- Dokumentieren Sie ihren Code!

## Aufgabe 2 (Clustering – Backtracking, 6+2 Punkte)

Wir definieren für eine Punktmenge  $A = x_1, \dots, x_m$  den Durchmesser  $r(A) := \max\{d(x_i, x_j) \mid x_i, x_j \in A\}$  wobei  $d$  irgend eine Abstandsfunktion ist.

Gegeben sei eine Menge  $n$  von Datenpunkten und es gelte die euklidische Abstandsfunktion  $d(x_i, x_j) = \|x_i - x_j\|$  zwischen allen Punktpaaren. Unterteilen Sie die Datenpunkte so in zwei Teilmengen  $A$  und  $B$ , dass  $\max\{r(A), r(B)\}$  minimal wird.



(a) Falsche Unterteilung

(b) Korrekte Unterteilung

Abbildung 1:

- a) Beschreiben Sie in Pseudo-Code einen Algorithmus zur Lösung dieser Aufgabe. Die Bestimmung des Durchmessers einer Menge von Punkten ist bereits gegeben, z.B folgendermaßen:

```

Function durchmesser(A)
Eingabe : Eine Punktmenge A
Ausgabe: Durchmesser dieser Punktmenge
d = 0
for i = 0 ... len(A) - 1 do
  for j = i+1 ... len(A) - 1 do
     $d_{ij} = \|x_i - x_j\|$ 
    if  $d_{ij} > d$  then
       $d = d_{ij}$ 
    end
  end
end

```

- b) Wo und wie kann mittels Precomputation in Ihrer Lösung Zeit gespart werden? Beziehen Sie auch die Funktion durchmesser in Ihre Überlegungen ein. Erläutern Sie ihre Lösung.

Bemerkung: Solch eine Unterteilung heißt Clustering. Dieses Verfahren kann man natürlich bei Bedarf rekursiv fortsetzen. Das Clustering ist eine wichtige Technik im sogenannten Maschinen-Lernen und dem Data-Mining.