Eidgenössische
Technische Hochschule
Zürich

Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Federal Institute of Technology at Zurich

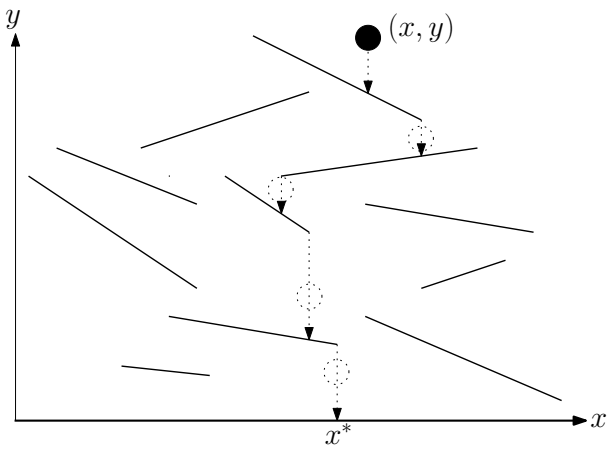
Institut für Theoretische Informatik
Peter Widmayer
Thomas Tschager
Antonis Thomas

1. Juni 2016

Datenstrukturen & Algorithmen Blatt 14 FS 16

Aufgabe 14.1 *Rollende Kugeln.*

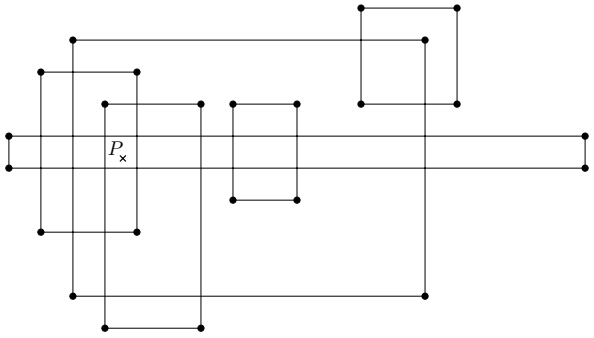
Wir betrachten eine Kugel, die über eine Anordnung von Holzrinnen hinunter rollt. Die Kugel wird an einem bestimmten Punkt losgelassen. Von dort fällt sie gerade herunter bis sie auf eine Rinne trifft. Ab da rollt sie diese Rinne hinunter, bis ans untere Ende der Rinne, und fällt dann wieder gerade nach unten. Dies wiederholt sich, bis die Kugel auf dem Boden auftrifft. Im Folgenden repräsentieren wir die Rinnen als Liniensegmente. Wir nehmen an, dass sich keine zwei Segmente berühren oder schneiden. Alle Endpunkte haben paarweise verschiedene x- und y-Koordinaten. Zudem ist kein Segment horizontal. Die Startposition der Kugel ist durch die Koordinate (x, y) gegeben.



Beschreiben Sie einen Algorithmus, der möglichst effizient ermittelt, an welcher x-Position x^* die Kugel auf den Boden trifft. Geben Sie die Laufzeit Ihres Algorithmus in Abhängigkeit der Anzahl Liniensegmente an.

Aufgabe 14.2 *Aufspiessen von orthogonalen Rechtecken.*

Seien n orthogonale Rechtecke gegeben, welche sich überlappen können. Es soll ein Punkt berechnet werden, für den die Anzahl aufgespiesster Rechtecke grösstmöglich ist. Ein Rechteck wird genau dann von einem Punkt aufgespiesst, wenn dieser Punkt in dem Rechteck (inklusive dessen Rand) enthalten ist. Im folgenden Beispiel ist P ein solcher Punkt, da er vier Rechtecke aufspiesst, und kein anderer Punkt mehr Rechtecke aufspiesst.



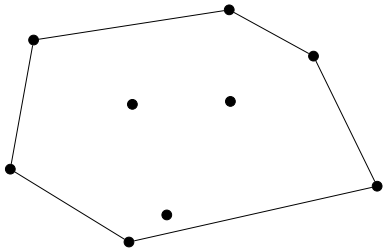
Beschreiben Sie einen möglichst effizienten Scanline-Algorithmus, der einen Punkt findet, der die meisten Rechtecke aufspiesst. Geben Sie zudem die Laufzeit Ihrer Lösung an.

Aufgabe 14.3 *B-Bäume.*

Geben Sie einen B-Baum der Ordnung 5 mit Höhe 3, sowie einen weiteren, neuen Schlüssel an, sodass beim Einfügen dieses neuen Schlüssels die Höhe des Baumes wächst.

Aufgabe 14.4 *Konvexe Hülle (**Programmieraufgabe**).*

In dieser Aufgabe soll Graham's Scan zur Berechnung der konvexen Hülle einer gegebenen Menge von Punkten implementiert werden. Seien $p_1, ..., p_n \in \mathbb{N}^2$ Punkte in der Ebene mit ganzzahligen Koordinaten und in allgemeiner Lage (d.h. keine drei von ihnen liegen auf einer Geraden). Das Ziel ist die Berechnung der *konvexen Hülle*, also der kleinsten konvexen Menge, die alle n Punkte enthält. Die folgende Abbildung zeigt ein Beispiel einer konvexen Hülle einer Punktemenge.



Eingabe Die erste Zeile der Eingabe enthält lediglich die Anzahl t der Testinstanzen. Danach folgt genau eine Zeile pro Testinstanz. Sie enthält die Folge $n, x_1, y_1, ..., x_n, y_n$. Der Wert $n \in \mathbb{N}$, $3 \leq n \leq 1000$, gibt die Anzahl der folgenden Punkte an, und für jedes i, $1 \leq i \leq n$, definiert das Paar $x_i, y_i \in \mathbb{N}_0$, $0 \leq x_i, y_i \leq 1000$, die x- und die y-Koordinate des i-ten Punkts.

Ausgabe Für jede Testinstanz soll lediglich eine Zeile ausgegeben werden. Sie enthält die Liste aller Koordinaten der Eckpunkte der konvexen Hülle im Uhrzeigersinn, beginnend mit den Koordinaten des am weitesten links liegenden Eckpunkts (kleinste x-Koordinate). Falls zwei Punkte in der Liste die gleiche x-Koordinate haben, dann starten wir mit demjenigen, der die kleinere y-Koordinate besitzt.

Beispiel

Eingabe:

```
2
3 1 1 2 4 3 9
5 0 0 0 3 2 3 2 0 1 1
```

Ausgabe:

```
1 1 3 9 2 4
0 0 0 3 2 3 2 0
```

Hinweis Für diese Aufgabe gibt es lediglich ein Testset.