
Algorithm Visualization in Teaching Spatial Data Algorithms

Jussi Nikander
jtn@cs.hut.fi

Helsinki University of Technology
Laboratory of Software Technology



Contents

- Background and motivation
- Algorithm visualization and animation
- TRAKLA2 system and Spatial Data Algorithm exercises
- Examples
- Preliminary experiences
- Final remarks



Background

- Basic course on spatial data algorithms at Helsinki University of Technology
- Students major in geoinformatics
 - Typically course is taken on the third study year
 - Students have some background in both geoinformatics and computer science
- The goal is to teach basic SDA on conceptual level
- Problem: Students do not learn very well
 - Do not grasp the main concepts and application areas of many algorithms discussed on the course

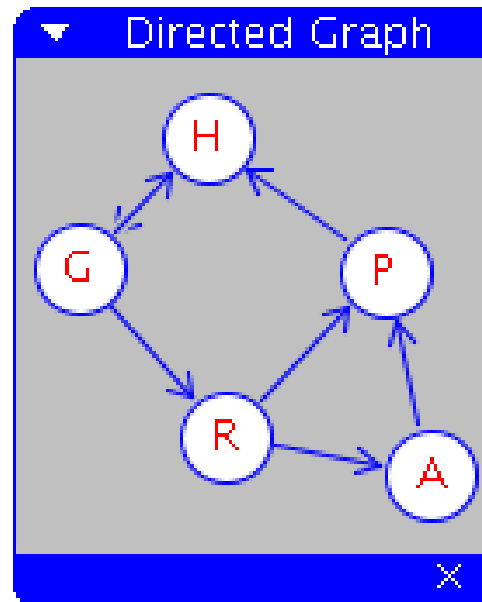
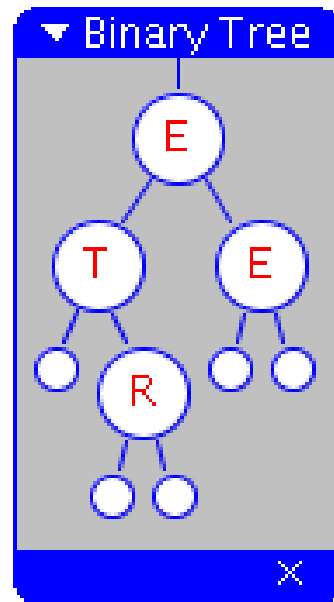
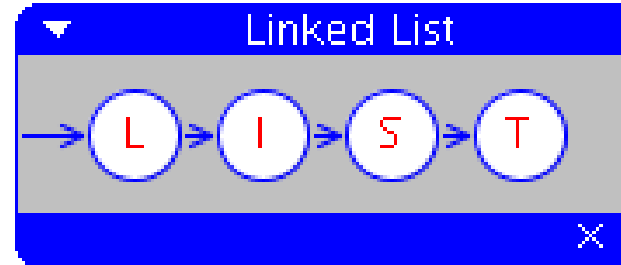
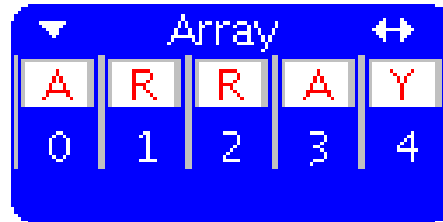


Proposed Solution

- Add **meaningful tasks** that help the students to build better mental models of the data structures and algorithms
- details:
 - Use **visualization**
 - Design tasks that can be solved **whenever** and **where-ever**
 - Make tasks **automatically assessed**
- Solution: **Visual algorithm simulation** with automatic assessment



Common views



Algorithm simulation

- manual manipulation of data structures in order to duplicate modifications a real algorithm would do
- Can be done using pen and paper...
- ...or in a system that supports the manipulation of data structures
 - In **visual algorithm simulation** the manipulation is done by modifying data structure visualizations
- Goal is to understand how an algorithm works on a conceptual level without having to dive into the implementation details



TRAKLA2

- A system for delivering and automatically assessing **visual algorithm simulation exercises**
 - Mainly covers basic data structures and algorithms
- A dynamic web environment and a Java applet
 - All course management in the web environment
 - Exercises solved using the Java applet
- Some features:
 - Supports multiple submissions
 - ◆ New input each time learner tries to solve an exercise
 - Automatic assessment and immediate feedback



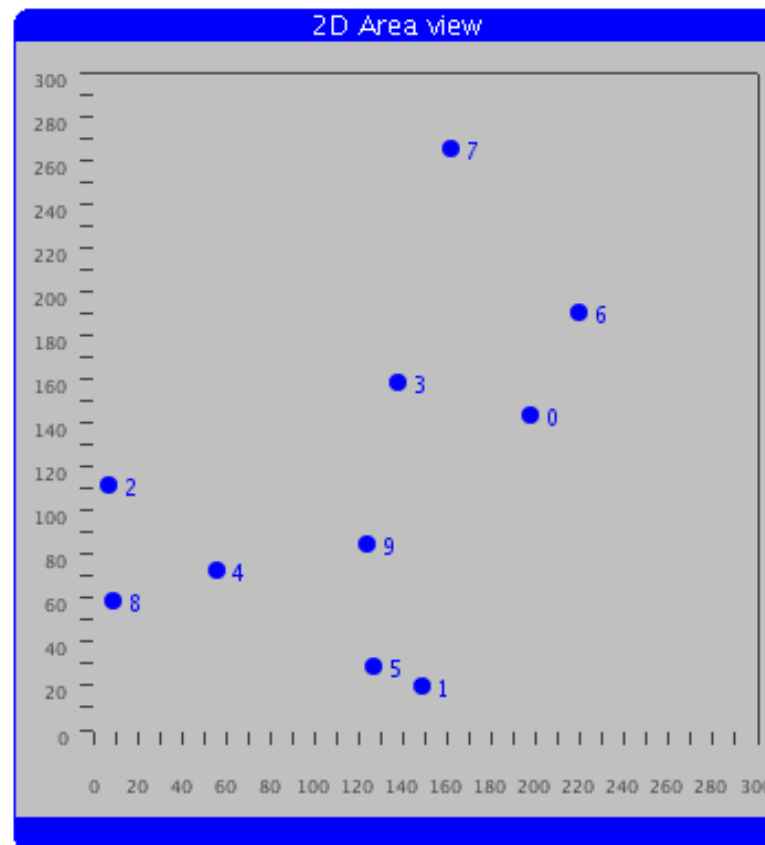
SDA in TRAKLA2

- First expansion of TRAKLA2 beyond basic data structures and algorithms
- Covers structures and algorithms for 2D data
 - Geoinformatics is the main application area
- Required several extensions to the framework
 - Support for multidimensional data items
 - 2D visualization of data items
 - ◆ Required for illustrating how data items relate to one other



Example

0	1	2	3	4	5	6	7	8	9
x 198.0	x 149.0	x 7.0	x 138.0	x 56.0	x 127.0	x 220.0	x 162.0	x 9.0	x 124.0
y 144.0	y 20.0	y 112.0	y 159.0	y 73.0	y 29.0	y 191.0	y 266.0	y 59.0	y 85.0
0	1	2	3	4	5	6	7	8	9



Spatial Exercises in T2

- Currently 11 exercises ready for deployment
 - Plus approximately a dozen exercises either under construction or in planning stages
- Automatically assessed, meaningful tasks that students can solve where-ever and whenever they want.
- All exercises can be solved visually



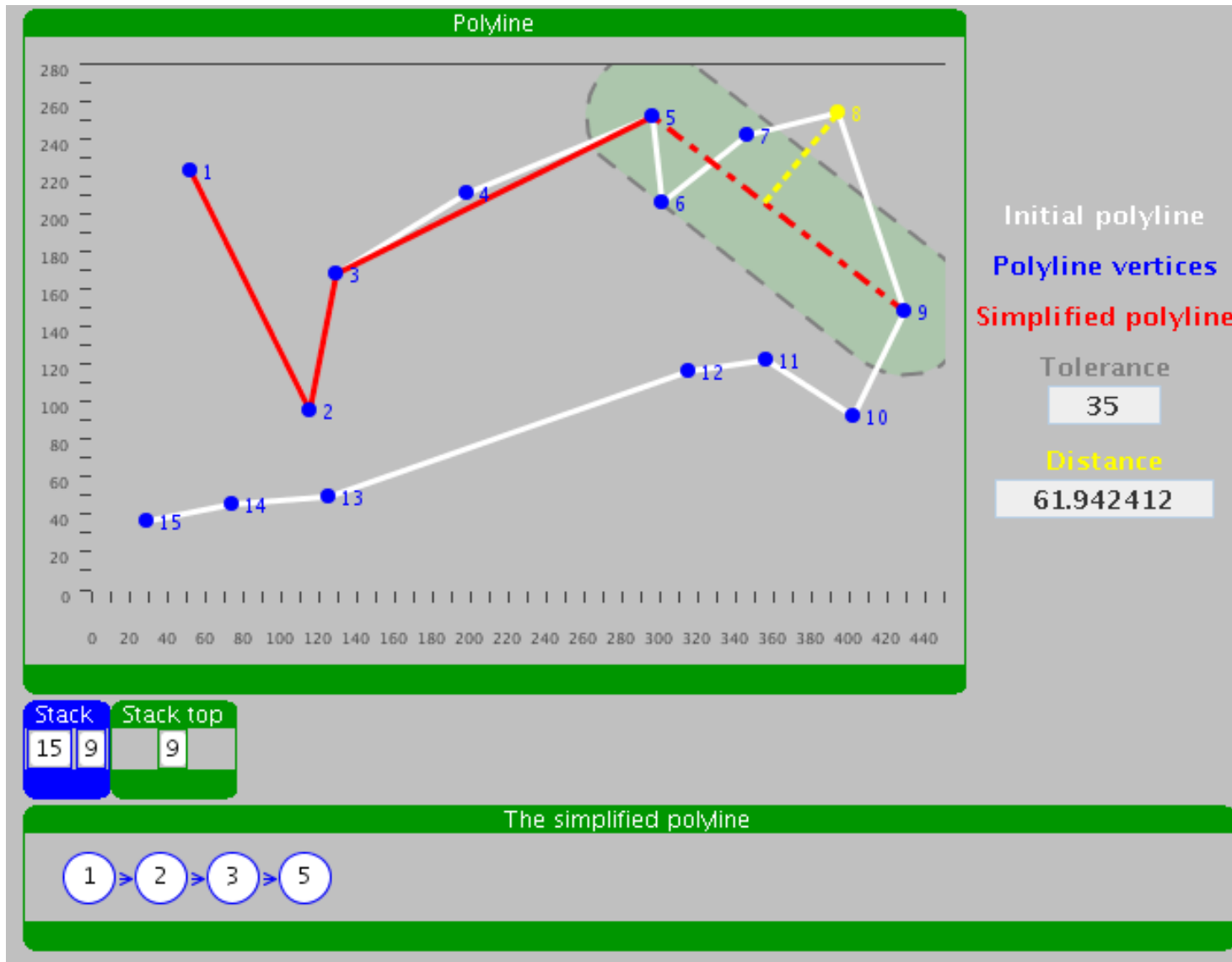
List of Exercises

8. Spatial Data Algorithms

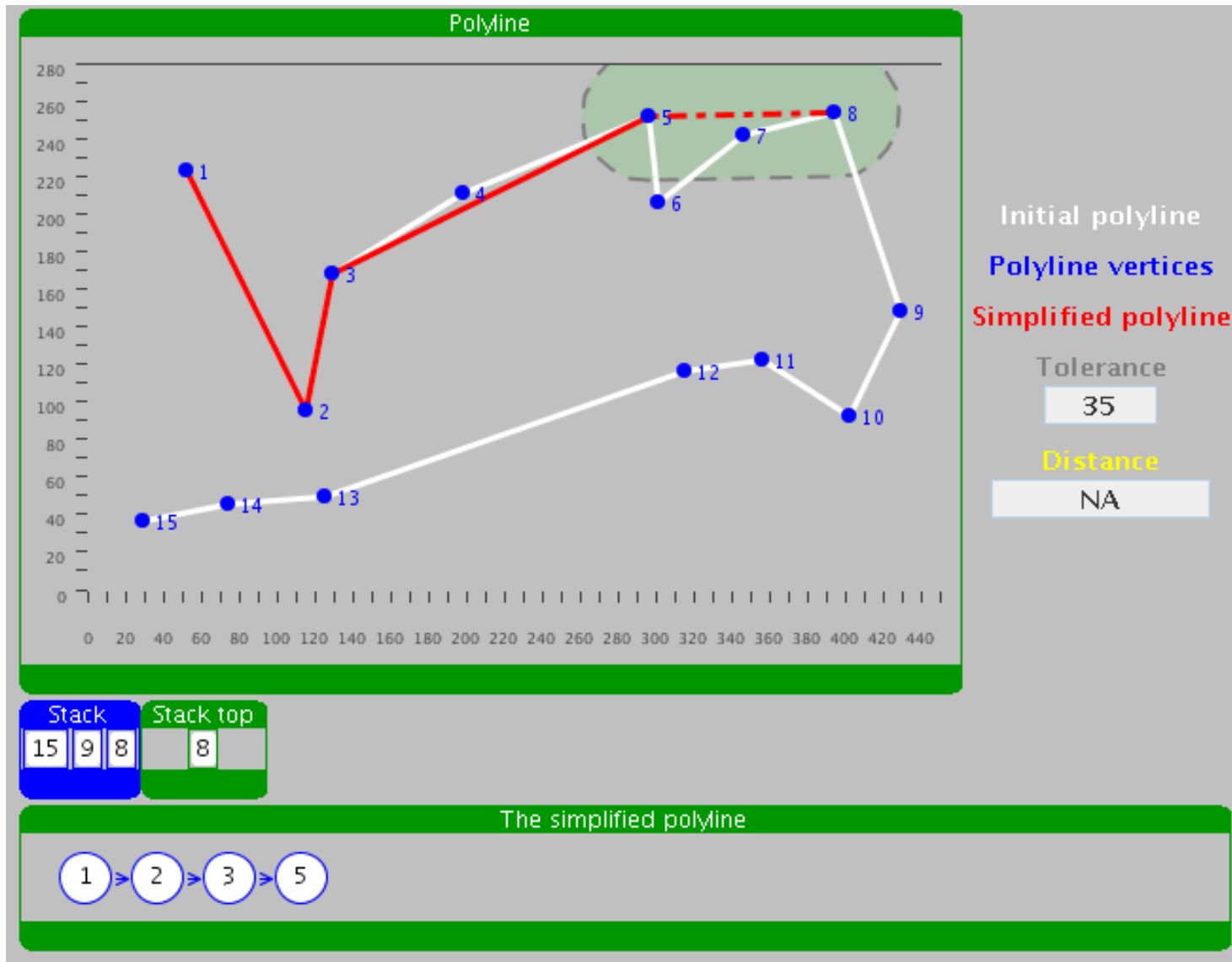
- ◆ Point in Polygon
- ◆ Point in Polygon with R-Tree
- ◆ Douglas-Peucker Line Simplification
- ◆ Closest pair of points
- ◆ Point-Region Quadtree Insert
- ◆ R-Tree Insert
- ◆ Line Intersections Using Line Sweep
- ◆ Visibility with Rotational Sweep
- ◆ Expanding Wave-method
- ◆ Adding a point to TIN
- ◆ Voronoi Construction



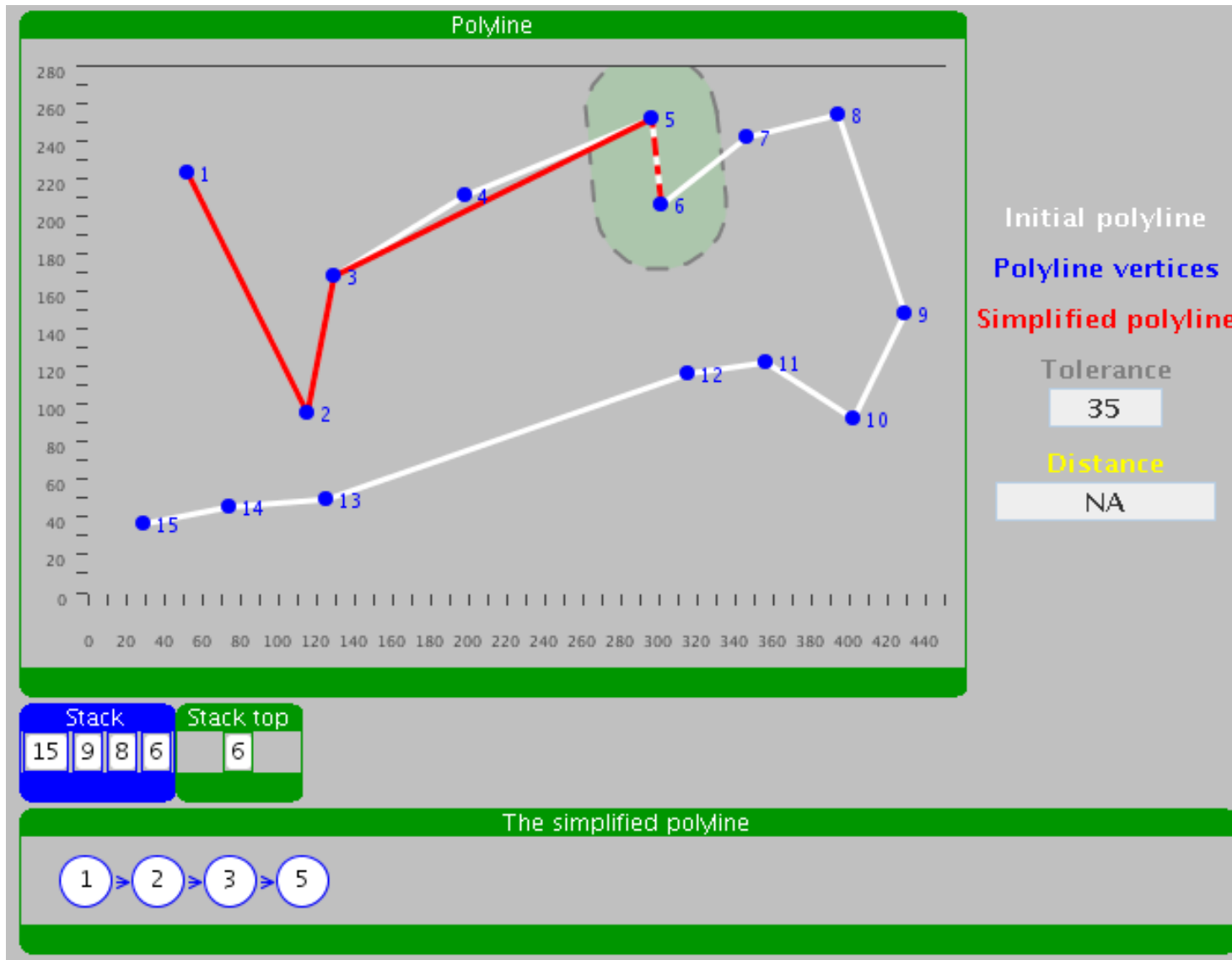
Example: Douglas-Peucker



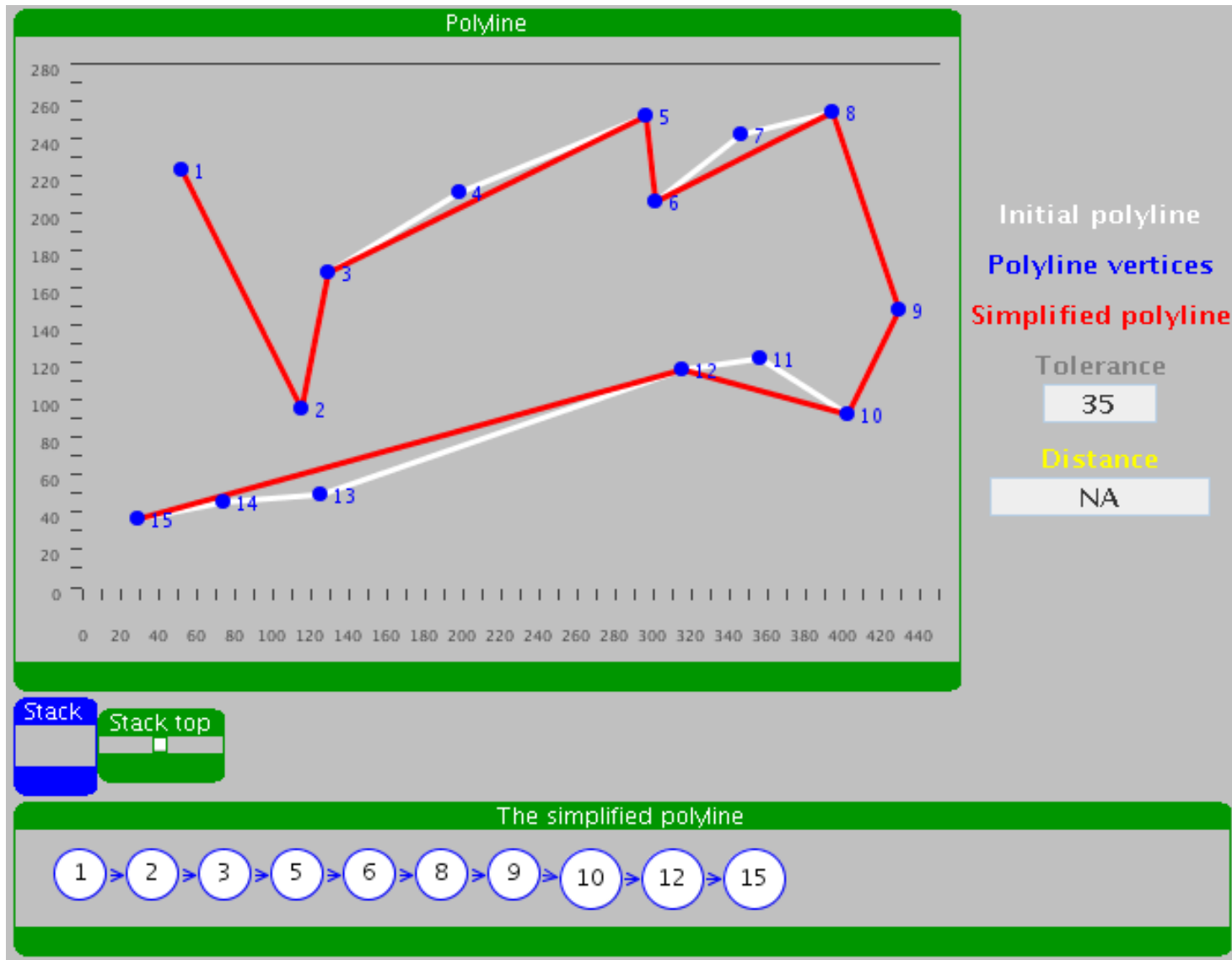
Example (cont.)



Example (cont.)



Example (end)



Example: Web environment



TRAKLA2

EXERCISES

EBOOK

SETTINGS

FEEDBACK

HELP

IN FINNISH

Jussi Nikander (49537E)

LOGOUT

Test course > Round 8: Spatial Data Algorithms > 3. Douglas-Peucker Line Simplification

Deadline 01.01.2009 00:00:00

Hide text Hide pseudo-code Open text in new window

Previous exercise Next exercise

Task Instructions

Simplify the line by simulating the Douglas-Peucker line simplification algorithm.

Douglas-Peucker

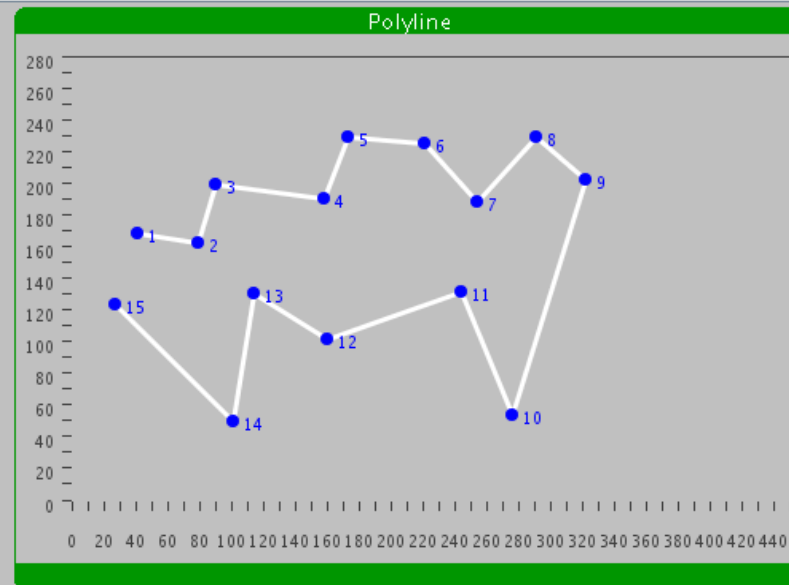
```
DOUGLAS_PEUCKER(Polyline l, int tolerance)
1. initialize list Q
2. initialize stack S
3. let v be the array of vertices in l
4. anchor = v[0]
5. floater = v[v.length - 1]
6. Q.append(anchor)
7. S.push(floater)
8. while (S is not empty)
9.   let seq be a line segment from anchor to floater
10.  maxd = 0
11.  farthest = floater
13.  i = index of anchor + 1
14.  while (i < index of floater)
15.    let d be the shortest distance from seq to v[i]
16.    if (d > maxd)
17.      maxd = d
18.      farthest = v[i]
19.    i = i + 1
20.  if (maxd <= tolerance)
21.    Q.append(S.pop())
22.    anchor = floater
23.    floater = S.peek()
24.  else
25.    floater = farthest
26.    S.push(floater)
27. return Q
```

Simplify the polyline following the Douglas-Peucker line simplification algorithm.

Font
14

Animator
◀ ◁ ▷ ▶

Exercise
Reset Model answer Submit



Initial polyline
Polyline vertices
Simplified polyline
Tolerance
35
Distance
NA

Stack Stack top

The simplified polyline

Applet exerciseApplet started

trakla.cs.hut.fi



Experiences

- Has been used on basic spatial data structures course at Helsinki University of Technology
- 9 TRAKLA2 exercises, 15 students on the course
- System divided student opinions
 - Most think the system is useful and exercises suitably challenging, however
- All students passed exercises
- Learning results have not been analyzed yet



Final remarks

- The system is freely available
 - Java applet and exercises have been published under GPL
 - Web environment will soon be published under Apache License
- You can try the exercises (and the web environment) at <http://www.cs.hut.fi/Research/TRAKLA2/exercises/index.html>
- Or just google TRAKLA2
- **Questions?**

