Dynamic Visualization of Algorithms for Solving the Travelling Salesman Problem (TSP) and TSP's Extensions to Real Word Scenarios on a Reactable Machine

Manushaqe Muco
manjola@mit.edu

HGS MathComp
Heidelberg Graduate School of Mathematical and Computational Methods for the Sciences

Ruprecht-Karls-Universität Heidelberg
Heidelberg, Germany

Supervisors: Prof. Katja Mombaur, Prof. Gerhard Reinelt

June 2013 – August 2013

Abstract

I present a tool for visualizing algorithms that solve the Travelling Salesman Problem (TSP), in a fun, dynamic and visually pleasing manner. TSP is extended to represent a simplified version of a real world scenario: travelling between a region of cities, where some roads experience traffic jams and constructions. The tool is meant to be used on a tangible user interface machine called reactable. However, the simulation can also be used on personal computers, though less fun. The tool is built in such a way that it can easily be extended to include other algorithms, aside from TSP.

1. Vision

I present a tool for visualizing algorithms that solve the Travelling Salesman Problem (TSP), in a fun, dynamic and visually pleasing manner. TSP is extended to represent a simplified version of a real world scenario: travelling between a region of cities, where some roads experience traffic jams and constructions. The tool is meant to be used on a tangible user interface machine called reactable. However, the simulation can also be used on personal computers, though less fun.

The Traveling Salesman Problem, deals with creating the ideal path that a salesman would take while traveling between cities. The solution to any given TSP would be the shortest way to visit a finite number of cities, visiting each city only once, and then returning to the starting point.

A reactable is a device that has a tangible user interface and was originally built as a musical instrument by Music Technology Group at the Universitat Pompeu Fabra, in Barcelona, Spain [1]. At Heidelberg University, the device was rebuilt by students of Prof. Mombaur. However, the reactable is not only a musical instrument: Its graphical interface can also be used to both input data as well as to display computational results in a visually pleasing manner.

The tool is built in such a way that it can easily be extended to include other algorithms, aside from TSP.

2. Elements of the System

Here I describe how the elements and design of the system looks in both simulation (computer) and on the reactable.

2.1. TSP Algorithms

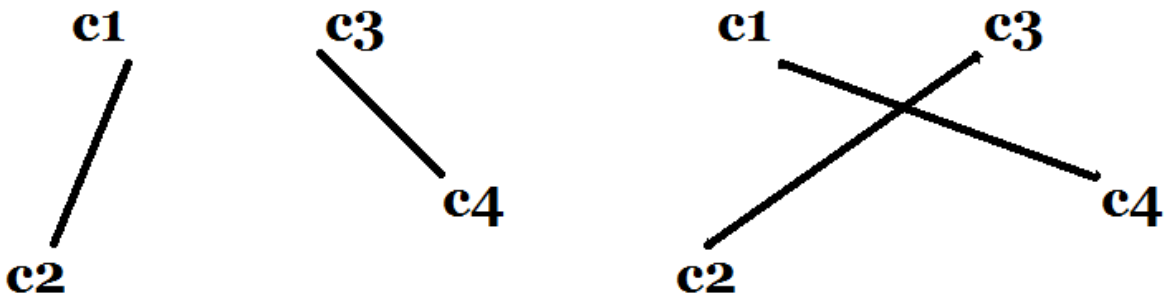
The Traveling Salesman Problem, deals with creating the ideal path that a salesman would take while traveling between cities. The solution to any given TSP would be the shortest way to visit a finite number of cities, visiting each city only once, and then returning to the starting point.

TSP is one of the most famous problems of combinatorial optimization and consists of finding a shortest Hamiltonian cycle/tour in a weighted graph. In graph theory, a Hamiltonian path is a path in an undirected or directed graph that visits each vertex exactly once. A Hamiltonian cycle is a Hamiltonian path that is a cycle.

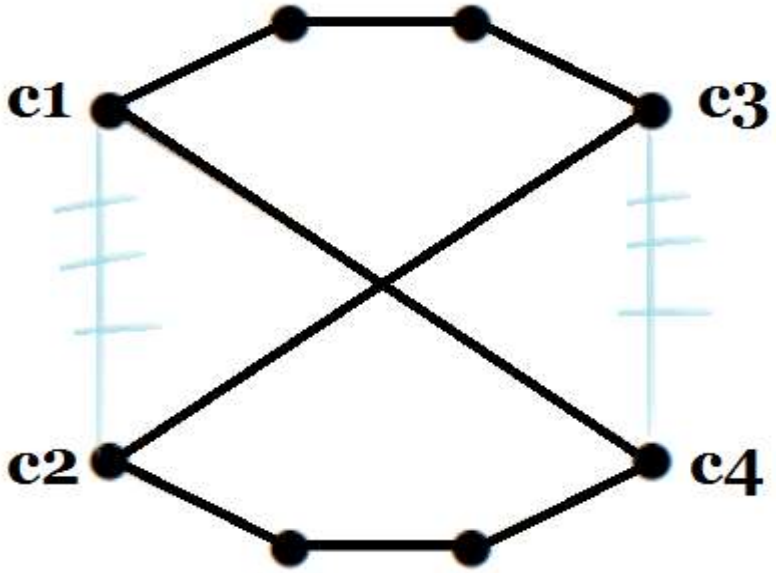
The algorithm I use to describe how the system works for the rest of the paper is called a 2-opt algorithm.

2.1.1. 2-opt

In optimization, 2-opt is a simple search algorithm first proposed by Croes in 1958 for solving TSP. A naïve implementation of 2-opt runs in O(n^2). This involves selecting an edge (c1, c2) and searching for another edge (c3, c4), completing a move only if dist(c1, c2) + dist(c3, c4) > dist(c2, c3) + dist(c1, c4). This moves is often referred to as a swap.



Picture: A 2-opt swap.
dist(c1, c2) + dist(c3, c4) > dist(c2, c3) + dist(c1, c4)

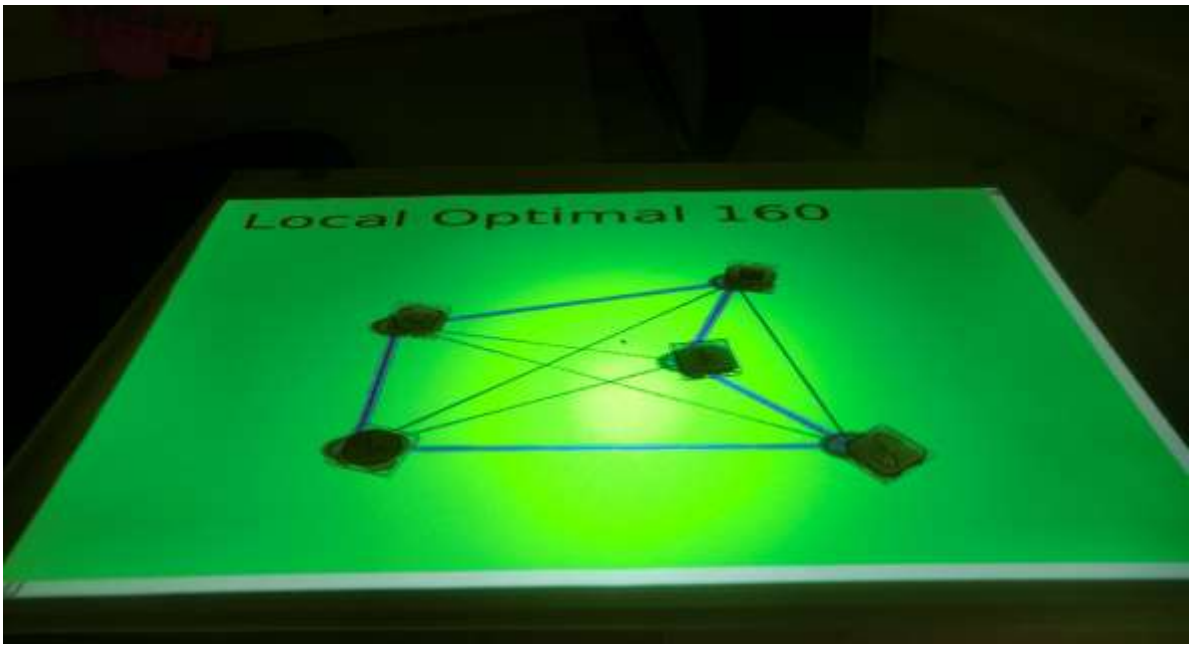


Picture: A 2-opt swap in a graph. Original edges in blue.

2.2. The Reactable Machine (FIX)
A reactable is an electro-acoustic musical instrument with a tangible user interface (TUI). In principle, a reactable is a table, whose surfaces uses a tone-generating computer. The reactable is a table with a translucent panel, which is used wherever possible in a darkened room. The tabletop acts as a backlight display. Below the tabletop is a camera that monitors the action on the table and a projector that projects feedback to the user on the tabletop. So that the projection of the projector does not interfere with the image processing, the camera operates in the infrared spectrum: the tabletop is illuminated by five IR laser diodes. In order to reduce unwanted reflections some polarizing filter is used [2].

The reactable has several blocks that can be placed on its surface. Each block has a unique ID that can be recognized by the camera and then passed to a software for further use. Blocks can be added, removed, rotated.

The use of the reactable is not limited to music. In fact, this project extends the use of the reactable to algorithm visualization, and traffic jam and road construction simulations.



Picture: Reactable photos

2.3. Simulator
For the simulation of the reactable, I use the TUIO software. More information about this software can be found on its webpage [3].

2.4. Graph
In a real world scenario, the graph is meant to represent the map of a region, where the nodes are cities, and edges are the roads connecting the cities. The TSP remains the same, however, modernity adds a little twist. Our modern salesman can travel by car. Now, he has to deal with traffic jams, road constructions, etc.

All edges are originally activated, meaning all roads are possible to cross. However, some roads may have a lot of traffic, and the salesman will want to avoid them. I use a special object to denote all the roads that have traffic jams. Those roads can no longer be included in the salesman's tour because of too much traffic.

Some roads may be under construction. The salesman cannot use these roads to travel. If a road is under construction, the edge will be deactivated, indicating that the salesman cannot use that road.

2.4.1. Nodes (or Normal Objects)
A node in the graph is meant to represent a city. Nodes are represented as circles in the graph. On the reactable, the nodes are normal blocks. I have recorded the ID of these blocks as node/normal objects.

2.4.2. Edges
The edge connects two nodes. In the graph, it is represented as a line connecting two nodes. On the reactable, it is represented as a line connecting two normal objects/blocks.

The length of the edge is its Euclidian distance units. The edge is black if the edge is activated (no construction on this road). The edge is light-blue if it is deactivated (there's construction happening on this road).

2.4.3. Special Objects
The special objects are represented as squares surrounded by a dashed circle. The circle represents the area that the object affects. If this area touches any of the edges, the edge cannot be included in the tour; the roads touched by the area have traffic jams.

On the reactable, the special objects are blocks, with IDs recorded as special.

2.4.4. Cursor
The cursor in the simulation is meant to represent the human fingers on the reactable.

2.5. Initializing the graph on a computer
By running the *main.py* file, the user can bring up the TUIO simulator. The blocks on the side can be nodes or special objects. The white square in the middle represents the reactable. The user can drag the blocks on the reactable surface using the cursor. Alongside, the graph representation of what the user is forming will be presented.

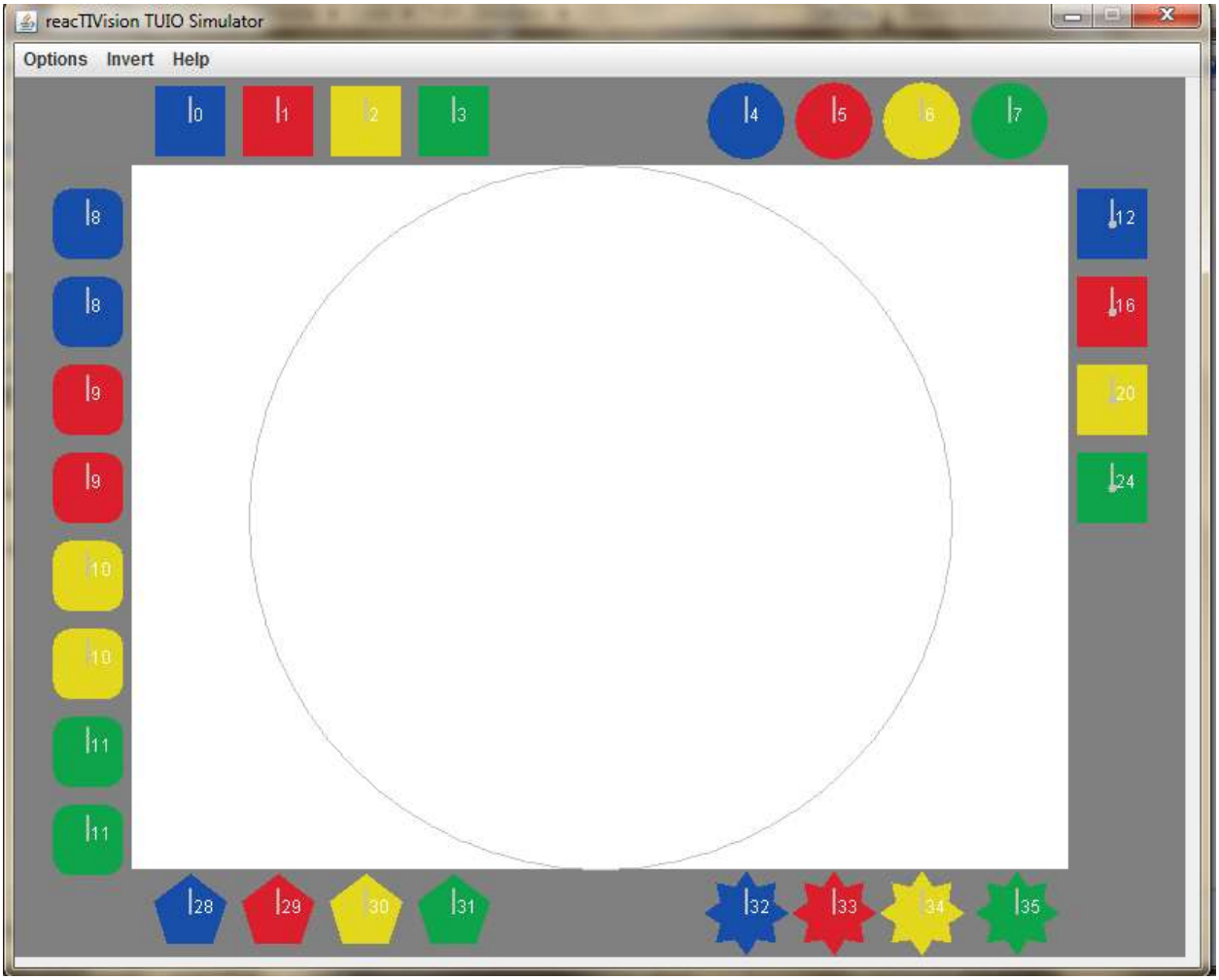


Figure: TUIO Simulator.

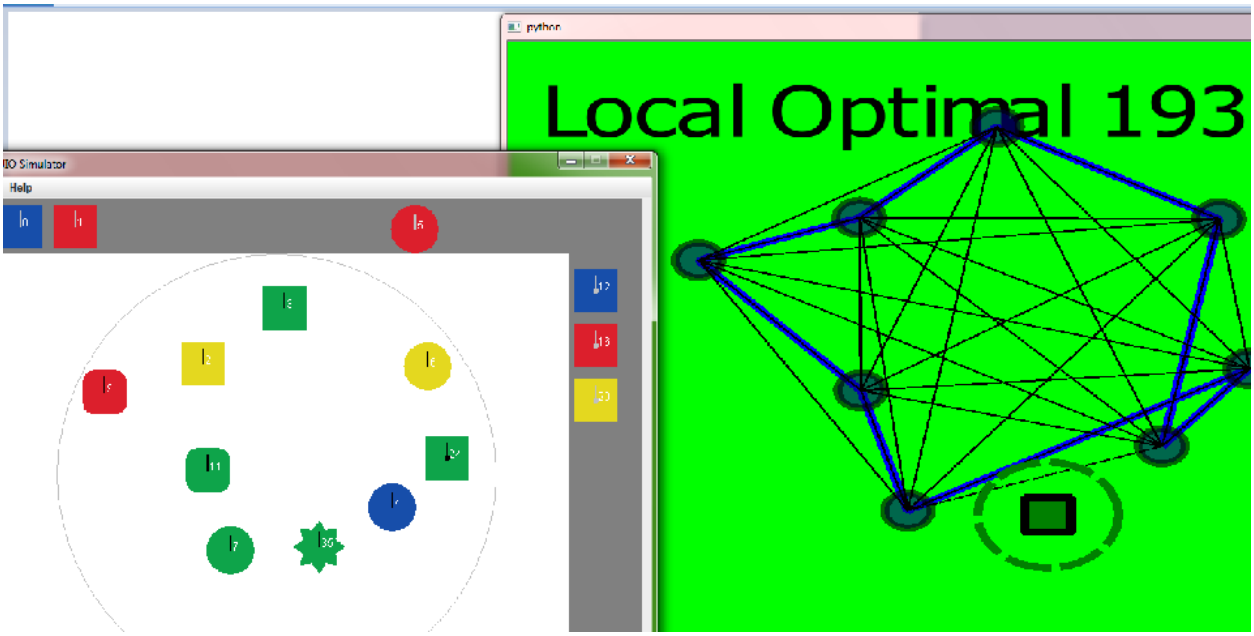


Figure: TUIO Simulator alongside the Graph Representation.

The background is light bluish while the algorithm is running to find a solution. The text will show *Current Solution*. When the algorithm has found the local optimal solution, the screen will turn green and the text will say *Local Optimal*, alongside with the value of the optimal tour. The tour is represented by edges colored in blue.

2.6. Initializing the graph on a reactable

When the user connects the computer with the tool/software to the reactable, he can now play with the actual reactable. The computer screen will show something similar to the next figure: scans of the blocks, called fiducials. The fiducials allow distinguishing a unique ID, location and rotation of blocks.

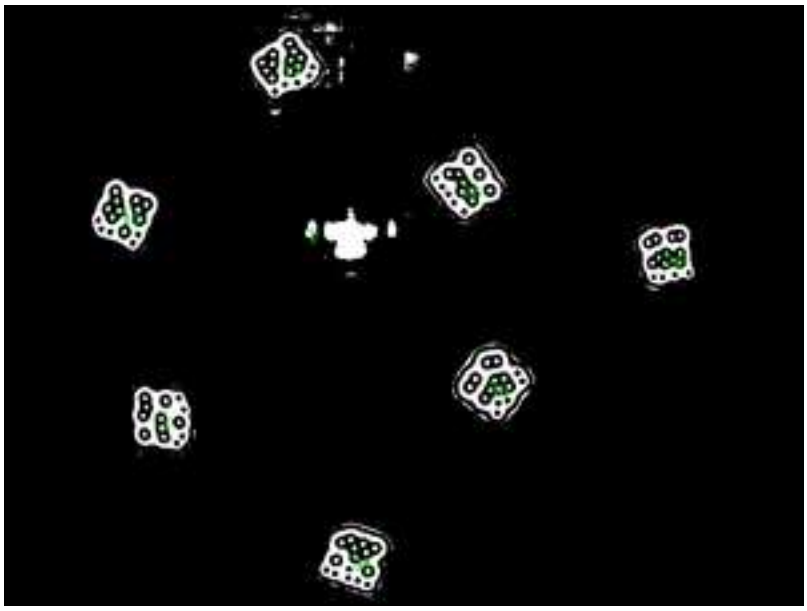


Figure: Image recognition of blocks; fiducials.

The real fun happens on the reactable. More than a user can play with the software. They can add blocks, change the blocks location, or rotation, remove blocks. The blocks can be nodes or special objects. The users can also use two fingers to deactivate an active edge (road becomes under construction), or activate a deactivated edge (road is no longer under construction). After each change, the users and the users can see how the tour changes.

Below I present some of the pictures of the reactable in use. I will also provide links to videos where you can see different aspects of the software in use. I highly recommend watching them, because it will give a better understanding of this interactive project. They are also fun to watch.

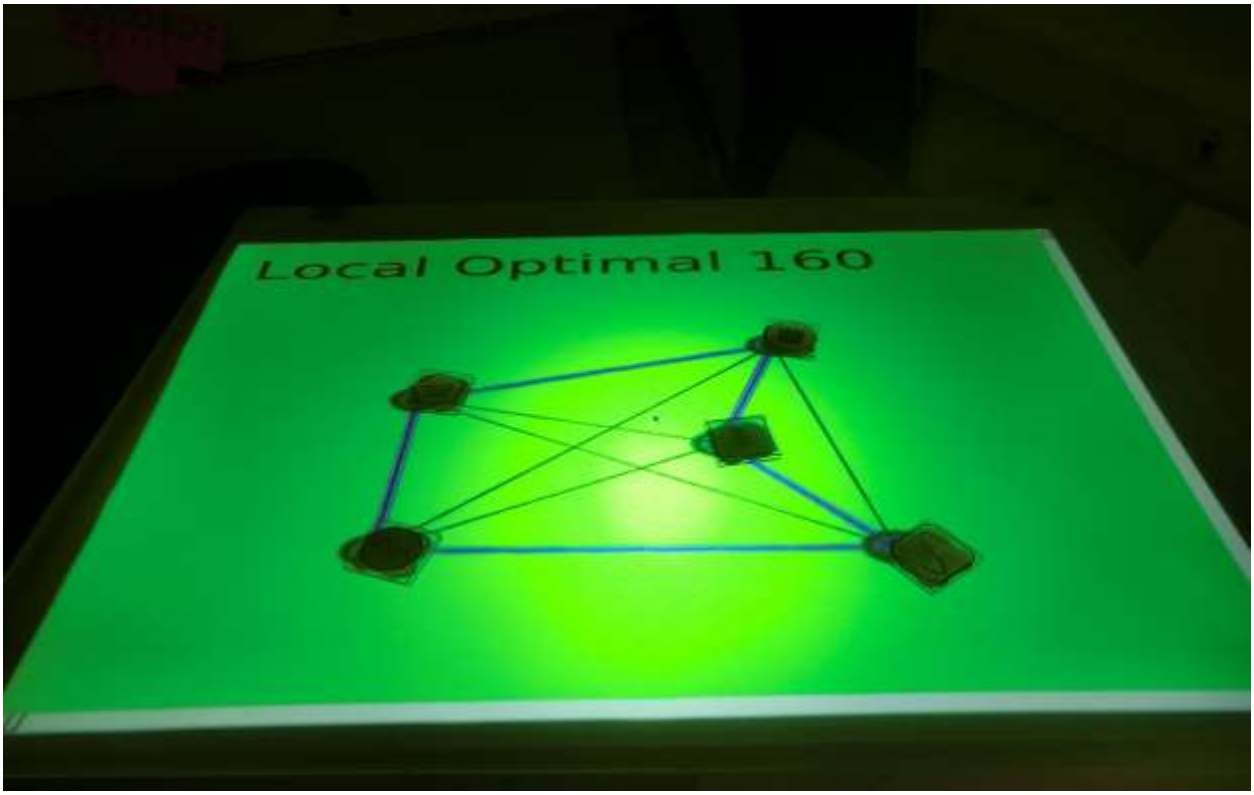


Figure: Algorithm has found the local optimal tour.

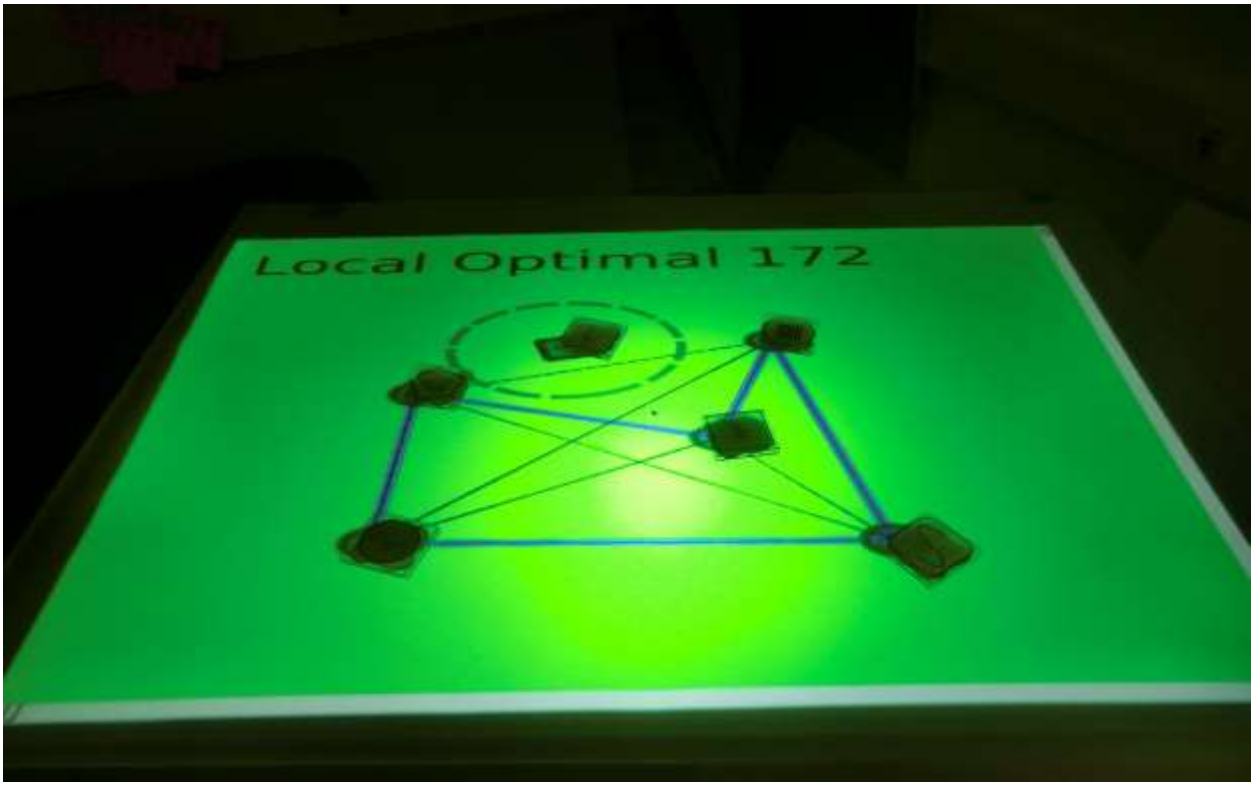


Figure: Algorithm has found local optimal. Notice the special object.

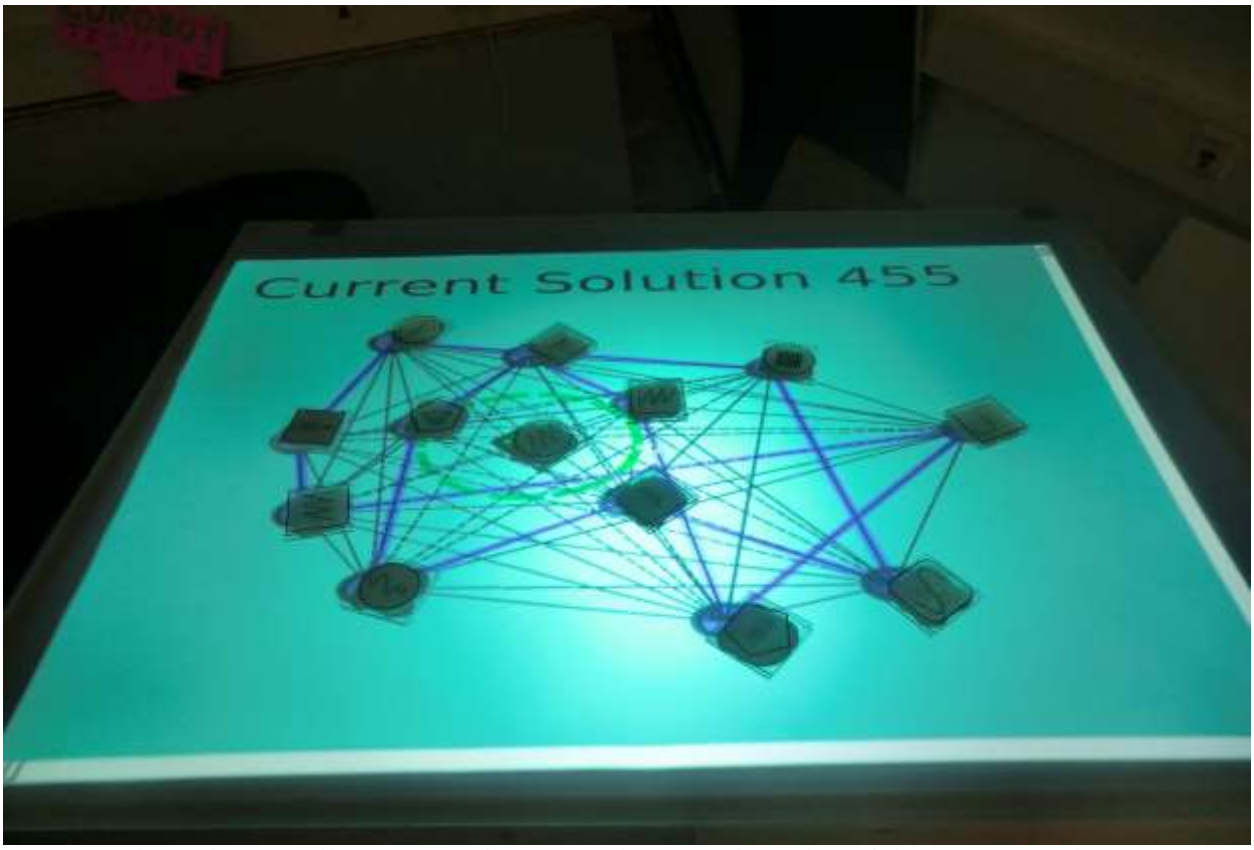


Figure: Algorithm is still running. The text is showing the value of the Current Solution. Notice the blue screen.

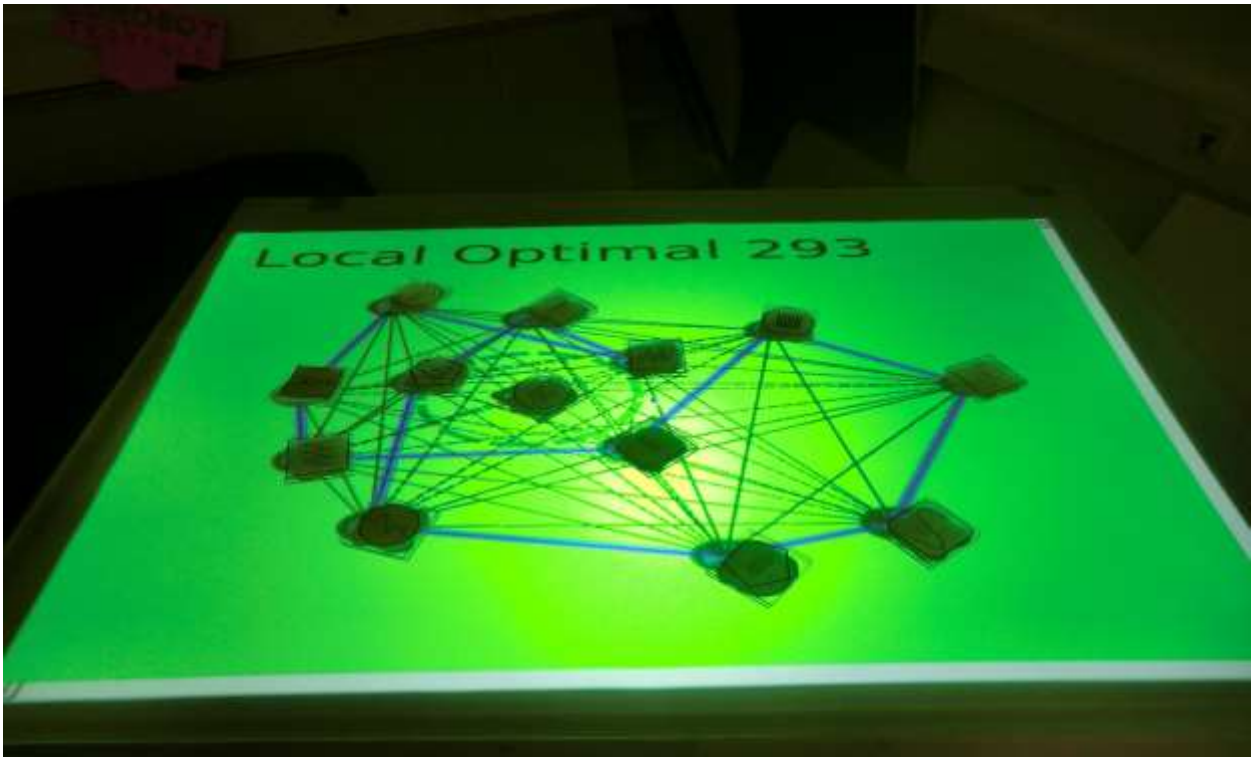


Figure: Algorithm has found a solution.

2.7. Videos

Trying with nodes/normal objects only. Also, graph modifications:
https://youtu.be/8AElEcP6RRI

Introducing the special objects:
https://youtu.be/77iGmgfHRLg

More special object:
https://youtu.be/818DeMoJ6jc

Deactivating edges:
https://youtu.be/h5u5ggaJFrQ

Re-activating edges:
https://youtu.be/WjZQ3Wd78zQ

Removing blocks:
https://youtu.be/MhEQ3EK-NF0

No solution found:
https://youtu.be/mh7j5G9A8_0

3. Code

The code can be found at:
https://github.com/manjola/TSP_visualize_on_reactable/tree/master/Code

In fact, the entire project can be found at:
https://github.com/manjola/TSP_visualize_on_reactable/

4. Considerations

When using the reactable, there is some noise that can cause the display to sometimes be slightly off the blocks and fingers. This is a point for future improvement.

5. News

The tool I built will be used as a pedagogical tool by the university when teaching algorithms. It will also be used at university open nights as a way to attract the public (especially kids) to research and computer science.

6. Contributions

Here I present my contributions to the project:

1. Created a tool for visualizing algorithms (Travelling Salesman Problem, etc.) in a fun, dynamic and visually pleasing manner, on both a tangible user interface reactable machine and personal computers.
2. Extended TSP to a represent a simplified version of a real world scenario: travelling between a region of cities, where some roads experience traffic jams and constructions.

3. Extended the use of a reactable to algorithm visualization, and traffic jam and road construction simulations.
4. Presented an example of the tool running on a reactable and computer, using a 2-opt TSP algorithm.
5. Created a pedagogical tool for visualizing and teaching algorithms.
6. Created a tool/demonstration to attract people (especially kids) to research and computer science.

Acknowledgment
This project was done as part of the Post-Bachelor program 2013 [4], a project summer school of HGS MathComp.

The program is targeted to students in their last year or immediately after their bachelor studies. In scientific projects participants get first-hand experience on doing own research in one of the key labs of the graduate school (applied mathematics, computer science, computational physics, systems biology ...). This serves as an introduction to a possible entry into the postgraduate and PhD programs at Heidelberg University.

I would like to thank you the people in charge of the program for considering my application, and the fact that I had enough technical and research background to participate, even though I was not yet a senior at the time.

References
[1] Reactable information: https://en.wikipedia.org/wiki/Reactable

[2] Reactable at Heidelberg University:
http://joanna.iwr.uni-heidelberg.de/projects/HOLODECK/projekt.html

[3] TUIO software: http://www.tuio.org/?software

[4] Post-Bachelor Program, HGS MathComp:
http://www.mathcomp.uni-heidelberg.de/programs/post-bachelor-program/