The Graph Burning Problem

Degree programme: BSc in Computer Science | Specialisation: Computer Perception and Virtual Reality
Thesis advisor: Prof. Dr. Erich Baur
Expert: Han van der Kleij (SBB)

During the corona pandemic one of the crucial questions was and is, how fast can the virus spread after an initial infection in a group of humans. To answer this question, it is necessary to analyse the contacts of a infected person. Those contacts could be with friends, with family members or with random people. If a person gets infected after such a contact, he becomes a virus spreader himself. The ‚Graph Burning Problem' is a mathematical model for this scenario.

In this work we investigate the ‚Graph Burning Problem'. We explain this mathematical problem in depth, its relevancy, approaches to solve it and unsolved conjectures. We implement a visualization tool to show graphs in general and to illustrate the workings of some known heuristic for this problem. Further, we investigate if the known heuristic can be optimized and if a new faster heuristic can be found.

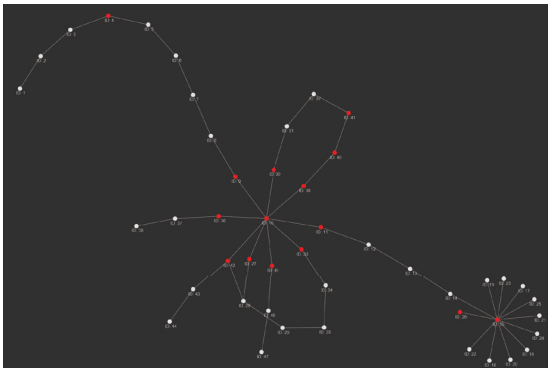
Explanation:
Graph burning is a discrete-time process in which nodes of a graph are burned one after the other. More specifically, initially, all vertices are unburned. Then, in each round, according to some strategy an unburned node is chosen to burn (if available). Once a vertex burns, each of its unburned neighbours becomes burned in the next round. The process ends if all vertices are burned.

The burning process highly depends on the strategy of choosing a node to burn. The goal is to burn all vertices of the graph in as few steps as possible.

Approach:
First, we implement a visualization tool. Then we implement known heuristics for this problem. We import a set of graphs from public graph repositories (SNAP and Network Data Repositories). We run the implemented algorithm on the predefined graph set and measure the performance for each heuristic. Further, we improve the heuristics and optimize their runtime. Lastly, we implement an own heuristic and made a performance comparison.

Raphael Klembowski

Results:
By making some assumption about the optimal burning number, a simple 3-approximation algorithm allows us to significantly reduce the runtime for the known heuristics. We visualize our results with tables and diagrams. We finally propose an optimized heuristic which performs optimal on each tested graph.



A graph after three timesteps. The fire sources are: (39, 26, 4).

Benchmark results after solution space reduction

Graphname	|V|	|E|	Number of runs BBGH	New Longest run BBGH	New Whole run BBGH
Netscience	379	914	2	9ms	10ms
Polblogs	643	2'281	3	18ms	27ms
Reed98	962	18'813	2	40ms	52ms
Mahindas	1'258	7'683	3	132ms	160ms
Chameleon	2'277	36'101	3	321ms	388ms
TVShow	3'892	17'262	4	872ms	1s 282ms
Ego-Facebook	4'039	88'234	2	1s 364ms	1s 481ms
Bitcoin	5'881	35'592	3	1s 670ms	1s 906ms
Squirrel	6'147	5'201	3	1s 793ms	2s 145ms
LastFM	7'624	27'806	4	3s 862ms	4s 843ms
Cite-DBLP	12'592	49'744	6	10s 214ms	14s 754ms
p2p	36'682	88'329	4	1min 26s 983ms	1min 40s 591ms
Gemsec-Deezer (HR)	54'573	498'202	3	5min 28s 294ms	6min 6s 53ms

Results after reducing the solution space.