Topics for licență final projects – Simon Cowell – Fall 2020

1) Topics in geometric group theory

In this project you will study topics of your own choosing from the 2017 book "Office Hours with a Geometric Group Theorist":

https://play.google.com/store/books/details?id=UV3yDQAAQBAJ

and explain them in your own words, with solved exercises, illustrations, or whatever you see fit to include. The contents of the book are as follows:

Groups; Metric spaces; Groups acting on trees; Free groups and folding; The ping-pong lemma; Automorphisms of free groups; Quasi-isometries; Dehn functions; Hyperbolic groups; Ends of groups; Asymptotic dimension; Growth of groups; Coxeter groups; Right-angled Artin groups; Lamplighter groups; Thompson's group; Mapping class groups; Braids.

2) Implementing the Agrawal-Kayal-Saxena (AKS) polynomial-time primality testing algorithm:

In 2002, the authors published the first ever deterministic algorithm of polynomial time complexity for determining whether an arbitrary given number is prime, whose proof of correctness does not depend on the generalised Riemann hypothesis, nor on any other mathematical conjecture. This answered, in the affirmative, the long-standing question of whether the primality decision problem was in the class "P":

https://people.csail.mit.edu/vinodv/COURSES/MAT302-S13/AKSpaper.pdf

In this project you will:

a) Implement the AKS algorithm in code.
b) Carry out empirical measurements of the time complexity of your code and plot the results.
c) Discuss the history of the problem, and the authors' analysis of their solution.

3) Implementing an anaglyph stereoscopic display with head-coupled perspective:

In 2007 researcher Johnny Lee at Carnegie Mellon University hacked a Wii remote and sensor bar to make a head-coupled perspective display:

http://www.johnnylee.net/projects/wii/

which produced a striking monocular illusion of depth. The idea to combine head-coupled perspective with an anaglyph stereoscopic display (with e.g. red and cyan glasses) is at least as early as Arthur's 1993 Master's thesis:

https://open.library.ubc.ca/cIRcle/collections/ubctheses/831/items/1.0051425

in which a mechanical head-tracking system was used.

In this project you will implement a head-coupled perspective display using the webcam of a laptop (i.e. without any extra hardware) to track the position and orientation of a pair of anaglyph glasses worn by the user. You will combine this with anaglyph stereoscopic 3D graphics which you will program, for example, using the OpenGL or DirectX graphics libraries.

4) Using a Random Walk to Analyse a Probabilistic Algorithm for the Boolean 2-Satisfiability Problem:

“2-SAT” is the problem of deciding whether there exists an assignment of True / False values to the variables in an expression such as

$$(x_1 \vee \bar{x}_2) \wedge (x_1 \vee x_3) \wedge (x_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee \bar{x}_2) \wedge (x_1 \vee x_2)$$

that makes the expression itself True. There are deterministic algorithms that solve this problem efficiently; in fact, in a time that increases only linearly with the number of variables. For example, the Aspvall-Plass-Tarjan algorithm:

http://www.math.ucsd.edu/~sbuss/CourseWeb/Math268_2007WS/2SAT.pdf

In this project, however, you will:

a) Implement the probabilistic algorithm described in Section 4.5.3 of Ross’s “Introduction to Probability Models, 10th Ed.” that either finds a satisfying assignment of values or else determines that, with high probability, there is no such assignment, and which, in a certain precise probabilistic sense, is of quadratic time complexity in the number of variables.
b) Analyse the (probabilistic) complexity of the probabilistic algorithm using a Markov chain.
c) Carry out empirical tests of your code on large random instances of the problem and plot your results.
d) Demonstrate empirically that this algorithm is no longer efficient when applied to the general “SAT” problem (which is NP-complete), in which the parentheses in the expression may contain more than 2 “literals” and explain this phenomenon analytically.
e) **(Optional)** Implement the Aspvall-Plass-Tarjan algorithm and compare its performance empirically with that of the probabilistic algorithm.

5) Implementing the “Fast Loaded Dice Roller” (FLDR) algorithm of Saad, Freer, Rinard and Mansingkha and discussing it:

In 2020 an algorithm was published that promises to greatly improve the generation of pseudorandom numbers:

https://arxiv.org/abs/2003.03830v2

“In 1976, computer scientists Donald Knuth and Andrew Yao introduced the fastest algorithm ever to simulate the roll of weighted dice in the most time-efficient manner. While efficient, the Knuth-Yao algorithm has a major drawback — it requires too much computer memory to store the information, making the method impractical for many real-world problems.

FLDR was designed to have the efficiency of the Knuth-Yao algorithm, using a fraction of the memory. Nearly as time efficient as the Knuth-Yao method, FLDR uses up to 10,000 times less memory storage while taking no more than 1.5 times longer per operation.”

From “How Rolling Loaded Dice Will Change the Future of AI”:

https://www.intel.com/content/www/us/en/research/blogs/mit-fast-loaded-dice-roller.html

In this project you will:

a) Implement the FLDR in code.
b) Carry out empirical measurements of the space and time complexities of your code and plot the results.
c) Discuss the author’s theoretical analysis of the complexity of their algorithm and compare it with your empirical results.