Randomized algorithms

Polynomial identity testing

- What is the polynomial identity testing problem? Why can it be reduced to deciding whether a given polynomial is the zero polynomial?
- Assume black-box access to a polynomial p of degree at most n:
  You can choose a query point s and get back p(s) as the result of your query.
    - How many points do you have to query in the worst-case in order to find out whether p is the zero polynomial?
    - How can you recover the coefficients of p in this model?
- How does randomness help in the polynomial identity testing problem? In particular, which success probability can you achieve using randomness?

Freivalds' algorithm

- Describe Freivalds' algorithm. Make sure to include the following:
    - What is the purpose of Freivalds' algorithm?
    - Why is it faster than computing matrix products?
    - Describe the random choices made by the algorithm. Depending on the random choices, when does the algorithm succeed and when does it fail?
    - What is the failure probability of the algorithm? Use probability theory to give an argument for your answer.

Karger's algorithm:

- What is a global min-cut in a graph?
- Describe Karger's algorithm to find a global min-cut in a graph.
- What is the failure probability of Karger's algorithm?
  Sketch an analysis using probability theory and highlight where exactly properties of an assumed min-cut are used in the analysis.
- Why does the success probability analysis of Karger's algorithm imply that any graph has at most $\binom{n}{2}$ global min-cuts?

The Monte Carlo method:

- Sketch how you can determine the number π by throwing darts.
- What does it mean for a randomized algorithm to (ε,δ)-approximate a value?
- How many samples are sufficient to (ε,δ)-approximate the mean μ of an indicator random variable X? What is the name of the bound used to prove this? (You don't need to prove this bound, but have a rough idea how the proof works.)
- Consider the following algorithm: To determine the number of satisfying assignments for a Boolean formula F with n variables, randomly sample some number T of

assignments. Among those sampled assignments, let S be the number of assignments that satisfy F. Then output S/T * 2^n.

  - Is the expected output of this algorithm correct?
  - What other problem is there?
- Describe the randomized approximation algorithm for counting satisfying assignments to DNFs discussed in class.
  - What exactly is the set it samples from?
  - Which condition does it check on such a sample?
  - How does this algorithm solve the problem encountered by the algorithm in the previous bullet point?

Primality testing

- What is a strong probable prime?
- What is the repeated squaring algorithm? What is its running time?
- Describe one algorithm that efficiently checks whether an integer is a prime or not w.h.p.

Algebraic algorithms:

- Roughly, what does the DeMillo-Lipton-Schwartz-Zippel lemma say?
- What is the difference between the two mathematical concepts ring and field?
- Name one combinatorial problem matrix determinants can be used to solve and describe roughly how it works.
- How computationally hard is it to compute an nxn symbolic matrix determinant with polynomials in several variables as entries with total degree bounded by O(n)? How computationally hard is it to compute an nxn numeric matrix determinant with elements from a finite field as entries?
- Describe one way of computing a numeric determinant.

Markov chains:

- What is a Markov chain?
- What is a stationary solution to a Markov chain?
- What is the cover time of a random walk on a graph?
- Show a tight asymptotic upper bound on the cover time of a random walk on an undirected non-bipartite connected graph.
- Show an example of a strongly connected directed graph in which the cover time is exponential.
- Describe how the algorithm for 2SAT can be analyzed as a random walk on a graph. What is the graph and what is its cover time?

Divide/Conquer and FFT

Divide & Conquer and the Master theorem

- Give the statement of the Master theorem for recurrence relations of the form $T(n) \leq aT(n/b) + O(n^c)$. Explain which cases can occur depending on a,b,c.
- Sketch a proof for one of the cases.

Integer Multiplication

- What is Karatsuba's algorithm? What is its running time?

~~Fast Fourier transform~~

- ~~What is the input and the output of the Fast Fourier Transform and its inverse?~~
  - ~~in terms of polynomials~~
  - ~~in terms of matrices~~
- ~~Describe the FFT algorithm. In particular, explain the following:~~
  - ~~When viewing FFT in terms of polynomials, what happens to the polynomial in the recursive step?~~
  - ~~Which recurrence relation captures the recursive step of FFT? Solve the relation.~~
  - ~~Which properties of roots of unity are relevant for FFT? In particular, why would FFT not work when the input polynomial is evaluated at arbitrary points?~~
- ~~How can you use FFT to multiply two polynomials?~~

~~K-SUM~~

- ~~What is the k-sum problem? Compare it to the Subset Sum problem. In particular, clarify which of these problems are NP-complete.~~
- ~~Describe the meet-in-the-middle approach for solving the k-sum problem on general real-valued inputs. What is the running time and why does it work?~~
- ~~How can you use FFT to solve the k-sum problem? Which requirements are needed for this to work with O(n log n) arithmetic operations? Would this approach also work for Subset Sum?~~

Algorithms based on matrix multiplication

Matrix multiplication

- Define the matrix multiplication problem. How many arithmetic operations do you need to solve this problem in the standard way?
- Describe a way to reduce $n \times n$ matrix multiplication recursively to 8 multiplications of $n/2 \times n/2$ matrices.
- How many recursive multiplications does Strassen's algorithm require, and what running time does this translate to in the end? Argue for the running time bound.
- Define the constant ω. What does the existence of Strassen's algorithm imply for ω? What are the best known lower and upper bounds on ω?

Matrix multiplication for cliques

- How can matrix multiplication be used for detecting and counting triangles?
- How can the algorithm for triangles be extended to detect cliques of a given size k? How does this compare to the brute-force algorithm?
- Can you use the clique algorithm to find/count independent sets?

Exponential-time algorithms

ETH and SETH

- Explain the statement of the exponential-time hypothesis ETH and the strong exponential-time hypothesis SETH. Consider the following:
    - Why do they imply P != NP?
    - Is an algorithm with running time $O(1.0001^n)$ for 3-SAT possible under ETH?
    - Is an algorithm with running time $O(1001^{n/log\, n})$ for 3-SAT possible under ETH?
    - Are either of these algorithms possible under SETH?
    - Briefly give two examples for concrete super-polynomial running times for 3-SAT that are ruled out under ETH.

- What is the sparsification lemma, and why is it useful?

- Describe the Orthogonal Vectors problem.
    - Which lower bound does SETH yield for this problem? Sketch the lower bound.
    - Is an $O(n^2/\log n)$ algorithm or an $O(n^2/\sqrt{n})$ time algorithm possible under SETH?

- Describe the k-Dominating Set problem.
    - Which lower bound does SETH yield for this problem? Sketch the lower bound.

Inclusion/Exclusion

- State and explain the inclusion-exclusion principle.

- Describe how to solve the Hamiltonian cycle problem using inclusion-exclusion.
    - Describe the role of walks and paths.
    - How can you count walks in polynomial time?
    - Why do you need to count them, and why is deciding existence not enough?
    - Briefly compare this approach to the Bellman-Held-Karp dynamic programming algorithm for Hamiltonian cycles. Also outline the DP algorithm.

- Describe how to count perfect matchings in a graph using inclusion-exclusion. Which structures are counted in the individual terms of the inclusion-exclusion formula? How can you speed this up on bipartite graphs, and what are the structures counted there?

- Consider an n-vertex graph G.
    - How is the chromatic number of G defined?
    - How can you compute the chromatic number in $O(3^n)$ time? The approach we've studied reduces the problem to counting certain structures. What are these structures and how fast can you count them?

Graph Structure

Chordal graphs

- What is a chordal graph and what is an interval graph?
- Show that every interval graph is a chordal graph.
- Show that a graph is chordal if and only if it has a perfect elimination ordering.
- Give a polynomial time algorithm to solve the Graph Coloring problem on chordal graphs. Argue for the correctness of the algorithm. What is the running time?
- What is a clique tree for a chordal graph? Show how to construct a clique tree for a chordal graph in polynomial time.

Solving problems on chordal graphs

- Describe how to solve the Maximum Independent Set problem on trees. How can you solve it on forests?
- What is a nice clique tree?
- Describe the Feedback Vertex Set problem. Give a polynomial time algorithm to solve this problem on chordal graphs using a nice clique tree. What is the running time of the algorithm?

Parameterized algorithms

Basics and Branching

- What is a parameterized problem? What is an FPT algorithm? For which parameter choices does an FPT algorithm run in polynomial time?
- Describe how to solve the Vertex Cover problem in $O(2^k n)$ running time, where k is the solution size.
- How can you improve the running time of the above algorithm?
- How can we solve Independent Set on 3-regular graphs in time $O(4^k n)$?
- Define a tournament. How can we solve Feedback Vertex Set on Tournaments in time $O(3^k n^3)$?

Kernelization

- What is a kernel for a parameterized problem?
- Argue that a parameterized problem has a kernel if and only if it is FPT
- What is a polynomial kernel?
- Give a kernel for Vertex Cover parameterized by the solution size k with $O(k^2)$ vertices
- Define the Edge Clique Cover problem. Give a kernel for Edge Clique Cover parameterized by the solution size k. Is this a polynomial kernel?

Treewidth

- Define the treewidth of a graph. Give examples for graph families with small and large treewidth (without proofs). Sketch the tree-decomposition for a grid.
- What is a nice tree-decomposition? Why is it useful?
- Describe how to solve the Maximum Independent Set problem on graphs of bounded treewidth.
  - What running time do you achieve?
  - Describe what information you store and how it is computed at the nodes of the tree-decomposition from previously computed information.
  - How to use this to conclude Vertex Cover is FPT parameterized by treewidth?
- Describe how to solve the 3-coloring problem on graphs of bounded treewidth.
- Describe how to solve the MaxCut problem on graphs of bounded treewidth.
- Describe the idea of a win-win approach to design parameterized algorithms.
- What does it mean that H is a *minor* of G? What does the Excluded Grid Minor Theorem say? How is it helpful to the win-win approach?
- What does the Planar Excluded Grid Minor Theorem say? How is it helpful to design subexponential parameterized algorithms on planar graphs? (Recall, in this case subexponential algorithms are those with running time $2^{O(\sqrt{k})} n^{O(1)}$ where k is the solution size.
- How can we solve Vertex Cover and k-Path in time $2^{O(\sqrt{k})} n$ on planar graphs?

Color-coding

- Describe the “color-coding” algorithm for deciding whether a graph has a path of length k. Make sure to explain the role of random colorings and walks. What changes if you want to decide the existence of cycles rather than paths?
- How can you determine the existence of a colorful k-path in a vertex-colored graph in time $(k + 1)! \cdot n^{O(1)}$ or $2^k n^{O(1)}$? (For the second running time, modify the DP for Hamiltonian paths.)

Approximation algorithms

Formalizing approximation problems

- What is an NP-optimization problem?
- What is the definition of approximation ratio?
- What is a PTAS and what is an FPTAS? Which one is more desirable in theory? Give examples for running times that match the definitions of PTAS and FPTAS algorithms.

Approximating vertex-cover

Describe the matching-based algorithm for finding a vertex-cover in a graph that is at most twice as large as the minimum vertex-cover. Include the following in your description:

- What does the existence of a k-edge matching imply for the minimum size of a vertex-cover?
- How does it matter whether you consider maximum matchings or maximal matchings in the algorithm? Which is the better choice?
- Give an infinite family of example graphs on which the algorithm indeed outputs only a 2-approximate solution. Also give a family of example graphs on which the algorithm outputs an optimal solution.

Coloring 3-colorable graphs

- Given a 3-colorable n-vertex graph, show how you can in polynomial time find a proper O(sqrt(n))-coloring of the graph.

Metric problems and TSP

- What does it mean that a graph has metric costs/weights?
- What is the Steiner Tree problem?
    - How can it be reduced to a version in which the input graph is required to have metric costs?
    - Give a 2-approximation for Metric Steiner Tree.
- Define the TSP and argue that it is NP-complete.
- Show that TSP remains NP-complete on metric instances.
- How do the general TSP and the metric TSP differ in terms of approximability? Argue that the general TSP cannot be approximated within a constant factor unless P=NP.
- Describe how to obtain a 2-approximation for the metric TSP. Highlight where you use the metric property.
- Describe how to obtain a (3/2)-approximation for the metric TSP.

Semidefinite Programming

- What is a semidefinite program? How fast can we find a solution to a semidefinite program in n variables?

- For a positive semidefinite matrix M, what is the name of the algorithm that computes a triangular matrix L so that LL^T=M? What is its running time?
- Name two combinatorial problems that can be approximated with the help of semidefinite programming.
- How do you get from the vectors obtained from a solution to the semidefinite program to a solution to the combinatorial problem?