Outline

1. What is Fine-grained Complexity?
   - Ingredients of a Complexity Theory
   - NP Theory

2. Problems, Instances and Complexity Parameters
   - 3-SUM Problem
   - Orthogonal Vectors Problem
   - Small Universe Hitting Set Problem
   - Edit Distance Problem
   - Maximum Length Chain of Subsets
   - All-Pairs Shortest Paths
   - Relationships among Problems I

3. Polynomial Method for Designing Algorithms

4. NP-complete Problems
   - Satisfiability Problem
   - NP-complete Graph Problems
   - Relationships among Problems II

5. Examples of Fine-grained Reductions
What is fine-grained complexity?

- Theory and techniques to reason about
  - exact worst-case complexities of deterministic or randomized algorithms that output exact solutions and
  - complexity relationships among them.
- What improvements can we expect over exhaustive search or standard algorithms?
- What are the obstructions that limit improvements?
- What principles explain the exact complexities of problems?
Ingredients of a Complexity Theory

- Problems and classes of problems
- Algorithms and design techniques
- Reductions among problems
- Hard and complete problems
- Conjectures
- (Conditional) Lower Bounds
- Interplay between upper and lower bounds
**NP Theory**

- **Problems**: Satisfiability, Max Independent Set, Hamiltonian Path, Colorability, Clique, Factoring, Graph Isomorphism, Primality, ...  
- **Classes**: P, NP, coNP, L, ...  
- **Notions of reductions**: Polynomial time reductions  
- **Complexity relationships**: The following problems (and many others) are polynomially equivalent. 
  - k-sat for $k \geq 3$, Colorability, Vertex Cover, Independent Set, Clique, ...  
- **Completeness**: 3-sat is complete for NP.  
- **Complexity conjecture**: $P \neq NP$.  
- **Conditional lower bounds**: None of the problems have a polynomial time algorithm (under the conjecture $P \neq NP$).
Fine-grained Complexity

- Shares almost some of the characteristics of the \textbf{NP}-theory
- \textbf{Problem-centric} rather than \textbf{complexity class-centric}.
- Strvies to determine the complexity as \textbf{exactly} as possible.
A **problem** is a set $\mathcal{I}$ of instances and a mapping to $\{0, 1\}^*$. Each instance comes with a **size parameter** and one or more **complexity parameters**. Parameters are mappings from instances to integers. Size parameter is a measure of the description length of an instance. Complexity of a problem (or an algorithm solving the problem) is expressed in terms of size and complexity parameters.
3-SUM Problem

3-SUM: Given a sequence of integers $x_1, x_2, \ldots, x_n$ where $x_i \in [0, 1, \ldots, d - 1]$, do there exist $i, j$ and $k$ such that $x_i + x_j = x_k$?

- Complexity parameters: $n$ and $d$
- Straightforward algorithm solves it in time $O(n^2 \log d)$.
- $O(n^2 \log \log^2 d / \log^2 d)$ algorithm by Baran, Demaine, Pătrașcu, 2005
**Orthogonal Vectors Problem**

**Orthogonal Vectors:** Given a sequence $A_1, \ldots, A_n$ of sets with elements from a universe of size $d$, do there exist $i \neq j$ such that $A_i \cap A_j = \emptyset$.

If the sets are thought of as characteristic vectors in $\{0, 1\}^d$, $A_i \cap A_j = \emptyset$ is equivalent to the proposition that the vectors $A_i$ and $A_j$ are orthogonal.

- Complexity parameters: $n$ and $d$
- Straightforward algorithm solves it in time $O(n^2 \log d)$.
- $O(n^2 - \frac{1}{O(\log c)})$ algorithm where $d = c \log n$ by Abboud and Williams, Yu (2015), Chan and Williams (2016).

**Orthogonal Vectors (Bipartite version):** Given two sequences $A_1, \ldots, A_n$ and $B_1, \ldots, B_n$ of sets with elements from a universe of size $d$, do there exist $i$ and $j$ such that $A_i \cap B_j = \emptyset$. 

---

**Introduction**
**Hitting Set**: Given two sequences $A_1, \ldots, A_n$ and $B_1, \ldots, B_n$ of sets with elements from a universe of size $d$, does there exist an $i$ such that for all $j A_i \cap B_j \neq \emptyset$.

- Complexity parameters: $n$ and $d$.
- Straightforward algorithm solves it in time $O(n^2 \log d)$.
- $O(n^{2 - \frac{1}{O(\log c)}})$ algorithm where $d = c \log n$ by Gao, Impagliazzo, Kolokolova and Williams (2017).
**Edit Distance Problem**

**Edit Distance**: Given two strings $x$ and $y$ over a fixed alphabet $\Sigma$, determine the edit distance between them, that is, the minimum number of insertions, deletions, and substitutions required to transform $x$ to $y$.

- Complexity parameters: $n = |x|$ and $m = |y|$.
- Standard dynamic programming algorithm has complexity $O(nm)$.
- ‘Four Russians Algorithm’: $n^2 / \log^2 n$, Masek and Paterson (1980) (when $n = m$)
**Maximum Length Chain of Subsets**: Given a sequence $A_1, \ldots, A_n$ of sets with elements from a universe of size $d$, find a maximum length subsequence $i_1 < i_2 < \cdots < i_k$ such that $A_{i_j} \subseteq A_{i_{j+1}}$ for $1 \leq j \leq k - 1$.

- **Complexity parameters**: $n$ and $d$
- Straightforward one-dimensional dynamic programming algorithm solves it in time $O(n^2 \log d)$.
- A better algorithm with complexity $n^2 - \frac{1}{O(\log c)}$ for $d = c \log n$, Künemann, Paturi and Schneider (2017) by reduction to Orthogonal Vectors problem.
**All-Pairs Shortest Paths Problem**

**All-Pairs Shortest Paths**: Given a undirected (or directed) graph $G = (V, E)$ with integer edge weights, determine the shortest path distances between every pair of vertices.

- Complexity parameters: the number of vertices: $n = |V|$, the number of edges: $m = |E|$, the range of edge weights: $d$
- Floyd-Warshall’s algorithm solves it in time $O(n^3 \log d)$.
- $O(n^3 \log d/2^{\Theta(\sqrt{\log n})})$ algorithm, Williams (2014).
- Is there an $\varepsilon > 0$ such that *All-Pairs Shortest Paths* problem can be computed in time $n^{3-\varepsilon}$ for $d = O(\log n)$?
Relationships among Problems

Problems: 3-sum, Orthogonal Vectors, Hitting Set, Maximum Length Chain of Subsets, and Edit Distance

- Is there an \( \varepsilon > 0 \) such that there is an \( n^{2-\varepsilon} \) algorithm for any of these problems?
- If the Orthogonal Vectors problem can be solved in time \( n^{2-\varepsilon} \) for some \( \varepsilon > 0 \), can one solve the Hitting Set problem in time \( n^{2-\delta} \) for some \( \delta > 0 \)?
- More generally, what 'fine-grained' reductions are possible among these problems?
  - Assume that problem \( A \) has a conjectured complexity \( T_A(n) \) and problem \( B \) \( T_B(n) \).
  - Assume that the complexity of \( A \) improved to \( T_A^{(1-\varepsilon)}(n) \) for \( \varepsilon > 0 \).
  - Can we infer if there will be an improvement in the complexity of \( B \)?
  - Reductions from \( B \) to \( A \) that enable the transfer of the improvement are fine-grained reductions.
What is Fine-grained Complexity? Problems, Instances and Complexity

Polynomial Method

Theorem (Abboud, Williams, and Yu, 2015)

For vectors of dimension $d = c \log n$, the bipartite Orthogonal Vectors problems can solved in $n^{2 - \frac{1}{O(\log c)}}$ time by a randomized algorithm that is correct with high probability.

Sketch:

- Find a suitable meta problem which has an improved algorithm over exhaustive search/evaluation
- Reduce the problem to the meta problem.
- Optimize the parameters.
Efficient Polynomial Evaluation on a Rectangle of Inputs

**Lemma (Williams 2014)**

Given a polynomial $P(x_1, \ldots, x_d, y_1, \ldots, y_d)$ over $\mathbb{F}_2$ with at most $n^{0.1}$ monomials and two sets of $n$ inputs $A = \{a_1, \ldots, a_n\} \subseteq \{0, 1\}^d$ and $B = \{b_1, \ldots, b_n\} \subseteq \{0, 1\}^d$, we can evaluate $P$ on all pairs $(a_i, b_j) \in A \times B$ in $O(n^2 \text{poly}(\log n))$ time.

**Proof:**
Circuit Satisfiability

**Circuit Sat**: Given a circuit $C$ from a circuit model (specified by certain parameters) on $n$ input variables $x = (x_1, \ldots, x_n)$, is there an assignment to the variables for which the circuit evaluates to 1?

- **Size** of the circuit is its description length.
- **Complexity parameter**: $n$, number of variables
- Satisfiability problem for a circuit $C$ can be solved in time $\text{size}(C)2^n$ by exhaustive search.
- Improvements are expressed as $\text{size}(C)2^{(1-\mu)n}$ where $\mu > 0$ is the savings over exhaustive search.
- Other complexity parameters: Treewidth
A Variety of Satisfiability Problems

- **$k$-SAT** for $k \geq 3$: Conjunction of disjunctions of literals where each disjunction contains at most $k$ literals.

- **CNF-SAT**: Conjunction of disjunctions of literals

- Number of clauses $m$ may also be useful as a complexity parameter for $k$-SAT and CNF-SAT.

- **FORMULA SATISFIABILITY**: Formula $F$ over the basis $\{\lor, \land, \neg\}$. 

NP-complete Graph Problems

- **Hamiltonian Path**: Given a graph \( G = (V, E) \), is there a hamiltonian path?
- **\( k \)-Colorability**: Given a graph \( G = (V, E) \), is \( G \) colorable with \( k \) or fewer colors?
- **Colorability**: Given a graph \( G = (V, E) \) and an integer \( k \), is \( G \) colorable with \( k \) or fewer colors?
- **Max Independent Set**: Given a graph \( G = (V, E) \) and an integer \( k \), does \( G \) have an independent set of size at least \( k \)?
- Complexity parameters: the number of vertices: \( n = |V| \), the number of edges: \( m = |E| \), the range of edge weights: \( d \)
- Potentially \( \log n! \), the size of the search space in the case of Hamiltonian Path.
Relationships among Problems

- All the previous problems have exponential-time algorithms, some have better exponential-time algorithms compared to exhaustive search.
- Which \textbf{NP}-complete problems have improved exponential-time algorithms?
- Is 3-\textsc{SAT} solvable in \textit{subexponential time}? How about 3-\textsc{Colorability}?
- If 3-\textsc{SAT} is solvable in subexponential time, is 4-\textsc{SAT} solvable in subexponential time?
- \textbf{k-SAT}: What is the exponential complexity of \textit{k-SAT} as \( k \rightarrow \infty \)?
- Is there an \( \varepsilon > 0 \) such that there is a \( 2^{(1-\varepsilon)n} \) algorithm for 5-\textsc{Colorability} or \textsc{Cnf-Sat} or \textit{k-sat} for all \( k \)?
Examples of Fine-grained Reductions

- If Orthogonal Vectors problem can be solved in time $n^{2-\varepsilon}$ for some $\varepsilon > 0$ for $d = \omega(\log n)$, there exists $\delta > 0$ such that $k$-SAT can be solved in time $2^{(1-\delta)n}$ for all $k$.

- If 3-SAT has a subexponential-time algorithm, then so does 4-SAT.
Reducing \textit{k-sat} to \textbf{Orthogonal Vectors}

- Assume that the \textbf{Orthogonal Vectors} problem can be solved in time $n^{2-\varepsilon}$ for $\varepsilon > 0$ for $d = \omega(\log n)$.
- Let $\varepsilon' = \varepsilon/3$ and $\phi$ be a k-CNF with $n$ variables for $k > 0$.
- Sparsify $\phi$ in $2^{\varepsilon'n}$ time into $2^{\varepsilon'n}$ many $k$-\textsc{sat} instances $\phi_i$ with at most $c_{\varepsilon'}n$ many clauses.
- For each $\phi_i$, construct two families $L$ and $R$ of sets which are subsets of a universe of size $c_{\varepsilon'}n$ where $|L| = |R| = N = 2^{n/2}$.
- $\phi_i$ is satisfiable if and only if there is a pair of sets $A \in L$ and $B \in R$ such that $A \cap B = \emptyset$.
- Total time for solving the satisfiability of $\phi$ is $2^{\varepsilon'n} + N^{2-\varepsilon}2^{\varepsilon'n} \approx 2^{(1-\varepsilon/6)n}$
- Since $k$ is arbitrary, this implies that \textbf{SETH} is false.
- There is no $\varepsilon > 0$ such that \textbf{Orthogonal Vectors} problem can be solved in time $n^{2-\varepsilon}$ for a universe of size $\omega(\log n)$. 

Reducing 4-SAT to 3-SAT under Subexponential-time Reductions

- Let $\varepsilon > 0$ be arbitrary.
- Apply Sparsification Lemma to the given 4-CNF $\phi$ to obtain a disjunction of $2^{\varepsilon n} \phi_i$ in time $2^{\varepsilon n}$ where each $\phi_i$ has linearly many clauses.
- Reduce 4-CNF $\phi_i$ to a 3-SAT formula with only linearly many new variables.
  \[
  (l_1 \lor l_2 \lor l_3 \lor l_4) = \exists y (l_1 \lor l_2 \lor y)(\bar{y} \lor l_3 \lor l_4)
  \]
- Now, a subexponential time algorithm for 3-SAT implies a subexponential time algorithm for $\phi_i$.
- Since $\varepsilon > 0$ is arbitrary, $\phi$ can be solved in subexponential-time.
Fine-grained Reductions

Definition (Fine-Grained Reductions ($\leq_{FGR}$))

Let $L_1$ and $L_2$ be languages, and let $T_1$ and $T_2$ be time bounds. We say that $(L_1, T_1)$ fine-grained reduces to $(L_2, T_2)$ (denoted $(L_1, T_1) \leq_{FGR} (L_2, T_2)$) if for all $\varepsilon > 0$, there is a $\delta > 0$ and a deterministic Turing reduction $M^{L_2}$ from $L_1$ to $L_2$ satisfying the following conditions.

(a) The time complexity of the Turing reduction without counting the oracle calls is bounded by $T_1^{1-\delta}$.

\[
\text{TIME}[M] \leq T_1^{1-\delta}
\]

(b) Let $\tilde{Q}(M, x)$ denote the set of queries made by $M$ to the oracle on an input $x$ of length $n$. The query lengths obey time bound: $\sum_{q \in \tilde{Q}(M, x)} (T_2(|q|))^{1-\epsilon} \leq (T_1(n))^{1-\delta}$.
Other Fine-grained Reductions

- **CNF-Sat** at $2^n$ is fine-grain reducible to **Edit Distance** at $n^2$.
- **Hitting Set** at $n^2$ is fine-grain reducible to **Orthogonal Vectors** at $n^2$.
- Many others
- It is **open** if **Orthogonal Vectors** (or **CNF-Sat**) is fine-grain reducible to **Hitting Set**?
- Is **Orthogonal Vectors** fine-grain reducible to **3-SUM**? **(open)**
Complexity Conjectures

- Let \( s_k = \inf\{\delta | \exists \ 2^{\delta n} \text{ algorithm for } k\text{-SAT}\} \) and \( s_\infty = \lim_{k \to \infty} s_k \).
- How does \( s_k \) behave?
- For best known algorithms, \( s_k = (1 - \frac{1}{O(k)}) \).
- **ETH** (Exponential Time Hypothesis): \( s_3 > 0 \)
- **SETH** (Strong Exponential Time Hypothesis): \( s_\infty = 1 \)
- What evidence supports ETH and SETH?
Assume that SETH holds, that is, there is no $\varepsilon > 0$ such that $k$-SAT can be solved in time $2^{(1-\varepsilon)n}$ for all $k$.

It follows that Orthogonal Vectors does not have a $n^{2-\delta}$ algorithm for $d = \omega \log n$.

What about lower bounds for NP-complete problems?

What about lower bounds for All-Pairs Shortest Paths?
Thank You