Two Big Questions:

P vs. NP and P vs. BPP

Jeff Kinne

University of Wisconsin-Madison

Indiana State University, March 26, 2010
Computational Complexity Theory

How much *time, memory space, etc.* are needed to solve problems?
Computational Complexity Theory

How much time, memory space, etc. are needed to solve problems?

- Is nondeterminism powerful?
Two Big Questions

Derandomization

Hierarchy Theorems

Computational Complexity Theory

How much time, memory space, etc. are needed to solve problems?

- Is nondeterminism powerful? $P$ vs. $NP$
Two Big Questions

Derandomization
Hierarchy Theorems

Computational Complexity Theory

How much time, memory space, etc. are needed to solve problems?

- Is nondeterminism powerful? P vs. NP
  - Conjecture: P \neq NP
Computational Complexity Theory

How much time, memory space, etc. are needed to solve problems?

- Is nondeterminism powerful? P vs. NP
  - Conjecture: P ≠ NP
  - Techniques: hierarchy theorems, others
Computational Complexity Theory

How much time, memory space, etc. are needed to solve problems?

- Is nondeterminism powerful? $P \neq NP$
  - Conjecture: $P \neq NP$
  - Techniques: hierarchy theorems, others

- Is randomness powerful?
Computational Complexity Theory

How much time, memory space, etc. are needed to solve problems?

- **Is nondeterminism powerful?** \( P \neq NP \)
  - Conjecture: \( P \neq NP \)
  - Techniques: hierarchy theorems, others

- **Is randomness powerful?** \( P \neq BPP, L \neq BPL \)
Computational Complexity Theory

How much **time, memory space, etc.** are needed to solve problems?

- **Is nondeterminism powerful?** $P \text{ vs. } NP$
  - Conjecture: $P \neq NP$
  - Techniques: hierarchy theorems, others

- **Is randomness powerful?** $P \text{ vs. } BPP, L \text{ vs. } BPL$
  - Conjecture: $P=BPP, L=BPL$
Computational Complexity Theory

How much **time, memory space, etc.** are needed to solve problems?

- **Is nondeterminism powerful?** $P$ vs. $NP$
  - Conjecture: $P \neq NP$
  - Techniques: hierarchy theorems, others

- **Is randomness powerful?** $P$ vs. $BPP$, $L$ vs. $BPL$
  - Conjecture: $P = BPP$, $L = BPL$
  - My work: derandomization, hierarchy theorems
Introducing two Big Questions
Time Complexity
Time Complexity

- **Exp Time (EXP)**
- **Poly Time (P)**

Decision Problem: yes/no questions
Time Complexity

- Poly Time (P)
  - factoring, NP-complete
  - sorting
  - shortest path

- Exp Time (EXP)
Time Complexity

- Poly Time (P)
- Exp Time (EXP)
- Sorting
- Shortest path
- ?
- Factoring, NP-complete
Time Complexity

**Decision Problem:** yes/no questions
Memory Space Complexity
Memory Space Complexity

- **Log Space (L)**
- **Poly Space (PSPACE)**
Memory Space Complexity

Amount of **working space memory** needed
Memory Space Complexity

Amount of **working space memory** needed
Memory Space Complexity

Amount of **working space memory** needed
Randomized Algorithm
Randomized Algorithm
Randomized Algorithm

Random Strings

```
01 10 10 10
11 01 10
yes
/no
01 10 10 10
```

Bounded error: Correct with probability $> 99\%$. 
Randomized Algorithm

Random Strings

good
bad

Bounded error: Correct with probability > 99%
Polynomial Identity Testing
Polynomial Identity Testing

\[ p(x) = x^3 \cdot (3x - x^2)^2 - x^4 \cdot (2x^3 + 5x) + x \cdot (4x^2 - x)^3 \]
Polynomial Identity Testing

- $p(x) = x^3 \cdot (3x - x^2)^2 - x^4 \cdot (2x^3 + 5x) + x \cdot (4x^2 - x)^3$
- **Do all terms cancel?**
Polynomial Identity Testing

- \( p(x) = x^3 \cdot (3x - x^2)^2 - x^4 \cdot (2x^3 + 5x) + x \cdot (4x^2 - x)^3 \)
- **Do all terms cancel?**
Polynomial Identity Testing

- $p(x) = x^3 \cdot (3x - x^2)^2 - x^4 \cdot (2x^3 + 5x) + x \cdot (4x^2 - x)^3$
- **Do all terms cancel?**
Multi-variate Polynomial Identity Testing
Multi-variate Polynomial Identity Testing

\[ p(x_1, x_2, x_3, x_4) = x_4^5 \cdot (x_1 - x_2 - x_3)^{30} + (x_4 + x_3 - x_1)^{15} \cdot (x_3 - x_2 + x_1)^{20} - (x_2 - x_3 + x_4)^{20} \cdot (x_2 + x_1)^{15} \]
Multi-variate Polynomial Identity Testing

- $p(x_1, x_2, x_3, x_4) = x_4^5 \cdot (x_1 - x_2 - x_3)^{30} + (x_4 + x_3 - x_1)^{15} \cdot (x_3 - x_2 + x_1)^{20} - (x_2 - x_3 + x_4)^{20} \cdot (x_2 + x_1)^{15}$

- Do all terms cancel?
Multi-variate Polynomial Identity Testing

- \( p(x_1, x_2, x_3, x_4) = x_4^5 \cdot (x_1 - x_2 - x_3)^{30} + (x_4 + x_3 - x_1)^{15} \cdot (x_3 - x_2 + x_1)^{20} - (x_2 - x_3 + x_4)^{20} \cdot (x_2 + x_1)^{15} \)

- Do all terms cancel?

Randomized Algorithm
Multi-variate Polynomial Identity Testing

- \( p(x_1, x_2, x_3, x_4) = x_4^5 \cdot (x_1 - x_2 - x_3)^{30} + (x_4 + x_3 - x_1)^{15} \cdot (x_3 - x_2 + x_1)^{20} - (x_2 - x_3 + x_4)^{20} \cdot (x_2 + x_1)^{15} \)

- Do all terms cancel?

Randomized Algorithm

- Pick point \((x_1, ..., x_4)\), each \(x_i \in \mathbb{R} S\)
Multi-variate Polynomial Identity Testing

- \( p(x_1, x_2, x_3, x_4) = x_4^5 \cdot (x_1 - x_2 - x_3)^{30} + (x_4 + x_3 - x_1)^{15} \cdot (x_3 - x_2 + x_1)^{20} - (x_2 - x_3 + x_4)^{20} \cdot (x_2 + x_1)^{15} \)

- **Do all terms cancel?**

Randomized Algorithm

- Pick point \((x_1, ..., x_4)\), each \(x_i \in \mathbb{R} \) \( S \)
- \( p \) non-zero, degree \( d \)
Multi-variate Polynomial Identity Testing

- \( p(x_1, x_2, x_3, x_4) = x_4^5 \cdot (x_1 - x_2 - x_3)^{30} + (x_4 + x_3 - x_1)^{15} \cdot (x_3 - x_2 + x_1)^{20} - (x_2 - x_3 + x_4)^{20} \cdot (x_2 + x_1)^{15} \)

- **Do all terms cancel?**

Randomized Algorithm

- Pick point \((x_1, \ldots, x_4)\), each \(x_i \in \mathbb{R} \cdot S\)

- **p non-zero, degree \(d\)** \(\Rightarrow\) \(\Pr[p(x_1, \ldots, x_4) = 0] \leq \frac{d}{|S|}\)
Randomized Algorithm

Random Strings

BPP: Bounded-error Probabilistic Poly time
BPL: log space
P? = BPP
Randomized Algorithm

**Random Strings**

- **good**
- **bad**

BPP: Bounded-error Probabilistic Poly time

$\mathbf{BPP}$: Bounded-error Probabilistic Poly time
Randomized Algorithm

- **BPP**: Bounded-error Probabilistic Poly time
- **BPL**: log space
Randomized Algorithm

- BPP: Bounded-error Probabilistic Poly time
- BPL: log space
- \( P \supseteq \text{BPP} \)
Nondeterministic Algorithm
Graph 3-Coloring
Graph 3-Coloring
Nondeterministic Algorithm
Nondeterministic Algorithm

Proof/Certificate

Graph coloring
Nondeterministic Algorithm

Potential Proofs

Proof/Certificate

NP: Nondeterministic Polynomial time

"NP-Complete" problems: 3-Coloring, TSP, Knapsack, ...

P \? = NP

NP \? = coNP
Nondeterministic Algorithm

- **Potential Proofs**
  - good
  - bad

- **Proof/Certificate**
  - graph coloring

- **NP**: Nondeterministic Polynomial time

- "NP-Complete" problems: 3-Coloring, TSP, Knapsack, ...
Nondeterministic Algorithm

**NP**: Nondeterministic Polynomial time

“NP-Complete” problems: 3-Coloring, TSP, Knapsack, ...
Nondeterministic Algorithm

- **NP**: Nondeterministic Polynomial time
- “NP-Complete” problems: 3-Coloring, TSP, Knapsack, ...
- $P \overset{?}{=} NP$
Nondeterministic Algorithm

- **NP**: Nondeterministic Polynomial time
- “NP-Complete” problems: 3-Coloring, TSP, Knapsack, ...
- \( P \equiv NP \)
- \( NP \equiv coNP \)
Two Big Questions

\[ P \ ? \ NP \]

Is finding proofs as easy as verifying them?

Is 3-coloring in Polynomial Time?
Two Big Questions

\[ P \ ? \ NP \]

Is finding proofs as easy as verifying them?

Is 3-coloring in Polynomial Time?

\[ P \ ? \ BPP \]

Does randomness truly add power?
Computational Complexity Theory

How much **time, memory space, etc.** are needed to solve problems?

- **Is nondeterminism powerful?** $P \text{ vs. } NP$
  - Conjecture: $P \neq NP$
  - Techniques: hierarchy theorems, others

- **Is randomness powerful?** $P \text{ vs. } BPP, L \text{ vs. } BPL$
  - Conjecture: $P=BPP, L=BPL$
  - My work: derandomization, hierarchy theorems
Derandomization
Naive Derandomization

Try all possible random bit strings – exponentially many

01 10 10 10
11 01 10
yes
/no
/p.math(/x.math1/comma.math /x.math2/comma.math /x.math3/comma.math /x.math4)
/x.math1/comma.math /period.math/period.math/period.math/comma.math /x.math4
Naive Derandomization

Random Strings

Try all possible random bit strings – exponentially many
Naive Derandomization

Try all possible random bit strings
Naive Derandomization

Try all possible random bit strings – exponentially many
Derandomization – the Standard PRG Approach
Derandomization – the Standard PRG Approach

Random Strings

Poly many strings to try

$\Rightarrow$ $O(\log n)$ seed, exp stretch
Derandomization – the Standard PRG Approach

Random Strings

- good
- bad
- pseudo-random

Poly many strings to try
⇒ $O(\log n)$ seed, exp stretch

$p(x_1, x_2, x_3, x_4)$

$01\ 10\ 10\ 10\ x_1, \ldots, x_4$

$11\ 01\ 10\ \text{yes/no}$
Derandomization – the Standard PRG Approach

Random Strings
- Good
- Pseudo-random
- Bad

PRG

Poly many strings to try $\Rightarrow O(\log n)$ seed, exp stretch

$p(x_1, x_2, x_3, x_4)$

$x_1, \ldots, x_4$
Derandomization – the Standard PRG Approach

Random Strings

- Good
- Bad
- Pseudo-random

Poly many strings to try
Derandomization – the Standard PRG Approach

Poly many strings to try $\Rightarrow O(\log n)$ seed, exp stretch
Derandomization – the Standard PRG Approach

Poly many strings to try $\implies O(\log n)$ seed, exp stretch
A New Approach – Typically-Correct Derandomization
A New Approach – Typically-Correct Derandomization

Random Strings

- good
- bad
- pseudo-random

PRG

$p(x_1, x_2, x_3, x_4)$

Input: $x_1, ..., x_4$

Output: yes/no
A New Approach – Typically-Correct Derandomization

Random Strings
- good
- pseudo-random
- bad

PRG

\[ p(x_1, x_2, x_3, x_4) \]

\[ x_1, \ldots, x_4 \]

Seed length \( n \), poly stretch \( \frac{18}{35} \)
A New Approach – Typically-Correct Derandomization

Random Strings

- good
- bad
- pseudo-random

Seed length $n$, poly stretch

PRG

$p(x_1, x_2, x_3, x_4)$

$x_1, \ldots, x_4$
Standard Use of PRG’s vs. Typ-Correct
## Standard Use of PRG’s vs. Typ-Correct Derandomization

<table>
<thead>
<tr>
<th>Standard Derandomization</th>
<th>Typ-Correct Derandomization</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Nisan &amp; Wigderson, ...]</td>
<td>[Kinne, Van Melkebeek, Shaltiel]</td>
</tr>
</tbody>
</table>

- **Always correct**
- **Small # mistakes**
- Run PRG many times
- Run PRG only once
- Need exponential stretch
- Need only poly stretch
- Conditional results
- Unconditional results: fast parallel time, streaming, communication protocols
### Standard Use of PRG’s vs. Typ-Correct

<table>
<thead>
<tr>
<th>Standard Derandomization</th>
<th>Typ-Correct Derandomization</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Nisan &amp; Wigderson, …]</td>
<td>[Kinne, Van Melkebeek, Shaltiel]</td>
</tr>
<tr>
<td>• Always correct</td>
<td>• Small # mistakes</td>
</tr>
</tbody>
</table>

- Always correct
- Small # mistakes
- Need exponential stretch
- Need only poly stretch
- Conditional results
- Unconditional results: fast parallel time, streaming, communication protocols
# Standard Use of PRG’s vs. Typ-Correct

<table>
<thead>
<tr>
<th>Standard Derandomization</th>
<th>Typ-Correct Derandomization</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Nisan &amp; Wigderson, …]</td>
<td>[Kinne, Van Melkebeek, Shaltiel]</td>
</tr>
<tr>
<td>• Always correct</td>
<td>• Small # mistakes</td>
</tr>
<tr>
<td>• Run PRG many times</td>
<td>• Run PRG only once</td>
</tr>
</tbody>
</table>
Standard Use of PRG’s vs. Typ-Correct

<table>
<thead>
<tr>
<th>Standard Derandomization</th>
<th>Typ-Correct Derandomization</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Nisan &amp; Wigderson, ...]</td>
<td>[Kinne, Van Melkebeek, Shaltiel]</td>
</tr>
<tr>
<td>• <strong>Always correct</strong></td>
<td>• Small # mistakes</td>
</tr>
<tr>
<td>• Run PRG many times</td>
<td>• Run PRG only once</td>
</tr>
<tr>
<td>• Need exponential stretch</td>
<td>• Need only poly stretch</td>
</tr>
<tr>
<td><strong>Standard Derandomization</strong></td>
<td><strong>Typ-Correct Derandomization</strong></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>[Nisan &amp; Wigderson, ...]</td>
<td>[Kinne, Van Melkebeek, Shaltiel]</td>
</tr>
<tr>
<td>• Always correct</td>
<td>• Small # mistakes</td>
</tr>
<tr>
<td>• Run PRG many times</td>
<td>• Run PRG only once</td>
</tr>
<tr>
<td>• Need exponential stretch</td>
<td>• Need only poly stretch</td>
</tr>
<tr>
<td>• Conditional results</td>
<td>• Unconditional results:</td>
</tr>
<tr>
<td></td>
<td>fast parallel time, streaming,</td>
</tr>
<tr>
<td></td>
<td>communication protocols</td>
</tr>
</tbody>
</table>
Computational Complexity Theory

How much time, memory space, etc. are needed to solve problems?

- **Is nondeterminism powerful?** $P$ vs. $NP$
  - Conjecture: $P \neq NP$
  - Techniques: hierarchy theorems, others

- **Is randomness powerful?** $P$ vs. $BPP$, $L$ vs. $BPL$
  - Conjecture: $P=BPP$, $L=BPL$
  - My work: derandomization, hierarchy theorems
Hierarchy Theorems
Hierarchy Theorems
Hierarchy Theorems

- Fix a model of computing
Hierarchy Theorems

- Fix a model of computing
  (deterministic, randomized, nondeterministic)
Hierarchy Theorems

- Fix a model of computing
  (deterministic, randomized, nondeterministic)
- Can we achieve more given more resources?
Hierarchy Theorems

- Fix a model of computing
  (deterministic, randomized, nondeterministic)
- Can we achieve more given more resources?
Hierarchy Theorems

- Fix a model of computing (deterministic, randomized, nondeterministic)
- Can we achieve more given more resources?
Hierarchy Theorems

- Fix a model of computing
  (deterministic, randomized, nondeterministic)
- Can we achieve more given more resources?
Fix a model of computing
(deterministic, randomized, nondeterministic)
Can we achieve more given more resources?
Hierarchy Theorems

- Fix a model of computing (deterministic, randomized, nondeterministic)
- Can we achieve more given more resources?
Hierarchy Theorems

- Fix a model of computing (deterministic, randomized, nondeterministic)
- Can we achieve more given more resources?
Fix a model of computing (deterministic, randomized, nondeterministic)
Can we achieve more given more resources?

$log^3 n \text{ space} = log n \text{ space}$
Fix a model of computing
(deterministic, randomized, nondeterministic)
Can we achieve more given more resources?
Hierarchy Theorems

- Fix a model of computing (deterministic, randomized, nondeterministic)
- Can we achieve more given more resources?
Fix a model of computing
(deterministic, randomized, nondeterministic)
Can we achieve more given more resources?

My work: hierarchy theorems for randomized algorithms
Computational Complexity Theory

How much time, memory space, etc. are needed to solve problems?

- **Is nondeterminism powerful?** P vs. NP
  - Conjecture: P ≠ NP
  - Techniques: hierarchy theorems, others

- **Is randomness powerful?** P vs. BPP, L vs. BPL
  - Conjecture: P = BPP, L = BPL
  - My work: derandomization, hierarchy theorems
Hierarchy Theorems for Deterministic Algorithms
Hierarchy Theorems for Deterministic Algorithms

All Algorithms

$A_1 \quad A_2 \quad A_3 \quad \ldots$

time $n$
Hierarchy Theorems for Deterministic Algorithms

All Algorithms

\(A_1\) \(A_2\) \(A_3\) \(\ldots\)

All Inputs

\(x_1\) \(x_2\) \(x_3\) \(\ldots\)

Time \(n\)
Hierarchy Theorems for Deterministic Algorithms

<table>
<thead>
<tr>
<th>All Inputs</th>
<th>A_1</th>
<th>A_2</th>
<th>A_3</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>x_1</td>
<td>A_1(x_1)</td>
<td>A_2(x_1)</td>
<td>A_3(x_1)</td>
<td></td>
</tr>
<tr>
<td>x_2</td>
<td>A_1(x_2)</td>
<td>A_2(x_2)</td>
<td>A_3(x_2)</td>
<td>...</td>
</tr>
<tr>
<td>x_3</td>
<td>A_1(x_3)</td>
<td>A_2(x_3)</td>
<td>A_3(x_3)</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

All Algorithms at time n
Hierarchy Theorems for Deterministic Algorithms

All Algorithms

\[ \begin{array}{c|c|c|c|c}
\text{x}_1 & A_1(x_1) & A_2(x_1) & A_3(x_1) & \cdots \\
\text{x}_2 & A_1(x_2) & A_2(x_2) & A_3(x_2) & \cdots \\
\text{x}_3 & A_1(x_3) & A_2(x_3) & A_3(x_3) & \\
\vdots & \vdots & \vdots & \vdots & \\
\end{array} \]
Hierarchy Theorems for Deterministic Algorithms

All Algorithms

\[ A_1(x_1) \quad A_2(x_1) \quad A_3(x_1) \quad \ldots \quad A_1(x_3) \quad A_2(x_3) \quad A_3(x_3) \]

All Inputs

x_1
x_2
x_3

\[ \text{time } n \]
\[ \text{time } n^2 \]

D

\[ A_1 \quad A_2 \quad A_3 \quad \ldots \quad D \]
## Hierarchy Theorems for Deterministic Algorithms

<table>
<thead>
<tr>
<th>All Inputs</th>
<th>A₁</th>
<th>A₂</th>
<th>A₃</th>
<th>...</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>x₁</td>
<td>A₁(x₁)</td>
<td>A₂(x₁)</td>
<td>A₃(x₁)</td>
<td></td>
<td>¬A₁(x₁)</td>
</tr>
<tr>
<td>x₂</td>
<td>A₁(x₂)</td>
<td>A₂(x₂)</td>
<td>A₃(x₂)</td>
<td>...</td>
<td>¬A₂(x₂)</td>
</tr>
<tr>
<td>x₃</td>
<td>A₁(x₃)</td>
<td>A₂(x₃)</td>
<td>A₃(x₃)</td>
<td></td>
<td>¬A₃(x₃)</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>

- For all inputs x₁, x₂, x₃, ..., there exists an algorithm A₁(x) from the set of all algorithms such that:
  - A₁(x₁) is true,
  - ¬A₁(x₂) is true,
  - ¬A₁(x₃) is true,
  - ...  

- For any algorithm D from the set of all algorithms, there exists an input x such that:
  - ¬A₁(x) is true.
Hierarchy Theorems for Deterministic Algorithms

Poly Time

$n^3$ time

$n^2$ time

$n$ time
Hierarchy Theorems for Deterministic Algorithms

- \( \log^3 n \) space
- \( \log^2 n \) space
- \( \log n \) space
Hierarchy Theorems for Nondeterministic Algorithms?
Hierarchy Theorems for Nondeterministic Algorithms?

All Nondet. Algorithms

A1 A2 A3 ...

x1 A1(x1) A2(x1) A3(x1) ...

x2 A1(x2) A2(x2) A3(x2) ...

x3 A1(x3) A2(x3) A3(x3) ...

... ...

D ¬A1(x1) ¬A2(x2) ¬A3(x3) ...

All Inputs

...
Hierarchy Theorems for Nondeterministic Algorithms
Hierarchy Theorems for Nondeterministic Algorithms

\[ A_1 \text{ time } n \]
\[ D \text{ time } n^2 \]
Hierarchy Theorems for Nondeterministic Algorithms

- $A_1$ time $n$
- $D$ time $n^2$

Inputs:
- $x_1$
- $0x_1$
- $00x_1$
- $\vdots$
- $0^{\ell-2}x_1$
- $0^{\ell-1}x_1$
- $0^\ell x_1$
- $\ell \approx 2^{|x_1|}$
Hierarchy Theorems for Nondeterministic Algorithms

\[ A_1(x_1) \]
\[ A_1(0x_1) \]
\[ A_1(00x_1) \]
\[ \vdots \]
\[ A_1(0^{\ell-2}x_1) \]
\[ A_1(0^{\ell-1}x_1) \]
\[ A_1(0^\ell x_1) \]

inputs \( \ell \approx 2^{|x_1|} \)

time \( n \)

time \( n^2 \)
Hierarchy Theorems for Nondeterministic Algorithms

\[ A_1(x) \]

inputs \[ \ell \approx 2^{|x_1|} \]

Derived from page 27 of the document.
Hierarchy Theorems for Nondeterministic Algorithms

\[ A_1(x_1) \quad A_1(0x_1) \]
\[ A_1(0x_1) \quad A_1(0^2x_1) \]
\[ A_1(00x_1) \quad A_1(0^3x_1) \]
\[ \vdots \quad \vdots \]
\[ A_1(0^{\ell-2}x_1) \quad A_1(0^{\ell-1}x_1) \]
\[ A_1(0^{\ell-1}x_1) \quad A_1(0^\ell x_1) \]
\[ A_1(0^\ell x_1) \quad \neg A_1(x_1) \]

\ell \approx 2^{\log_2 x_1}

inputs \neq \}\
Hierarchy Theorems for Nondeterministic Algorithms

Assume $A_1$ the same as $D$ on all inputs
Hierarchy Theorems for Nondeterministic Algorithms

Assume $A_1$ the same as $D$ on all inputs.

- $A_1(x_1) = A_1(0x_1)$
- $A_1(0x_1) = A_1(0^2x_1)$
- $A_1(00x_1) = A_1(0^3x_1)$
- $A_1(0^{\ell-2}x_1) = A_1(0^{\ell-1}x_1)$
- $A_1(0^{\ell-1}x_1) = A_1(0^\ell x_1)$
- $A_1(0^\ell x_1) = \neg A_1(x_1)$
Hierarchy Theorems for Nondeterministic Algorithms

Assume $A_1$ the same as $D$ on all inputs
Hierarchy Theorems for Nondeterministic Algorithms

Assume $A_1$ the same as $D$ on all inputs.
Hierarchy Theorems for Nondeterministic Algorithms

Poly Time

\[ n^3 \text{ time} \]

\[ n^2 \text{ time} \]

\[ n \text{ time} \]
Hierarchy Theorems for Nondeterministic Algorithms

$\log^3 n$ space

$\log^2 n$ space

$\log n$ space
Hierarchy Theorems for Randomized Algorithms?

What if \( \Pr[A_1(x_1) = \text{"yes"}] \approx 0.5 \)?

Then \( D \) does not have bounded error, not valid.
### Hierarchy Theorems for Randomized Algorithms?

<table>
<thead>
<tr>
<th>All Randomized Algorithms</th>
<th>time $n$</th>
<th>D</th>
<th>time $n^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_1(x_1)$</td>
<td>$A_2(x_1)$</td>
<td>$A_3(x_1)$</td>
<td>...</td>
</tr>
<tr>
<td>$A_1(x_2)$</td>
<td>$A_2(x_2)$</td>
<td>$A_3(x_2)$</td>
<td>...</td>
</tr>
<tr>
<td>$A_1(x_3)$</td>
<td>$A_2(x_3)$</td>
<td>$A_3(x_3)$</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

All Inputs

- $x_1$
- $x_2$
- $x_3$
- ...

What if $\Pr[A_1(x) = \text{"yes"}] \approx 0.5$?

Then $D$ does not have bounded error, not valid.
## Hierarchy Theorems for Randomized Algorithms?

<table>
<thead>
<tr>
<th>All Inputs</th>
<th>All Randomized Algorithms</th>
<th>time $n$</th>
<th>time $n^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>$A_1(x_1)$</td>
<td>$A_2(x_1)$</td>
<td>$A_3(x_1)$</td>
</tr>
<tr>
<td>$A_1(x_2)$</td>
<td>$A_2(x_2)$</td>
<td>$A_3(x_2)$</td>
<td>$\neg A_1(x_2)$</td>
</tr>
<tr>
<td>$x_3$</td>
<td>$A_1(x_3)$</td>
<td>$A_2(x_3)$</td>
<td>$A_3(x_3)$</td>
</tr>
<tr>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
<td>$\vdots$</td>
</tr>
</tbody>
</table>

| $A_1$       | $A_2$                     | $A_3$     | $\ldots$   | $D$       |
|-------------|---------------------------|-----------|-------------|
| $\neg A_1(x_1)$ | $\neg A_2(x_2)$         | $\neg A_3(x_3)$ | $\vdots$   | $\vdots$ |

**What if** $\Pr[A_1(x_1) = \text{“yes”}] \approx 0.5$?
Randomized Algorithm

Random Strings

Bounded error: Correct with probability > 99%
# Hierarchy Theorems for Randomized Algorithms?

<table>
<thead>
<tr>
<th>All Randomized Algorithms</th>
<th>( A_1 )</th>
<th>( A_2 )</th>
<th>( A_3 )</th>
<th>...</th>
<th>( D )</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Inputs</td>
<td>( A_1(x_1) )</td>
<td>( A_2(x_1) )</td>
<td>( A_3(x_1) )</td>
<td>...</td>
<td>( \neg A_1(x_1) )</td>
</tr>
<tr>
<td>( x_1 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( x_2 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>( \neg A_2(x_2) )</td>
</tr>
<tr>
<td>( x_3 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>( \neg A_3(x_3) )</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>

- **What if** \( \Pr[ A_1(x_1) = \text{“yes”} ] \approx .5 \)?

\[ A_1^{(n)} \]
### Hierarchy Theorems for Randomized Algorithms?

<table>
<thead>
<tr>
<th>All Randomized Algorithms</th>
<th>( A_1 )</th>
<th>( A_2 )</th>
<th>( A_3 )</th>
<th>...</th>
<th>( D )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>( A_1(x_1) )</td>
<td>( A_2(x_1) )</td>
<td>( A_3(x_1) )</td>
<td>...</td>
<td>( \neg A_1(x_1) )</td>
</tr>
<tr>
<td>( x_2 )</td>
<td>( A_1(x_2) )</td>
<td>( A_2(x_2) )</td>
<td>( A_3(x_2) )</td>
<td>...</td>
<td>( \neg A_2(x_2) )</td>
</tr>
<tr>
<td>( x_3 )</td>
<td>( A_1(x_3) )</td>
<td>( A_2(x_3) )</td>
<td>( A_3(x_3) )</td>
<td>...</td>
<td>( \neg A_3(x_3) )</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

#### What if \( \Pr[A_1(x_1) = \text{“yes”}] \approx 0.5? \)

- Then \( D \) does not have **bounded error**, not valid
Hierarchy Theorems for Randomized Algorithms?

...
Hierarchy Theorems for Randomized Algorithms?

\[ \text{inputs} \quad \ell \approx 2^{|x_1|} \]

- \( x_1 \)
- \( 0x_1 \)
- \( 00x_1 \)
- \( \vdots \)
- \( 0^{\ell-2}x_1 \)
- \( 0^{\ell-1}x_1 \)
- \( 0^\ell x_1 \)

- \( A_1(x_1) \)
- \( A_1(0x_1) \)
- \( A_1(00x_1) \)
- \( \vdots \)
- \( A_1(0^{\ell-2}x_1) \)
- \( A_1(0^{\ell-1}x_1) \)
- \( A_1(0^\ell x_1) \)

- \( A_1(0x_1) \)
- \( A_1(0^2x_1) \)
- \( A_1(0^3x_1) \)
- \( \vdots \)
- \( A_1(0^{\ell-2}x_1) \)
- \( A_1(0^{\ell-1}x_1) \)
- \( \neg A_1(x_1) \)

- \( A_1 \) (time \( n \))
- \( D \) (time \( n^2 \))
Hierarchy Theorems for Randomized Algorithms?

Make sure $D$ has bounded error.
Hierarchy Theorems for Randomized Algorithms?

Make sure $D$ has bounded error – 1 bit of advice
Hierarchy Theorems for Randomized Algorithms?
Yes, for algorithms with 1 bit of advice!
Hierarchy Theorems for Randomized Algorithms?

- Yes, for algorithms with 1 bit of advice!
Hierarchy Theorems for Randomized Algorithms?

- Yes, for algorithms with 1 bit of advice!
Yes, for algorithms with 1 bit of advice!
Hierarchy Theorems for Randomized Algorithms?

- Yes, for algorithms with 1 bit of advice!

- My work [Kinne, Van Melkebeek]
Hierarchy Theorems for Randomized Algorithms?

- Yes, for algorithms with 1 bit of advice!

- My work [Kinne, Van Melkebeek]
  Memory Space hierarchies: randomized, quantum, ...
Computational Complexity Theory

How much **time, memory space, etc.** are needed to solve problems?

- **Is nondeterminism powerful?** $P \neq \text{NP}$
  - Conjecture: $P \neq \text{NP}$
  - Techniques: hierarchy theorems, others

- **Is randomness powerful?** $P = \text{BPP, L=BPL}$
  - Conjecture: $P=\text{BPP, L=}\text{BPL}$
  - My work: derandomization, hierarchy theorems
The End, Thank You!
The End, Thank You!

Slides available at:
http://www.kinnejeff.com/GoSycamores/
(or E-mail me)

More on my research (slides, papers, etc.) at:
http://www.kinnejeff.com/