

Universal Methods for Derandomization

Lecture 1

Randomness in computation:
five examples and a **universal**
task.

1

Randomness in Computation

- Example 1: **Breaking symmetry.**
- Example 2: **Finding witnesses.**
- Example 3: **Monte Carlo integration.**
- Example 4: **Approximation algorithms.**
- Example 5: **The probabilistic method.**

2

QuickSort

```
QuickSort(a){  
  x = select(a);  
  first = QuickSort(less(a,x));  
  last = QuickSort(greater(a,x));  
  return first.x.last;  
}
```

3

Variations of QuickSort

select(a) = a[1] (or **a[n/2]**)

- Worst case time $\Theta(n^2)$
- Average case time $\Theta(n \log n)$

select(a) = a[random(1..n)]

- Worst case *expected* time $\Theta(n \log n)$

select(a) = median(a) ← Derandomization

- Worst case time $\Theta(n \log n)$

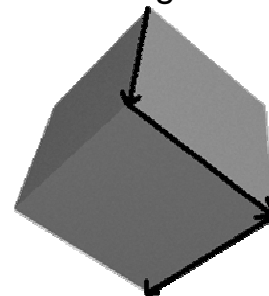
4

QuickSort

- For designing derandomized version of QuickSort detailed knowledge of randomized QuickSort and its analysis is needed – **Can this be avoided?**
- Even though we had to be clever, we didn't redesign the algorithm – we merely replaced the random element with a specific element. **This will be typical.**
- Derandomized QuickSort retains asymptotic complexity of randomized QuickSort but is less practical – Sad but seemingly unavoidable – in general we will even be prepared to accept a **polynomial** slowdown....

5

The Simplex Method for Linear Programming



6

The Simplex Method

- Pivoting rule: Rule for determining which neighbor corner to go to.
- Several rules: **Largest Coefficient, Largest Increase, Smallest Subscript, ...**
- All known deterministic rules have worst case exponential behavior.
- **Random Pivoting** is **conjectured** to be worst case expected polynomial.

7

The Simplex Method

- Can we prove **Random Pivoting** to lead to expected polynomial time simplex method? Very deep and difficult question, related to Hirsch' conjecture.
- **If Random Pivoting** is expected polynomial, can we then find a polynomial deterministic pivoting rule? **To be dealt with!**

8

Example 2: Finding Witnesses

- Is $(x-y)(x+y) = (x+x) - (y+y)$?
- Is $(x-y)(x+y)(z-u)+u-(v+u)(x-y) + uz = (y+z)(u-v)+u +(uv-uz)+(u-v)$?

9

Equivalence of Arithmetic Expressions

Obvious Deterministic Algorithm:

- Convert to sum-of-terms by applying distributive law.
- Collect terms.
- Compare.

Exponential complexity.

10

Randomized Algorithm

- Let n be the total length of the expressions. Pick random values a, b, c, \dots in $\{1, \dots, 10^n\}$ for the variables.
- If the two expressions evaluate to equal values, say "yes", otherwise say "no".

Polynomial complexity

11

Analysis

- (a, b, c, \dots) is a potential **witness** of the inequivalence of the expressions.
- If the expressions are equivalent, the procedure says "yes" with probability 1.
- If they are not, the procedure says "no" with probability at least 9/10 by the Schwartz-Zippel Lemma.
- The procedure has **one-sided error probability** at most 1/10.
- Arithmetic Expression Equivalence is in **coRP**.

12

Amplification by repetition

- Repeat t times with independently chosen values. Return “no” if at least one test returns “no”.
- If expressions are equivalent, the procedure returns “yes” with probability 1.
- If expressions are *not* equivalent, the procedure returns “yes” with probability at most 10^{-t} .

13

Derandomizing RP algorithms

- Can we find a polynomial time algorithm which always finds a witness or reports none exists?
- Can we achieve error probability 10^{-t} using much less than $\approx tr$ random bits, where r is the number of bits used to achieve error probability $1/10$?

14

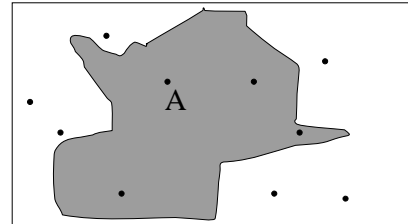
Complexity classes

- Why will I not focus much on **RP**, **BPP**, **coRP**, **ZPP**,...?
- Equivalence of Arithmetic Expressions is essentially the *only* important decision problem for which we have an efficient Monte Carlo and no known efficient deterministic algorithm (Primality used to be the other example)!

15

Example 3: Monte Carlo Integration

What is the area of A? Approximately 4/10.



16

Monte Carlo Integration

```
volume(A,m){
  c := 0;
  for(j=1; j<=m; j++)
    if(random(U) ∈ A) c++;
  return c/m;
}
```

17

Chernoff Bound

Let X_1, X_2, \dots, X_m be independent 0-1 variables. Let $X = \sum X_i$ and $\mu = E[X]$.

$$\Pr[|X-\mu| \geq d] \leq e^{-2d^2/n}$$

18

Analysis of Monte Carlo Integration

Let $m = 10 (1/\epsilon)^2 \log(1/\delta)$.

Chernoff Bounds



With probability at least $1 - \delta$, $\text{volume}(A, m)$ correctly estimates the volume of A within additive error ϵ .

19

Derandomizing Monte Carlo Integration

- Can we find an efficient deterministic algorithm which estimates the volume of A within additive error $1/100$ (with A given as “procedure parameter”)?
- Can we achieve error probability δ using much less than $\approx \log(|U|) \log(1/\delta)$ random bits?

20

Complexity Classes

- **P=BPP** is *not* known to imply that Monte Carlo Integration can be derandomized.

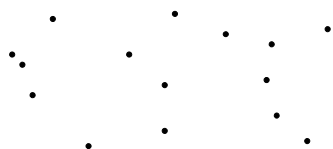
21

Example 4: Approximation Algorithms and Heuristics

- **Randomized Approximation Algorithms:** Solves (NP-hard) optimization problem with specified expected **approximation ratio**.
- **Example:** Goemans-Williamson MAXCUT algorithm.
- **Randomized Approximation Heuristics:** No approximation ratio, but may do well on many natural instances.

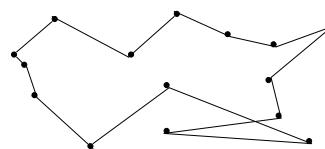
22

What is the best Traveling Salesman Tour?



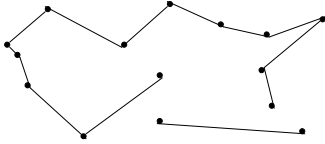
23

Local Search



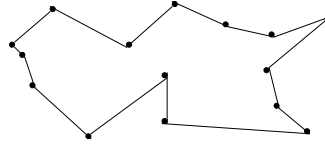
24

Local Search



25

Local Search



26

Local Search

```
LocalSearch(x){  
  y := feasible solution to x;  
  while(exists z in N(y) with  
    val(z) < val(y))  
    y := z;  
  return y;  
}
```

27

Local Search

- Local Search often finds good solutions to instances of optimization problems, but may be trapped in bad **local optima**.
- Randomized versions of local search often perform better.

28

Simulated Annealing

```
SimulatedAnnealing(x, T){  
  y := feasible solution to x;  
  repeat{  
    Pick a random neighbor z of y;  
    y := z with probability  
      min(1, exp((val(y)-val(z))/T));  
  }until(tired);  
  return the best y found;  
}
```

29

Derandomizing approximation algorithms and heuristics.

- Can a randomized approximation algorithm with a known approximation ratio be converted into a deterministic approximation algorithm with similar ratio?
- Can a randomized approximation heuristic which **behaves well on certain instances** be converted into a deterministic heuristic which **behaves well on the same instances**?

30

Example 5: The Probabilistic Method

- **Erdős 1947:** To prove the **existence** of a certain combinatorial object, prove that a **randomly** chosen object has the desired property with **positive** probability.
- Example: **Ramsey Graphs.**
- **Computational version:** Prove further that property is obtained with probability **close to 1**. Then we have a randomized algorithm for **constructing** the object.

31

Constructing Hard Truth Tables

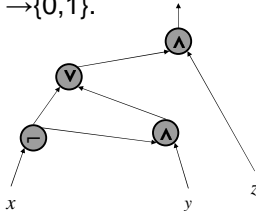
- A **truth table** tabulates $f:\{0,1\}^n \rightarrow \{0,1\}$.

x	y	z	$f(x,y,z)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

32

Circuits

- A (feed-forward) **circuit** may compute $f:\{0,1\}^n \rightarrow \{0,1\}$.



33

Upper bounds on circuit size

Every function $f:\{0,1\}^n \rightarrow \{0,1\}$ may be computed

- by a circuit of size $O(n2^n)$
 - **DNF.**
- by a circuit of size $O(2^n)$
 - **Decision tree.**
- by a circuit of size $O(2^n/n)$
 - **Dynamic Programming.**
- by a circuit of size $(1+o(1))(2^n/n)$
 - **with quite a bit of effort (Lupanov, 1965).**

34

Constructing Hard Functions

If a function $f:\{0,1\}^n \rightarrow \{0,1\}$ is chosen at random, the probability that it can be computed with a circuit with fewer than $2^n/2n$ gates is much less than 2^{-n} .

Proof:

#functions is 2^{2^n}

#circuits of size less than s is $(3(n+s)^2)^s$

35

Constructing Hard Functions

- A trivial efficient randomized algorithm can construct a truth table requiring circuits of size $2^n/2n$ (and even $(1+o(1))(2^n/n)$) with very small error probability.
- Can a **deterministic** algorithm efficiently construct a truth table requiring circuits of size $2^{n/100}$?

36

Complexity Theory \approx
**Study of generic (or universal)
 tasks.**

**Is there a single task capturing
 all (or most of) previous
 examples?**

37

A Universal Task

Finding Hay in a Haystack
 (Black Box version)

Given **Black Box** $T: \{0,1\}^n \rightarrow \{0,1\}$ with
 $\mu(T) \geq \frac{1}{2}$, find x so that $T(x)=1$.

$$\mu(T) = \#\{x \mid T(x)=1\} / 2^n$$

Want: Algorithm polynomial in n .

38

What is a Black Box?

- Computational object T representing a map.
- Only allowed operations on T are *queries* for values $T(x)$ for arbitrary x .
- No access to any representation of T or any additional information about T (except domain and co-domain).

39

Universality of Haystack task

- If we can find hay in a haystack **efficiently** **deterministically** or **with small error probability** **but using few random bits**, we can immediately do similarly for:
- Example 1 (**breaking symmetry**): $T(x)=1$ if x is a pivot sequence leading to the optimum vertex.
- Example 2 (**finding witnesses**): $T(a)=1$ if a is a witness for the non-equivalence of the expressions.
- What about Examples 3,4,5? **To be dealt with later..**

40

Unfortunately....

- There is no efficient deterministic algorithm for the haystack task in the black box version.

Proof:

- The deterministic algorithm queries $T(x_1), T(x_2), \dots, T(x_t)$ for *fixed* x_1, x_2, \dots, x_t (depending only on n)
- It fails to find hay for $T(y)=0$ iff y in $\{x_1, x_2, \dots, x_t\}$.

41

Preview

- We can't find hay in a black box haystack deterministically, but we *will* be able to save random bits (to be seen later)

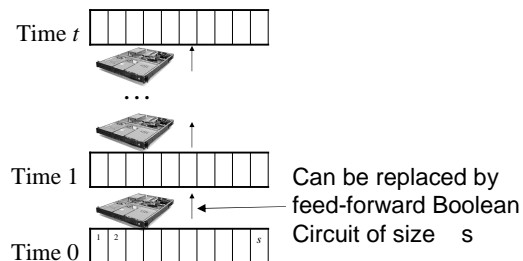
42

Opening the Black Box

- We are only interested in T if we have an **efficient algorithm** for computing T .
- **Theorem:** If $T: \{0,1\}^n \rightarrow \{0,1\}$ can be computed by an algorithm using space s and time t then T can be computed by a **circuit** of size roughly $s t$.

43

The Tableau Method



44

A Universal Task: Finding Hay in a Haystack (Circuit version)

Given **circuit** $C: \{0,1\}^n \rightarrow \{0,1\}$ with $\Pr(C=1) \geq \frac{1}{2}$, find x so that $C(x)=1$.

Want: Algorithm polynomial in n and size of C .

45

Hypothesis H

There exists polynomial procedure **findHay** taking as input a circuit $C: \{0,1\}^n \rightarrow \{0,1\}$ so that

1. **findHay**(C) is in $\{0,1\}^n$.
2. If $\Pr(C) \geq \frac{1}{2}$ then $C(\text{findHay}(C))=1$.

46

Fact

The constant $\frac{1}{2}$ can be replaced with

- any constant strictly between 0 and 1,
- n^{-k} (quite close to 0), or
- $1-2^{-n^{1-1/k}}$ (very close to 1),

without changing truth value of Hypothesis H.

This is not the end of the story – we shall do even better later in the week!

47

Numerical integration revisited

Density Estimation: Given circuit $C: \{0,1\}^n \rightarrow \{0,1\}$, estimate $\Pr(C)$ within additive error ϵ .

Desired: Algorithm running in time polynomial in C and $1/\epsilon$.

48

Finding Hay Derandomizes Monte Carlo Integration

Hypothesis H



An efficient deterministic algorithm for density estimation exists.

49

Credits

- Proof due to Sipser, Lauterman 1983 (**proving a statement relating probabilistic computation to the polynomial hierarchy**).
- Theorem due to Andreev, Clementi, Rolim 1995 (**proving slightly different theorem**) and Buhrman and Fortnow, 1999 (**noting that the Lauterman proof implies the theorem**).

50