Lecture 5. The Incompressibility Method

- A key problem in computer science: analyze the average case performance of a program.

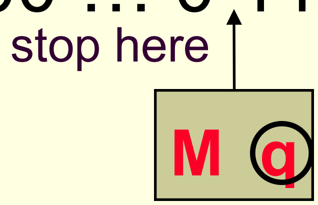
- Using the Incompressibility Method:
  - Give the program a random input of length n, say of complexity n- log n (or sometimes complexity n).
  - Analyze the program with respect to this single and fixed input. This is usually easier than average case using the fact this input is almost incompressible.
  - If we used complexity n- log n, the running time for this single input is the average case running time of all inputs, since a (1-1/n)th fraction of all inputs have this high complexity!

Formal language theory

- Example: Show $L=\{0^k1^k \mid k>0\}$ not regular. By contradiction, assume that DFA M accepts L.

  Choose k so that $C(k) \gg 2|M|$. Simulate M:

$$\overbrace{000 \ldots 0}^{k}\ \overbrace{111 \ldots 1}^{k}$$



  $C(k) < |M| + |q| + O(1) < 2|M|$. Contradiction. ■

- Remark. Generalizes to iff condition: more powerful & easier to use than “pumping lemmas”.

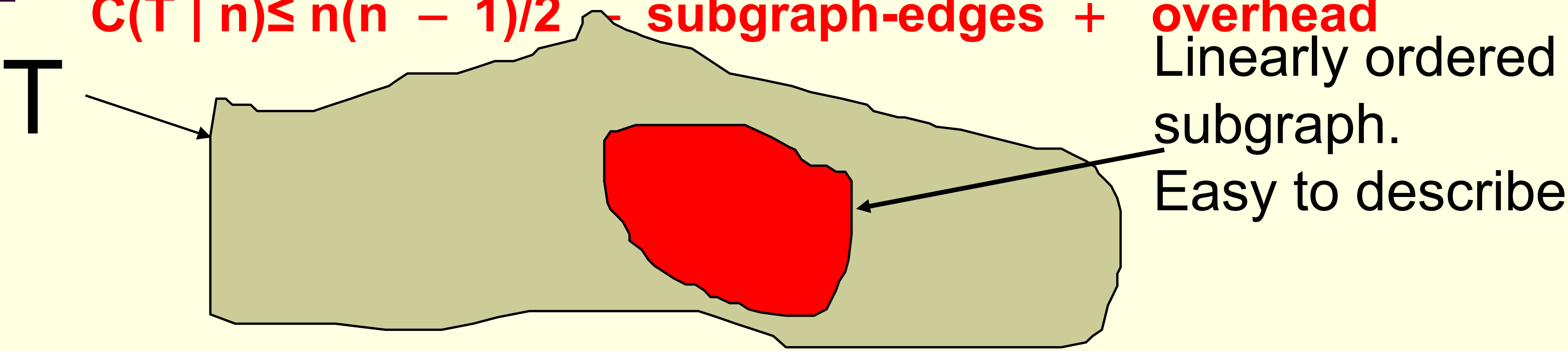
Combinatorics

- Theorem. There is a tournament (complete directed graph) T of n players that contains no large transitive subtournaments (>1 + 2 log n).

Proof by Picture: Choose a random T.

- One bit codes a directed edge, each tournament is encoded in string of $n(n-1)/2$ bits, and each string of $n(n-1)/2$ bits codes a tournament. Choose T such that $C(T \mid n) \geq n(n-1)/2$.
- If there is a large transitive subtournament on $v(n)$ nodes, then a large number of edges are given for free! Subgraph-edges = $v(n)(v(n)-1)/2$. Overhead = $v(n) \log n$. Overhead ≥ subgraph edges since ■

$$C(T \mid n) \leq n(n-1)/2 - \text{subgraph-edges} + \text{overhead}$$



Combinatorics

- Theorem. Let $w(n)$ be the largest integer such that every tournament T has disjoint node sets A and B both of cardinality $w(n)$ such that AxB is a subset of the ordered edge set of T. Then, $w(n) \leq 2 \log n$.
- Proof. Choose T with $C(T|n) \geq n(n-1)/2$.
- Add descriptions A and B in $2\, w(n) \log n$ bits (in lexicographic order, say).
- Save bits describing edges between A and B in $w(n)^2$ bits.
- Add – Save ≥ 0. QED

Graphs

- Consider undirected labeled graphs.
- A clique is a subset of nodes with edges between every pair.
- An anticlique is a subset of nodes without edges between any pair.
- Encode graph G s.t. The set of node pairs are lexicographically ordered without repetition, {i,j} with $i < j$, and the corresponding bit is 1 if there is an edge, and 0 otherwise.
- Theorem. There is an undirected labeled graph G on n nodes that contains no clique or anticlique on $>1+2 \log n$ nodes.
- Proof. Let G be an undirected labeled graph of high Kolmogorov complexity, $C(G|n) \geq n(n-1)/2$. The proof is now isomorphic to that of the transitive subtournaments.

Graphs

- Lemma. A fraction of at least $1 – 1/2^{d(n)}$ of all labeled undirected graphs on n nodes have $C(G|n,d) \geq n(n-1)/2 - d(n)$.
- Proof. There are at most $2^{n(n-1)/2 – d(n)} - 1$ programs of length $< n(n-1)/2 - d(n)$. QED
- Remark. Hence a property that holds for such graphs holds with high probability and in expectation (on average).
- Lemma. All nodes of a graph with $d(n)=o(n)$ have degree
- $n/2 \pm o(n)$.
- Proof. Choose G s.t. $C(G|n) \geq n(n-1)/2 - d(n)$. For every node i, the scattered substring of bits corresponding to {i,j} or {j,i} has complexity $\geq n-d(n)- 2 \log n$, since otherwise its description + description i +the literal remainder of G|n gives a description of G|n of length $< n(n-1)/2 – d(n)$. Let $d(n)=o(n)$.
- Since the substring has complexity $\geq n-o(n)$, we have by similar reasoning to that of the last frame of lecture 2 that the substring contains $n/2 \pm O(\sqrt{o(n)n}) = n/2 \pm o(n)$ bits 1, and hence node i has degree $n/2 \pm o(n)$. QED

Graphs

- Lemma. All graphs with $d(n)=o(n)$ have diameter 2.

- Proof. Diameter 1 is a complete graph G with $C(G|n)=O(1)$.
- Assume there is a shortest path of length >2 between nodes i,j.
- Add identity of nodes i,j in 2 log n bits.
- Save $n/2-o(n)$ bits from omitting edge bits (k,j) (which are all 0) for every k for which there is an edge (i,k). There are $>n/2-o(n)$ of them by previous lemma. QED

- Remark. There is some discrepancy between add and save here. We can in fact strengthen the theorem to show that all such graphs have $n/4 -o(n)$ disjoint paths of length 2 between every pair of nodes.

Unlabeled Graphs

- # of labeled undirected graphs on n nodes is 2^{n(n-1)/2}.
- Theorem (Harary, Palmer 1973) # of unlabeled undirected graphs on n nodes is asymptotic to 2^{n(n-1)/2} / n!

- Proof by incompressibility (Sketch). There are n! ways to relabel a graph on n nodes for every graph. But, for example, the complete graph stays the same under every relabeling. So the automorphism group of that graph has cardinality n! A Kolmogorov random graph stays the same only under identity relabeling. Its automorphism group has cardinality 1 (such graps are called rigid.)
- By incompressiblity we estimate the number of graphs (what is their minimum complexity and maximum complexity) which have automorphism groups of given cardinality. This gives the theorem.

- QED

Fast adder

- Example. Fast addition on average.
  - Ripple-carry adder: n steps adding n-bit numbers.
  - Carry-lookahead adder: 2 log n steps (divide-and-conquer).
  - Burks-Goldstine-von Neumann (1946): log n expected length of carry sequence, so log n expected steps.

```
S= x⊕y; C= carry sequence;
while (C≠0) {
     S= S⊕C;
     C= new carry sequence; }
```

Average case analysis: Fix x, take random y s.t. $C(y|x) \geq |y|$

x = … u1 …
y = … û1 …,

(Max such u is precise carry length) Low order bits right.

û is complement of u

If $|u| > \log n$, then $C(y|x)<|y|$. Average over all y, get log n. QED

Sorting

- Given n elements (in an array). Sort them into ascending order.
- This is the most studied fundamental problem in computer science.

- Shellsort (1959): p passes. In each pass, compare in subarrays (length related to increment) adjacent elements and move larger elements to the right (Bubblesort) so that the large elements `bubble’ to front.
- Open for over 40 years: a nontrivial general average case complexity lower bound of Shellsort?

Shellsort Algorithm

- Using p increments $h_1, \dots, h_p$, with $h_p=1$
- At k-th pass, the array is divided in h_k separate sublists of length n/h_k (taking every h_k-th element).
- Each sublist is sorted by insertion/bubble sort.

- Application: Sorting networks --- $n \log^2 n$ comparators, easy to program, competitive for medium size lists to be sorted.

Shellsort history

- Invented by D.L. Shell [1959], using $p_k = n/2^k$ for step k. It is a $\Theta(n^2)$ time algorithm
- Papernow&Stasevitch [1965]: $O(n^{3/2})$ time by destroying regularity in Shell's geometric sequence.
- Pratt [1972]: All quasi geometric sequences use $O(n^{3/2})$ time .$\Theta(n\log^2 n)$ time for p=(log n)^2 with increments 2^i3^j.
- Incerpi-Sedgewick, Chazelle, Plaxton, Poonen, Suel (1980's) – best worst case, roughly, $\Theta(n\log^2 n / (\log \log n)^2)$.
- Average case:
  - Knuth [1970's]: $\Theta(n^{5/3})$ for p=2
  - Yao [1980]: p=3 characterization, no running time.
  - Janson-Knuth [1997]: $O(n^{23/15})$ for p=3.
  - Jiang-Li-Vitanyi [J.ACM, 2000]: $\Omega(pn^{1+1/p})$ for every p.

Shellsort Average Case Lower bound

Theorem. p-pass Shellsort average case $T(n) \geq pn^{1+1/p}$

Proof. Fix a random permutation Π with Kolmogorov complexity $nlogn$. I.e. $C(\Pi) \geq nlogn$. Use Π as input. (We ignore the self-delimiting coding of the subparts below. The real proof uses better coding.)

For pass i, let $m_{i,k}$ be the number of steps the kth element moves. Then $T(n) = \Sigma_{i,k}\, m_{i,k}$

From these $m_{i,k}$'s, one can reconstruct the input Π, hence

$$\Sigma \log m_{i,k} \geq C(\Pi) \geq n \log n$$

Maximizing the left, all $m_{i,k}$ must be the same (maintaining same sum). Call it m. So $\Sigma\, m = pnm = \Sigma_{i,k}\, m_{i,k}$ Then,

$\Sigma \log m = pn \log m \geq \Sigma \log m_{i,k} \geq nlogn \Rightarrow m^p \geq n$.

So $T(n) = pnm > pn^{1+1/p}$. ■

Corollary: p=1: Bubblesort $\Omega(n^2)$ average case lower bound. p=2: $n^{3/2}$ lower bound. p=3, $n^{4/3}$ lower bound (4/3=20/15); and only $p=\Theta(\log n)$ can give average time $O(n \log n)$.

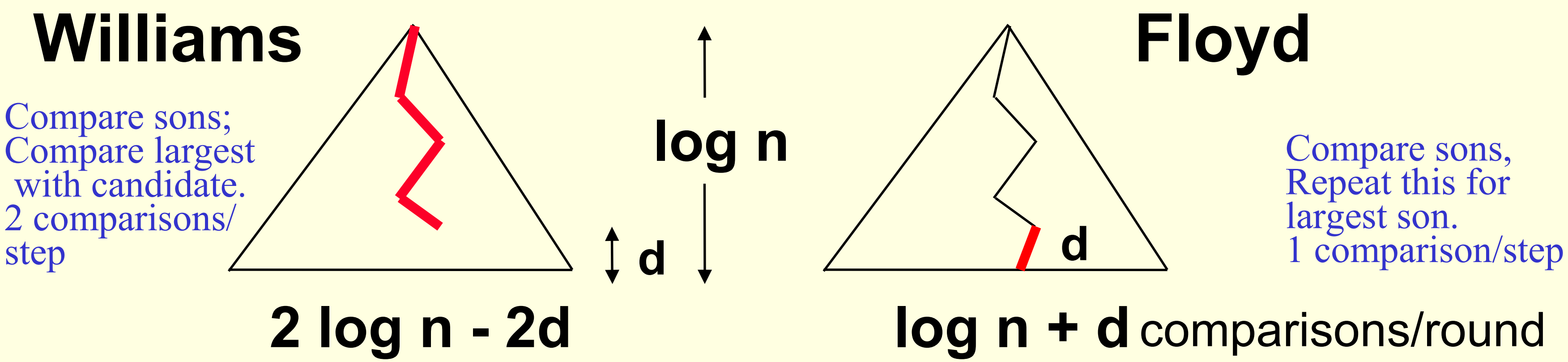
Heapsort

- 1964, JWJ Williams [CACM 7(1964), 347-348] first published Heapsort algorithm
- Immediately it was improved by RW Floyd.
- Worst case O(n logn).
- Open for 40 years: Which is better in average case: Williams or Floyd? (choose between n log n and 2n log n)
- R. Schaffer & Sedgewick (1996). Ian Munro provided the solution here.

Heapsort average analysis (I. Munro)

- Average-case analysis of Heapsort.

Heapsort: (1) Make Heap. O(n) time.
(2) Delete max at root, restore heap, repeat.



Fix random heap H, C(H) > n log n. Simulate Step (2). Each round, encode the red path in log n -d bits. The n paths describe the heap! Hence, total n paths, length $\geq$ n log n, hence d must be a constant Floyd takes n log n comparisons, and Williams takes 2n log n. ■

A selected list of results proved by the incompressibility method

- $\Omega(n^2)$ for simulating 2 tapes by 1 (30 years)
- k heads > k-1 heads for PDAs (15 years)
- k one-ways heads can't do string matching (13 yrs)
- 2 heads are better than 2 tapes (40 years)
- Average case analysis for heapsort (30 years)
- k tapes are better than k-1 tapes. (20 years)
- Many theorems in combinatorics, formal language/automata, parallel computing, VLSI
- Simplify old proofs (Hastad Lemma).
- Shellsort average case lower bound (40 years)

More on formal language theory

Lemma (Li-Vitanyi) Let $L \subseteq V^*$, and $L_x=\{y: xy \in L\}$. Then L is regular implies there is c for all x,y,n, let y be the n-th element in L_x, we have $C(y|x) \leq C(n)+c$.

Proof. Like example. QED.

Example 2. $\{1^p : p \text{ is prime}\}$ is not regular.

Proof. Let p_i, i=1,2 …, be the list of primes. Then p_{k+1} is the first element in L_{Pk}, hence by Lemma, $C(p_{k+1}|p_k) \leq O(1)$. Impossible since $p_{k+1}-p_k \to \infty$ for $k \to \infty$

QED

Characterizing regular sets

- For an lexicographic enumeration of $\Sigma^*=\{y_1,y_2, \dots\}$, define characteristic sequence $X= X_1 X_2 \dots$of

  $L_x=\{y_i : xy_i \in L\}$ by

  $X_i = 1$ iff $xy_i \in L$

Theorem. L is regular iff there is a c for all x,n,

$$C(X_{1:n}|n) < c$$

Proof. L is regular (finite-state) iff L is the union of finitely many disjoint sets $\{x\}L_x$ (The Myhill-Nerode Theorem). Hence every X of L_x is a recursive sequence. This shows the `if' side. The `only if' side depends on a sophisticated lemma, see textbook.