Algebraic Techniques

Maryam Babaei

- introduction
- Fingerprinting and Freivalds' Technique
- Verifying polynomial identities
- Perfect matching in graphs
- Verifying equality of strings
- A Comparison of Fingerprinting Techniques
- Pattern Matching
- Verifying Graph Non-Isomorphism

introduction

- **The problem:**
  - given a large Universe U and elements a , b $\in$ U
  - decide a = b has a deterministic complexity at least log |U|
  - in some cases this may be very slow

- **The idea:**
  - pick a random mapping $M: U \to V$ such that w.h.p.:

    $$a = b \Leftrightarrow M(a) = M(b)$$
  - M(a) = M(b) can be evaluated faster than a = b
  - we call M(x) the fingerprint of x

Fingerprinting and Freivalds' Technique

Problem 1:

Let A, B, C be n ×n matrices, We want to verify that
AB = C

A randomized algorithm

- ❑ Pick a random vector $r \in \{0,1\}^n$
- ❑ Compute x = Br, y = Ax = ABr and z = Cr
- ❑ If y = z then

  "yes" AB = C
- ❑ **Else**

  "no" AB≠ C

Time complexity: $O(n^2)$

- if $AB=C \Rightarrow y=z$
- if $AB \neq C \not\Rightarrow y \neq z$
- algorithm errs only if $AB \neq C$ but $y=z$
- this is a One-sided error Monte-Carlo algorithm

Theorem 7.1: Let A, B, and C be n x n matrices over F such that $AB \neq C$ Then for r chosen uniformly at random from $\{0,1\}^n$, $pr[ABr = Cr] < \frac{1}{2}$

- Let D = AB - C. We know that $D \neq 0$
- Pr[y=z]=pr[Dr=0]
- Let **d** be the vector consisting of the entries in the first row of **D** that has a non zero entry.

$$pr[Dr = 0] \leq pr[d^T r = 0]$$

$$\mathrm{d}^{\mathrm{T}} r = 0 \Leftrightarrow r_1 d_1 = -\sum_{i=2}^{k} d_i r_i \Rightarrow r_1 = \frac{-\sum_{i=2}^{k} d_i r_i}{d_1}$$

- for each choice of the values $r_2, \ldots, r_n$ there is only one value for r_1 that could have caused $d^T r = 0$

- Since r_1 is uniformly distributed over a set of size 2

$$pr[r_1 = v] \leq \frac{1}{2}$$

principle of deferred decisions

- we first fixed the choices for $r_2, \ldots, r_n$ and then considered the effect of the random choice of r_1 given those choices.
- After k iterations the probability of error is $\frac{1}{2^k}$
- This can be generalized for verifying any matrix identity. However it only makes sense if at least one of the matrices is not explicitly provided.

Verifying polynomial identities

- Comparing polynomials is trivial if they are explicitly given in the same form

$$x^2 - 2x - 3 = x^2 - 2x - 3$$

$$(x-1)(x+3) = x^2 - 2x - 3$$

Problem: Given polynomials $p_1(x), p_2(x), p_3(x)$ verify that $p_1(x) * p_2(x) = p_3(x)$

- $p_1(x)$ *and* $p_2(x)$ are of degree at most n.
- $p_3(x)$ is of degree at most 2n.
- mult in O(n log n) using FFT(fast fourier transform).
- evaluation at fixed point in O(n) time.

A randomized algorithm

- Choose a value r $\in s \subseteq F \; with \; |s| \geq 2n+1$
- Evaluate $Q(r) = p_1(r) * p_2(r) - p_3(r)$

Theorem: if Q(x)=0 then Q(r)= 0. Otherwise

$$pr(Q(r) = 0 \mid Q(x) \neq 0) \leq \frac{2n}{|s|}$$

Proof:

Q(x) is not all zero.Due to the fundamental theorem of algebra we also know that Q(x) has at most 2n distinct roots.

$$pr(Q(r) = 0 \mid Q(x) \neq 0) \leq \frac{2n}{|s|}$$

$$\det M = \sum_{\pi \in S_n} sgn(\pi) \prod_{i=1}^{n} M_{i\,\pi(i)}$$

Definition 7.1:

- The Vandermonde matrix $M(x_1, x_2, \ldots, x_n)$ is defined in terms of the indeterminates $x_1, x_2, \ldots, x_n$ such that $M_{ij} = x_i^{j-1}$ that is

$$M = \begin{pmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{n-1} \\ \vdots & \vdots & \vdots & \vdots & \\ 1 & x_n & x_n^2 & \cdots & x_n^{n-1} \end{pmatrix}$$

$$\det M = \prod_{j<i} (x_i - x_j)$$

- Computing the determinant of this symbolic matrix is prohibitively expensive since it has n! terms
- we will formulate this as the problem of verifying that the polynomial

$$Q(x_1, x_2, \dots, x_n) = \det M - \prod_{j<i}(x_i - x_j)$$

- substitute random values for each x_i and check whether Q = 0.
- the determinant can be computed in polynomial time for specified values of the variables $x_1, x_2, \dots, x_n$

- Multivariate polynomial $Q(x_1, x_2, \ldots, x_n)$
- degree of term: sum of variable degree
- total degree Q: max degree of term

Theorem 7.2(Schwartz-Zippel Theorem) :

Let $Q(x_1, x_2, \ldots, x_n) \in F[x_1, \ldots, x_n]$ be a multivariate polynomial of total degree d. Fix any set $S \subseteq F$, and let $r_1, \ldots, n_n$ be chosen independently and uniformly at random from S. Then

$$pr[Q(r_1, \ldots, r_n) = 0 \mid Q(x_1, \ldots, x_n) \not\equiv 0] \leq \frac{d}{|S|}$$

Proof:

by induction on the number of variables n.

base case: n = 1 (univariate polynomial)

$$pr[Q(r) = 0 \,|Q(x) \not\equiv 0] \leq \frac{d}{|S|}$$

induction hypothesis:

$$pr[Q(r_1, \dots, r_{n-1}) = 0 \,|Q(x_1, \dots, x_{n-1}) \not\equiv 0] \leq \frac{d}{|S|}$$

Consider the polynomial $Q(x_1, x_2, \dots, x_n)$ and factor out the variable x_1 :

$$Q(x_1, \dots, x_n) = \sum_{i=0}^{k} x_1^i Q_i(x_2, \dots, x_n)$$

Where k<=d is the largest exponent of x_1 in Q.

- We know that $Q_k(x_2, \dots, x_n) \neq 0$, otherwise k would not be the largest exponent of x_1 in Q.
- The total degree of $Q\,(x_2, \dots, x_n)$ is at most d-k.
- With induction hypothesis:

$$pr[Q_k(r_2, \dots, r_n) = 0] \leq \frac{d-k}{|s|}$$

- We assume the $Q_k(r_2, \dots, r_n) \neq 0$
- Consider the following polynomial:

$$q(x_1) = Q(x_1, r_2, \dots, r_n) = \sum_{i=0}^{k} x_1^i Q_i(r_2, \dots, r_n)$$

- univariate polynomial q has degree k and, by our assumption, is not identically zero.
- Our base case now implies that:

$$pr[q(r_1) = Q(r_1, r_2, \ldots, r_n) = 0 | Q_k(r_2, \ldots, n_n) \neq 0] \leq \frac{k}{|s|}$$

we have :

$$pr[Q_k(r_2, \ldots, r_n) = 0] \leq \frac{d-k}{|s|}$$

$$pr[q(r_1) = Q(r_1, r_2, \ldots, r_n) = 0 | Q_k(r_2, \ldots, n_n) \neq 0] \leq \frac{k}{|s|}$$

Note: $pr(E_1) \leq pr(E_1 | \bar{E}_2) + pr(E_2)$

Therefore:

$$pr[Q(r_1, \ldots, r_n) = 0 | Q(x_1, \ldots, x_n) \neq 0] \leq \frac{k}{|s|} + \frac{d-k}{|s|} = \frac{d}{|s|}$$

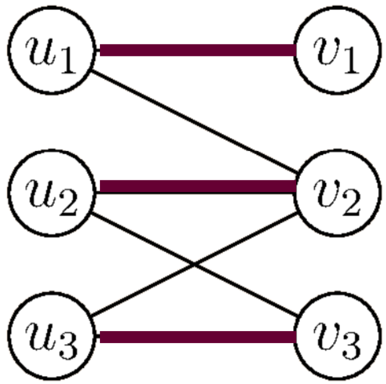
Perfect matching in graphs

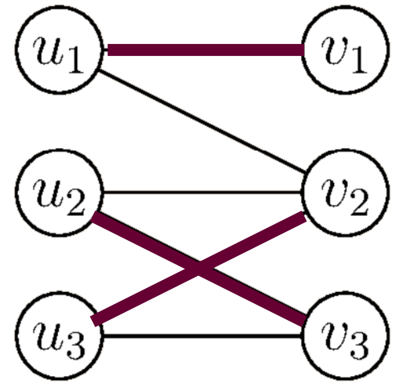
- Consider a bipartite Graph G(U, V,E) with the independent sets of vertices U=$\{u_1, u_2, \ldots, u_n\}$ and

V =$\{v_1, v_2, \ldots, v_n\}$

- A matching M⊆ E is a set of edges such that each vertex occurs at most once in M.
- A perfect matching is a matching of size n.
- A perfect matching can be viewed as a permutation from U into V.

$$\pi \in s_n \Rightarrow \left(u_i, v_{\pi(i)}\right) \qquad 1 \leq i \leq n$$





$$\pi_1 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix} \qquad \pi_2 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix}$$

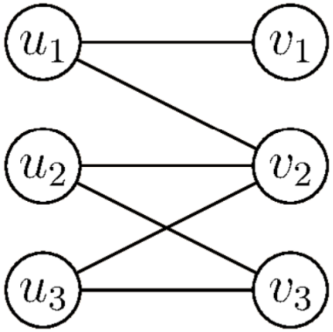
$$\det M = \sum_{\pi \in s_n} sgn(\pi) \prod_{i=1}^{n} M_{i\,\pi(i)}$$

Theorem 7.3 (Edmonds' Theorem):

Let A be the n x n matrix obtained from G(U, V,E) as follows:

$$A_{ij} = \begin{cases} x_{ij} & (u_i, v_j) \in E \\ 0 & (u_i, v_j) \notin E \end{cases}$$

Define the multivariate polynomial $Q(x_{11}, x_{12}, \ldots, x_{nn})$ as being equal to det(A). Then, G has a perfect matching if and only if Q ≠ 0.



$$A = \begin{pmatrix} x_{11} & x_{12} & 0 \\ 0 & x_{22} & x_{23} \\ 0 & x_{32} & x_{33} \end{pmatrix}$$

$$\det A = x_{11}x_{22}x_{33} - x_{11}x_{23}x_{32}$$

Proof:

The determinant of A is given by:

$$\sum_{\pi \in s_n} sgn(\pi) A_{1,\pi(1)} A_{2,\pi(2)} \cdots A_{n,\pi(n)}$$

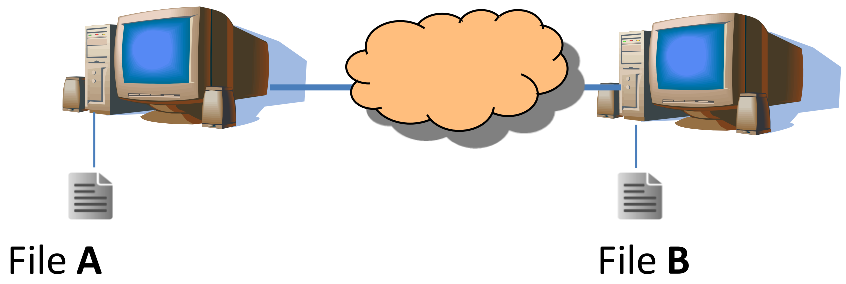
The determinant is non-zero iff there is a permutation for wich all A_{ij} are not zero. This is a perfect matching.

$\square$

- Note that the determinant can be computed in O(n3) time by Gaussian elimination.

- The determinant is a multivariate polynomial with n^2 variables of degree n.
- run the Schwartz-Zippel on det(A).
- We simply have to fill in random values for the x_{ij} and evaluate it.
- Construction a perfect matching deterministically takes $m\sqrt{n}$ where m = |E|.
- This method can be used for a parallel algorithm.

Verifying equality of strings



- **Aim:** To determine if file **A** identical to file **B** by communicating fewest bits ?
- Given two strings represented by the bit sequences $(a_1, a_2, \dots, a_n)$ and $(b_1, b_2, \dots, b_n)$
- we want to check them for equality without comparing all the bits.
- define: $$a = \sum_{i=1}^{n} a_i 2^{i-1} \quad , \quad b = \sum_{i=1}^{n} b_i 2^{i-1}$$

A randomized algorithm

- A chooses a prime p ∊ [2… T] uniformly at random
- A sends (p , a mod p) to B
- B computes b mod p
- B returns the result of checking a mod p=b mod p

- Now we only need to compare log p bits.

Theorem: for any number k let $\pi(k)$ be the number of distinct primes less than k is $\frac{k}{\ln k}$

Lema: The number of distinct prime divisors of any number less than 2^n is at most n.

Proof:

Each prime number is greater than 1. If N has more n distinct prime divisors, then $N > 2^n$

$\square$

- Let c = |a-b| Fingerprint fails only when $c \neq 0$ and p divides c.
- Since $a, b < 2^n$ we know that $c < 2^n$
- Choose a threshold T larger than n.
- Number of prime smaller T is $\frac{T}{\ln T}$

- At most n can be divisors of c and cause fingerprint strategy to fail.
- Pick a random prime p smaller than T.
- The number of bits needed for tranmission is o(log T)
- Choose T=tn log tn.

Theorem 7-5: $pr\left[F_p(a) = F_p(b) \middle| a \neq b\right] \leq \frac{n}{\pi(T)} = o(\frac{1}{t})$

Proof:

$$\frac{n}{\pi(T)} = \frac{n \ln T}{tn \log tn} = \frac{\ln T}{t \log tn} = \frac{(\ln tn + \ln \log tn}{t \log tn}$$

$$= \frac{1}{t} + \frac{\ln \log tn}{t \log tn} = o(\frac{1}{t})$$

A Comparison of Fingerprinting Techniques

- verify the equality of two strings $a = (a_1, \dots, a_n)$ and $b = (b_1, \dots, b_n)$ with same alphabet.
- encode the alphabet symbols using the set of numbers r = {0, 1, ... ,k-1}, where k = $|\Sigma|$

Define :

$$A(z) = \sum_{i=0}^{n-1} a_i z^i \qquad B(z) = \sum_{i=0}^{n-1} b_i z^i$$

- two strings are polynomials with integer coefficients and degree at most n.

$$(a_1, \dots, a_n) = (b_1, \dots, b_n) \Leftrightarrow A(z) = B(z)$$

fingerprinting technique1:

- Fix some prime p > 2n,k. A(z),B(z) are polynomials over field Z_p and evaluate A(z), B(z) at a random point $r \in Z_p$

fingerprinting technique2:

- Fix z =2 and choose a random prime p. Evaluate A(2),B(2) using arithmetic modulo p.

- **both techniques reduce the problem of comparing n bits to that of comparing log n bits**

Pattern Matching

- Given a string of text $X = x_1x_2 \dots x_n$ and a pattern , $Y = y_1y_2 \dots y_m$ where $m \leq n$, determine whether or not the pattern appears in the text.
- The pattern occurs in the text if there is a $j \in \{1,2,\dots,n-m+1\}$ *such that for* $1 \leq i \leq m$, $x_{j+i-1} = y_i$
- **Deterministic algorithm**
  - Trivial algorithm O(mn)
  - Knuth-morris-pratt algorithm O(m+n)
- **Randomized monte carlo algorithm**
  - O(m+n) time and error probability $< \frac{1}{n}$

- Define the string $X(j) = x_j x_{j+1} \ldots x_{j+m+1}$ as the sub-string of length m in X that starts at position j.
- A match occurs if there is a choice of j

  $1 \leq j \leq n - m + 1 \quad that \quad Y = X(j)$
- randomized algorithm will choose a fingerprint function F and compare F(Y) with each of the fingerprints F(X(j)).
- An error occurs if F(Y) = F(X(j)) but $Y \neq X(j)$
- for any string $Z \in \{0,1\}^m$, interpret Z as an m-bit integer and define $F_p(Z) = Z \bmod p$

A randomized algorithm

- pick a random prime p $\in$ [2, ..., T]
- compute $F_p(Y) = Y \bmod p$
- for j = 1; j <= n − m + 1; j = j + 1 do
- Compute $F_p(X(j))$
- If $F_p(Y) = F_p(X(j))$
- then output “match?” and halt
- output “no match!”

- By Theorem 7.5,the probability of such a false match is:

$$pr[F_p(Y) = F_p(X(j))|\ Y \neq X(j)] \leq \frac{m}{\pi(T)} = O(\frac{m \log T}{T})$$

- the probability that a false match occurs for any of the at most n values of j is O((nmlogT)/T).
- With $T = n^2 m \log(n^2 m)$

$$pr[\ a\ false\ match\ occurs] = O\left(\frac{nmlog(n^2 m \log(n^2 m))}{n^2 m \log(n^2 m)}\right)$$

$$= O\left(\frac{1}{n} + \frac{\log\log(n^2 m)}{n \log(n^2 m)}\right) + O\left(\frac{1}{n}\right)$$

- For 1<= j<= n-m+1

$$X(j+1) = 2\left[X(j) - 2^{m-1}x_j\right] + x_{j+m}$$

- then $F_p(X(j+1)) = 2\left[F_p(X(j)) - 2^{m-1}x_j\right] + x_{j+m} mod\ p$

- given the fingerprint of X(j), the incremental cost of computing the fingerprint of X(j + 1) is 0(1)
- the total time is O(n + m)
- **Theorem 7.6**: The Monte Carlo algorithm for pattern matching requires O(n + m) time and has a probability of error 0 (1/ n) .

Las Vegas algorithm

- When a false match occurs, we detect it and abandon the whole process and then use the O(nm) running time deterministic algorithm to find match.

- The new algorithm does not make any errors and has expected running time:

$$T(n) = (1 - \frac{1}{n})O(m+n) + \frac{1}{n}\ O(nm) = O(m+n)$$

Verifying Graph Non-Isomorphism

Definition 7.2:

Let $G_1(V, E_1)$ and $G_2(V, E_2)$ be two graphs on the same set of labeled vertices V = {1, ... ,n}. The two graphs are said to be isomorphic if there exists a permutation $\pi \in s_n$ such that an edge (i,j) $\in E_1$ if and only if the edge $(\pi(i), \pi(j)) \in E_2$; the permutation π is referred to isomorphism from G_1 to G_2 . Two graphs are non-isomorphic there does not exist any isomorphism from one graph to the other.

Verifier V:

- picks index $i \epsilon$ {1,2} and permutation $\sigma \epsilon s_n$, both uniformly at random;
- computes H = $\sigma(G_i)$;
- specifies H to the prover P and asks for an index j such that H is isomorphic to G_i ;

Prover P: responds with an index j;

Verifier V: if j = i then it accepts that G_1 and G_2 are non-isomorphic, else it rejects.

Theorem 7.7:

If G_1 and G_2 are non-isomorphic, an honest prover P can ensure that V will accept; otherwise, for any (possibly maliciously dishonest) prover P',

the probability that V accepts is 1/2.