Great Theoretical Ideas In Computer Science		
Anupam Gupta		CS 15-251 Fall 2005
Lecture 14	October 13, 2005	Carnegie Mellon University

Polynomials, Secret Sharing, And Error-Correcting Codes

$P(X) = \_X^3 + \_X^2 + \_X^1 + \_$

Polynomials in one variable over the reals

$P(x) = 3x^2 + 7x - 2$

$Q(x) = x^{123} - \frac{1}{2}x^{25} + 19x^3 - 1$

$R(y) = 2y + \sqrt{2}$

$S(z) = z^2 - z - 1$

$T(x) = 0$ $W(x) = \pi$

No Formal Power Series this time, just finite polynomials.

Representing a polynomial

A degree-d polynomial is represented by its (d+1) coefficients:

$P(x) = a_d x^d + a_{d-1} x^{d-1} + \ldots + a_1 x^1 + a_0$

The numbers $a_d, a_{d-1}, \ldots, a_0$ are the coefficients.

E.g. $P(x) = 3x^4 - 7x^2 + 12x - 19$

Coefficients are:

Are we working over the reals?

We could work over any "field"
(set with addition, multiplication, division defined.)

E.g., we could work with the rationals, or the reals.

Or with Z_p, the integers mod prime p.

In this lecture, we will work with Z_p

The Set Z_p for prime p

$Z_p = \{0, 1, 2, \ldots, p-1\}$

$Z_p^* = \{1, 2, 3, \ldots, p-1\}$

Simple Facts about Polynomials

Let P(x), Q(x) be two polynomials.

The sum P(x)+Q(x) is also a polynomial.
(i.e., polynomials are "closed under addition")

Their product P(x)Q(x) is also a polynomial.
("closed under multiplication")

P(x)/Q(x) is not necessarily a polynomial.

Evaluating a polynomial

Suppose:

$P(x) = a_d x^d + a_{d-1} x^{d-1} + \ldots + a_1 x^1 + a_0$

E.g. $P(x) = 3x^4 - 7x^2 + 12x - 19$

$P(5) = 3\times5^4 - 7\times5^2 + 12\times5 - 19$

$P(-1) = 3\times(-1)^4 - 7\times(-1)^2 + 12\times(-1) - 19$

$P(0) = -19$

The roots of a polynomial

Suppose:

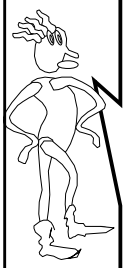
$P(x) = a_d x^d + a_{d-1} x^{d-1} + \ldots + a_1 x^1 + a_0$

Definition: r is a "root" of P(x) if P(r) = 0

E.g., $P(x) = 3x + 7$ root = -(7/3).

$P(x) = x^2 - 2x + 1$ roots = 1, 1

$P(x) = 3x^3 - 10x^2 + 10x - 2$ roots = 1/3, 1, 2.



The Single Most Important Fact About Low-degree Polynomials

A <u>non-zero</u> degree-d polynomial P(x) has at most d roots.

A Crucial Implication

Two polynomials P(x) and Q(x) of degree at most d.
Suppose $x_1, x_2, \ldots, x_{d+1}$ are d+1 points such that

$P(x_k) = Q(x_k)$ for all k = 1,2,…,d+1

Then P(x) = Q(x) for <u>all values</u> of x.

Proof:

If you give me pairs $(x_1, y_1), (x_2, y_2), \ldots, (x_{d+1}, y_{d+1})$

then there is <u>at most one</u> degree-d polynomial P(x) such that

$P(x_k) = y_k$ for all k

Hmm: <u>at most one</u>.

So perhaps there are no such degree-d polynomials with

$P(x_k) = y_k$

for all the d+1 values of k

Lagrange Interpolation

Given any (d+1) pairs $(x_1, y_1), (x_2, y_2), \ldots, (x_{d+1}, y_{d+1})$

then there is <u>exactly one</u>
degree-d polynomial P(x) such that

$P(x_k) = y_k$ for all k

k-th "Switch" polynomial

k-th "Switch" polynomial

Adding them together

The Lagrange Polynomial

Example

Input: (0,1), (1,2), (2,9)

Switch polynomials

$h_1(x) = (x-1)(x-2)/(0-1)(0-2) = \frac{1}{2}(x-1)(x-2)$
$h_2(x) = (x-0)(x-2)/(1-0)(1-2) = x(x-2)/(-1)$
$h_3(x) = (x-0)(x-1)/(2-0)(2-1) = \frac{1}{2}x(x-1)$

$P(x) = 1 \times h_1(x) + 2 \times h_2(x) + 9 \times h_3(x)$
$= 3x^2 - 2x + 1$

To recap:

If you give me pairs
$(x_1, y_1), (x_2, y_2), \ldots, (x_{d+1}, y_{d+1})$

then there is exactly one
degree-d polynomial P(x) such that

$P(x_k) = y_k$ for all k

(And I can find this polynomial P(x)
using Lagrange interpolation.)

Two different representations

$P(x) = a_d x^d + a_{d-1} x^{d-1} + \ldots + a_1 x^1 + a_0$
can be represented either by

a) d+1 coefficients
$a_d, a_{d-1}, \ldots, a_2, a_1, a_0$

b) Its value at any d+1 points
$P(x_1), P(x_2), \ldots, P(x_d), P(x_{d+1})$
(e.g., P(0), P(1), P(2), …, P(d+1).)

Converting between the two representations

Coefficients to Evaluation:

Evaluation to Coefficients:

Some representations are better for some operations

Adding two polynomials:

P(x) can be represented by
a) d+1 coefficients
$a_d, a_{d-1}, \ldots, a_2, a_1, a_0$
b) Its value at any d+1 points
$P(x_1), P(x_2), \ldots, P(x_d), P(x_{d+1})$
(e.g., P(0), P(1), P(2), …, P(d+1).)

Some representations are better for some operations

Multiplying two polynomials:

P(x) can be represented by
a) d+1 coefficients
$a_d, a_{d-1}, \ldots, a_2, a_1, a_0$
b) Its value at any d+1 points
$P(x_1), P(x_2), \ldots, P(x_d), P(x_{d+1})$
(e.g., P(0), P(1), P(2), …, P(d+1).)

And some representations are better for other operations

Evaluating the polynomial
at some other point:

P(x) can be represented by
a) d+1 coefficients
$a_d, a_{d-1}, \ldots, a_2, a_1, a_0$
b) Its value at any d+1 points
$P(x_1), P(x_2), \ldots, P(x_d), P(x_{d+1})$
(e.g., P(0), P(1), P(2), …, P(d+1).)

The value-representation is tolerant to "erasures"

I want to send you a polynomial P(x) of degree d.

Suppose your mailer drops my emails once in a while.

Now hang on a minute!

Why would I ever want to send you a polynomial?

The value-representation is tolerant to "erasures"

I want to send you a polynomial P(x) of degree d.

Suppose your mailer drops my emails once in a while.

Say, I wanted to send you a message "hello"
I could write it as
"8 5 12 12 15"

and hence as
$8x^4 + 5x^3 + 12x^2 + 12x + 15$

The value-representation is tolerant to "erasures"

I want to send you a polynomial P(x) of degree d.

Suppose your mailer drops my emails once in a while.

I could evaluate P(x) at (say) n > d+1 points and send
<k, P(k)>
to you for all k = 1, 2,…,d, …,n.

As long you get at least (d+1) of these,
choose any (d+1) of the ones you got, and reconstruct P(x).

But is it tolerant to "corruption" ?

I want to send you a polynomial P(x).

Suppose your mailer **corrupts** my emails once in a while.

E.g., suppose $P(x) = 2x^2 + 1$, and I chose n = 4.
I evaluated P(0) = 1, P(1) = 3, P(2) = 9, P(3) = 19.
So I sent you <0,1>, <1, 3>, <2, 9>, <3,19>

Corrupted email says <0,1>, <1, 2>, <2, 9>, <3, 19>

You choose <0,1>, <1,2>, <2,9>
and get Q(x) =

Error-Detecting Representation

The above scheme does detect errors!

If we send the value of degree-d polynomial P(x)
at $n \geq d+1$ different points,

$\langle x_1, P(x_1)\rangle$, $\langle x_2, P(x_2)\rangle$, … , $\langle x_n, P(x_n)\rangle$

then we can detect corruptions
as long as there fewer than (n-d) of them

Why? If only n-d-1 corruptions, then d+1 correct points!

Also Error **Correcting** Representation

As long as fewer than (n-d)/2 corruptions
then can get back the original polynomial P(x) !!!

Error Correcting Codes (ECCs)

(We don't need to know which ones are corrupted.
Just that there are < (n-d)/2 corruptions.)

We can do this in class if we have enough time at the end…

Now for something quite different…

Secret Sharing

Missile has random secret number S encoded into its hardware. It will not arm without being given S.

n officers have memorized a private, individual "share".

Any k out of n of them should be able to assemble their shares so as to obtain S.

Any ≤ k-1 of them should not be able to jointly determine any information about S.

A k-out-of-n secret sharing scheme

Let S be a random "secret" from Z_p

Want to give shares $Z_1, Z_2, \ldots, Z_n$ to the n officers such that:

a) if we have k of the Z_i's, then we can find out S.

b) if we have k-1 Z_i's, then any secret S is equally likely to have produced this set of Z_i's.

Our k-out-of-n S.S.S.

Let S be a random "secret" from Z_p

Pick k-1 random coefficients $R_1, R_2, \ldots, R_{k-1}$ from Z_p

Let $P(x) = R_{k-1} x^{k-1} + R_{k-2} x^{k-2} + \ldots + R_1 x^1 + S$

For any j in {1,2,…,n}, officer j's share $Z_j = P(j)$

Our k-out-of-n S.S.S.

Let S be a random "secret" from Z_p
Pick k-1 random coefficients $R_1, R_2, \ldots, R_{k-1}$ from Z_p
Let $P(x) = R_{k-1} x^{k-1} + R_{k-2} x^{k-2} + \ldots + R_1 x^1 + S$
For any j in {1,2,…,n}, officer j's share $Z_j = P(j)$

P(0) = where P hits y-axis = S.

P(x) chosen to be a random degree k-1 polynomial given that f hits the y-axis at S.

Since S is random, each such polynomial is equally likely to be chosen

Our k-out-of-n S.S.S.

Let S be a random "secret" from Z_p
Pick k-1 random coefficients $R_1, R_2, \ldots, R_{k-1}$ from Z_p
Let $P(x) = R_{k-1} x^{k-1} + R_{k-2} x^{k-2} + \ldots + R_1 x^1 + S$
For any j in {1,2,…,n}, officer j's share $Z_j = P(j)$

If k officers get together, they can figure out P(x)
And then evaluate P(0) = S.

Our k-out-of-n S.S.S.

Let S be a random "secret" from Z_p
Pick k-1 random coefficients $R_1, R_2, \ldots, R_{k-1}$ from Z_p
Let $P(x) = R_{k-1} x^{k-1} + R_{k-2} x^{k-2} + \ldots + R_1 x^1 + S$
For any j in {1,2,…,n}, officer j's share $Z_j = P(j)$

If k-1 officers get together, they know P(x) at k-1 different points.

For each value of S', we can get a unique polynomial P' passing through their points, and P'(0) = S'.

And so each S' equally likely!!!

Study Bee

Polynomials
- Fundamental Theorem of polynomials:
  - Degree-d polynomial has at most d roots.
  - Two different deg-d polys agree on ≤ d points.

Lagrange Interpolation:
- Given d+1 pairs (x_k, y_k), can find unique poly P such that $P(x_k) = y_k$ for all these k.
- Gives us alternative representation for polys.

Many Applications of this representation
- Error detecting/correcting codes
- Secret sharing.