Lecture # 12

Lecturer: Prof. Allan Borodin

Scribe: Yeleiny Bonilla

Agenda:

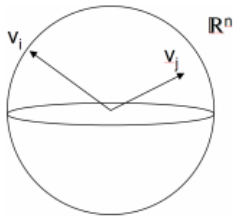
I. Finish discussion of Vector Program Relaxation for Max-Cut problem.
II. Briefly discuss same approach for Max-2-Sat.
III. The constructive Lovasz Local Lemma (LLL).
IV. The Miller – Rabin primality testing.
V. Multiplicative update.

I. Vector Program Relaxation for Max Cut problem

We can write the Max Cut problem as:

$$\text{maximize } \sum_{i,j\in E} w_{ij}\, 1/2(1-v_i v_j) \quad (1)$$
$$\text{subject to } ||v_i||^2 = 1$$
$$v_i \in R^n$$

Consider the vectors are in a sphere



*

We are going to take a random unit vector $r \in R^n$. Set $y_i = +1$ if $(v_i\, r \geq 0)$

As the inner product of two vectors is equal $v_i v_j = \cos(\theta_{ij})$ and $0 \leq \theta \leq \pi$. Then substituting in equation (1), we can re-write the problem as:

$$\text{maximize } \sum_{i,j\in E} w_{ij}\, 1/2(1-\cos(\theta_{ij}))$$

Where $\cos(\pi) = -1$
$\cos(\pi/2) = 0$
$\cos(0) = 1$

Claim: The expected value of the rounded solution will be

$$E[\text{rounded solution}] \geq \alpha \sum_{i,j\in E} w_{ij}\, 1/2(1-\cos(\theta_{ij}))$$

Main Claim: $\text{Prob}_r [v_i \text{ and } v_j \text{ are separated by r}] = \theta_{ij} / \pi$ (Proof by picture above *)

➔ To get the desired expectation we want $\theta_{ij} / \pi \geq \alpha ((1 - \cos (\theta_{ij})) /2)$
$\alpha = \min_{0 \leq \theta \leq \pi} 2/\pi * (1 - \cos (\theta))/ \theta) \geq 0.87856$

II. Vector Program Relaxation for Max-2-Sat problem

$y_0 \ldots\ldots y_i \in_{0 \leq i \leq n} \{+1, -1\}$
$y_i \sim$ propositional x_i

Interpretation: $x_i = \{$true if $y_i = y_0$
false if $y_i = - y_0\}$

We are trying to maximize:

$$\text{maximize}_{\tau} \sum_C \text{val}_{\tau} (C)$$

val(C) -> {1 if C/ τ = true
0 if C/ τ = false}

If $C = x_i$ then $\text{val}(C) = (1+ y_i y_0) /2$

If $C = \bar{x}_i$ then $\text{val}(C) = 1 – (yi\ y_0)/2$

And if $C = x_i \vee x_j$ then

$\text{val}(C) = 1 – \text{val} (\bar{x}_i) * \text{val} (\bar{x}_j)$
$= a ((1 + y_i y_0) /2) + b ((1 - y_i y_0) /2)$

maximize $\sum_{i \in j} a_{ij} (1+ y_i y_j) + b_{ij} (1- y_i y_j)$

Relaxing $yi\ y_j$ to $v_i . v_j$, with $v_i \in R^{n+1}$, $\| v_i\| =1$, $0 \leq i \leq n$.

Doing the calculations we get the same $\alpha \sim 0.87856$ approximation.

III. The constructive Lovasz Local Lemma (LLL)

Having a series of events $E_1,\ldots., E_m$

Prob $[E_i] = p < 1$

Prob $[\bar{E}_1 \wedge \bar{E}_2 \wedge \ldots.. \wedge \bar{E}_m] > 0$, if all E_i are independent

Suppose that each E_i occurs with probability at most p, and such that each event is independent of all the other events except for at most d of them.

If

$$ep(d+1) \leq 1$$

(Where e = 2.718... is the base of natural logarithms), then there is a nonzero probability that none of the events occurs.

Consider an instance of the LLL,

$F= C_1 \wedge C_2 \wedge \ldots \wedge C_m$

F in an exactly K – CNF,

$E_i = C_i$ is not satisfied by a random τ, then

Prob $[E_i] = 1/2^k$

$\bar{E}_i$ means C_i is satisfied, then

Prob $[\bar{E}_1 \wedge \bar{E}_2 \wedge \ldots \wedge \bar{E}_m]$ = Prob $_\tau$ [F/ τ is satisfied] >0

Let *d* be the number of clauses that share a variable (with a given clause C), then to apply the LLL we want:

$d + 1 \leq 2^k / e$ which implies that $d < 2^k/e$.

The Constructive Proof:
(Proof by Moser and Gabor, Tardos + Moser)

Chose any random τ

Solve,
while there exists a clause C not satisfied by τ
Call Fix(C)
end while

Fix(C)

Randomly set the bits in C
While there is a neighboring clause D that is unsatisfied
Call Fix (D)
End while

Theorem: Let r be the size of neighborhood. If r is not to big: $r \leq 2^k/8$, then with high probability the algorithm terminates in O (mlogm) calls to Fix(C) and hence found a satisfying assignment.

Proof:

Suppose the algorithm takes at least *s* recursive calls to fix,

n + s*k bits describes algorithm up to the s^{th} cell at which time we have some true assignment τ'.

Using Kolmogorov complexity, we state the fact that *most random strings cannot be compressed.*

Now we say that if r is sufficiently small k – log v – c > 0

Then we can describe these n + s*k bits in a compressed way.

n bits to describe τ'.
s calls to fix
C_1 ….. C_s clauses being fixed.

Claim: Solve has at most *m* clauses.

Any C' satisfied before Fix(C) that is called in Solve remains satisfied.

Claim: We can recover original n + s*k bits using

n	+	m logm	+	s (logr + c) ≥	n + s*k
(for τ')		*(calls to fix in solve)*			

m logm ≥ s (k – logr – c)

(Note: Here it is not proved, but the algorithm does not always terminates)

IV. Primality testing

Some background in primality tests and authors:

- *The **Solovay–Strassen primality test**, developed by Robert M. Solovay and Volker Strassen, is a probabilistic test to determine if a number is composite or probably prime. It has been largely superseded by the Miller–Rabin primality test, but has great historical importance in showing the practical feasibility of the RSA cryptosystem.*
- *The **Miller–Rabin primality test** or **Rabin–Miller primality test** is in an algorithm which determines whether a given number is prime, similar to the Fermat primality test and the Solovay–Strassen primality test. Its original version, due to Gary L. Miller, is deterministic, but the determinism relies on the unproven generalized Riemann hypothesis; Michael O. Rabin modified it to obtain an unconditional probabilistic algorithm.*

In general those authors gave a one sided randomized algorithm.

Prob[ALG says N prime | N composite] ≤ ½

Prob [ALG says N composite | N prime] = 0

➔ Composite testing ∈ RP

Except for a very small (but still infinite) class of numbers, there is a very simple randomized algorithm:

➔ Fernat's little theorem: N prime implies $a^{N-1} = 1 \bmod N$, where gcd(a,N) = 1

Lagrange's Theorem: If S is a subgroup of a group G, then |S| divides |G|.

Our group: $Z^*_N = \{a \mid \gcd(a,N) = 1$, under mullt' mod N$\}$

$S = \{a \mid \gcd(a,N) = 1$ and $a^{n-1} \bmod N = 1$, $1 \le a \le n-1\}$ is a subgroup.

False test: choose $a \in Z^*_N$ randomly.

If $a^{k-1} \bmod N = 1$ -> Output prime

Else -> Output composite

Carmichael number (AKA false primes):

For all $a \in Z^*_N$, $a^{k-1} \bmod N = 1$, and N is composite.

N- Carmichael -> $N=N_1N_2N_3$, N_i square free. Gcd (N_i, N_J) = 1

- If N is prime, then Z^*_N is a field, and 1 has exactly 2 square roots {-1, +1}
- If N is odd then N-1 is even -> $N-1 = 2^t u$, with *u* odd

Algorithm:

Choose $a \in Z^*_N$ randomly,

$x_0 = a^u \bmod N$, $x_t = a^{N-1} \bmod N$

for i=1 until t

$x_i = x^2_{i-1} \bmod N$

if x_i does not belong to {-1, +1} then output composite.

end for

if $x_i \neq 1$ then output composite

else output prime.

Chinese remainder theorem (CRT):

$\gcd(N_1N_2) = 1$ then for all u, v, $N = N_1N_2$

Exist λ: $\lambda = u \bmod N_1$

$\lambda = v \bmod N_2$ and $0 \leq \lambda < N_1 N_2$.

V. Multiplicative update

Suppose we have n experts who every day are predicting the value of a {0, 1} event.

Let m^t_i be the number of errors by experts i in first t days. We want an algorithm that will do well compared to the best expert.

w_i will be the weight of i^{th} expert. Initially $w^0_i = 1$.

Algorithm:

For j=1 until t (for each day)

For each **i**

if prediction of expert **i** on day **j** is wrong then $w^j_i = w^{j-1}_i (i- \mathcal{E})$

else $w^j_i = w^{j-1}_i$

end for

end for

Output: the weighted majority of the expert predictions.

Claim: Let m^t be the number of errors the algorithm make. Then $m^t \leq 2\ln n/\mathcal{E} + 2(1+ \mathcal{E})m^t_i$, for all i.