Topics in Algorithms 2005
Max Cuts

Ramesh Hariharan
Undirected Unweighted Max Cut

- **Cut:** Partition of the vertex set
- **Max Cut:** Number of edges between the two sides of the partition should be maximized
Status

- Max Cut is NP-Complete!! Unlike Min-cut.

- If we can solve this problem we can solve the 3-SAT problem and we can solve any problem which can be solved in non-deterministic polynomial time by a Turing Machine.
Show that Max Cut is NP-Complete:

- Reduce from 3-SAT

- Given a 3-SAT formula $f$ find a graph $G$ and a number $k$ such that
  - $f$ is sat implies $G$ has a cut of size $k$.
  - $f$ is not sat implies $G$ has no cut of size $k$ or larger.
Reduction from 3-SAT

Reduce from 3-SAT:

- Put 2 vertices for each variable $x$, one for $x$ and one for $\overline{x}$
  - Connect pairs of vertices $x, \overline{x}$ by $k$ edges

- Put one vertex for each clause
  - Connect literals in that clause to it

The Max Cut will always split complement literal pairs why??

- Ideally, the Max Cut should now equal the number of clauses satisfied plus $nk$ where $n$ is number of variables

Problems!
- A clause could have more than 1 satisfying literal
- Clause vertices could get split as well
Reduction from 3-SAT

Reduce from 3-SAT:

- We need to ensure that each satisfied clause contributes the exact same amount to each cut and each unsatisfied clause contributes a lesser amount. How do we ensure this??

- We need to handle splitting of clause vertices. How?

Trick

- Work with heterogeneous 3-SAT instead
- 3-SAT in which satisfiability requires the assignment to have at least one true and one false in each clause
- Heterogeneous 3-SAT is also NP-Complete. Exercise??
Reduction from hetero-3-SAT

Reduce from hetero-3-SAT with m clauses and n variables:

- Put 2m vertices for each variable \( x \), m for \( x \) and m for \( \overline{x} \)
  - There are 2 vertices associated with each clause
  - Connect all \( m^2 \) pairs of vertices \( x, \overline{x} \) by k edges each forcing them to come on different sides

- For each clause
  - Connect the three literals for this clause together.

- The Max Cut is \( kmn^2 + 2 \text{#satisfiable clauses} \). Why??
Approximation Algorithms

- Can we find a near optimal solution in polynomial time?

- E.g., What if we take a random cut? How does it compare to the min-cut?
  - Show that the expected number of edges in the random cut is $\frac{\text{#edges}}{2}$. Exercise!!
  - Thus a random cut is a expected factor $\frac{1}{2}$ approximation
Approximation Algorithms

- How about a greedy algorithm. Switch sides for a vertex if it improves the cut.
  - Will this terminate??
  - What approximation factor does it get??
Beating the $\frac{1}{2}$ factor

- Can you write max-cut as a linear/concave program (so local and global maxima are the same)?

$$\text{Max}_{e=(i,j)} \frac{(1-x_i.x_j)}{2}$$
For each vertex $x_i$, $x_i = -1$ or $x_i = 1$

- Issues
  - Not Linear. Is the objective function concave?
  - Hoe does one relax the -1,1 constraints?
    - Linear relaxation: $x_i >= -1$ and $x_i <= 1$
  - How does one solve a concave problem in polynomial time?
  - How does one round linear solutions to integer solutions?
Concavity

- A twice differentiable function is concave if and only if its Hessian is negative semidefinite, i.e., for vectors $v$

\[ \delta^2 f / \delta x_i \delta x_j \leq 0 \]
Maximizing Concave Functions

There are polynomial time algorithms to maximize concave function over any convex set (e.g., one obtained by intersection of linear inequalities)

- Reference: Solving Convex Programs by Random Walks
  By Bertsimas and Vempala
Beating the $\frac{1}{2}$ factor

$$\text{Max}_{e=(i,j)} \frac{(1-x_i \cdot x_j)}{2}$$

For each vertex $x_i$, $-1 \leq x_i \leq 1$

Issues

- How does one round linear solutions to integer solutions?
- Think about this? Not sure what is known.
Another Relaxation

$$\text{Max}_{e=(i,j)} \frac{1-x_i.x_j}{2}$$

For each vertex $x_i$, $-1 \leq x_i \leq 1$

- Relax $X_i$s to be vectors or length 1
- Issues
  - Is the feasible region convex?
  - How does one round?
SemiDefinite Formulation

\[ \text{Max}_{e=(i,j)} \frac{(1-d_{ij})}{2} \]

- The feasible region is convex
- The objective function is linear
- So solvable in polynomial time
- In addition, a nice geometric interpretation
- Due to Goemans and Williamson

\( d_{ij} \) is pos semidefinite
SemiDefinite Formulation

\[ \max_{e=(i,j)} \frac{(1-d_{ij})}{2} \]

- Rounding
  - Solving the program yields a set of unit length vectors
  - Use a random hyperplane through the origin to split the vectors into 2; this gives a cut
  - How does the size of this cut compare to that of the semidef program optimum?
SemiDefinite Formulation

\[ \text{Max}_{e=(i,j)} \frac{1-d_{ij}}{2} \]

\[ d_{ij} \text{ is pos semidefinite} \]

\[ d_{ij} = x_i \cdot x_j \]

- The probability of the edge e appearing in the cut is \( \frac{\text{arccos}(d_{ij})}{\pi} \)
- So the approx factor is \( 2 \frac{\text{arccos}(d_{ij})}{\pi} \left(1 - d_{ij}\right) \)
- How low can \( \frac{2\theta}{\pi} \left(1 - \cos \theta\right) \) be?? Exercise??
References

- Bertsimas, Vempala on convex programming
- Goemans-Williamson’s paper on Max-Cut
Exercise

Max-2-SAT

Given a 2 SAT formula find an assignment which satisfies as many clauses as possible

Can you give a simple randomized algorithm followed by a SemiDefinite relaxation and then Rounding algorithm?
Thank You