SNU CSE UROP

Approximation Algorithms for Max-Cut on Structural/Geometric Classes of Graphs

Euiwoong Lee (University of Michigan)

- Max-Cut is the following basic combinatorial optimization problem:
  - Input: An undirected graph $G = (V, E)$.
  - Output: $S \subseteq V$.
  - Goal: Maximize the *value* of S, which is the number of edges between S and $V \backslash S$.
- Max-Cut has been central to the theory of computation. It is one of the first 21 problems proved to be NP-complete by Karp in 1971. Therefore, it is natural to study *approximation algorithms*, which are polynomial-time algorithms to find an approximately optimal solution. Several natural ideas lead to a 0.5-approximation, meaning that they are guaranteed to find S whose value is at least half of the optimal value.
- In 1994, Goemans and Williamson designed a beautiful approximation algorithm based on semidefinite programming (SDP), which yields a much better approximation ratio of $\alpha_{GW} := \min_{a \in [-1,+1]} \frac{2 \arccos(a)}{\pi(1-a)} \approx 0.878$. Surprisingly, in 2004, other researchers proved that no other polynomial-time algorithm can achieve a better approximation ratio assuming a complexity conjecture called the Unique Games Conjecture!
- So, the story is complete when we allow every undirected graph as a valid input. What happens if we restrict the input graph G to be from a special class of graphs? There are numerous classes of graphs researchers have studied in several fields, including planar graphs (more generally minor-free graphs), interval and chordal graphs (more generally perfect graphs), and various geometric intersection graphs. Surprisingly, the optimal approximability of Max-Cut is wide open for many of these classes of graphs (e.g., a 0.88-approximation is not known while a 0.99-approximation is not ruled out).
- The goal of this project is to design approximation algorithms or prove a hardness of approximation result for Max-Cut on interesting classes of graphs.
- Prerequisite: Basic knowledge in design and analysis of algorithms, mathematical maturity, and interest in approximation algorithms; background knowledge in graph theory or linear algebra is a plus but not required.