Seoul
National
University

Probabilistic Method

Name: Chong-kwon Kim

SCONE
Lab.

Introduction

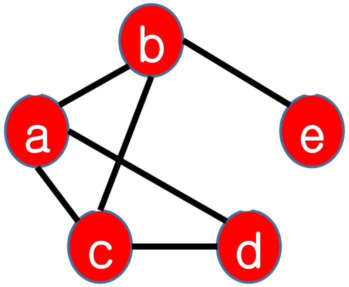
- Probabilistic method is a powerful way to prove the existence of certain objects (events)

- Basic Idea
  - If a certain object can be selected with positive probability (can be very small), then this object must exist

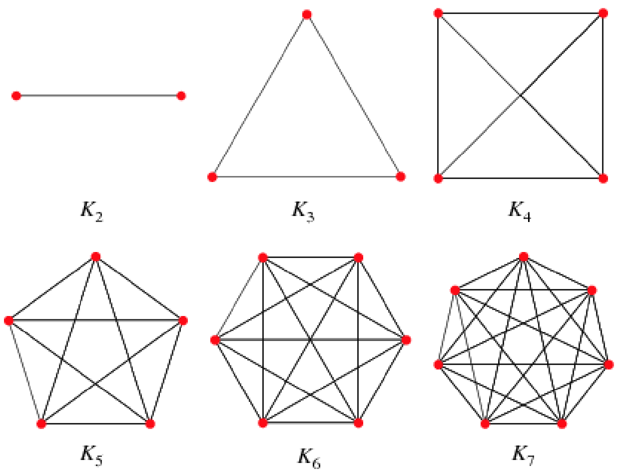
- Two techniques
  - Basic counting
  - Expectation

Graph Edge Coloring

- An example of basic counting techniques
- Graph – G(V, E)
  - V: Set of vertices (nodes)
    - {a, b, c, d, e}
  - E: Set of edges
    - {(a,b), (a,c), (a,d), (b,c), (b,e), (c,d)}



- Complete graph (Clique)
  - A graph where all node pairs are connected
  - If edges (a,e), (b,d), (c,e), (d,e) are added, then the graph becomes a complete graph
  - # edges in a complete graph of |V| = n
- k-Clique: K_k
  - A clique of k vertices



Graph Edge Coloring

- Assume we color edges with one of *two* colors
- K_k is monochromatic
  - All $\binom{k}{2}$ edges have the same color

- Theorem
  - If $\binom{n}{k} \cdot 2^{-\binom{k}{2}+1} < 1$, then it is possible to color the edges of K_n such that it has no monochromatic K_k subgraphs

- Proof
  - Let S be a sample space of all possible coloring
    - $2^{\binom{n}{2}}$ instances
  - Choose one instance (G) randomly from S
  - Equivalently, color each edge randomly with the equal probability

Graph Edge Coloring

- # K_k subgraphs in Kn
  - $N = \binom{n}{k}$ K_k
- Consider i-th K_k subgraph
- Ai: Event that i-th K_k subgraph is monochromatic
  - $\Pr(A_i) = 2 \cdot (\frac{1}{2})^{\binom{k}{2}}$
- E: Event that no monochromatic K_k subgraph

$E \equiv \bar{A}_1 \cap \bar{A}_2 \cap \cdots \cap \bar{A}_N$

- $\Pr(E) = \Pr(\overline{A_1 \cup A_2 \cup \cdots \cup A_N})$

$$= 1 - \Pr(A_1 \cup A_2 \cup \cdots \cup A_N)$$

$$\geq 1 - \sum_i \Pr(A_i)$$

$$= 1 - \binom{n}{k} \cdot 2^{-\binom{k}{2}+1} > 0$$

Example – Graph Edge Coloring

- Consider n=1000, k=20
  - First note that $n < 2^{k/2}$ at n=1000, k=20
  - To show that $\binom{n}{k} \cdot 2^{-\binom{k}{2}+1} < 1$ at n=1000, k=20
  - $\binom{n}{k} \cdot 2^{-\binom{k}{2}+1} \leq \frac{n^k}{k!} \cdot 2^{-\binom{k}{2}+1}$

$$< \frac{{2^{k/2}}^k}{k!} \cdot 2^{-\binom{k}{2}+1}$$

$$= \frac{2^{\frac{k}{2}+1}}{k!} < 1$$

Note: $n \leq 2^{k/2}$ for k≥3

Monte-Carlo vs. Las Vegas

- **Las Vegas**

Las Vegas will never cheat you

  - A random algorithm is called Las Vegas if it always produces the correct answer
  - Example
    - Random QuickSort
  - Usually, running time depends on random choices

- **Monte-Carlo**

Probability that random coloring has monochromatic K_{20}

$$\frac{2^{\frac{k}{2}+1}}{k!} < 8.5 \cdot 10^{-16}$$

  - May give wrong answer sometimes
  - Example
    - Testing the equivalence of two polynomials
  - Usually, running time is constant
  - To convert Monte-Carlo to Las Vegas, need to add a checking procedure
    - If wrong answer probability is p then retry 1/p on average

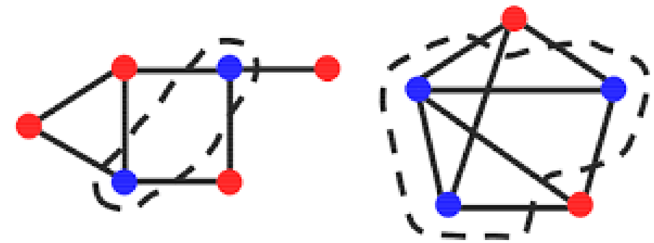
Expectation Argument

- Lemma:
  - Let X be a random variable and E[X] = μ,
  - then $\Pr(X \geq \mu) > 0$ and $\mathbf{Pr(X \leq \mu) > 0}$
- Proof
  - $\mu = E[X] = \sum x \cdot \Pr(X = x)$
  - Suppose $\Pr(X \geq \mu) = 0$

    $\mu = \sum x \cdot \Pr(X = x) = \sum_{x<\mu} x \cdot \Pr(X = x) < \mu$

Example – Large Cut

- Definitions:
  - Cut: A cut is a partition of a graph into two disjoint vertices sets V_1 and V_2



  - Size (Value, Capacity) of a cut: Size of a cut (V_1, V_2) is # edges with one end point in V_1 and another in V_2
    - More generally, sum of weights of edges connecting V_1 and V_2
  - Max Cut: A cut with maximum size

Max Cut is an NP-Hard problem

NP-Hard problem
- No efficient (polynomial) solution
- Should enumerate all possible combinations

Large Cut

- Assume a graph with n vertices and m edges
  - Obviously, Max Cut $\leq m$
- How about the *Lower bound of Max Cut* ?
- Claim: Lower bound of MaxCut is m/2

  More formally, there *must be* a cut whose size is at least m/2
- Proof
  - Construct a random cut (V_1, V_2)
    - Assign each vertex randomly into V_1 or V_2 with equal probabilities
  - Let X be the size of the random cut
  - E[X]?

How many random cuts?

Large Cut

- Proof – Cont.
  - Claim $E[X] = m/2$

    Let $X_i = 1$, if i–th edge belongs to the cut

    0, ow

    $E[X_i] = \frac{1}{2}$ (Why ??)

    $E[X] = \sum E[X_i] = m/2$

    ➔ From the previous lemma, $\Pr(X \geq m/2) > 0$

Large Cut

- Previously, we proved the existence of a cut whose size is at least m/2
- *How easy* to find such a large cut?

- Again, Let X be the size of a random cut
- Let p = Pr(X $\geq$ m/2)
  - m/2 = E[X]

$$= \sum_{x<m/2} x \cdot \Pr(X = x) + \sum_{x \geq m/2} x \cdot \Pr(X = x)$$

$$\leq (1\text{-}p)(m/2\text{-}1) + pm$$

➔Pr(X $\geq$ m/2) = p $\geq$ 1/ (m/2 + 1)

On average, (m/2 + 1) trials

- Repeatedly try random cuts until find one whose size is at least m/2

➔ Find a sub-optimal at least half of the optimal

Another Example – MAXSAT

- Definitions
  - A literal is a Boolean variable or a negated Boolean variable
    - E.g. x, ¬y
  - A clause is literals connected with ∨
    - E.g. (x ∨ ¬y)
  - A SAT formula is an expression consists of clauses connected by ∧
    - E.g. (¬ x ∨ y ∨ z) ∧ (x ∨ ¬y) ≡ (−x + y + z)•(x+ −y)

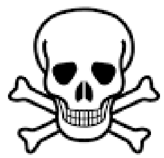
Also called CNF (Conjunctive Normal Form)

  - A SAT formula is satisfiable if there is an assignment of variables to T/F such that the value of the formula is TRUE

(¬ x ∨ y ∨ z) ∧ (x ∨ ¬y) is satisfiable
(¬x ∨ y) ∧ (¬x ∨ ¬y) ∧ (x) is not satisfiable

MAXSAT

- Determining the **satisfiability** of a SAT formula is **NP-Complete**



Caution: Do not spend too much time to find efficient solutions for the SAT problem

- **Maximum Satisfiability (MAXSAT)**
  - Maximize the # TRUE clauses
- MAXSAT is **NP-Hard** also
  - ➔ Finding the optimal solution is generally very time-consuming

- Can we get reasonable good sub-optimal solutions?
- Let m = # clauses in a SAT formula
  - Obviously, MAXSAT $\leq$ m

MAXSAT

- Theorem

  Let k = # literals in the smallest clause

  Then, there is an assignment that satisfies at least **$m \cdot (1-1/2^{k})$** clauses

- Proof

  Assume uniformly random assignment of T/F to each literal

  Let k_i be the number of literals in i-th clause

  Pr(i-th clause is TRUE) = $(1-1/2^{k_i})$ (Why??)

  # TRUE clauses = $\sum$ Pr(i-th clause is TRUE)

  $= \Sigma(1-1/2^{k_i}) \geq m \cdot (1-1/2^{k})$