Matching is as easy as Matrix Inversion

Mulmuley, Vazirani and Vazirani

Isolating Lemma

- Definition - A set system (S, F) consists of a finite set S of elements, $S = \{x_1, x_2, \dots, x_n\}$, and a family F of subsets of S i.e. $F = \{S_1, S_2, \dots, S_k\}, S_j \subseteq S \; for \; all \; 1 \leq j \leq k$
- Let us assign a weight $w_i \; to \; each \; element \; x_i \in$ S and let us define the weight of set S_j to be the sum of weight of all elements present in it
- Lemma - Let (S, F) be a set system whose elements are assigned integer weights chosen uniformly and independently from $[1, 2n]$ .Then $Probability[There \; is \; a \; unique \; minimum \; weight \; set \; in \; F] \; \geq \frac{1}{2}$

Proof of isolation lemma

- Set weights of all elements other than x_i. Define the threshold of weight for x_i to be any real number a_i such that if $w_i \leq a_i$ then x_i is contained in some minimum weight subset, S_j and if $w_i > a_i$ then x_i is in no minimum weight subset.
- If $w_i < a_i$, then the element x_i must be in every minimum weight subset. Thus, ambiguity regarding presence of x_i occurs if $w_i = a_i$, since this occurs if a minimum weight subset contains x_i and another minimum weight subset does not contain x_i . Let us define that if this occurs, x_i is a singular element

Proof of isolation lemma

- The threshold of the given element x_i was defined without reference to it's weight. Hence, both threshold and the weight associated with x_i are independent of each other.
- $Probability[w_i = a_i] \leq \frac{1}{2n}$
- S contain n elements, hence
  $Probability\ [There\ exists\ a\ singular\ element\ in\ S] \leq\ n\ * \frac{1}{2n}$
- Hence, $Probability[Existence\ of\ singular\ element] \leq 1/2$

Isolation Lemma conclusion

- As proved in earlier slide, with a probability of at least $\frac{1}{2}$, no element is singular
- If there is no singular element, there exists a unique minimum weight set
- Hence, the probability of existence of a unique minimum weight set is at least ½
- By, similar argument existence of unique maximum weight set is also at least ½

Valiant Vazirani Theorem

- This theorem concluded that the complexity of finding solutions to SAT instances having unique solutions is NP Hard under randomized reductions
- The SAT problem, which checks if a satisfying assignment exists for given Boolean expression can be reduced in polynomial time to CLIQUE problem.
- The CLIQUE problem checks for existence of a clique of given size k in a given Graph G
- Clique is a subset of vertices in G such that every two vertices are connected by an edge between them.

Valiant Vazirani theorem proof

- We try to prove the theorem using isolating lemma by showing a randomized reduction from CLIQUE to UNIQUE CLIQUE.
- Assign an independent weight w(v) to each vertex v of the graph G(V, E), which is a value in the range $[1, 2|V|]$.
- By applying isolating lemma, the maximum weight clique will be unique in graph with probability of at least ½

Valiant Vazirani theorem proof

- We have a graph $G = (V, E)$, where we aim to find a $CLIQUE$ of size k
- Let us generate a transformed graph G' such that, corresponding to vertex v present in G, G' will have $2nk + w(v)$ vertices with a clique on them.
- Also, for each edge (u, v) present in G, corresponding vertices of u and v are joined by an edge in G'
- Let r be an random integer in the range $[1, 2nk]$ and let $k' = 2nk^2 + r$.
- We have now transformed the problem to finding a clique of size k' in G'

Valiant Vazirani Theorem Proof

- By applying the isolating lemma,
- If the Graph G does not contain a CLIQUE of size k, the Graph G’ will not have a UNIQUE CLIQUE
- If the Graph G has a CLIQUE of size k, the Probability of Graph G’ having a UNIQUE CLIQUE of size k’ is higher than $\frac{1}{4n}$

NC

- NC is the set of decision problems which are solvable in polylogarithmic time on a parallel computer with polynomial number of processors.
- A problem with input size n exists in NC if there exist constants c and k such the problem can be solved in $O((\log n)^c)$ time using $O(n^k)$ parallel processors
- NC is the class of problems which can be efficiently solved on a parallel computer

RNC, RNC^2

- RNC is a class which extends the NC class with access to randomized computation
- RNC^2 is a complexity class which contains problems which can be solved with one sided error by processors which are polynomial to size of input in polylogarithmic time.
- Many problems like Matrix Inverse computation are part of the RNC^2 complexity class.
- Matrix Inverse can be computed as, given matrix M, we can compute it’s adjoint in parallel, which can then be used to invert the matrix