

Graph Algorithms

BU CS Theory Seminar
Fall 2004

What you need to know ?



- Basic Grover Algorithm
 - $F : \{0, \dots, N-1\} \rightarrow \{0,1\}$
 - “F” given as a black box – can only ask $F(i)$?
 - Find i such that $F(i)=1$
 - General method
 - Initialize
 - Repeatedly apply *Grover iteration* for k times (each iteration calls “F” once)
 - Perform measurement
 - Verify solution (one more call to “F”)

Unique Solution

- Unique i_0 such that $F(i_0)=1$
- [A fast quantum mechanical algorithm for database search. '96] There *exists* $m < \sqrt{2N}$ such that
 - Prob[success after m Grover iterations] $> \frac{1}{2}$
- [Quantum mechanics helps in searching for a needle in a haystack, 97] $m < \sqrt{N}$
- Prob of success might go down if more than m iterations are performed

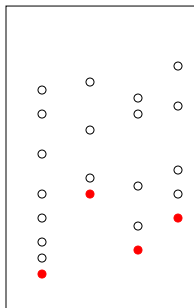
Tight bounds [BBHT98]

- [Unique solution] If $\sin^2\theta=1/N$, Prob=1 for $m=(\pi-2\theta)/4\theta$
 - Prob of failure at most $1/N$ if $m=\lfloor \pi/4\theta \rfloor$
 - For large N , $m \approx \pi/4\sqrt{N}$
- [t solutions, t is known]
 - For $m=\lfloor \pi/4\theta \rfloor$ ($\sin^2\theta=t/N$), Prob of failure $\leq t/N$
 - $m \leq \pi/4 \sqrt{N/t}$
- [t solutions, t *unknown*]
 - There is a randomized algorithm which finds a solution (chosen uniformly at random) in expected time $4.5\sqrt{N/t}$

Search Methods Used

- t solutions, $t > 0$, returns a random solution after an expected number of at most $4.5\sqrt{N}/t$ queries
- Returns a solution (if exists) with probability at least $(1-\epsilon)$, uses $O(\sqrt{[n \log 1/\epsilon]})$ queries
- Finding minimum
 - Find i such that $f(i)$ is minimum among $\{f(1), \dots, f(N)\}$
 - Current threshold index – j (= uniformly at random)
 - Repeat
 - Find index i such that $f(i) < f(j)$
 - Set $j=i$
 - $\text{prob}[\text{index } j \text{ will have rank } r] = 1/r$
 - For j of rank r , search takes $c\sqrt{N}/(r-1)$ – expected #queries to find minimum = $O(\sqrt{N})$

[SMALLEST_VAL_DISTINCT_TYPE] For function $f: \{1, \dots, N\} \rightarrow \mathbb{Z}^+$, type $g: \{1, \dots, N\} \rightarrow \mathbb{Z}^+$ and integer d , find subset $S \subseteq [N]$ consisting of d elements of distinct types which map to minimum possible function values.2



Element x is *good* for a set I , if there exists an element y in I s.t. $f(x) < f(y)$ and

- $g(x)=g(y)$ [x and y have same type, so x can replace y safely]
- **OR** $g(x) \notin g(I)$ [x has unknown type for I] and $f(y)$ is maximum in I

1. Initialize I as a set of d elements – with spurious elements having unique types and maximal values
2. Repeat...
 - a. Search for a *good* element j for current I
 - b. Replace a suitable element in I by j

SMALLEST_VAL_DISTINCT_TYPE

- Let I_k be the d -element subset and T_k be the set of good elements for I_k during k^{th} iteration
 - Maximum function value in I is always greater than the maximum function value in T (T – the good set for I , changes as I changes).
 - Elements replaced from I can never belong to any T in future – so T always shrinks in fact $T_{k+1} \subset T_k$
 - If element j from T is chosen, then it can only affect elements with higher function value than $f(j)$
- After an expected number of $O(\sqrt{dN})$ iterations (query to f), there are no good elements for I

Minimum Spanning Tree

- Borůvka's algorithm
 - Start with n spanning trees (single vertex each)
 - While #spanning trees > 1
 - Find edges e_1, \dots, e_j such that e_i is a minimum weight edge leaving T_i
 - Add the edges e_1, \dots, e_j to the trees, forming larger trees
 - After $\log(n)$ steps, only spanning tree left
- For edge (u, v) , let $g(u, v) =$ index of the tree to which u belongs; $f(u, v) =$ weight of the edge
- Find set e_1, \dots, e_j of edges of distinct types with minimum possible weights
- Classically $(E \lg V)$; use quantum search – (\sqrt{EV})

Connected Components

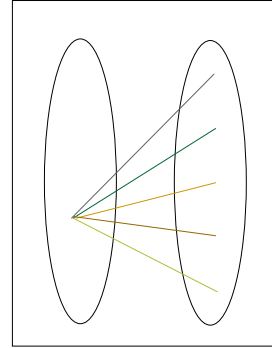
- Given undirected graph G , return a set of connected components $\{C_1, \dots, C_k\}$ such that total degree for each component $C_i \leq |C_i|^2$
 - Set of edges in conn. comp. – A (= empty)
 - Set of vertices not yet chosen – S ($=V$)
 - #components constructed so far – k ($=0$)
 - While (S not empty)
 - $v \leftarrow$ highest degree vertex in S ; $D=\{v\}$
 - For each neighbour w of v ,
 - If $w \in C_j$ ($j \leq k$), add D to C_j , remove D from S ; start next iteration of the while loop
 - Else, add w to D , (v,w) to A
 - $k=k+1$; $C_k=D$; remove D from S
 - Output k , A , $\{C_1, \dots, C_k\}$
 - $O(n)$ queries; $\deg(w) \leq \deg(v) \leq C_j$ for each comp. till then

Strong Connectivity

- Directed spanning tree
 - Edge set – A (=empty)
 - Set of reachable vertices from A – S ($=\{u\}$)
 - Stack of vertices to be processed – T ($=\{u\}$)
 - While (T not empty)
 - $u \leftarrow$ top(T)
 - $v \leftarrow$ neighbour(u) not in S , with prob $> 1-(1/20n)$
 - Uses $O(\sqrt{d(u)} \log n)$ queries
 - If v not found, remove u from T
 - Else, add (u,v) to A , add v to S , push v onto T
 - uses $O(\sqrt{nm} \log n)$; prob at least 9/10
- Strong connectivity – $O(n^{3/2} \sqrt{\log n})$

Lower Bound

- $L \subseteq \{0,1\}^*$ is a decision problem; allowed queries – i^{th} bit of instance
- Adversary chooses a relation R
 - $X \subseteq L$: set of positive instances
 - $Y \subseteq L'$: set of negative instances
 - $R \subseteq X \times Y$, such that
 - Every x is related to at least m diff y ; every y is related to at least m' diff x – so there are lots of neighbours to x , y
 - For any i , x is related to at most k different y which differ at entry i from x ; similarly for y – so there are not many neighbours which differ at some specific position
 - Then, need $\Omega(\sqrt{mm'}/kk')$ Q. queries for L



Lower Bound

- SMALLEST_VAL_DISTINCT_TYPE
 - Consider $d \times k$ boolean matrices M with a single 0 in each row ($N=dk$)
 - $f(id+j) = M[i, j]$
 - $g(id+j) = i$ if $f(id+j)=0$, d otherwise
 - Finding $d+1$ smallest values of different types have same lower bound as finding the positions of zeroes
 - $\Omega(\sqrt{dN})$ queries
- MST
 - $\Omega(\sqrt{nm})$ queries
- Strong Connectivity
 - $\Omega(\sqrt{nm})$ queries