Approximate Nearest Neighbors in Hamming Model

Algorithmic Problems Around the Web #6

Yury Lifshits
http://yury.name

CalTech, Fall'07, CS101.2, http://yury.name/algoweb.html

Self-Reduction in a Nutshell

Problem: $(1+\varepsilon)$-approximate *l*-range queries in *d*-dimensional Hamming cube

Self-Reduction in a Nutshell

Problem: $(1+\varepsilon)$-approximate l-range queries in d-dimensional Hamming cube

- Apply embedding $\{0,1\}^d$ into $\{0,1\}^k$ such that l-neighbors usually fall within $\delta_1 k$ from each other, while $(1+\varepsilon)l$-far objects are embedded at least $\delta_2 k$ from each other
- Precompute all $(\frac{\delta_1+\delta_2}{2})k$-neighbors for every point in $\{0,1\}^k$
- In search step, embed q and explicitly check all precomputed $(\frac{\delta_1+\delta_2}{2})k$-neighbors

RP: Inner product test

Single test:

- Choose random subset of positions of size $\frac{1}{2l}$
- Randomly assign 0 or 1 to every of them, the rest assign to 0, call the resulting vector r
- $h_r(p) = r \cdot p$

RP: Inner product test

Single test:

- Choose random subset of positions of size $\frac{1}{2l}$
- Randomly assign 0 or 1 to every of them, the rest assign to 0, call the resulting vector r
- $h_r(p) = r \cdot p$

Claim: there exist constants $\delta_1 > \delta_2$

- $H_d(p, s) \leq l \Rightarrow Pr[h(p) = h(q)] \geq \delta_1$
- $H_d(p, s) \geq (1 + \varepsilon)l \Rightarrow Pr[h(p) = h(q)] \leq \delta_2$

RP: Preprocessing

Inner product mapping:

- Choose k random tests $r_1, \ldots, r_k$
- Map every p into $A(p) = h_{r_1}(p) \ldots h_{r_k}(p)$

RP: Preprocessing

Inner product mapping:

- Choose k random tests $r_1, \ldots, r_k$
- Map every p into $A(p) = h_{r_1}(p) \ldots h_{r_k}(p)$

Data Structure

- Apply inner product mapping to all strings in database
- For every $v \in \{0, 1\}^k$ precompute all $(\frac{\delta_1+\delta_2}{2})k$-neighbors

RP: Search

- Compute $A(q) = h_{r_1}(q) \ldots h_{r_k}(q)$
- Retrieve and explicitly check all $(\frac{\delta_1+\delta_2}{2})k$-neighbors of $A(q)$

RP: Search

- Compute $A(q) = h_{r_1}(q) \dots h_{r_k}(q)$
- Retrieve and explicitly check all $(\frac{\delta_1+\delta_2}{2})k$-neighbors of $A(q)$

Analysis:

- Chances to miss true l-neighbor: $\exp(-\frac{\delta_1-\delta_2}{2\delta_1}k)$
- Chances to waste time on $(1+\varepsilon)l$-far neighbor: $\exp(-\frac{\delta_1-\delta_2}{2\delta_1}k)$

RP: Search

- Compute $A(q) = h_{r_1}(q) \dots h_{r_k}(q)$
- Retrieve and explicitly check all $(\frac{\delta_1+\delta_2}{2})k$-neighbors of $A(q)$

Analysis:

- Chances to miss true l-neighbor: $\exp(-\frac{\delta_1-\delta_2}{2\delta_1}k)$
- Chances to waste time on $(1+\varepsilon)l$-far neighbor: $\exp(-\frac{\delta_1-\delta_2}{2\delta_1}k)$

Thus we should take near-logarithmic k which lead to polynomial size of $\{0,1\}^k$ to be NN-precomputed

RP: Formal Claim

Theorem (Kushilevetz, Ostrovsky, Rabani, 1998)

Consider $(1+\varepsilon)$*-approximate* l*-range search in* d*-dimensional Hamming cube. Then for every* μ *there is a randomized algorithm with (roughly)* $d^2 polylog(d, n)$ *query time and* $n^{\mathcal{O}(\varepsilon^{-2})}$ *preprocessing space. For every query this algorithm answers correctly with probability at least* $1-\mu$

Thanks for your attention! Questions?

References

Course homepage http://simsearch.yury.name/tutorial.html

Y. Lifshits
The Homepage of Nearest Neighbors and Similarity Search
http://simsearch.yury.name

A. Andoni, P. Indyk
Near-Optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions. FOCS'06
http://web.mit.edu/andoni/www/papers/cSquared.pdf

E. Kushilevitz, R. Ostrovsky, Y. Rabani
Efficient Search for Approximate Nearest Neighbor in High Dimensional Spaces. STOC'98
http://www.cs.technion.ac.il/~rabani/pss/Publications/KushilevitzOR98.ps.gz