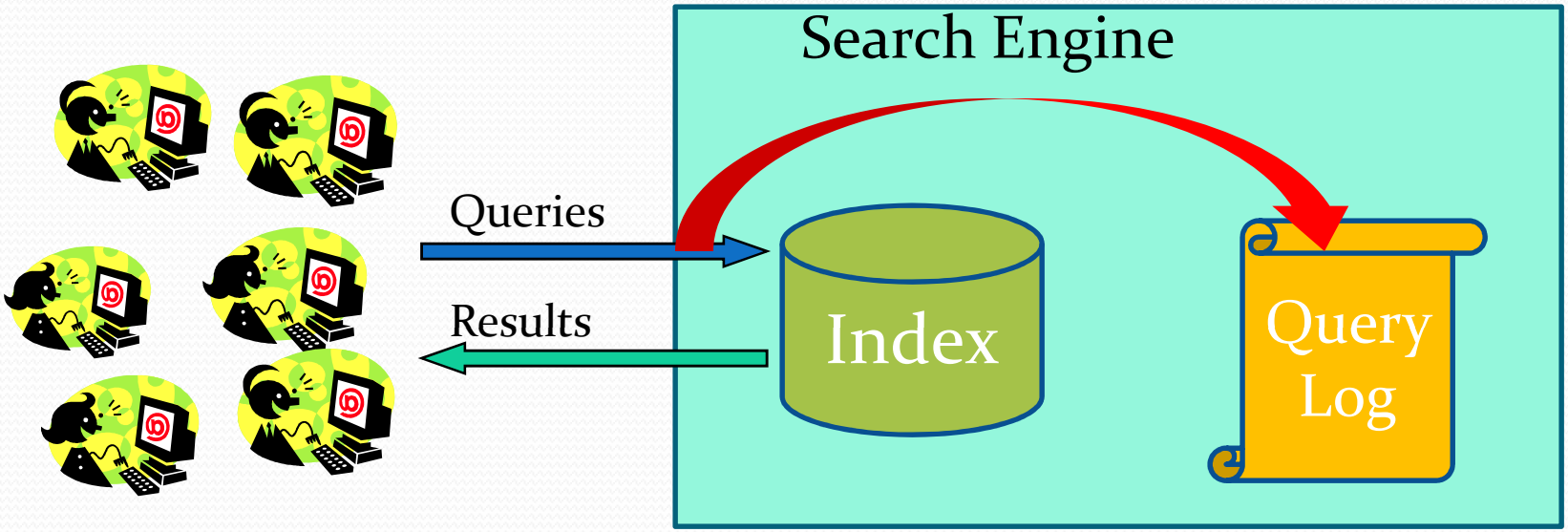
Mining Search Engine Query Logs via Suggestion Sampling

Ziv Bar-Yossef
Technion and Google

Maxim Gurevich
Technion

Search Engine Query Logs



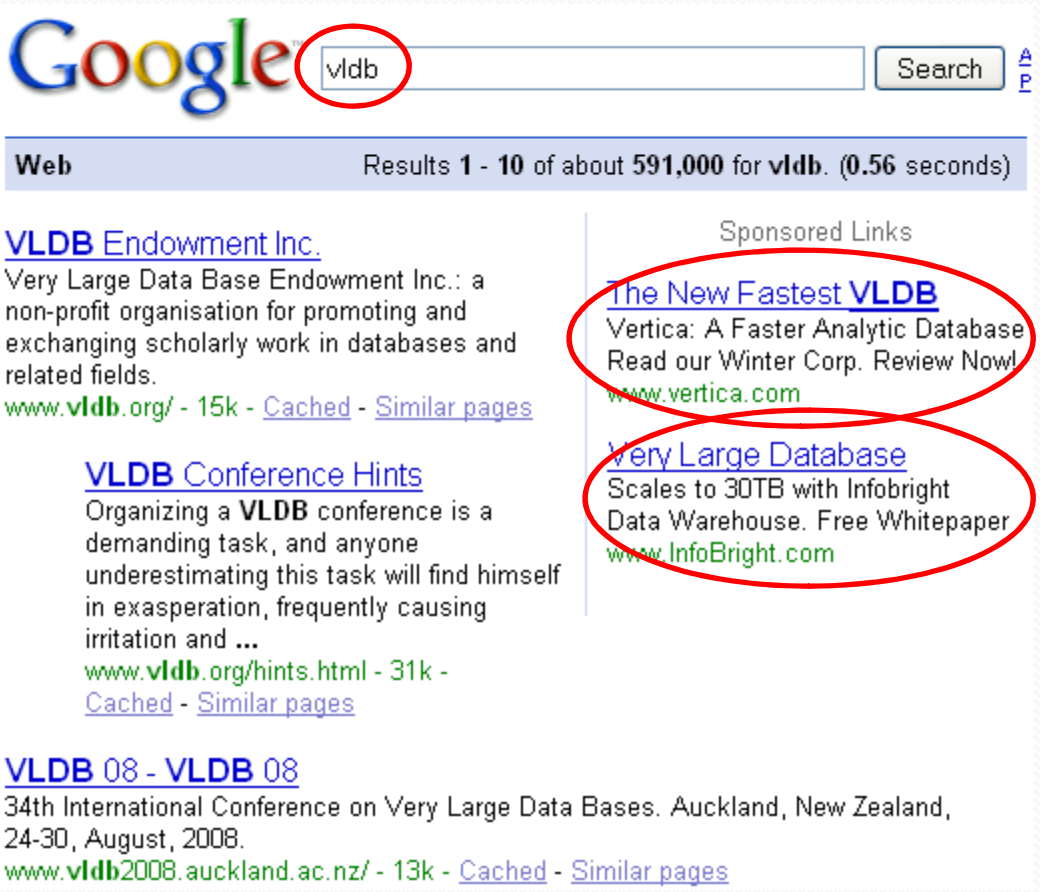
- Used by search engines to improve search results
- Contains private information
  - Of users **and** of the search engine itself
- Not disclosed by search engines

Applications of Query Log Analysis

- Keyword based advertising
- Search quality evaluation
- User modeling

Keyword Based Advertising

- Compare keyword popularity
- Track keyword popularity over time
- Find related keywords



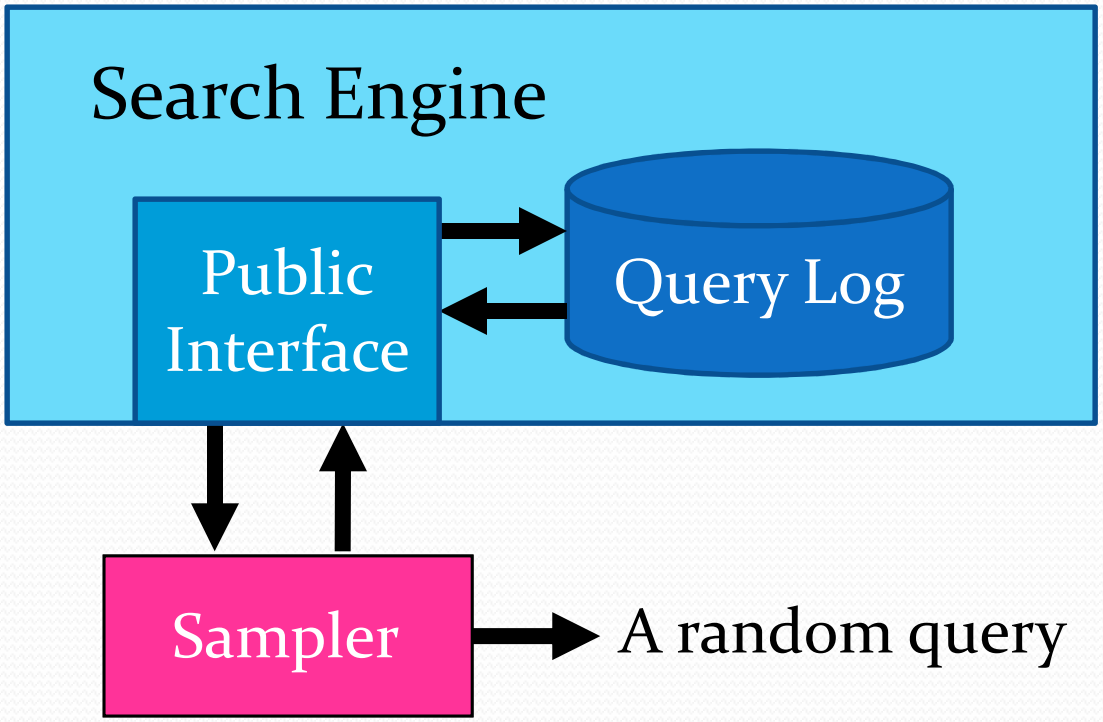
Search Quality Evaluation

- Estimate the amount of undesirable content sent to users
  - Spam
  - Stale results
  - Non-existent results
  - Pornography
  - Hate materials
  - Virus contaminated pages
  - …
- Estimate search engine bias towards
  - Authoritative sources
  - Certain domains
  - Certain languages
  - …

User modeling

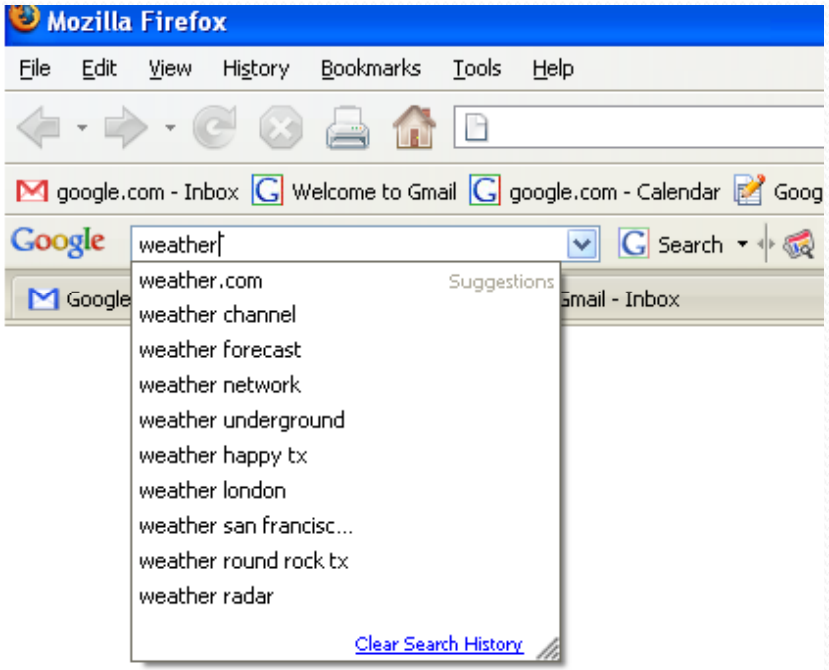
- Distribution of query types [Broder 02]
  - Navigational
  - Informational
  - Transactional
- Density of commercial queries
- Fraction of geographical queries
- …

External Query Log Mining



- Sampling (uniform or by popularity)
- Computing aggregate (privacy preserving) functions

Suggest: Trapdoor to Query Logs



- Query auto-completion
  - Suggests query completions
    - More popular first
  - Offered by major search engines
- Backed by a hidden underlying database
  - Assumption: Derived from query logs

Our Contribution

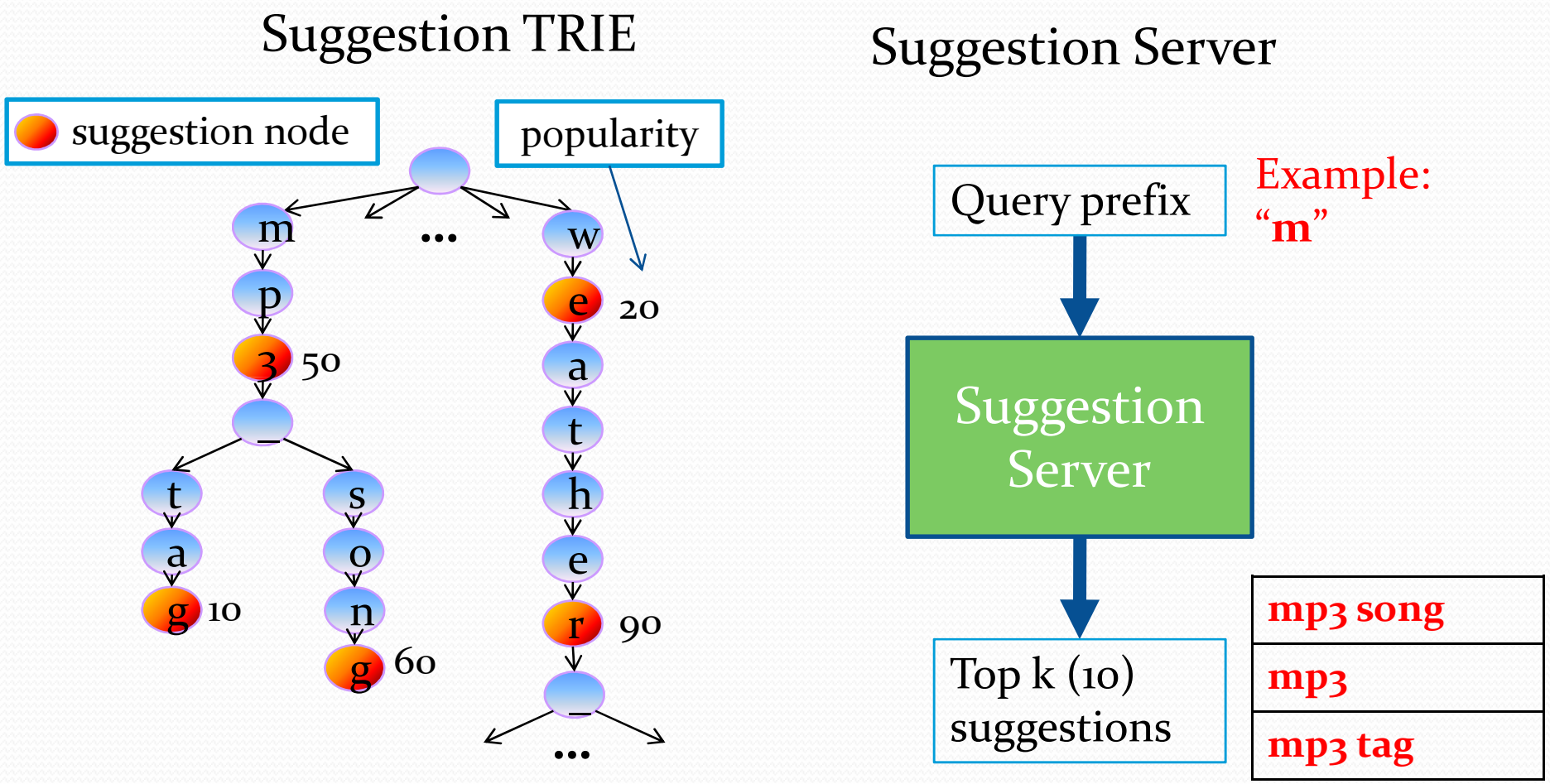
- Algorithm for sampling queries uniformly from the query log using the suggestion service
  - Practical (few suggestion requests)
  - Unbiased
- Algorithm for sampling queries from the query log proportionally to their popularity using the suggestion service
  - Practical (few suggestion requests)
  - Slightly biased

Focus of this talk

Related Work

- Sampling documents from search engine index [BarYossef et al 06,07, Broder et al 07, …]
  - Different problem and setting
- Sampling from B-trees [Wong et al 80, Olken et al 89,95]
  - B-tree specific assumptions
  - Inefficient for query log mining
- Sampling from databases behind web forms [Dasgupta et al 07]
  - Different setting, inefficient for query log mining
- Uniform sampling of combinatorial structures [Jerrum et al. 86]
  - Theoretical, the basis of our sampling algorithm

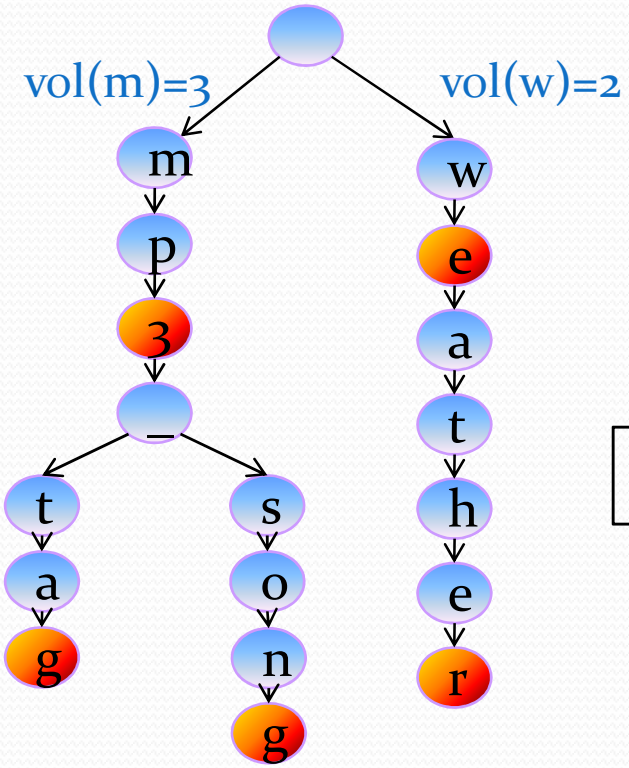
Uniform Suggestion Sampling

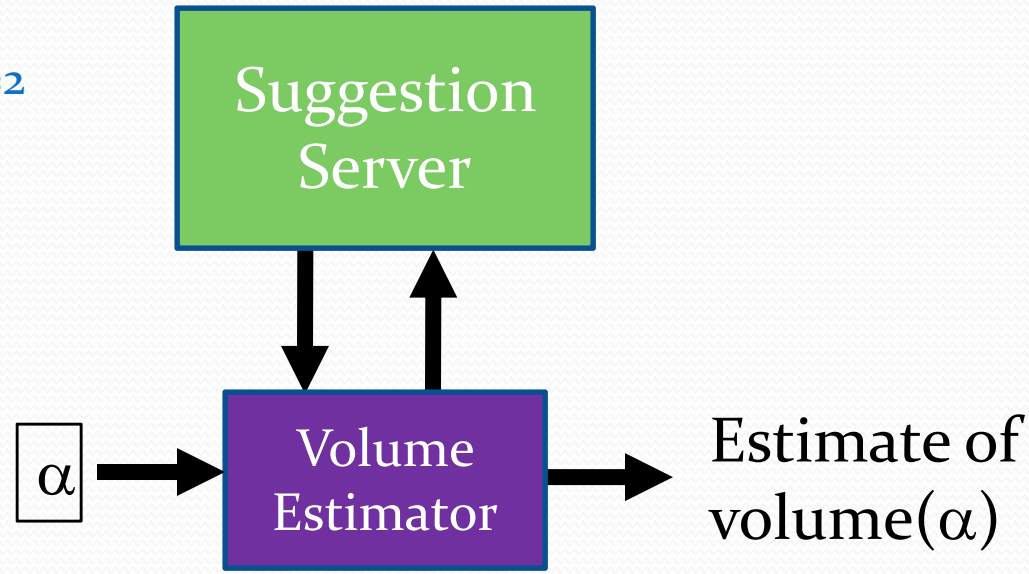


- Goal: Sample suggestion nodes uniformly

Volume Estimators

- Define: volume(α) = # of suggestions starting with α



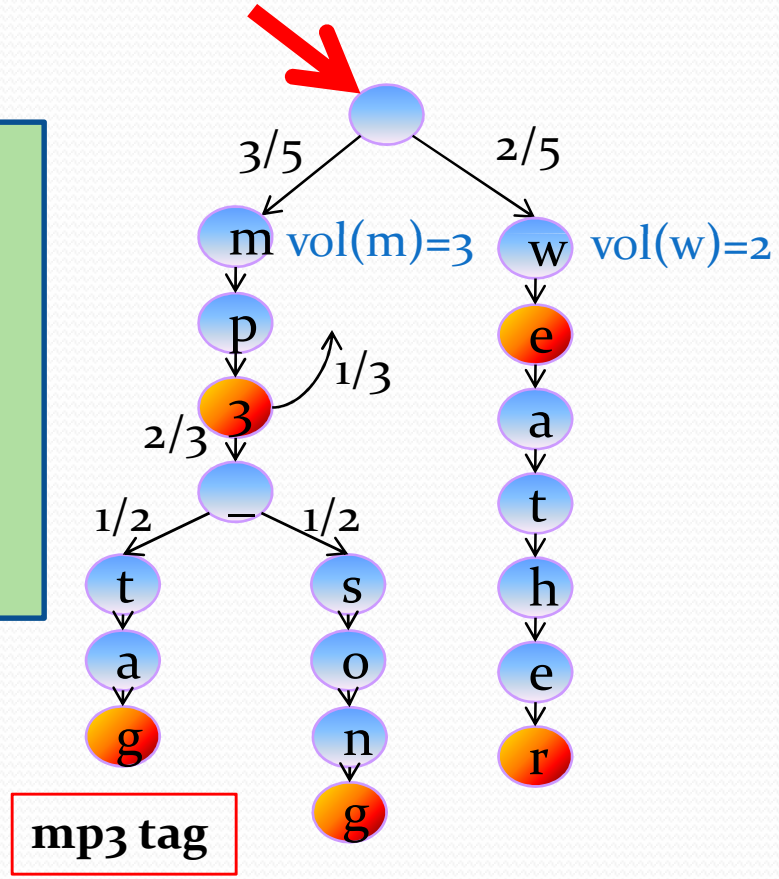


Random Walk Tree Sampling

- Assumption: we have a prefect volume estimator

- current = root
- while true
  - If current is a suggestion node
    - Return current with probability 1/volume(current)
  - Go to child x with probability ∝ volume(x)

- Theorem: If volumes are accurate, the samples are uniform



How To Estimate Volumes

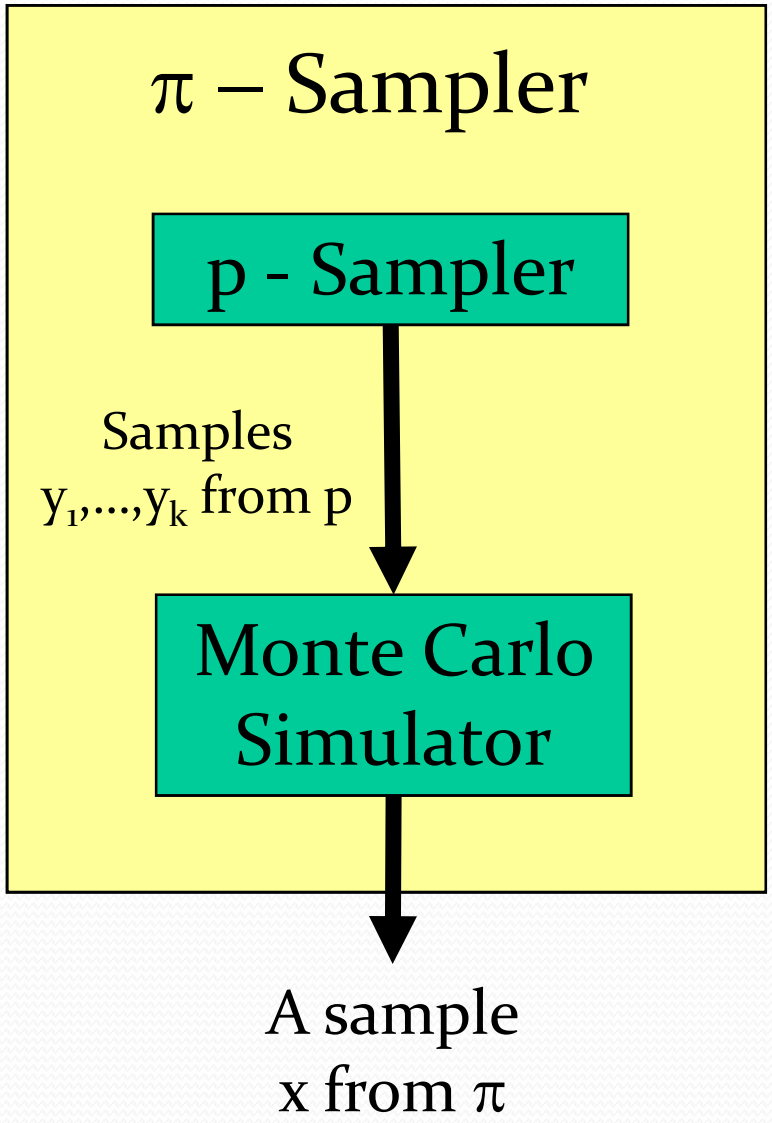
- Input: Prefix string α
- Output: $\approx$ # of suggestions starting with α
- Naïve estimator: volume(α) $\approx$ # of suggestions the server returns on α
- Popularity based estimator: volume(α) $\approx$ popularity(most popular suggestion for α)
  - Rationale: Power Law distribution of popularity (procedure for popularity estimation is included in the paper)
- Sample based estimator: volume(α) $\approx$ normalized number of suggestions for α in a previously available query log
- Final estimator aggregates all the three results

Caveat

- Random Walk Tree Sampler assumed volumes are known perfectly
- We can only approximate volumes (heuristically)

- Suggestion samples are not uniform

Monte Carlo Stochastic Simulation



- π = target distribution = uniform distribution on suggestions
  - Can deal with other target distributions as well
- p = trial distribution on S
  - Can compute p(x) for each x
  - $p \neq \pi$ but support(p) should contain support(π)
  - p should be easy-to-sample-from

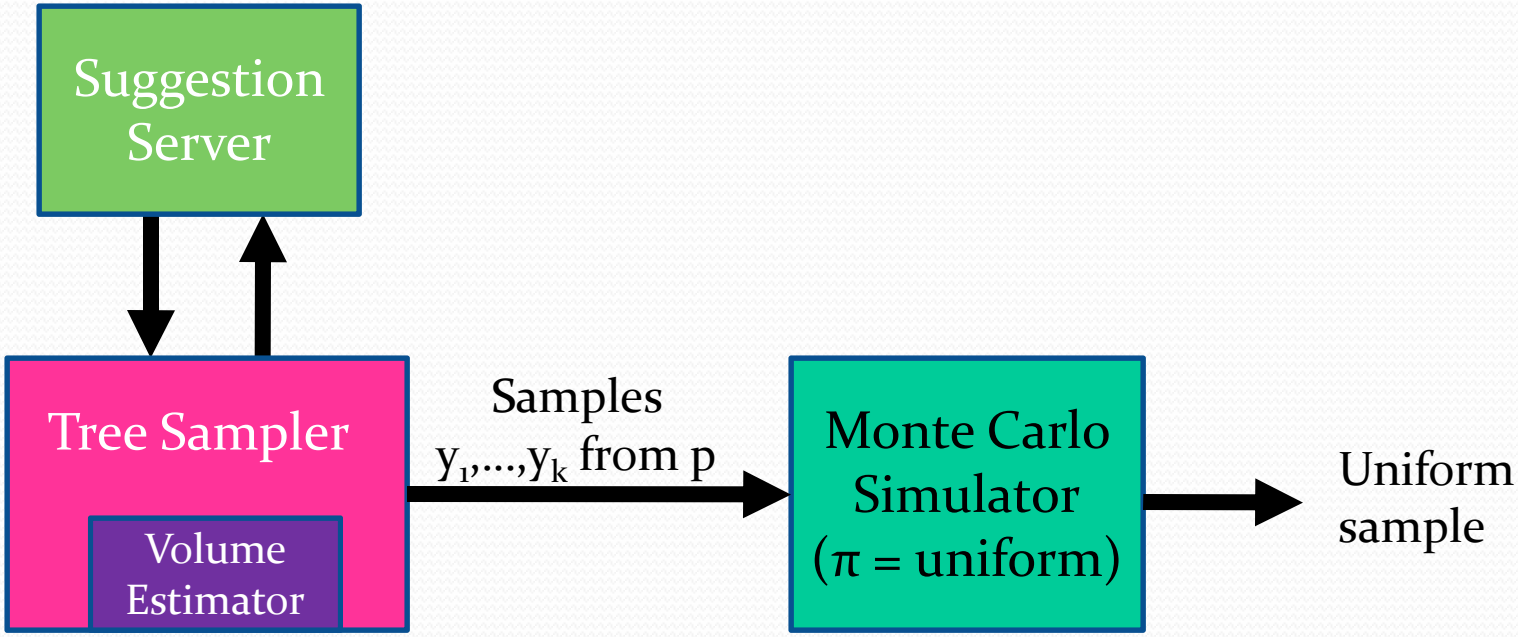
Rejection Sampling

[von Neumann 63]

- accepted := false
- while (not accepted)
  - Sample suggestion q from p
  - Calculate p(q) and π(q)
  - Toss a coin whose heads probability is $\frac{\pi(q)}{C \cdot p(q)}$
  - if coin comes up heads, accepted := true
- return q

- $\Pr(q \text{ accepted}) = p(q) \cdot \frac{\pi(q)}{C \cdot p(q)} \propto \pi(q)$

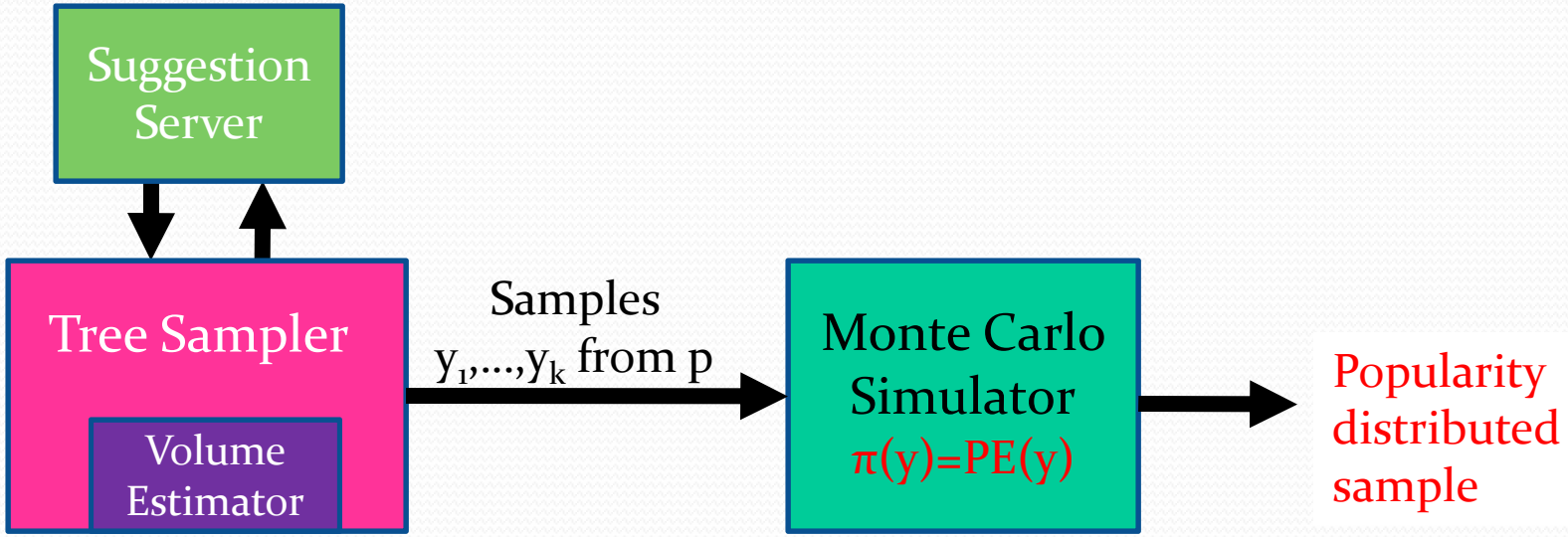
Recap – Uniform Sampling



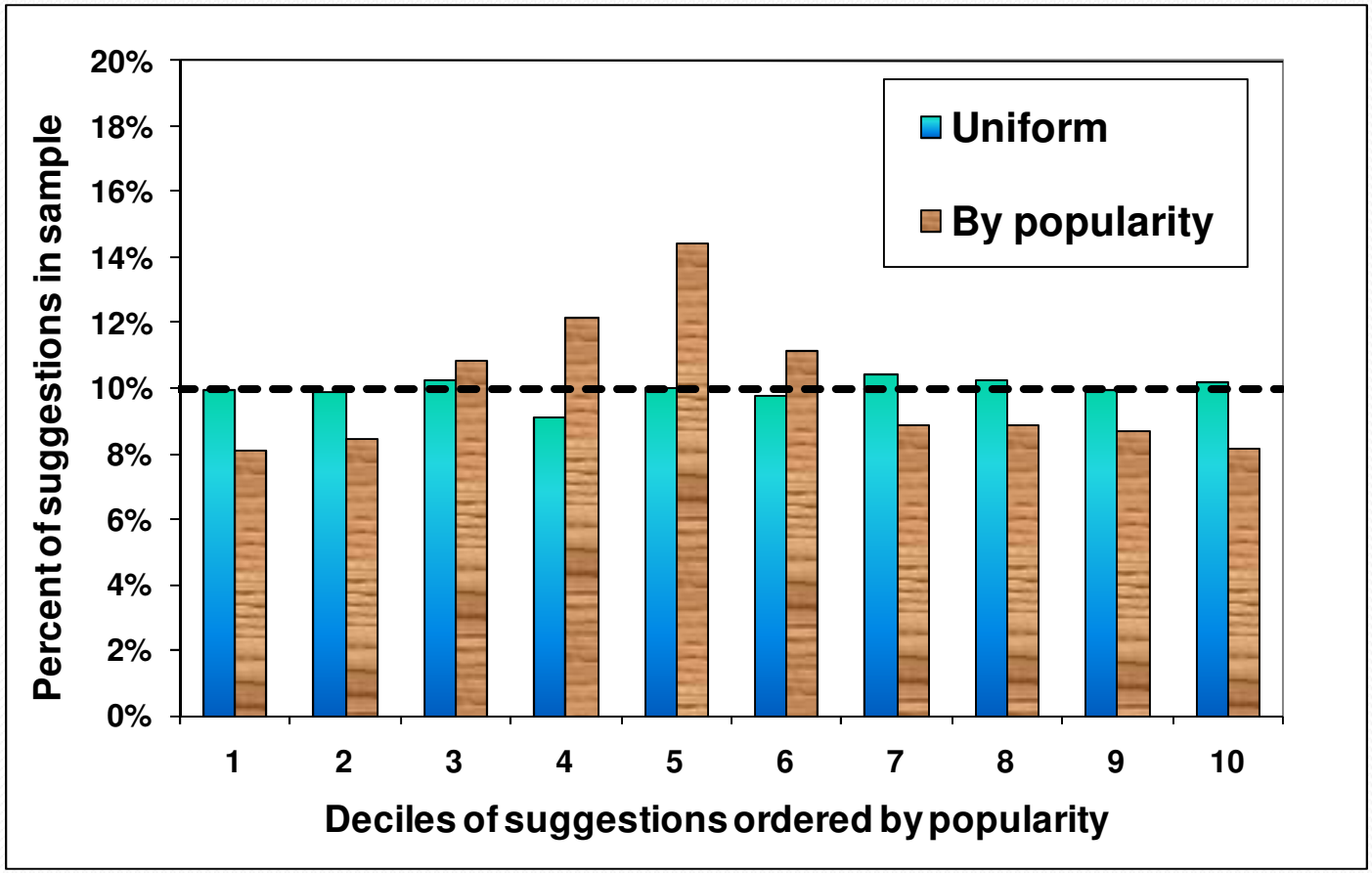
- ~6000 suggestion server requests per uniform sample

Popularity based suggestion sampling

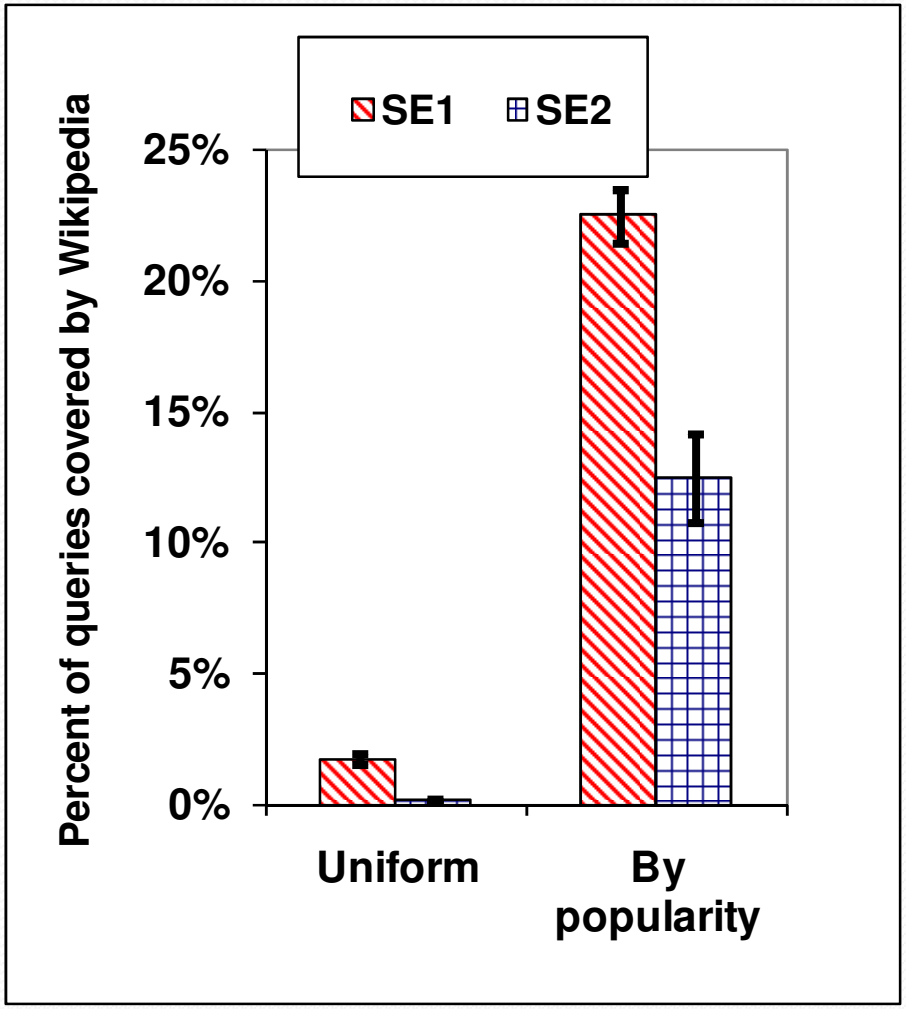
- Assumptions
  - Popularity is distributed according to Power Law
  - The Power Law exponent is known apriori
- Basic building block: Popularity Estimator (PE)



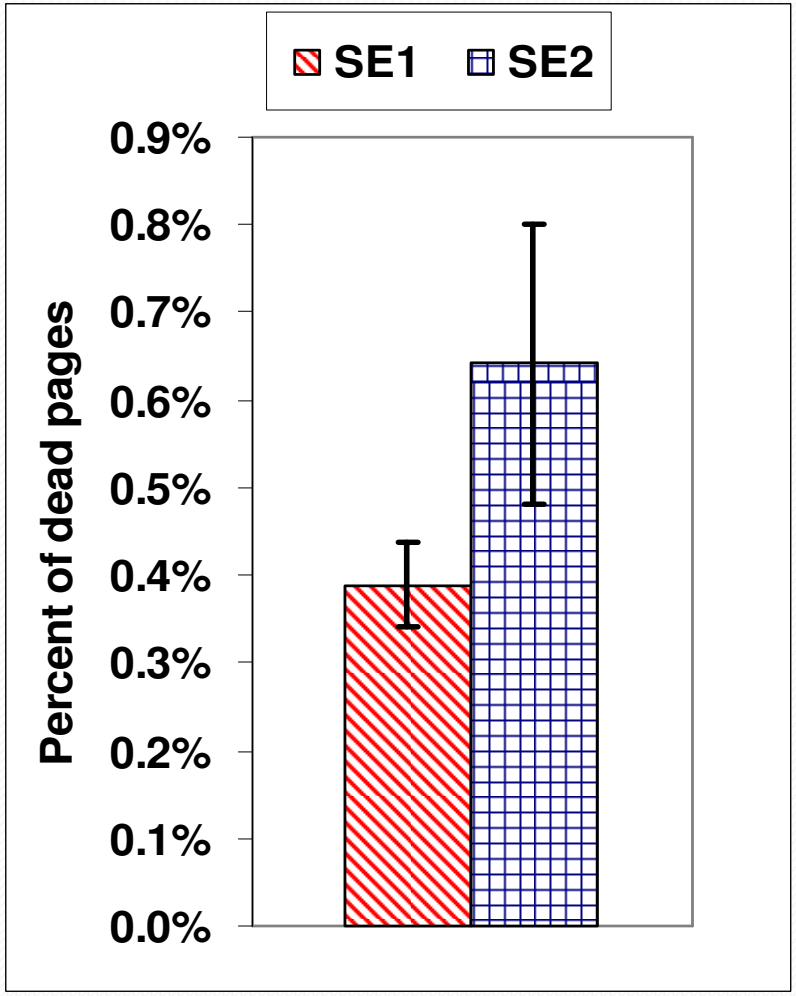
Sampling Bias



Coverage of sampled queries by Wikipedia



Percent of non-existent search results



Conclusions

- Algorithms for sampling queries randomly from a search engine query log
  - Uniformly or by popularity
  - Useful for keyword based advertising, search engine evaluation, user behavior studies
  - Via public suggestion interface only
  - Practical (accurate and efficient)

Thank You