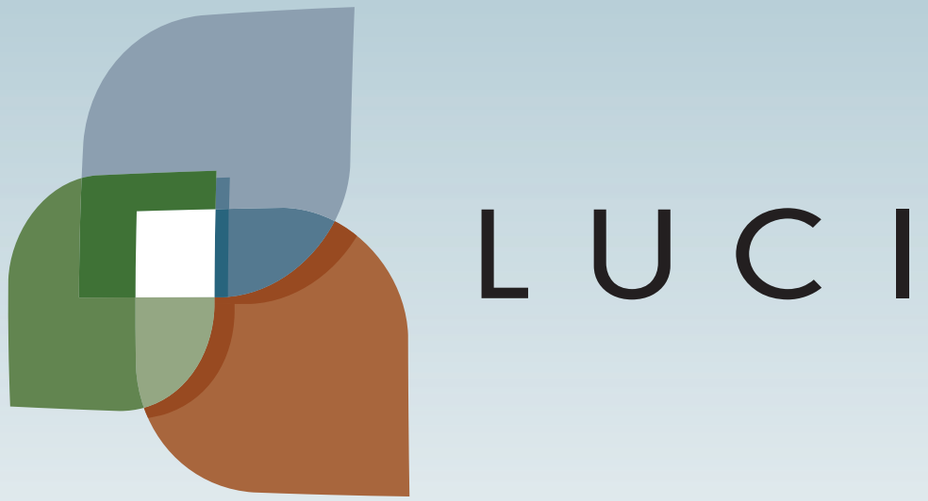
# Summary

- PageRank
  - Static score based on random walk
  - Long-term visit rates
  - Calculated based on link structure..
    - Not from actual processes wandering the web
- PageRank for topics
  - Know this
- HITS
  - PageRank for hub and authority model
  - Optional

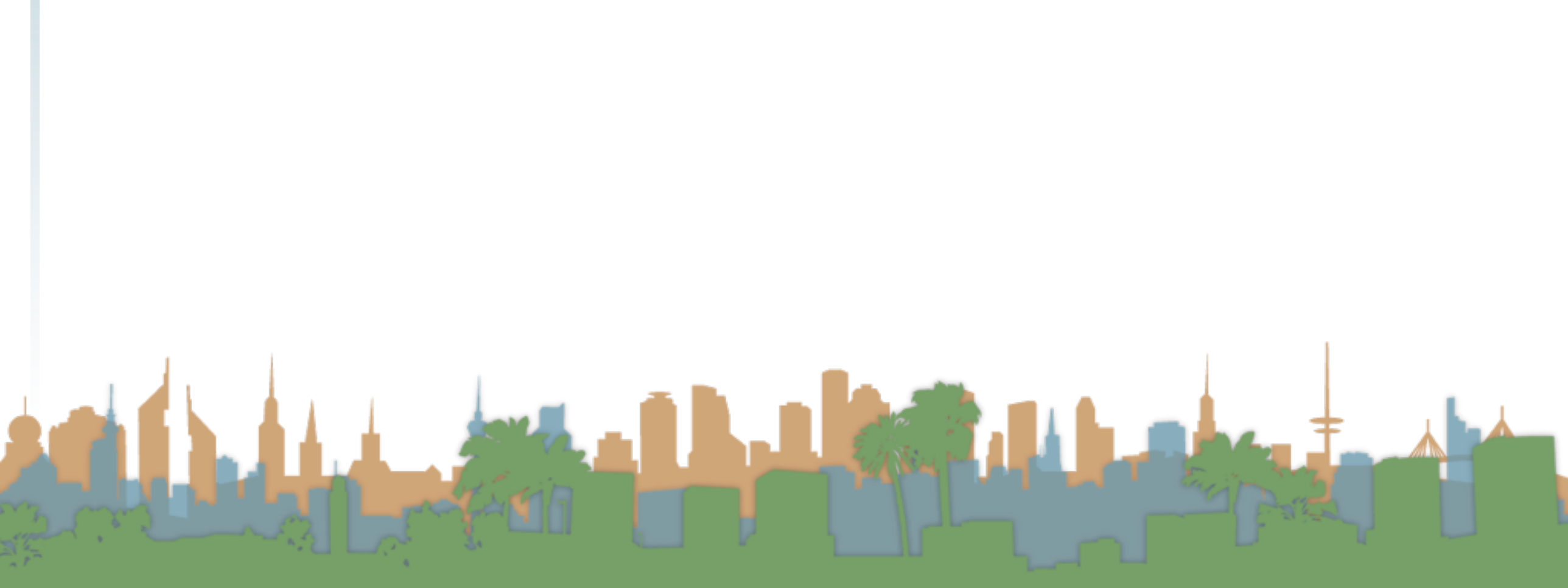
next...



LUCI

# Calculate Cosine Similarity Score

- Input
  - Query
  - Posting List
- Output
  - List of 10 top ranked documents



# Calculate Cosine Similarity Score

- Remember what this is about
  - A query as a vector
  - A corpus as a term-document matrix
    - Where each document is a column in the matrix

$$sim(q,d) = \frac{\vec{V}(q) \cdot \vec{V}(d)}{|\vec{V}(q)||\vec{V}(d)|}$$

# Calculate Cosine Similarity Score

- We are not going to calculate the similarity score of a query with every document
  - That would be inefficient.
  - Many scores are zero.
- We are not going to actually create a term-document matrix
- The posting list has all the information that we need to calculate the similarity scores

# Calculate Cosine Similarity Score

- We are going to calculate the cosine similarity score, but in a clever way.
- Here are some constants we will need:
  - The number of documents in the posting list (aka corpus).
    - Figure this out when creating the corpus (new thing)
  - The document frequency of a term
    - This should be the number of items in a row of the posting list. (each term has its own row)
- The term frequency of a term in a document.
  - Different for every term document pair.

# Calculate Cosine Similarity Score

- Steps
  - Get a query from the user
  - Convert it to TF-IDF scores

$$tfidf(t,q) \quad = \quad WTF(t,q) * log\left(\frac{|corpus|}{df_{t,q}}\right)$$

$\mathrm{WTF}(t,q)$
1 **if** $tf_{t,q} = 0$
2 **then** $return(0)$
3 **else** $return(1 + log(tf_{t,q}))$

# Calculate Cosine Similarity Score

- "UCI Informatics Professors"
  - 3 terms {"UCI", "Informatics", "Professors"}
  - 3 TF-IDF scores
    - Size of the corpus comes from the posting list
    - The document frequency of "UCI" comes from the number of entries in the posting list for "UCI"
    - use 1 if your posting list is too small
  - The term frequency is 1/3

$$tfidf(\text{“}UCI\text{”}, \text{“}UCI\ Informatics\ Professors\text{”}) \quad = \quad 1 + log(1/3) * log\left(\frac{|corpus|}{df_{\text{“}UCI\text{”}}}\right)$$

# Calculate Cosine Similarity Score

- Steps
  - Get a query from the user
  - Convert it to TF-IDF scores
  - Create a data structure that is indexed by documents
    - Which will accumulate scores for the documents
      - so like, Scores = new Hashmap<String,Double>()

# Calculate Cosine Similarity Score

- Steps
  - Get a query from the user
  - Convert it to TF-IDF scores
  - Create a data structure that is indexed by documents
    - Which will accumulate scores for the documents
      - so like, Scores = new Hashmap<String,Double>()
- For each term in the query
  - Get the posting list for the term
  - For each document that has that term we are going to update the entry in Scores

# Calculate Cosine Similarity Score

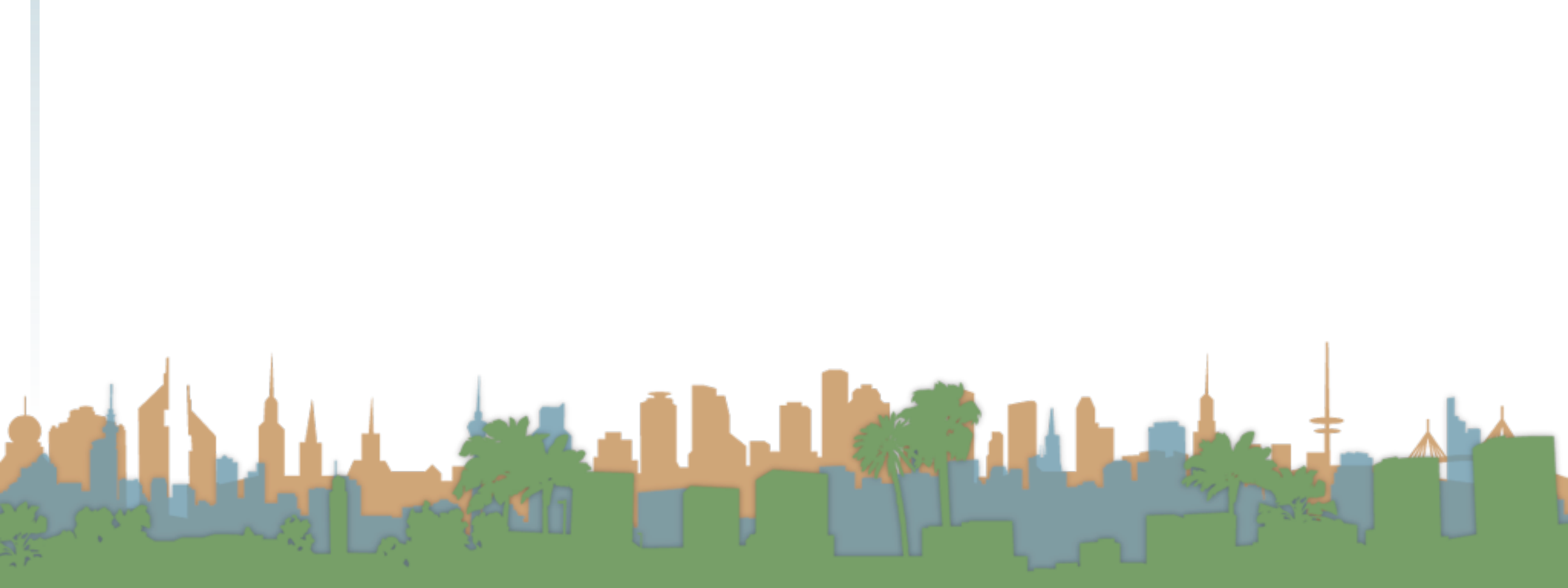
- Steps
  - For each term in the query
    - Get the posting list for the term
    - For each document that has that term we are going to update the entry in Scores
      - Scores[d] += TF-IDF(term,query) * TF-IDF(term, document)

# Calculate Cosine Similarity Score

- At the end of this we will have the data structure Scores
- Which for “UCI Informatics Professors” required looking up 3 posting lists
- Finally the scores must be normalized so we can compare them against each other.
- Create a new data-structure like Scores called Magnitude
- For each term in the entire posting list
  - For each document represented in Scores
    - Magnitude[document] += TF-IDF(term, document)^2

# Calculate Cosine Similarity Score

- Now we have Scores and Magnitude
- Now we calculate the highest rankings
- For each document in Scores
  - Double x = Scores[document]/sqrt(Magnitude[document])



# Calculate Cosine Similarity Score

- Summary
  - Get query from user, transform to TF-IDF
  - Pull out a few postings to calculate scores
  - Look at every positing to calculate magnitudes
  - Calculate final scores
  - Output URLs and scores of highest documents

# Calculate Cosine Similarity Score

$\textsc{CosineScore}(q)$
1 $\quad\textsc{Initialize}(Scores[d \in D])$
2 $\quad\textsc{Initialize}(Magnitude[d \in D])$
3 $\quad$**for** $each\ term(t \in q)$
4 $\quad\quad$**do** $p \leftarrow \textsc{FetchPostingsList}(t)$
5 $\quad\quad\quad df_t \leftarrow \textsc{GetCorpusWideStats}(p)$
6 $\quad\quad\quad \alpha_{t,q} \leftarrow \textsc{WeightInQuery}(t, q, df_t)$
7 $\quad\quad\quad$**for** $each\ \{d, tf_{t,d}\} \in p$
8 $\quad\quad\quad\quad$**do** $Scores[d] += \alpha_{t,q} \cdot \textsc{WeightInDocument}(t, q, df_t)$
9 $\quad$**for** $d \in Scores$
10 $\quad\quad$**do** $\textsc{Normalize}(Scores[d], Magnitude[d])$
11 $\quad$**return** $top\ K \in Scores$