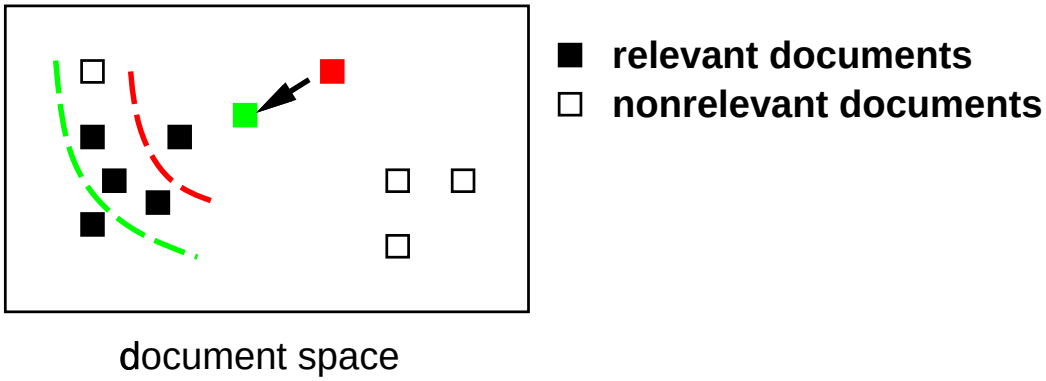
Contents

1 Relevance Feedback

1.1 Basic Assumptions

- Similar documents are close to each other in the vector space.
- A better query can be formulated by moving the original query closer to the relevant documents and farther away from the nonrelevant documents.



Of course, this is not a fail-safe method. Moving the query towards the relevant documents may at the same time moving the query towards the nonrelevant documents.

- Given the relevant and nonrelevant document sets, the ideal query is:

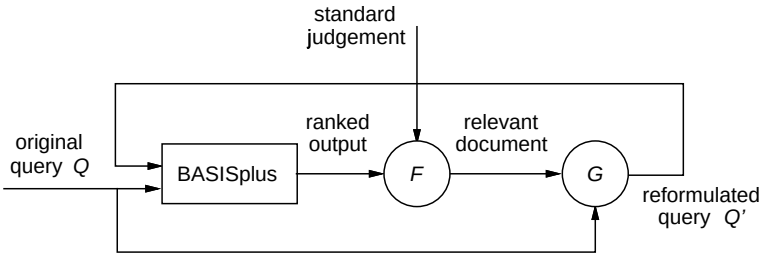
$$Q_{opt} = k\left(\frac{1}{R}\sum_{Rel}\frac{D_i}{|D_i|} - \frac{1}{N-R}\sum_{Nonrel}\frac{D_i}{|D_i|}\right).$$

- In relevance feedback, the user identifies some subset of relevant and nonrelevant documents, R' and N', respectively, from the retrieved items:

$$\begin{aligned} Q^{i+1} &= Q^i + \frac{1}{|R'|}\sum_{D_i\in R'} D_i - \frac{1}{|N'|}\sum_{D_i\in N'} D_i \\ &= Q^i + \alpha\sum_{D_i\in R'} D_i - \beta\sum_{D_i\in N'} D_i, \end{aligned}$$

where $0 \le \alpha, \beta \le 1$.

- The document vectors D_i's should have been normalized by the magnitudes of the vectors as in the previous formula. It seems that normalization factors are routinely dropped as a matter of convenience, which is, of course, not the right thing to do.



- F accepts relevance judgement from the user and produces as output sets of relevant and nonrelevant documents identified by the user.

- G implements the feedback formula.

1.2 Parameters to be Determined

- What are the values of α and β?
  - Equal weight: $\alpha = \beta = 0.5$. I.e., relevant and nonrelevant documents have equal contribution to the reformulated query.
  - $\alpha = 1, \beta = 0$. I.e., only relevant documents are used in reformulating the query.
- How many documents to use in R' and N'?
  - Use all relevant and nonrelevant documents.
  - Use all relevant documents and the highest-ranked nonrelevant documents.
- Should the entire document vector be used? The essence is to extract and include only those content-bearing terms from the chosen documents.

  - Use only high-frequency or high-weight terms from the documents (relevant or nonrelevant). (Using the entire vector may introduce too many terms in the reformulated query.)
  - Use terms adjacent to the hits.
  - More conservative: Terms common in all documents in R' (and N' correspondingly).

1.3 Query Splitting

- When the set of relevant documents R' (or nonrelevant documents) is not clustered, the feedback mechanism would not work.
- When R' has no cluster at all, there is nothing we can do.
- When R' has multiply clusters, examine the retrieved relevant documents and detect for multiple clusters. If multiple clusters are found, formulate a new query for each of the clusters. The same formula can be used.

1.4 Document Space Modification

- Instead of modifying the user query, the document vectors (i.e., the index terms and weights of the documents) are modified according to user feedback.
- Operation: Move relevant document vectors closer to the query and nonrelevant document vectors farther from the query.
- The operation affects the document vectors *permanently*. Therefore, each movement must be small enough so that a single erroneous movement (i.e., due to erroneous judgement or malicious user actions) won't affect the collection too much.
- Documents adapt to user feedback. Documents not identified as relevant by any user tend to cluster together. Thus, the method can be used as a *document retirement policy*.
- Results are not repeatable. Note that methods based on $tf \times idf$ weights may produce unrepeatable results, but the effect is not as obvious as when the document vectors are modified directly.
- Performance evaluation is difficult to conduct.

Query splitting and document space modification are rarely used in document retrieval systems. Query splitting requires the user to specify a large number of relevant documents in order for cluster identification to work. Document space modification, on the hand, is expensive to implement, and, more importantly, we don't know what is the right way to modify the documents.

1.5 Performance Evaluation Methods

- Basic principle: Relevance feedback should be able to promote previously unseen relevant documents to higher ranks so that they can be retrieved eventually. That is, eventually those relevant documents are promoted to ranks higher than the cutoff point.

- Documents identified as relevant by user will *likely* be ranked the highest after the feedback process. Since the relevance information is supplied to the system directly from the user, this improvement in performance doesn't reflect the system's ability in improving the quality of retrieval. Therefore, the documents previously specified by the user as relevant must be discarded when retrieval performance (i.e., precision and recall) of the feedback mechanism is evaluated.

1.5.1 Partial Rank Freezing

- Retrieved documents identified as relevant are frozen at their current rank.

	initial	feedback	
Rank	Doc Id	Doc Id	
1	10	70	
2	90	90	
3	40	20	← erroneously promoted
4	20	45	
5	65	65	
6	70	130	
7	88	120	
8	17	10	← correctly demoted
9	45	40	← correctly demoted
10	30	17	
...	...		

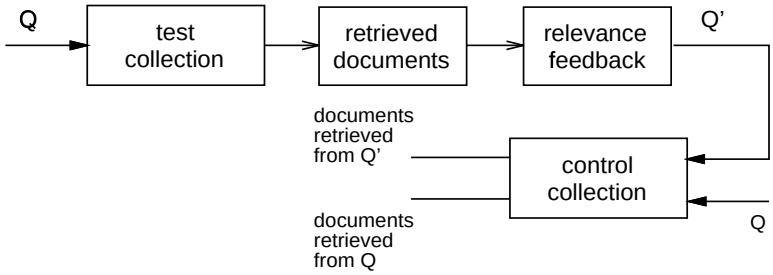
- Only top 5 documents are returned to the user.
- Red documents are relevant documents identified by the user.
- Blue documents are relevant documents not seen by the user. to higher ranks.

1.5.2 Pruning of Relevant Documents

- Documents are ranked as usual in the feedback process.
- Remove documents which are previously identified by user as relevant.
- Move all documents after the pruned documents up to fill the ranks vacated by the pruned documents
- Evaluate the precision and recall of the final output.
- This method also takes into account the effect on the relevant documents used in the feedback process.
- The freezing method may be too artificial in that the frozen documents may create abnormal behavior in the resulting precision and recall graph.

1.5.3 Test and Control Collections

- Split the document collection into 2 sets.
- Derived modified query Q' from the test collection.
- Apply the reformulated query Q' to the control collection.

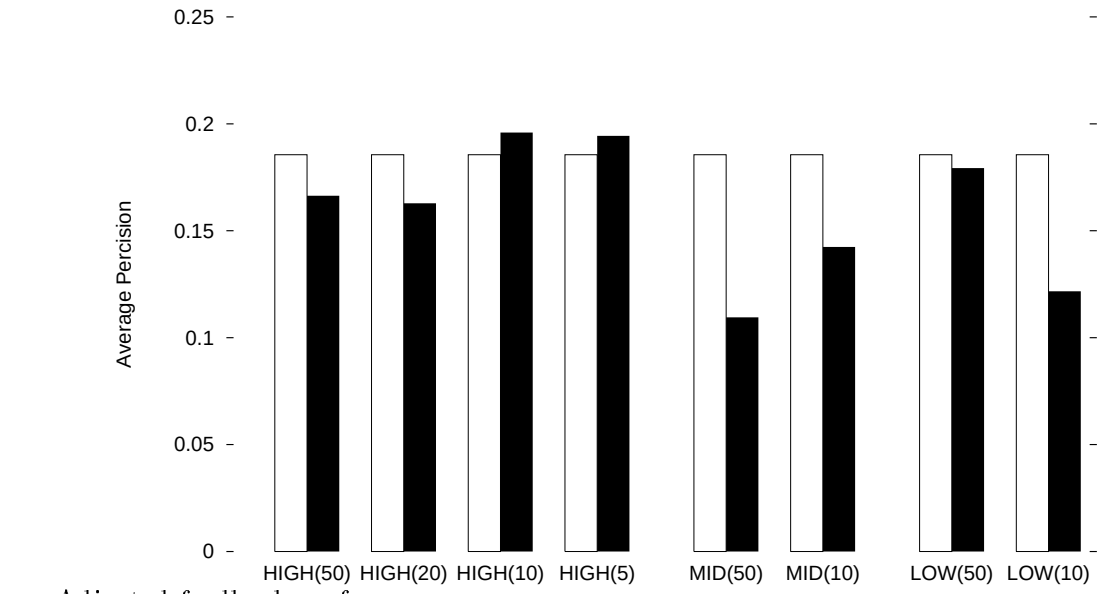


1.6 Performance of Feedback Methods

- Not discussed in the textbook: including a large piece of text, *even from a relevant document*, nondiscriminatively into the original query may adversely affect retrieval effectiveness.
- Experiments performed:
  1. Terms are extracted from relevant documents according to their occurrence frequencies: high, mid, or low frequency. No grammatical boundary is used. That is, words are extracted regardless of where they appear in the document.
  2. Same as above, but terms are selected from the same grammatical units (e.g., sentences or paragraphs) containing at least one query term.
  3. Same as (2), but terms are selected in the neighborhood of the hits.

NO CONTEXT	HIGH	MID	LOW	N/A
CONTEXT	HIGH	MID	LOW	HITS

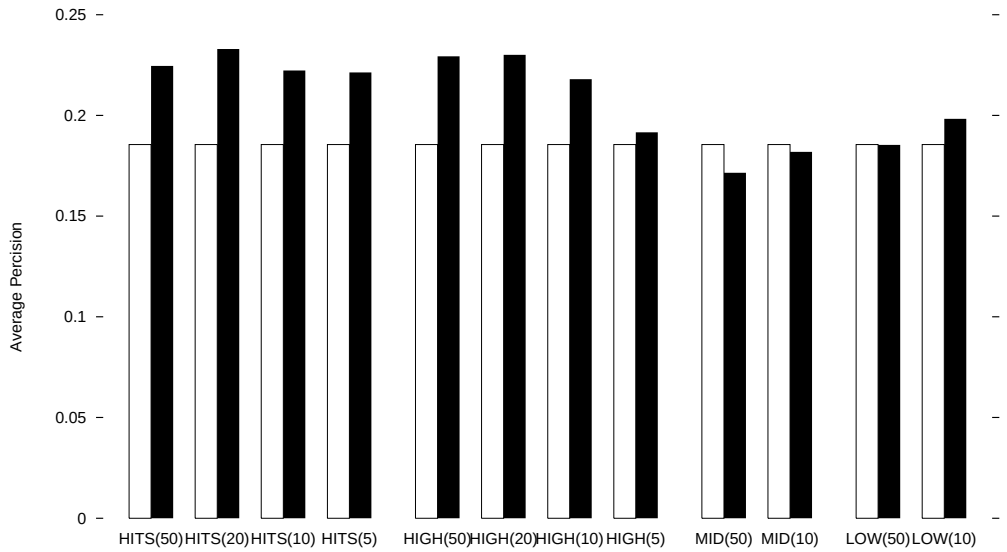
1.6.1 Feedback Performance with Context Units Ignored



Adjusted feedback performance
Adjusted base performance

- Without context units, terms all over the documents will be taken in generating the new query. Therefore, terms must be selected carefully.
- Only HIGH(5) and HIGH(10) show an improvement in the feedback process. All other parameters show a degradation on the performance.

1.6.2 Feedback Performance using Sentences as Context Units

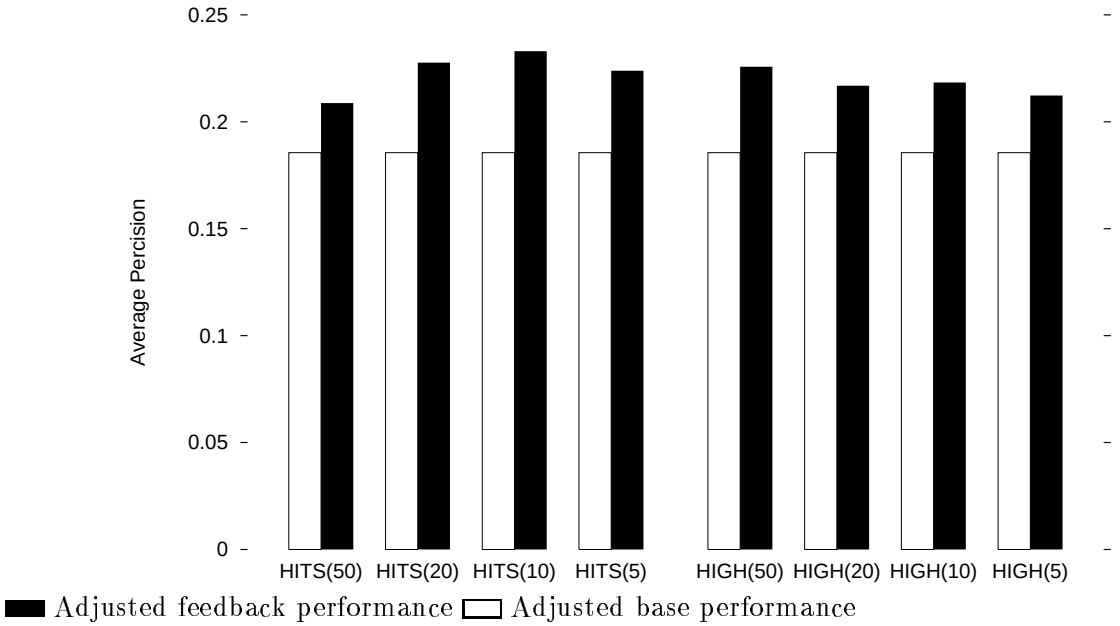


Adjusted feedback performance Adjusted base performance

- The sentence context narrows down the text to a region where terms in the region are useful for feedback even though they occur at a very low frequency (LOW(10)) or at an extremely high frequency (HIGH(20) and HIGH(50)).

- HIT(n) means terms within n terms on either side of a hit are selected. HIT performs quite well for all four parameters.

1.6.3 Feedback Performance using Paragraphs as Context Units



- broadening the context to the paragraph unit doesn't help much.

2 Search Heurisics on Inverted Files

2.1 Full Ranking versus Partial Ranking

- Full ranking obtains the exact set of top documents.
- Partial ranking attempts to find an approximate set of top documents with the goal of maximizing the accuracy of the approximation and minimizing the processing cost.
- For full ranking, the fast algorithms described in the previous section cannot produce any saving since, almost always, all query terms must be processed in order to obtain full ranking.

2.2 Heuristical Methods based on Partial Ranking

- Order of searching: How do we order the query terms so that partial document scores are accumulated as fast as possible?
- Stopping criteria: How do we estimate the accuracy achieved and when to stop the search?

$$
\begin{array}{rcccccccccc}
D & = & < & 1 & , & 4 & , & 0 & , & 4 & , & 4 & , & 0 & , & 9 & > \\
Q & = & < & 0 & , & 1 & , & 0 & , & 0 & , & 1 & , & 0 & , & 1 & > \\
idf & = & & & & \uparrow 2.5 & & & & \uparrow 3 & & & & \uparrow 2 & &
\end{array}
$$

- Ordered by term number:

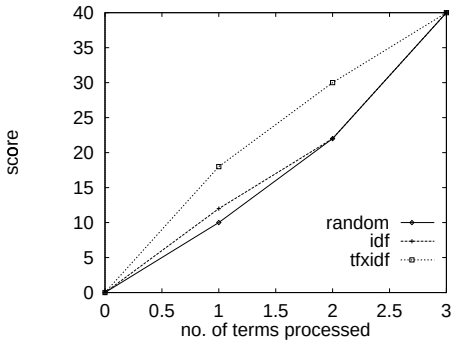
  score = 10 → 22 → 40

- Ordered by idf's (length of posting lists):

  score = 12 → 22 → 40

- Ordered by tf's and then by idf's

  score = 18 → 30 → 40



2.3 Search Heuristics

- The L method: Query terms are processed by ascending order of df. I.e., shortest postings list first. The df values must be stored in the index file. A typical entry in the inverted file looks like:

term t_i	df_i	⟶	$d_j, tf_{j,i}$	$d_k, tf_{k,i}$	$\cdots$

- The W method: Query terms are processed by descending order of $idf_i \times tf_{max,i}$ values, where $tf_{max,i}$ is the largest weight in the postings list corresponding to query term t_i.

term t_i	df_i	$tf_{max,i}$	⟶	$d_j, tf_{j,i}$	$d_k, tf_{k,i}$	$\cdots$

- The SW method: Postings lists are divided into pages and processing is done on a per-page basis, as opposed to per-list. Pages are processed in descending order of $idf_i \times tf_{max,i,j}$, where $tf_{max,i,j}$ is the largest weight in the page j of the postings list corresponding to query term t_i.

term t_i	df_i	$tf_{max,i,1}$	$tf_{max,i,2}$	$\cdots$	⟶	$d_j, tf_{j,i}$	$d_k, tf_{k,i}$	$\cdots$

- One can note that:
  - The clue is to find the page/list which will produce the largest increment to the document scores after the postings on that page/list has been processed.
  - The heuristics are using more and more precise information in finding the page/list that will produce the largest increment.
  - In the extreme case, all tf values are stored in the index file forming a 2-dimensional array containing all the information (i.e., tf and idf values) needed in computing the document scores.

term t_i	df_i	$tf_{j,i}$	$tf_{k,i}$	$\cdots$	⟶	d_j	d_k	$\cdots$
term t_j	df_j	$tf_{p,j}$	$tf_{q,j}$	$\cdots$	⟶	d_p	d_q	$\cdots$

2.4 Performance Criteria

- Retrieval Accuracy:

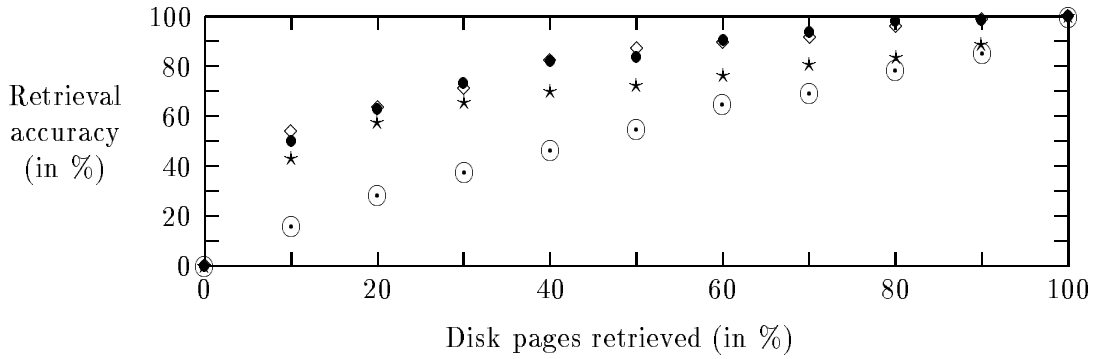
$$Ra = \frac{\mid Q_i \cap Q_f \mid}{\mid Q_f \mid} \cdot 100\%$$

where Q_f is the final set of top documents obtained when all of the query terms are processed, and Q_i is the set of top documents obtained after processing $i\%$ of the total disk accesses.

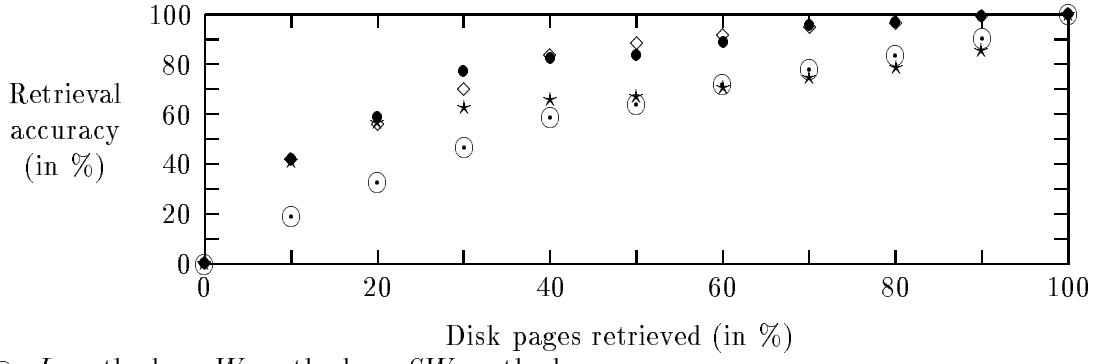
- Precision and Recall: Plot the precision and recall graph based on Q_i and compared to the precision and recall graph based on Q_f; compute the percentage of error introduced by Q_i.
- Note that retrieval accuracy doesn't consider the difference in ranking within Q_i and Q_f.
- Considering that Q_f is not the "perfect" result, errors introduced into the ranking will have a smaller effect on precision and recall.

2.5 Performance on the CACM collection

50 query terms per query:

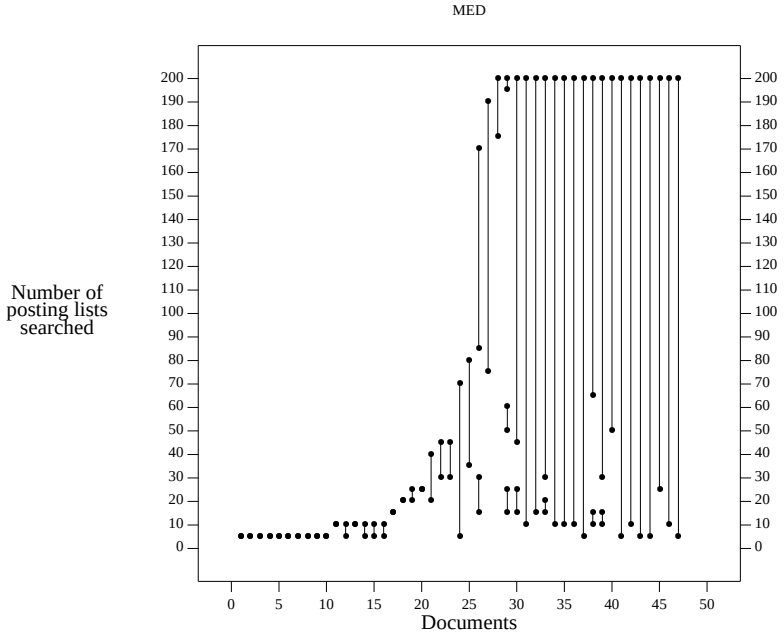


100 query terms per query:



⊙: L method, ⋆: W method, ◇: SW method,
120 bytes/page, •: SW method, 240 bytes/page.

2.6 Temporal Movement of Documents



- All documents that entered the top-20 range are monitored.
- The top-20 documents are records after every 5 query terms are processed.
- After processing all query terms, the documents are sorted by their document scores for clarity of presentation on the graph.

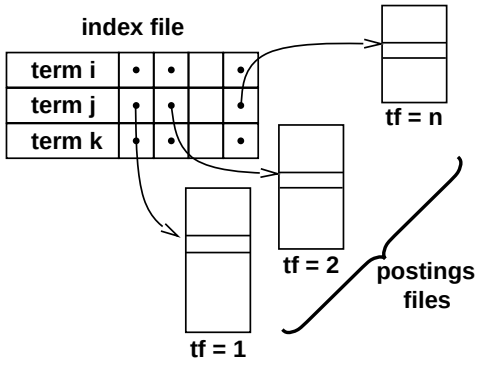
2.7 Estimation of Retrieval Accuracy

- For any heuristics, we must decide when an acceptable, albeit approximate, solution is reached. A method is needed to estimate the retrieval accuracy so that processing can stop when a certain accuracy is reached.
- Estimation based on document movement:
  - Use a heuristics to predict the likelihood that a document will be in the final result (i.e., top T documents).
  - Define a stable region t, where $t < T$, so that once a document is in that region we assume it will be in the final result.
  - Define a candidate region (typically equals to T). When a document stays in that region long enough, it will be assumed to be in the final result.
  - Based on the estimation, we can calculate the retrieval accuracy periodically and terminate the process when the accuracy reaches the expected level.
- By calibration:
  - Use a sample collection and submit random queries to it. In this case, we know the fi
  - For each query, find the top T documents at each interval and compare it with the final top T documents, from which the retrieval accuracy can be calculated.

- Obtain a linear interpolation relating retrieval accuracy and some easily measurable quantity (e.g., percentage of disk accesses performed).

2.8 Partitioned Inverted File Structure

- Knowing the exact term frequencies will help in:
  - accumulating partial scores as fast as possible
  - accurate estimation of retrieval accuracy. This is because increments on partial scores are smoother and more predictable.
- The question is to represent term frequencies so that it can be easily accessible and not consuming too much storage. Here is one solution, called the partitioned inverted file:



- Each "slice" of postings list corresponds to one term frequency. Processing should start with the highest term frequency (i.e., $tf = n$).

3 Review Questions in Class

- Does context (paragraph, sentence) help in:
  - obtaining better precision and recall in relevance feedback?
  - obtaining better precision and recall in document ranking?
- Paragraph-based retrieval, dynamic indexing, and navigation as a special case of querying.
- Application of paragraph-based similarity in detecting identical documents.