MARS: Applying Multiplicative Adaptive User Preference Retrieval to Web Search

Zhixiang Chen & Xiannong Meng
U.Texas-PanAm & Bucknell Univ.

Outline of Presentation

- Introduction -- the vector model over R+
- Multiplicative adaptive query expansion algorithm
- MARS -- meta-search engine
- Initial empirical results
- Conclusions

Introduction

- Vector model
  - A document is represented by the vector **d** = (*d1, … dn*) where *di*'s are the relevance value of i-th index
  - A user query is represented by **q** = (q1,…,qn) where qi's are query terms
  - Document **d**' is preferred over document **d** iff **q•d** < **q•d**'

Introduction -- continued

- Relevance feedback to improve search accuracy
  - In general, take user's feedback, update the query vector to get closer to the target **q**(k+1) = **q**(k) + **a**1•**d**1 + … + **a**s•**d**s
  - Example: relevance feedback based on similarity
  - Problem with linear adaptive query updating: converges too slowly

Multiplicative Adaptive Query Expansion Algorithm

- Linear adaptive yields some improvement, but it converges to an initially unknown target too slowly
- Multiplicative adaptive query expansion promotes or demotes the query terms by a constant factor in i-th round of feedback
  - promotes: q(i,k+1) = (1+f(d)) • q(i,k)
  - demotes: q(i, k+1) = q(i,k)/(1+f(d))

MA Algorithm -- continue

```
while (the user judged a document d)
{
  for each query term in q(k)
    if (d is judged relevant)          // promote the term
       q(i,k+1) = (1+f(di)) • q(i,k)
    else if (d is judged irrelevant) // demote the term
       q(i, k+1) = q(i,k) / (1+f(di))
    else     // no opinion expressed, keep the term
       q(i, k+1) = q(i, k)
}
```

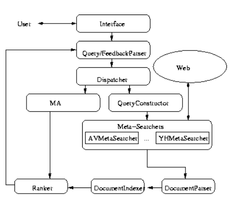
MA Algorithm -- continue

- The f(di) can be any positive function
- In our experiments we used
  f(x) = 2.71828 • weight(x)
- where x is a term appeared in di
- We have detailed analysis of the performance of the MA algorithm in detail in another paper
- Overall, MA performed better than linear additive query updating such as Rocchio's similarity based relevance feedback in terms of time complexity and search accuracy
- In this paper we present some experiment results

The Meta-search Engine MARS

- We implemented the algorithm MARS in our experimental search engine
- The meta-search engine has a number of components, each of which is implemented as a module
- It is very flexible to add or remove a component

The Meta-search Engine MARS
-- continue

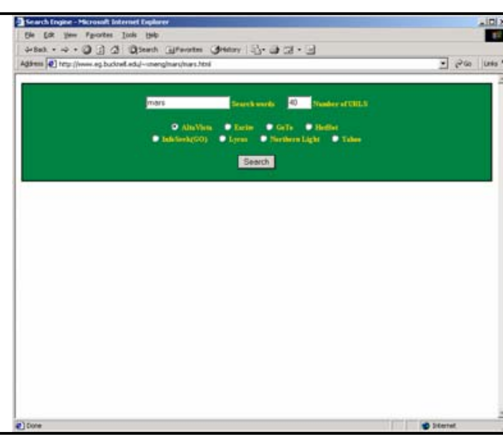


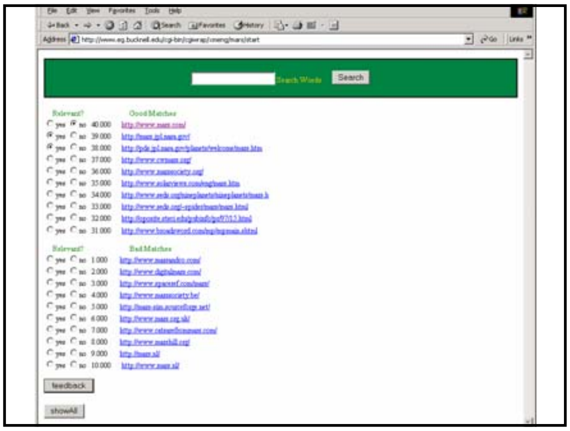
The Meta-search Engine MARS
-- continue

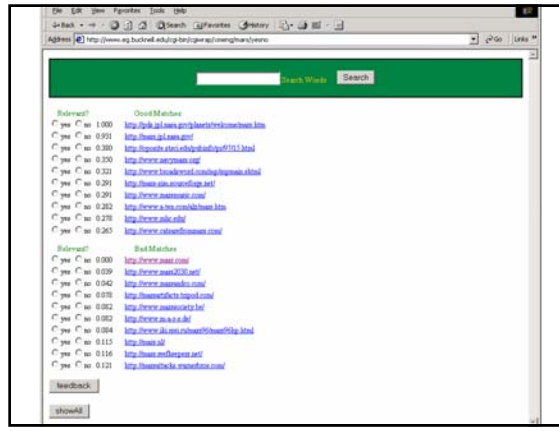
- User types a query into the browser
- The QueryParser sends the query to the Dispatcher
- The Dispatcher determines whether this is an original query, or a refined one
- If it is the original, send the query to one of the search engines according to user choice
- If it is a refined one, apply the MA algorithm

The Meta-search Engine MARS
-- continue

- The results either from MA or directly from other search engines are ranked according to the scores based on similarity
- The user can mark a document relevant or irrelevant by clicking the corresponding *radio button* at the MARS interface
- The algorithm MA refines document ranking by either promoting or demoting the query term







Initial Empirical Results

- We conducted two types of experiments to examine the performance of MARS
- The first is the response time of MARS
  – The initial time retrieving results from external search engines
  – The refine time needed for MARS to produce results
  – Tested on a SPARC Ultra-10 with 128 M memory

Initial Empirical Results --continue

- Initial retrieval time:
  – mean: 3.86 seconds
  – standard deviation: 1.15 seconds
  – 95% confidence interval 0.635
  – maximum: 5.29 seconds
- Refine time:
  – mean: 0.986 seconds
  – standard deviation: 0.427 seconds
  – 95% confidence interval: 0.236
  – maximum: 1.44 seconds

Initial Empirical Results --continue

- The second is the search accuracy improvement
  – define
    - A: total set of documents returned
    - R: the set of relevant documents returned
    - Rm: set of relevant documents among top-m-ranked
    - m: an integer between 1 and |A|
    - recall rate = |Rm| / |R|
    - precision = |Rm| / m

Initial Empirical Results --continue

– randomly selected 70+ words or phrases
– send each one to AltaVista, retrieving the first 200 results of each query
– manually examine results to mark documents as relevant or irrelevant
– compute the precision and recall
– use the same set of documents for MARS

Initial Empirical Results --continue

Recall	(200, 10)	(200, 20)	Precision	(200,10)	(200,20)
AltaVista	0.11	0.19		0.43	0.42
MARS	0.20	0.25		0.65	0.47

Initial Empirical Results --continue

- Results show that the extra processing time of MARS is not significant, relative to the whole search response time
- Results show that the search accuracy is improved by in both recall and precision
- General search terms improve more, specific terms improve less

Conclusions

- Linear adaptive query update is too slow to converge
- Multiplicative adaptive is faster to converge
- User inputs are limited to a few iterations of feedback
- The extra processing time required is not too significant
- Search accuracy in terms of precision and recall is improved