

VISUAL 2008

10th International Conference on Visual Information Systems

An Architectural Paradigm for Collaborative Semantic Indexing of Multimedia Data Objects

*Alice Chan, Clement Leung, Jiming Liu,
Hong Kong Baptist University, Hong Kong*

(speaker) Alfredo Milani, University of Perugia, Italy

Outline

- **Introduction**
- **Collaborative Indexing Approach**
- **Index Score Updating Algorithm**
- **Genetic Variations for Object Discovery**
- **Architecture Overview**
- **Experimental Results**
- **Conclusions**

Introduction: motivations

Texts are built by writers using words, i.e. semantic symbols, images are first taken and then given a semantic by the observer

- Searching of multimedia data lags behind since:
 - current technological limitations,
 - costly & time-consuming
 - Preindexing need to know search terms in advance
- Image retrieval:
 - **Concept-based** (e.g. title, keywords, and caption)
 - Content-based (e.g. size, colours, and textures)

The meaning and relevance of images/multimedia can depends on the observer and can change over the time

Introduction: Goal

- to provide a methodology for
- supporting *semantic visual information search* in the framework of a
 - Web 2.0 **interactive** approach: exploiting *user generated content, user interaction* and *visual classification capabilities*

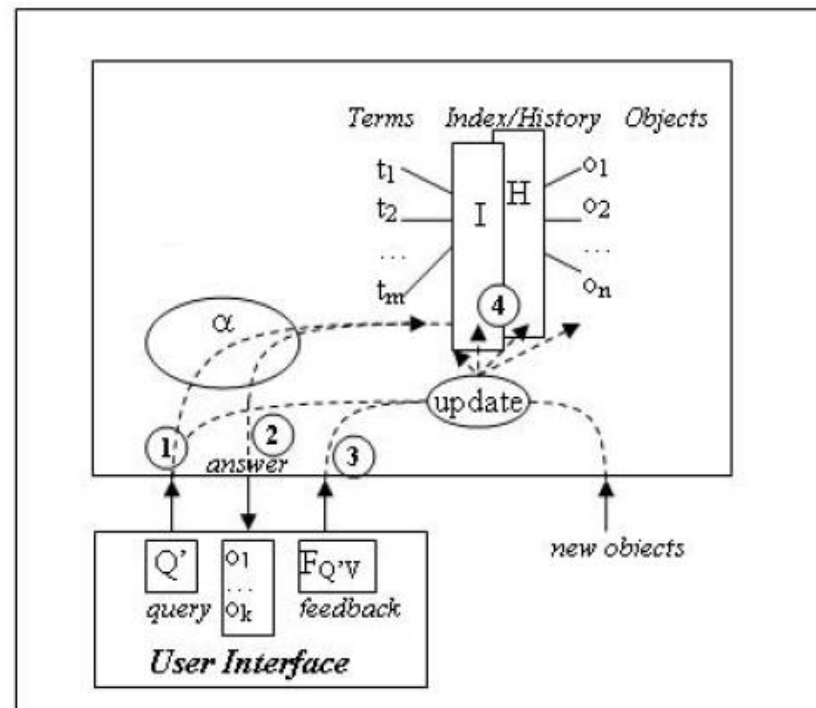
Collaborative Indexing Approach

- Indexing of **semantic contents** of multimedia objects
- Exploiting **explicit and/or implicit user feedback**
- Let consider:
 - A set of data objects $\{O_j\}$
 - Each O_j has an index set
$$I_j = \{e_{j1}, e_{j2}, \dots, e_{jM}\}$$
 - Each index element e is a triple
$$e_{jk} = (t_{jk}, s_{jk}, o_j)$$

where t_{jk} is a search *term*
 o_j is an *object* in the repository
 s_{jk} is the **score** of o_j with respect to t_{jk}

Collaborative Indexing (continued)

- **Users/system loop:**
query, answer, feedback, update



new objects enter the repository at lowest layer

new terms enter the index by new term installation

Index Score Updating Algorithm

- query input: search terms $Q(T_1, \dots, T_n)$
- query output: k multimedia objects O_1, O_2, \dots, O_k
- Increase the score s_{ix} of index element $e(t_i, s_{ix}, O_x)$ $\forall t_i \in T_1, \dots, T_n$ when
 - user select O_x in the query result list (*implicit feedback*), or
 - receive **+ve** *explicit feedback* from user
- Related index scores increased by $\Delta+$
- Promotion of index term T to the next higher level

Index Score Updating Algorithm (continued)

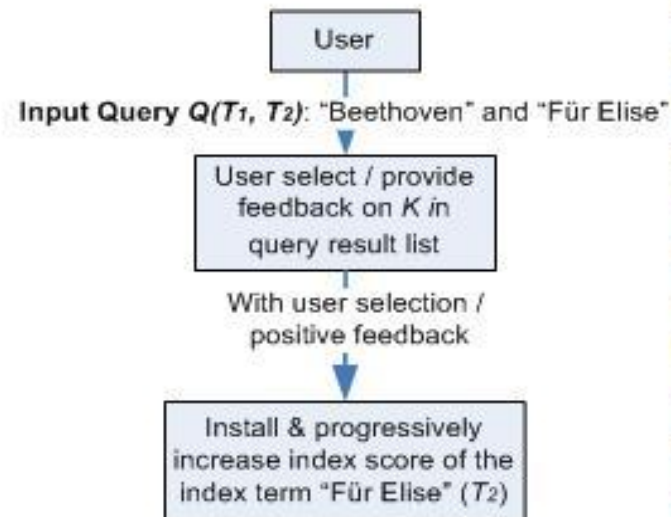
- query input: search terms $Q(T_1, \dots, T_n)$
- query output: k multimedia objects O_1, O_2, \dots, O_k
- Decrease the score when
 - User don't select any O_x in the query result list (implicit feedback), or
 - Receive -ve *explicit feedback* from user
- Related index scores decreased by Δ -
- Drop of T to the next lower level

Installing New Index Terms: propagation

- Consider:
 - an object O_J is minimally indexed with T_1
 - input query: $Q(T_1, T_2)$
 - a list of query result is returned
 - add T_2 to O_J when user select O_J in result list
 - after progressive usage: T_2 would be properly indexed
- For example:

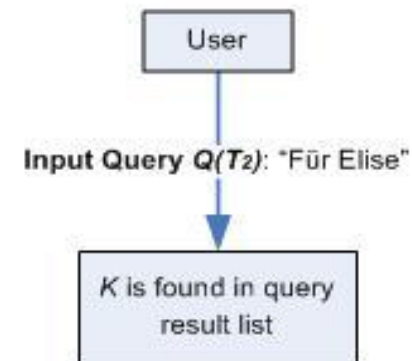
Initially

Data Object K is indexed with the index term T_1 "Beethoven". K can be only searched by using a query that consists of T_1 .



After progressive usage

When the index score on the index term T_2 "Für Elise" for K reaches the required threshold, T_2 becomes a proper index of K , such that, K becomes searchable by a query that consists of T_2 .



Drawbacks

The *repository size* is much larger than the *query answer size*: some objects have not many chance to be shown to the user

The straightforward “prize the best one” strategy main drawbacks:

- sort of **local maximum** situation
- the first elements shown in the first answers **tend to prevail** even on the best relevant objects which had no the chance to be shown in the early stages
- **new elements** have no chance to be submitted to the user for evaluation

Genetic Variations for Objects Discovery

- Genetic Algorithms (GA)
 - allow to discover 'hidden' data objects
 - object O_j ranking in the **query answer** for term T is randomized with *probability proportional to index score S_j*

$$P_j = f(S_j), \quad j = 1, 2, \dots, M,$$

where $f(\)$ is a monotonically increasing function,
and M is the no. of objects in search result

- a particular simple form for this function:

$$P_j = \frac{S_j}{\sum_{i=1}^n S_i}, \text{ where } j = 1, 2, \dots, M.$$

Genetic Variations for Objects Discovery (continued)

- For example:

Object ID (O_j)	Query Score (S_j)	Probability value (P_j)
O_1	S_1	P_1
O_2	S_2	P_2
O_3	S_3	P_3
...
O_M	S_M	P_M

Architecture Overview

Multimedia Information Search Architecture Overview

User Level

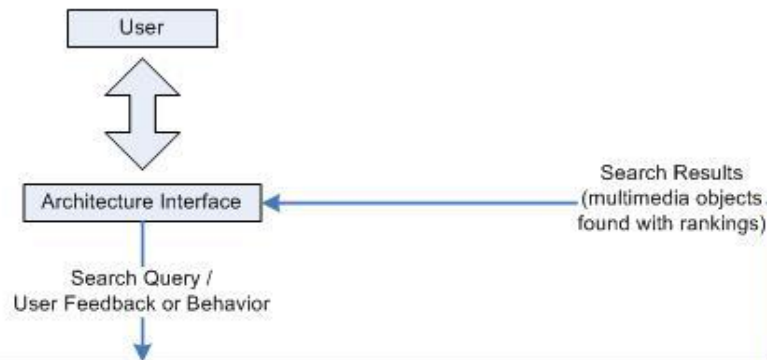
User is the one who interact with the search architecture via the architecture interface. They can:

- Submit queries to search multimedia objects
- Get query results from the architecture interface
- Provide relevance feedback on query results

Interface Level

Architecture Interface acts as a bridge between user and the database. It captures:

- User query input
- User selections on the search results
- User relevance feedback



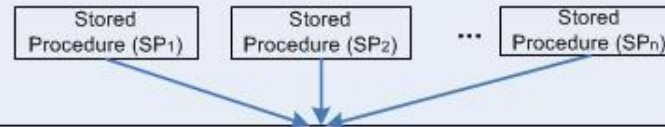
Database Level

DB consists of:

- Multimedia objects that can be searched by user
- An index hierarchy and an index table that maintains the weights of index terms to multimedia objects
- Stored Procedures that performs all DB related processing

Query Component

Responsible for all DB related process, e.g., multimedia object retrieval, search result ranking, index score update, index insertion, etc.



Index Component

Index level is supported by the index hierarchy and the index table that links the relationship between multimedia objects and its indexes

Index Hierarchy

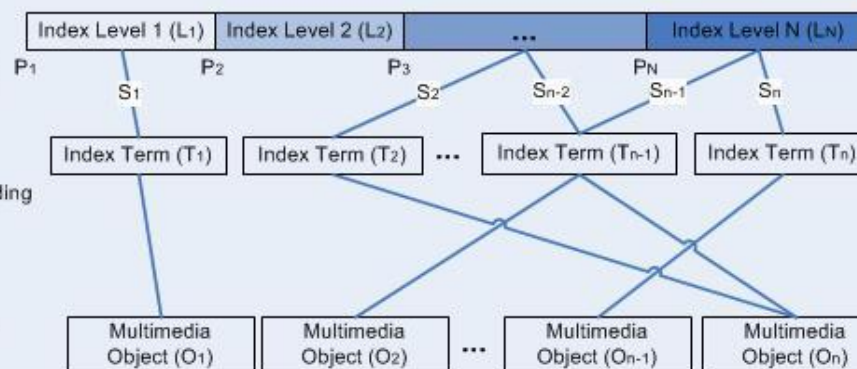
Index Hierarchy partitions index levels with parameters values on index scores

Index Terms

Index Terms are linked with related multimedia objects with its corresponding index score

Multimedia Objects

Multimedia objects are indexed with different index terms with its corresponding score

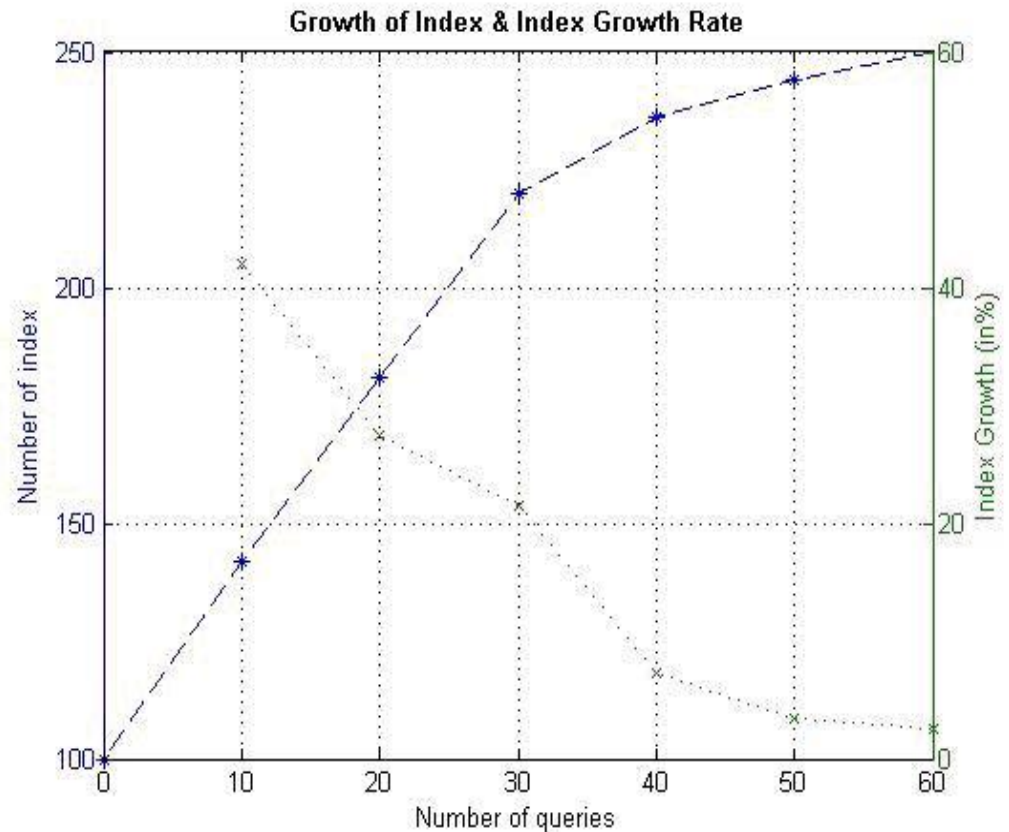


Experimental Results

- Index Growth
 - initially: 100 minimally indexed multimedia data objects (*pop/classic music objs*)
 - 60 queries were performed

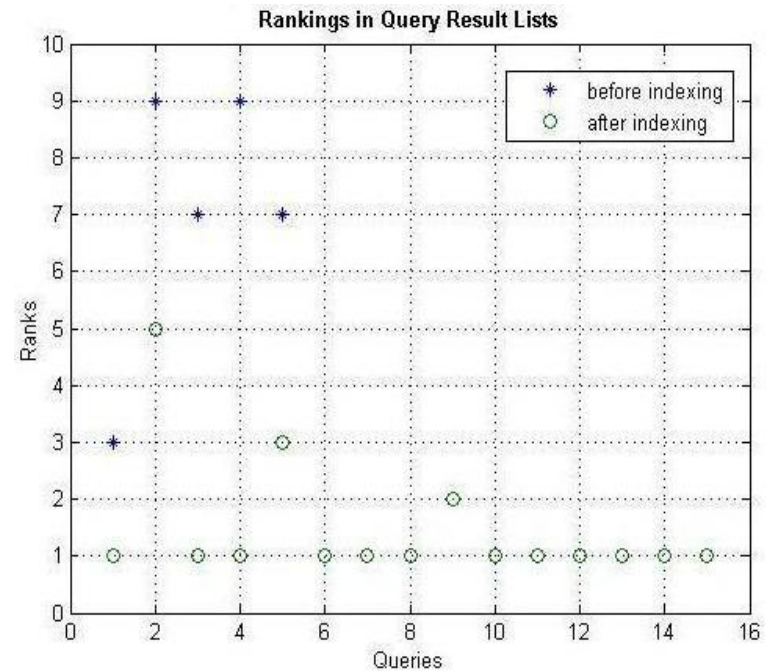
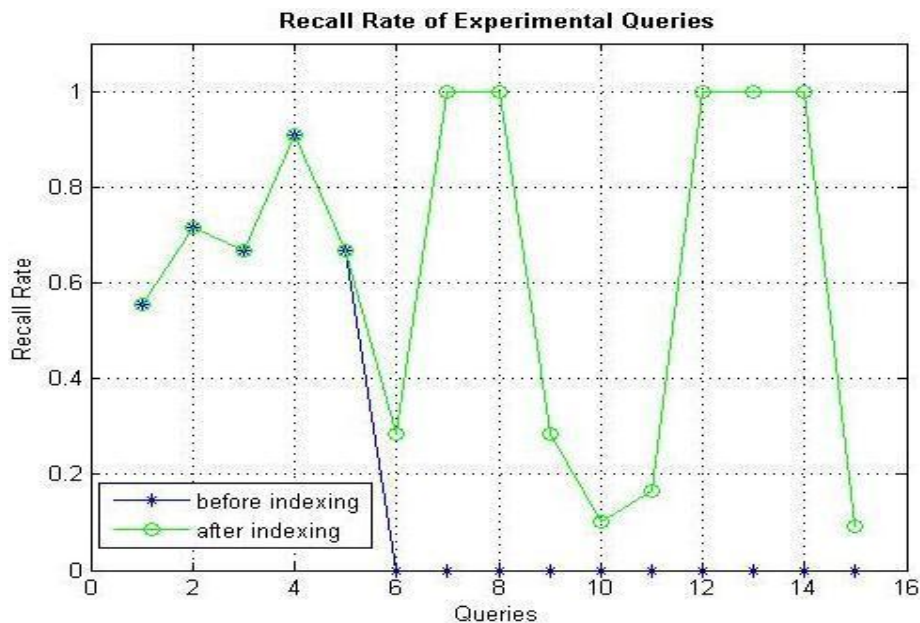
Result

- as no. of queries increases, no. of index terms in higher index layer also increases



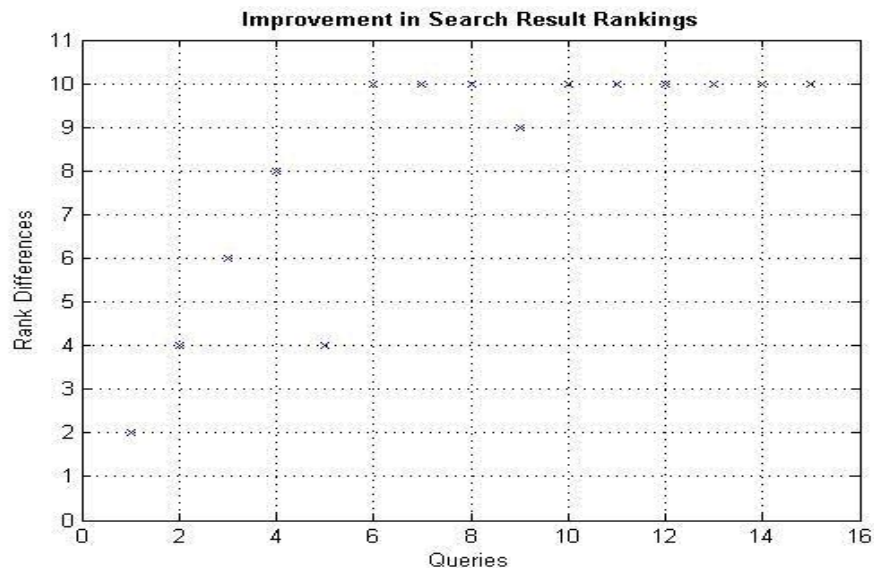
Experimental Results (continued)

- Effect of Indexing
 - before indexing: perform 15 queries
 - indexing through 60 queries
 - after indexing: perform the same set of 15 queries for comparison
 - objects can be found after collaborative indexing



Experimental Results (continued)

- Improvement in search results based on cutoff at 10
 - consistently produces significant improvement in search performance
 - **new** meaningful index **terms** will gradually be promoted to the most significant level of index hierarchy



Experiments: issues&drawbacks

- Users are **not willing to collaborate** into experiments
- Explicit user feedback is **difficult** to be obtained

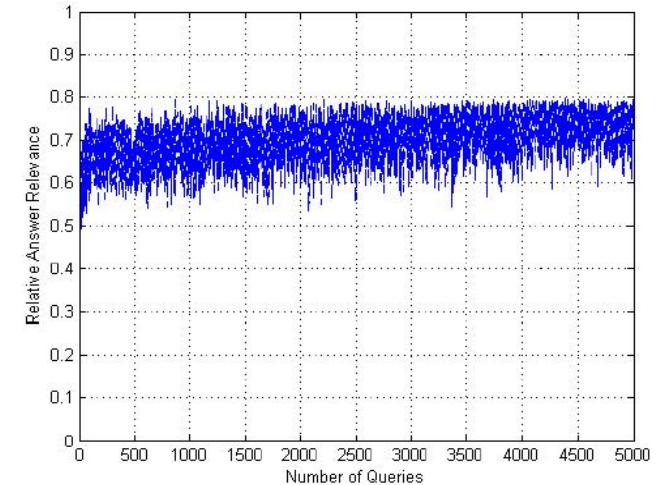
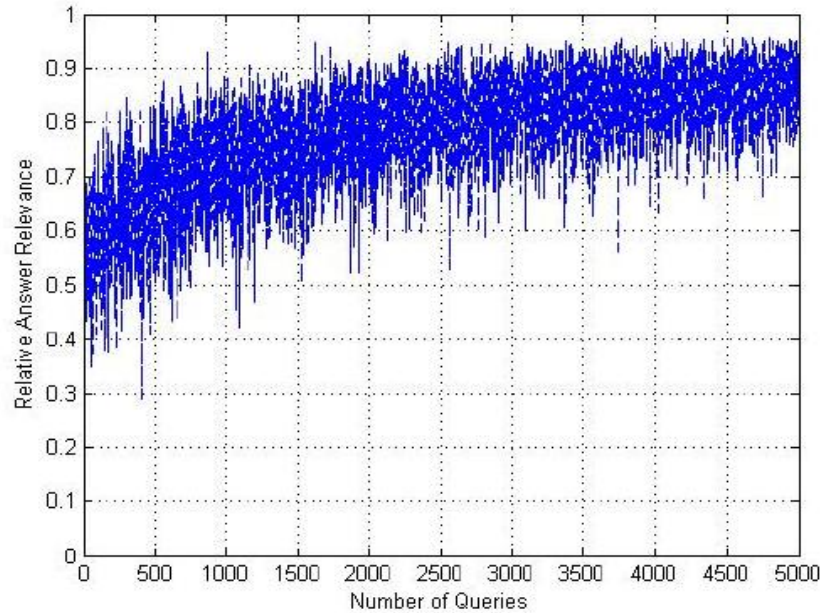
- Experiments with human users are **biased**
 - influenced by personal taste/attitudes
 - not reproducible

- Experiments are **fundamental** to:
 - assess the effectiveness of the approach
 - tune the parameters of the collaborative indexing systems (genetic variations etc.)

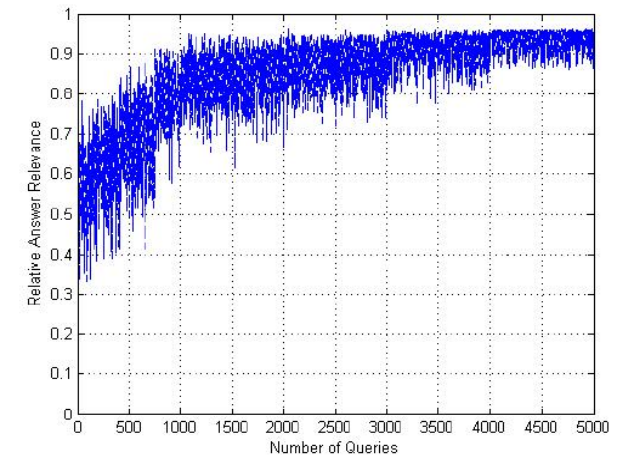
Experiments: user simulation & hidden relevance score

- Basic Idea:
 - for each term/object pair is an ***hidden ideal score*** assigned
 - ***queries*** are randomly generated and submitted to the system
 - the collaborative indexing algorithm increment/decrements its own ***index scores*** and returns list of objects as query answer
 - the user implicit feedback is simulated by using the ***hidden score***
- **Performance indicators:** distance between ***hidden score*** and collaborative discovered ***index score*** distribution

Experiments: some recent results



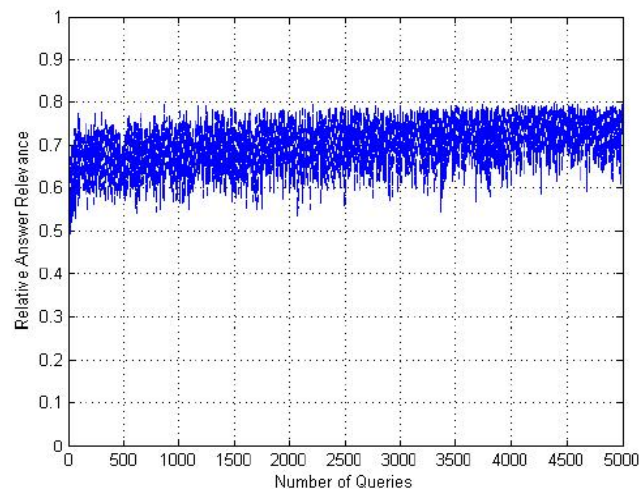
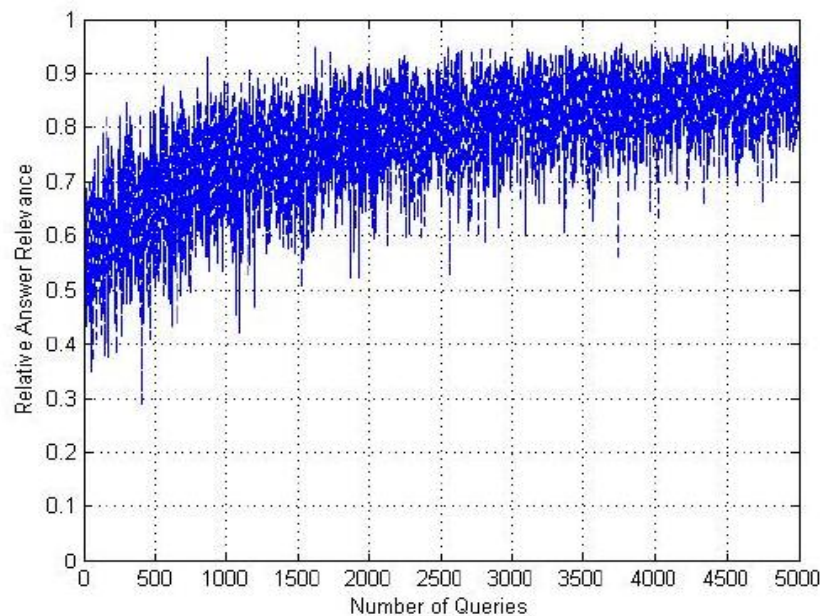
(e) 5th run : static elitism 70%



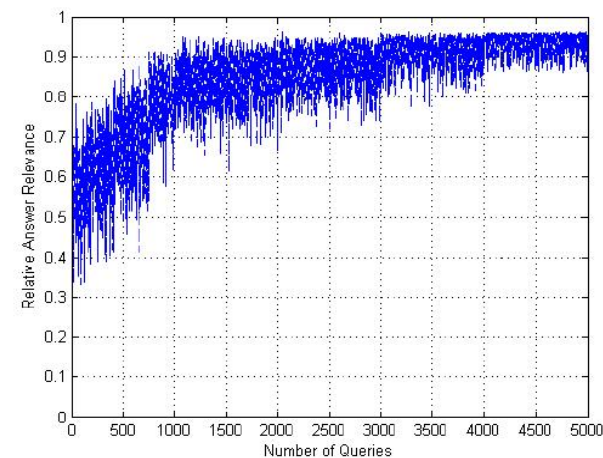
(g) 7th run : dynamic elitism

- 5000 queries, answer size 10
- 1000 objects
- other GA techniques e.g. dynamic elitism

Experiments: some recent results



(e) 5th run : static elitism 70%



(g) 7th run : dynamic elitism

- 5000 queries, answer size 10
- 1000 objects

Conclusions

the ***Collaborative Semantic Indexing*** approach:

- suitable for semantic searching & discovery of multimedia data objects
- implicit/explicit feedback management
- convergence & adaptation, moving object thru index layers
- community based implicit indexing:
 - different communities attribute different relevance to the same objects (cultural reasons, user goals etc.)
- evolutionary and dynamic environment:
 - object relevance change over the time
 - new objects are added to the repository
- systematic tests by hidden values, user simulation

Thank You!

...any questions?!?