# interactive querying

## locating and discovering information

Alan Dix

School of Computing, Staffordshire University, Stafford, UK
alan@soc.staffs.ac.uk

http://www.hiraeth.com/alan/topics/QbB/

The once distinct information retrieval, database and hypertext technologies are converging and overlapping. In addition, a plethora of interactive visualisation techniques have been developed over recent years. This paper will examine the nature of interactive querying and retrieval in order to understand the common features between apparently diverse retrieval techniques and in so doing help to address the needs of the emerging hybrid information repositories. This analysis will applied to Query-by-Browsing, an intelligent database interface based on a hybrid of IR, database and AI techniques, originally developed some years ago, and now being redeveloped to use evolutionary algorithms.

**Keywords:** database interfaces, information retrieval, machine learning, browsing, data exploration, genetic algorithms

## Introduction

Traditionally the information retrieval community and database community worked on very different kinds of data and using very different retrieval techniques. IR focused on free text documents with keyword and similarity-based retrieval. Databases focused on structured data with precise formal queries whether expressed in a program-like (e.g. SQL) or tabular (e.g. query by example) fashion. Hypertext retrieval stands opposed to both these approaches with its network structure and directed browsing for retrieval.

However, these barriers have been dissolving. Commercial databases now include large text fields with free-text search. The web although starting out as a hypertext is increasingly borrowing traditional IR techniques for search engines as well as pioneering new technologies such as crawlers and recommender systems [Resnick, 1997]. In addition, the chaos of the web has lead to a demand for greater structure and semantics. This has been partly satisfied by 'META' tags, but will be extended radically as XML becomes widely used, which will allow database-like queries over published XML document types.

If we are to develop mechanisms for effective query and retrieval from the emerging hybrid information storage systems, then we must understand the similarities in the semantic models of databases, IR and hypertext. Furthermore, we need to understand how these fit into the human process of interactive retrieval.

Some years ago I developed an intelligent database querying system called Query-by-Browsing (QbB) [Dix, 1992, 1994]. Although aimed at traditional databases, its focus was on the selection and relevance ranking of specific records – far more similar to IR techniques. However, the differences between the two domains were important. In particular, for database retrieval it is important that the retrieved records are not just a useful or suitable set, but they are *precisely* the right set. This makes it important to be able to feedback not just the selected records, but also the query *generated* by the system to retrieve them.

QbB is in the process of being redeveloped in order to include different machine learning algorithms, both to improve the interactive style and to extend the kinds of data managed by it. A sound understanding of the interactive querying process is thus essential.

In this paper, we will begin by looking at the convergence of IR, database and web technology in a little more detail.

Next, we shall look at Query-by-Browsing (QbB), both the general principle and also several quite different implementations. These differ in internal algorithms and user-interface details, but are clearly the 'same' idea. Finally, we will look at the process of interactive querying in detail, exposing the similarities and differences between various modes of interaction and retrieval. This will allow us to clarify the essential features of QbB.

## Convergence of technologies

In the past databases used to only dealt with short fixed-length character strings, most now allow large text fields ranging from 'abstract' sized text (4096 bytes), to medium document sized (64k) or effectively unlimited. These may be stored using different technology and may be indexed using techniques more familiar in IR document repositories. For example, Oracle's SQL*TextRetrieval allows full free-text searching of text fields and external documents [Oracle, 1992].

Standalone hypertext systems have often included different forms of search and index facilities. However, the fact that they are 'designed' have meant that users had a ready made structure within which to explore. With the exception of dictionaries and encyclopaedias, search facilities in hypermedia have often been 'icing on the cake' rather than the default navigation mechanism.

The web has effectively bridged the hypertext–IR gap in that there is no global structure or designed model. Hence IR techniques have been increasingly been used in search engines such as Yahoo, AltaVista and InfoSeek – including not just keyword based search, but also inter-document similarity measures. In addition, we are seeing a new collaborative view of information retrieval with recommender systems based on users commonality of interests. To confuse things further, web content is increasingly being generated from databases [Dix, 1998].

The chaos of the web has lead to a demand for greater structure and more semantics. This is partly satisfied by the (often under-used) 'META' tags which encode key-information such as keywords, short descriptions, and inter-relationship with other documents. However, a more dramatic change will occur as XML becomes widely used allowing the semantic content of pages to be separated from their layout and the formulation of database-like queries over published XML document types [Mace, 1998].

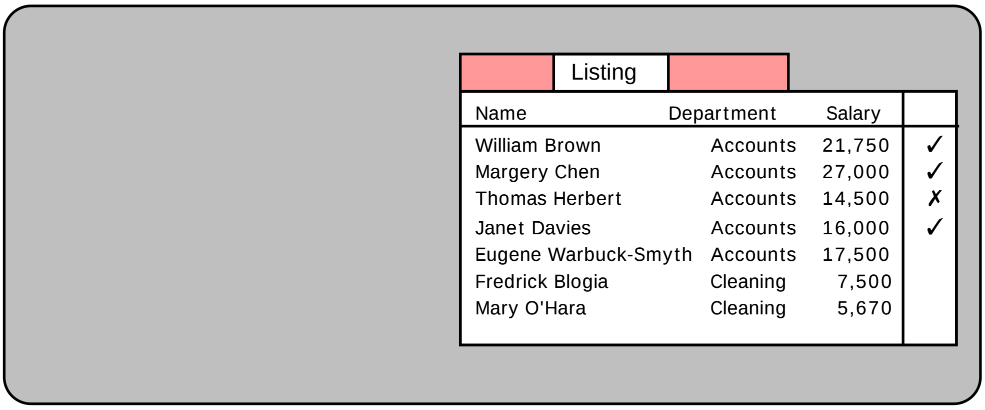
Finally, from the HCI and graphics research community have come a whole range of interactive visualisation techniques. Some of these are addressed at established problems in IR (e.g. "scatter/gather browser" [Pirolli, 1996] and 3D representations of document spaces such as 'Cone Trees' [Robertson, 1991] and "populated information terrains" [Benford, 1994]), databases (e.g. visual query languages for relational [Haw, 1994; Papantonakis, 1995; Benzi, 1998] and object databases [Chavda, 1997; Murray, 1998]) and hypertext (e.g. node graphs [Halasz, 1987] and "Disk Trees" [Chi , 1998]). Others address new problems (e.g. interactive parameter selection in "Dynamic Queries" [Ahlberg, 1992, 1994a] and "Attribute Explorer" [Tweedie, 1994, 1996]). Furthermore, media databases are forcing new solutions to search pictorial [Jain, 1997], audio [Wilcox, 1998] and video [Cristel, 1998] data collections; and even virtual worlds need to be indexed and navigated [Elvins, 1998].

## Query-by-Browsing

Query-by-Browsing (QbB) is an intelligent database querying system which has been developed over several years [Dix, 1992, 1994]. Although aimed at traditional relational databases, its focus is on the selection and relevance rating of specific records – far more similar to IR techniques. However, the differences between the two domains are important. In particular, for database retrieval it is important that the retrieved records are not just a useful or suitable set, but they are *precisely* the right set. This makes it important to be able to feedback not just the selected records, but also the query generated by the system to retrieve them.

### how it works

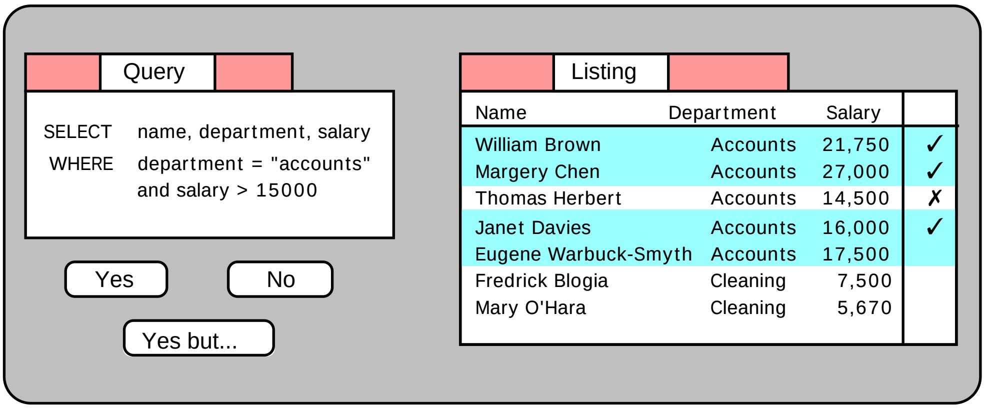
The system is aimed at users who want a precise a precise set of items. For example, a user may be selecting all employees of a certain type in order to give them a pay rise – an approximate answer will not do. However, although the users can identify precisely which items are required, they may not know how to articulate this as a general query.



| Name | Department | Salary | |
|---|---|---|---|
| William Brown | Accounts | 21,750 | ✓ |
| Margery Chen | Accounts | 27,000 | ✓ |
| Thomas Herbert | Accounts | 14,500 | ✗ |
| Janet Davies | Accounts | 16,000 | ✓ |
| Eugene Warbuck-Smyth | Accounts | 17,500 | |
| Fredrick Blogia | Cleaning | 7,500 | |
| Mary O'Hara | Cleaning | 5,670 | |

**Figure 1.** QbB – user ticks interesting records

Initially the users of QbB are faced with a list of all the records in the relevant table (figure 1). They browse this table and select records which are of interest (marked ✓) and those which aren't (marked ✘). After a while the system infers a query which would account for the users' choices. The system then presents the query to the user for confirmation (figure 2). If the system has found the correct query then the user confirms it. If not, then the user can mark more records or enter other dialogues with the system to refine the query.



```
SELECT   name, department, salary
WHERE    department = "accounts"
         and salary > 15000
```

| Name | Department | Salary | |
|---|---|---|---|
| William Brown | Accounts | 21,750 | ✓ |
| Margery Chen | Accounts | 27,000 | ✓ |
| Thomas Herbert | Accounts | 14,500 | ✗ |
| Janet Davies | Accounts | 16,000 | ✓ |
| Eugene Warbuck-Smyth | Accounts | 17,500 | |
| Fredrick Blogia | Cleaning | 7,500 | |
| Mary O'Hara | Cleaning | 5,670 | |

**Figure 2.** QbB – system highlights inferred selection

As can be seen in the screenshot, the feedback is given in two forms, a textual query and highlighting of all the records which satisfy the query. The need for both is directly related to the form of output required from the process and the expected user group. As a precise set of records is required, the user needs to be sure that exactly those records required are in the final set. Some description of the final result as the user cannot check them all individually. Even if the user is a novice, it is fair to assume that the textual form of the query is largely recognisable even if the user could not generate it, but there are various ambiguities between natural language and formal queries (even when rendered as natural language!), especially when considering Boolean connectives. Hence the need for the highlighted items in the listing to confirm the user understanding of the query. At different phases in the interaction, the user may focus on one or other of the output windows.

Note that in "Query by Example" [Zloof, 1975] the user uses a report-like query template, but is still editing a query. In contrast Query-by-Browsing really is "by example".

## first implementations of QbB

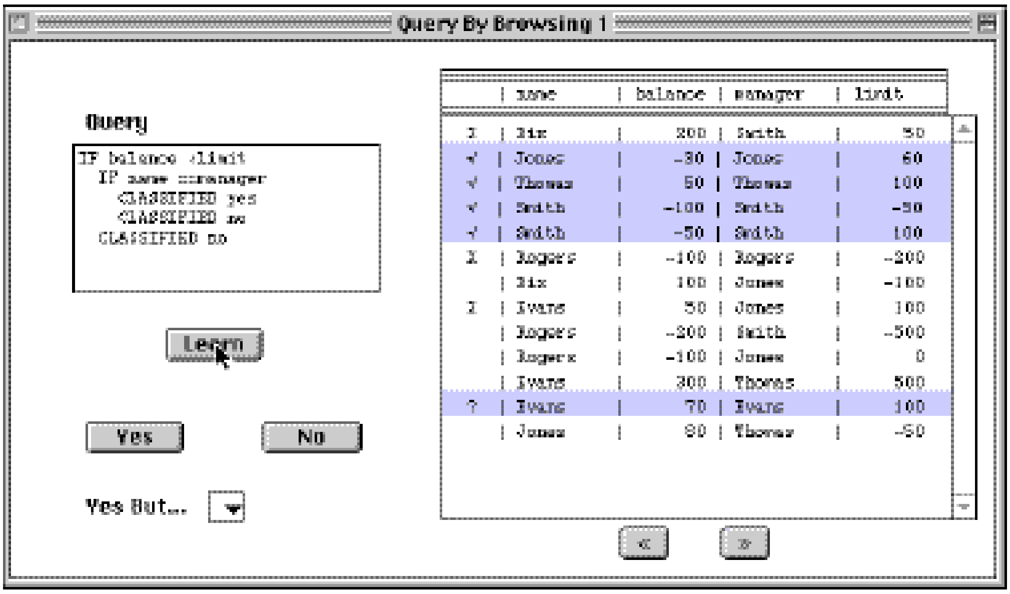
Figures 1 and 2 are simulated screen shots taken from the first description of QbB in 1992. At this stage it was not envisaged as a real system, but as a 'thought experiment' to demonstrate some of the interaction problems that may arise when pattern recognition techniques are used in user interfaces.

It was two years before the first implementation of QbB was produced first on a Sun workstation by a student Andrew Patrick and then re-implemented on a Macintosh in order to make it easier to demonstrate [Dix and Patrick, 1994]. A screen shot of the Macintosh version is shown in figure 3.

As envisaged in the original 1992 paper this used Quinlan's ID3 algorithm to infer a decision tree [Quinlan, 1986a]. However, the basic algorithm was extended to allow constraints between attributes such as:

```
balance  <  overdraft_limit
```

This simple, but powerful, extension in expressiveness is still surprisingly rare amongst concept learning algorithms.



```
IF balance <limit
  IF name =manager
    <CLASSIFIED yes
    <CLASSIFIED no
  CLASSIFIED no
```

|  | name | balance | manager | limit |
|---|---|---|---|---|
| X | Dix | 200 | Smith | 50 |
| √ | Jones | -30 | Jones | 60 |
| √ | Thomas | 50 | Thomas | 100 |
| √ | Smith | -100 | Smith | -50 |
| √ | Smith | -50 | Smith | 100 |
| X | Rogers | -100 | Rogers | -200 |
|  | Dix | 100 | Jones | -100 |
| X | Evans | 50 | Jones | 100 |
|  | Rogers | -200 | Smith | -500 |
|  | Rogers | -100 | Jones | 0 |
|  | Evans | 300 | Thomas | 500 |
| ? | Evans | 70 | Evans | 100 |
|  | Jones | 80 | Thomas | -50 |

**Figure 3.** QbB – Macintosh implementation

Recall that the original purpose of QbB was to expose potential problems in applying pattern recognition in the user interface. It turned out that this first implementation was embarrassingly good! Although it demonstrated all the expected interaction problems, it was actually a pleasure to use, normally generating sensible queries! In actual use, the times it produced confusing results were outweighed by the uncanny ability of the algorithm to guess what you wanted.

Despite this, there were residual problems (precisely those predicted). In particular, the algorithm could generate strange queries, which did match the data, but were clearly not sensible. Furthermore, the interaction in these circumstances was, as predicted, complex. It is clearly always going to be hard to communicate "its something like that, but not quite" to the computer. However, in addition, there were three specific problems related to the choice of ID3:

- ID3 queries are not 'natural'. It is hard to bias ID3 to sensible queries that are more natural for users to understand. Some versions of ID3 have been developed with a bias towards disjunctive normal form [Quinlan 1988b], but it would also be good to bias towards the use of specific attributes .
- ID3 is deterministic. It always generates the same query for a specific example set (you can't say "try again").
- ID3 is not incremental. If the user selects one or two additional records the algorithm is likely to generate a completely different decision tree.

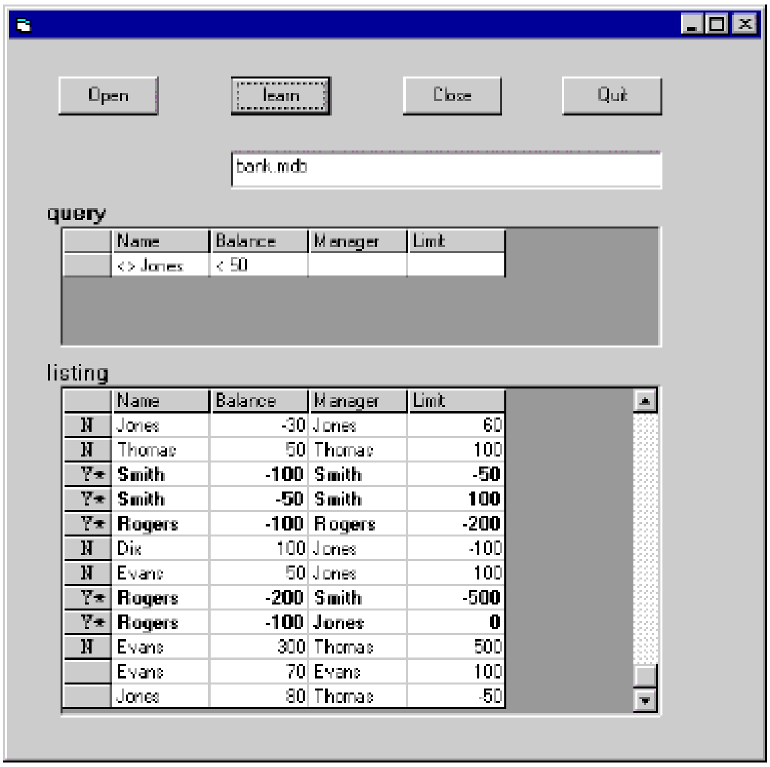
In contrast, a human helper would generate queries which used some attributes and types of query more than others (e.g. money columns in a bank database), and would respond by incrementally modifying the query if the user said it was nearly right, or by trying a completely different approach if asked.

It is possible to 'tweak' the basic algorithms, for example versions of ID3 have been developed with a bias towards disjunctive normal form [Quinlan 1988b], but it is far easier to incorporate these into a less-deterministic techniques such as simulated annealing or genetic algorithms.

### recent development

QbB is currently being redeveloped (after a few years gap!) to use such algorithms, to make it more widely accessible, and as a platform to experiment with new machine learning algorithms. Some differences are pragmatic: the current version is written in Visual Basic and can operate any ODBC database whereas the earlier version required a bespoke file format. However, there are also deeper differences: the query generated is in a disjunctive normal form, it is displayed in a Query-by-Example tableau format and a genetic algorithm is being used to infer the query.

Figure 4 shows a screen shot of this version of QbB. In this version the user selected records are indicated by 'Y' or 'N' in the left margin and the system selected records are indicated by the bold lines and the asterisk in the left margin.



bank.mdb

query

| | Name | Balance | Manager | Limit |
|---|---|---|---|---|
| | <> Jones | < 50 | | |

listing

| | Name | Balance | Manager | Limit |
|---|---|---|---|---|
| N | Jones | -30 | Jones | 60 |
| N | Thomas | 50 | Thomas | 100 |
| **Y*** | **Smith** | **-100** | **Smith** | **-50** |
| **Y*** | **Smith** | **-50** | **Smith** | **100** |
| **Y*** | **Rogers** | **-100** | **Rogers** | **-200** |
| N | Dix | 100 | Jones | -100 |
| N | Evans | 50 | Jones | 100 |
| **Y*** | **Rogers** | **-200** | **Smith** | **-500** |
| **Y*** | **Rogers** | **-100** | **Jones** | **0** |
| N | Evans | 300 | Thomas | 500 |
| | Evans | 70 | Evans | 100 |
| | Jones | 80 | Thomas | -50 |

**Figure 4.** QbB – Visual Basic implementation

The use of genetic algorithms generates fresh problems, some soluble, but some more fundamental.

- The current version doesn't allow inter-attribute comparisons which were so powerful in the first version. This reduction in expressiveness was purely a simplification to speed initial development and will be corrected as the current version develops.
- Choosing appropriate parameters (mutation rates, numbers of generations etc.) to allow a suitable level of incremental change is quite difficult. Often the algorithm fixates on a non-optimal query. This is partly related to the very small training set sizes (records that the user has selected), but is also probably a matter of fine-tuning the underlying algorithms.
- The mechanism for generating the disjunctive normal form is to breed a population of conjunctions and at the end choose a subset which best cover the data. This is far simpler and (faster) than making each member of the population

be a full disjunctive query, but has an unfortunate side effect. If there are several clusters of records that need to be selected, each needing a different conjunctive query, the population tends to develop so that all the members match the same (largest) cluster. In practice, the algorithm tends to produce a single conjunction.

This last problem has been noted by others using symbolic genetic algorithms with various 'fixes'. One alternative is to make the individual population members more complex queries (e.g. disjunctive normal, decision tree, raw SQL). However, this would dramatically increase the learning time and has been observed by others to produce rather Byzantine rules.

An alternative approach is to introduce competition similar to that fund in natural selection in the wild. The development of these *artificial ecosystems*, or *eco-algorithms* is currently being undertaken with colleagues at the University of Birmingham.

**would the real QbB ...**

The simulated screenshot in figure 2 and the two real screenshots of the Macintosh and Visual Basic version of QbB in figures 3 and 4 all show the same stage n QbB, when the system has inferred a query. However, the three are quite different. At a superficial level they have a different look-and-feel because of the nature of the platforms and development environments used. However, they also differ in the nature of the query. The simulated system in figure 2 shows an SQL-like query, figure 3 has a decision tree and figure 4 a Query-by-Browsing style tableau. Also because they use different algorithms the queries they infer and their reaction to user input will be different. However, they clearly represent the same concept.

## Interactive querying

The imprecise and exploratory nature of most IR tasks has naturally lead to an interactive style of 'querying'.. The user enters keywords, gets results, refines the keywords etc. In contrast, the formal nature of traditional database queries favour a pre-specified batch approach: formulae query – get results – stop. However, real experience of the latter belies this. Either because it is difficult to formulate the right query, or because one is searching for an informally-specified result (as in many IR tasks), the database query process is often more cyclic: query – get results – refine query ... etc.

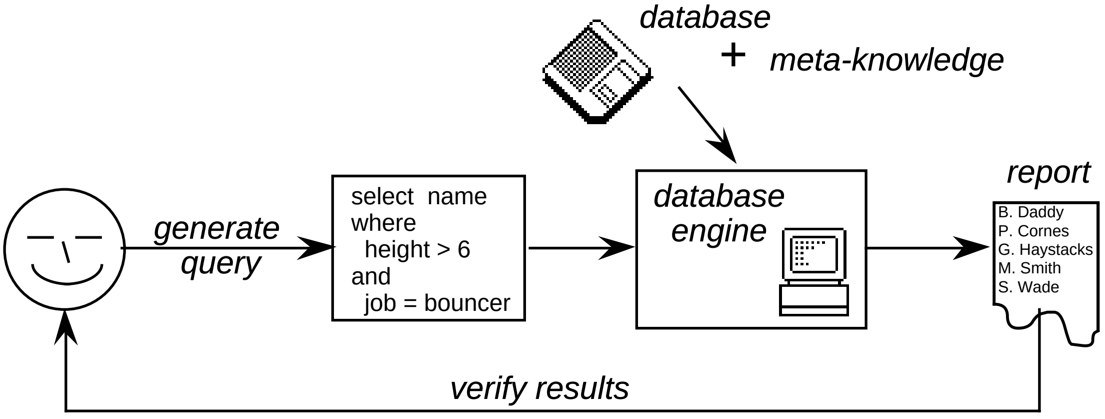


**Figure 5.** traditional database interaction

This form of exploratory interaction is even more pronounced in web navigation. Users will follow links and use search engines, not because there is a specific page or group of pages out there, but in order to discover if there is information on a topic and to 'see what is there'. This exploratory behaviour is evident in the frequent use of the back button (surveys measure between 30% [Tauscher, 1997] and 40% [Catledge, 1995] of user interactions) – although some of these will be 'undoing' mistakes many represent 'hub and spoke' or depth-first search [Tauscher, 1997]

Like IR and database systems, web browsing consists of a cyclic interaction: follow a link – get results – go deeper or go back and try a different link (refine query). Indeed all are examples of the general Norman 'execute–evaluate' cycle of interaction [Norman , 1988].

## Understanding the problem

We need to look at the interactive retrieval process at two levels.

First we need to look at the *outside* picture:

- what kind of **data** are we dealing with (e.g. numeric, text, hierarchical)
- what kind of **result** do we want (in terms of its structure)
- how are our **goals** formulated (precise, unknown, exploratory)

Then we need to look *inside* the interactive loop:

- who **starts** it (a blank screen for a user query, or an initial display)?
- what sort of **user input**/feedback does the user give to the system (SQL query, relevance feedback on records, parameter values)?
- what **processing**/matching/filtering does the system (select records on query, ranking by semantic association, machine learning)?
- what **output** does it produce (single records, set of records, clusters, ?
- what sort of **display** does the system produce (lists of records, summary/aggregate data, graphical views)?

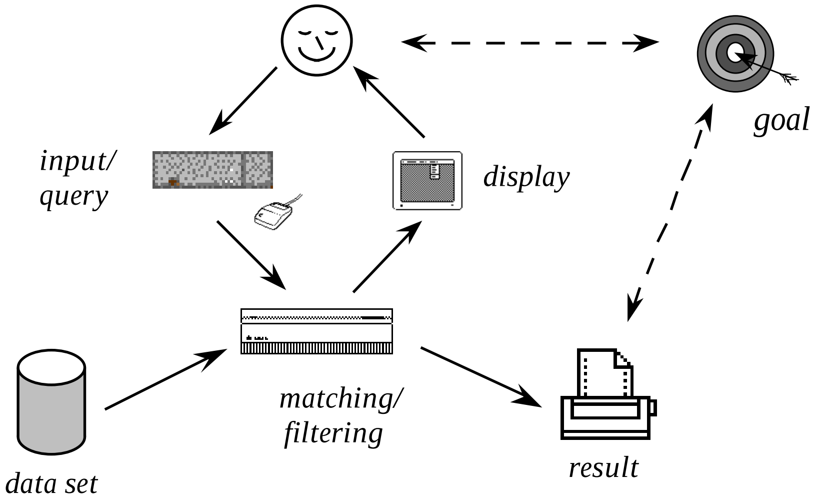


**Figure 6.** The interactive querying process

Let's compare traditional database , IR, hypertext and QbB using this process:

| | **database** | **IR** | **hypertext /www** | **QbB** |
|---|---|---|---|---|
| **dataset** | set of records | set of texts | linked pages | set of records |
| **final result** | set of records | small set of texts | collection of pages | query + matching records |
| **goal** | specific records | something relevant | specific page or learn during process | specific records or reusable query |
| **start** | blank screen | blank screen | home page or search page | listing of database |
| **user input** | query | keywords or relevance feedback | follow link | small set of classified records |
| **processing** | select records | rank on keywords or semantic association | fetch named page | machine learning |
| **output** | set of records | ranking of records (+ keywords) | single page | query + matching records |
| **display** | listing | count of matches or small set of ranked texts | page contents | SQL / QbE query + listing |

In each case, the final result and intermediate output are similar, but with some differences. In particular, in both IR and hypertext the actual result is not just the last output, but some collection of selected texts / pages (using a computer assisted mechanism such as bookmarks, or simply paper and pencil). Furthermore, the goal may mean that one part id important. In QbB the ultimate goal may either to have the selected records (to print or further process), or the query for future use. In hypertext this is particularly interesting as the real benefit may have accrued during the process.

In all cases it is also important to recognise the importance of the **dynamics** of the process, how the interactive process feels to the user. As we noted, in the case of hypertext, this may be the whole purpose of interaction. Also, for QbB one of the reasons for looking at different learning algorithms is to improve the dynamics of interaction, not the expressiveness of efficiency of query inferencing. Also, the need for rapid feedback in QbB rules out certain slow algorithms and puts a limit on the number of generations and population size for genetic algorithms.

Of course, from the viewpoint of QbB the really interesting thing is that it effectively reverses the roles of input and output. In all cases, the nature of input and output was either a query (if we include keywords in that category) and some collection (perhaps single) of records / texts however in a database the processing is of the form:

query → collection of records

whereas in QbB this is:

collection of records → query

We can see that the versions of QbB differ in the style of machine learning in the 'processing' row and in the type query visualisation used (and to a lesser extent the kind of query generated), but have the same distinctive 'back-to-front' style of interaction.

Note that output column for IR has "(+keywords)". This is because some IR systems generate a set of (ranked) keywords based on relevance feedback. Note that this is precisely the analogue of QbB, reversing the keywords to documents mode of processing.

## Summary

Looking at the interactive querying process helps us see the distinctive nature of Query-by-Browsing which turns the traditional database processing cycle inside-out. However, it also emphasises the similarity between different forms of interactive querying. In particular, QbB has deep similarities of structure with relevance feedback systems for free text suggesting that it will extend well to hybrid database systems.

## and more ...

Both the Macintosh and Visual Basic versions of QbB can be downloaded from:

http://www.hiraeth.com/alan/topics/QbB/

Watch this page for a more detailed description of the stages interactive querying process, for prototypes incorporating different algorithms and for news on the development of eco-algorithms and QbB in Java.

## References

C. Ahlberg, C. Williamson and B. Shneiderman (1992). Dynamic queries for information exploration: an implementation and evaluation. *Proceedings of CHI'92*, ACM Press. pp. 619–626.

C. Ahlberg and B. Shneiderman (1994a). Visual information seeking: tight coupling of dynamic query filters with starfield displays. *Proceedings of CHI'94*, Boston, ACM Press. pp. 313–317.

C. Ahlberg and B. Shneiderman (1994b). The Alphaslider: a compact and rapid selector. *Proceedings of CHI'94*, Boston, ACM Press.

S. Benford and J. Mariani (1994). Virtual environments for data sharing and visualisation – populated information terrains. *Proceedings of IDS'94: The 2nd International Workshop on User Interfaces to Databases*, Lancaster, UK, Springer Verlag. pp. 168–182.

F. Benzi, D. Maio and S. Rizzi (1998). VisTool: a visual tool for querying relational databases. *Proceedings of AVI98*, , ACM Press. pp. 258–260.

L. Catledge and J. Pitkow (1995). Characterizing browsing strategies in the World-Wide Web. *Proceedings of the Third International World Wide Web Conference*, http://www.igd.fhg.de/www/www95/papers/, Darmstadt, Germany.

M. Chavda and P. Wood (1997). Towrads an ODMG-compliant visual query language. *Proceedings of the 23rd Interbational Conference of Very Large Databases,* , pp. 456–465.

E. H. Chi, J. Pitkow, J. Mackinlay, P. Pirolli, R. Gossweiler and S. K. Card (1998). Visualizing the evolution of web ecologies. *CHI '98. Conference proceedings on Human factors in computing systems,* , ACM Press. pp. 400–407.

M. G. Cristel, M. A. Smith, C. R. Taylor and D. B. Winkler (1998). Evolving video skims into useful multimedia abstractions. *CHI '98. Conference Proceedings on Human Factors in Computing Systems,* , ACM Press. pp. 171–178.

A. Dix (1992). Human issues in the use of pattern recognition techniques. *Neural Networks and Pattern Recognition in Human Computer Interaction*, Eds. R. Beale and J. Finlay. Ellis Horwood. pp. 429-451.

A. Dix and A. Patrick (1994). Query By Browsing. *Proceedings of IDS'94: The 2nd International Workshop on User Interfaces to Databases,* Lancaster, UK, Springer Verlag. pp. 236–248.

A. Dix (1998). The Active Web – Part 1. *Interfaces,* **38**:18–21. Summer 1998.

T. T. Elvins, D. R. Nadeau, R. Schul and D. Kirsh (1998). Worldlets: 3D thumbnails for 3D browsing. *CHI '98. Conference proceedings on Human factors in computing systems,* , ACM Press. pp. 163–170.

G. W. Furnas (1986). Generalised fisheye views. *Proceedings of CHI'86,* ACM Press.

G. W. Furnas and J. Zacks (1994). Multitrees: enriching and reusing hierarchical structure. *Proceedings of CHI'94,* Boston, ACM Press.

J. Goldstein and S. F. Roth (1994). Using aggregation and dynamic queries for exploring large data sets. *Proceedings of CHI'94,* Boston, ACM Press.

P. D. Gray, K. W. Waite and S. W. Draper (1990). Do-it-yourself iconic displays. *Interact'90,* Elsevier.

F. Halasz, T. Moran and R. Trigg (1987). NoteCards in a nutshell. *Proceedings of the CHI+GI.* ACM, New York. pp. 45-52.

D. Haw, C. Goble and A. Rector (1994). GUIDANCE: making it easy for the user to be an expert. *Proceedings of IDS'94: The 2nd International Workshop on User Interfaces to Databases,* Lancaster, UK, Springer Verlag. pp. 19–43.

R. Jain (1997). Introduction – special issue on Visual Information Manangement. *Communications of the ACM,* **40**(12):30-32.

S. Mace, U. Flohr, R. Dobson and T. Graham (1998). Weaving a better web. *Byte,* March 1998. pp. 58–68.

J. D. Mackinlay, G. G. Robertson and S. K. Card (1991). The perspective wall: detail and context smoothly integrated. *Proceedings of CHI'91,* ACM Press.

N. Murray, N. Paton and C. Goble (1998). Kaleidoquery: a visual query language for object databases. *Proceedings of AVI98,* , ACM Press. pp. 247–257.

D. A. Norman (1988). *The Psychology of Everyday Things.* Basic Books.

Oracle (1992). *SQL*TextRetrieval Developers Guide Version 2.0.* Oracle Corporation.

A. Papantonakis and P. J. H. King (1995). Syntax and semantics of Gql: a graphical query language. *Journal of Visual Languages,* **6**:3–25.

P. Pirolli, P. Schank, M. Hearst and D. C (1996). Scatter/gather browsing communicates the topic structure of a very large text collection. *Proceedings of CHI'96,* Vancouver, ACM Press. pp. 213–220.

J. R. Quinlan (1986a). Induction of decision trees. *Machine Learning,* **1**(1).

J. R. Quinlan (1986b). Simplifying decision trees. *International Journal of Man-Machine Studies,* **27**(4):221–234.

P. Resnick and H. R. Varian (1997). Special Issue on Recommender Systems. *CACM,* **40**(3):56–89.

G. G. Robertson, S. K. Card and J. D. Mackinlay (1991). Cone Trees: Animated 3D Visualisation of Hierarchical Information. *Proceedings of CHI'91 Conference of Human Factors in Computing Systems,* , ACM Press. pp. 184–194.

L. Tauscher and S. Greenberg (1997). How people revisit web pages: empirical findings and implications for the design of history systems. *International Journal of Human Computer Studies,* **47**(1):97–138.

L. Tweedie, R. Spence, R. Boghal and D. Williams (1994). The Attribute Explorer. *Video Proceedings nd Conference Companion, CHI'94,* , ACM Press. pp. 435–436.

L. Tweedie, R. Spence, H. Dawkes and H. Su (1996). Externalising Abstract Mathematical Models. *Proceedings of CHI'96,* Vancouver, ACM Press. pp. 406–412.

L. D. Wilcox, B. N. Schilit and N. Sawhney (1997). Dynomite: a dynamically organized ink and audio notebook. *Proceedings of CHI'97,* Atlanta, ACM Press. pp. 186–193.

M. M. Zloof (1975). Query by example. *Proc. AFIPS National Computer Conf. 44,* , AFIPS Press, New Jersey. pp. 431–438.