The Information Quest: A Dynamic Model of User's Information Needs

Paul B. Kantor, Endre Boros, Benjamin Melamed, Vladimir Meñkov
Rutgers University
4 Huntington St, New Brunswick, NJ 08903
kantor@scils.rutgers.edu http://aplab.rutgers.edu/ant/

Abstract

In networked information environments, using server-browser architectures, nearly all information finding episodes become extended interactions between the user and the system. In this setting the system needs some way to "understand" what the user is seeking, as this goal adapts and is modified during a session or a series of sessions. We describe a formal model, in which the model of the user's quest is represented as a generalized abstract "response function" representing the user's response to the information delivered by the system. Representing this response as $u(n) = Q(S(n-1))$ shows that the user's utterance $u(n)$ at a time step n is determined according to the user's "response function" Q by the materials $S(n-1)$ that had been presented up through the previous time step $n-1$.

The entire history of materials presented thus plays a role in determining the user's response, providing a very rich probe into the precise nature of the user's information quest, here represented by the rule Q. We show how this gives rise naturally to a new model for assimilating relevance feedback information, and to the concept of itineraries in the information network. Finally the concept of an information quest Q, provides a natural framework for considering the time dependence of information about the user's needs, and for various models of information aging. The use and effectiveness of this concept are illustrated with data collected in the Ant World Project at Rutgers.

1 Introduction

The Ant World Project at Rutgers (Ant World, 1999; Kantor et al., 1999), simulates the laying of information-bearing traces in the World Wide Web. Since the links of the World Wide Web are incorporeal, these traces are actually stored in a database maintained at the Ant World server site (aplab.rutgers.edu (Ant World, 1999)). The contents of that database include a quest identification number, representing the quest of the user who left the mark, a pair of URLs indicating where the link comes from and where it goes to, and a user judgment, which varies according to the particular judgment scheme being tested at the moment. Unlike popularity-ranking systems such as Direct Hit (Direct Hit, 1999), which try to infer the user's opinion based on indirect feedback, such as the time he spends viewing a document, the Ant World asks the user to explicitly provide his judgment of the usefulness of the document. In early implementations the judgment is drawn on a 5 point scale from "extremely useful" to "worthless".

The ID of the quest Q is linked in the database to the quest's descriptions, provided by the user at the beginning of the quest: the *short description*, or SD (Q), and an optional *long description*, or LD (Q). The

SD is a very short text, similar to a search engine query, for example "data fusion information retrieval". The LD is an extended natural language text, resembling a TREC query, e.g. "I am interested in learning more about data fusion and information retrieval, and in particular on non-parametric methods for estimating the most effective rule for combining several IR systems".

These short and long quest descriptions provide an initial representation of what this particular individual user wants on this particular occasion. If we think of them in terms of the quest response function, $Q(S)$, which represents the way in which the user with this particular quest responds to what the system S has revealed so far, this is in fact the response to the empty set: $Q(\emptyset)$. Even in this raw form, obviously it contains more information than that on which most web search engines base their retrieval. In actual use the Ant World system in fact does a preliminary retrieval from the base of stored quests, using this information as a query. Before examining the details of that retrieval mechanism, we consider what happens as individual pages are viewed and judged.

As a new page is seen and judged, its contents are noted in a database, which is in effect an inverted index to all the pages that have been seen so far. In addition, the judgment is attached to that information together with the user's quest ID.

When the quest is created, the pairs $(\sigma_S, \mathrm{SD}\,(Q))$ and $(\sigma_L, \mathrm{LD}\,(Q))$ stored in the database constitute the quest profile. (The σ_S and σ_L are special symbols indicating that the texts marked by them are an SD and an LD). As time progresses and the user visits more pages, the quest profile is elaborated by the addition of pairs of the form (J, d). In each such pair J is a tuple representing the user's judgment, and d is the full text of the document that the user is seeing when he or she makes this judgment. In a sense, σ_S and σ_L play the role of special judgments.

The judgment J is a tuple because not only the "judgment label" (the label on the button in the console window (Kantor et al., 1999) the user clicked when he judged the document), but also the configuration which determines the rule in effect at the time that judgment was posted (such as what is the set of available judgment buttons). In addition, for time sensitive information, the tuple must also contain the time τ at which the judgment was posted. Thus the judgment J is

$$J = (s, v, \tau) \tag{1}$$

where s is the "judgment label" (which is an element the set of possible judgment labels allowed in the configuration v), v is the system configuration (which determines the judgment options set) in effect when that judgment was made, and τ is the clock time at which the judgment was posted.

This information forms the basis for our investigations in quest description and quest matching. As noted, each such judgment can be represented as the response of the user to the system's presentation of the document represented by D, given everything that the user has experienced and can remember during the course of this quest, and from her entire outside life.

2 Quest Matching

As judgments are applied to individual pages, we could, if we had a suitable differencing technology, infer that the judgment applies to that which is new and different about the given page. One such trivial technology is to subtract, from the vector representing the given page, a normalized vector representing the sum of previously seen pages. However, to represent the user's mental processes, each earlier page ought to be weighted according to the attention that the user gave to it, a quantity which is unknown to us. Hence, at present, we do not attempt differencing.

For matching of quests to each other, which is needed in order to recall judgments posted by users whose quests most nearly match the present quest, we adopt a hybrid vector retrieval system. Note that the quest

Q is defined as a mapping from the space of system responses (pages delivered) to the space of judgments. This mapping can be represented by its graph, as a set of ordered pairs. The set of ordered pairs is what we store and manipulate in our database. The representation of the present quest denoted by Q, plays the role of a query in conventional information retrieval, and the collection of stored quests Q' represents the corpus or document collection. In general then we are concerned with measuring the quest similarity score $\mathrm{Sim}\,(Q, Q')$ between the present quest and any particular stored quest Q'. Given any specific choice of the similarity function $\mathrm{Sim}\,(\cdot,\cdot)$ the system compiles the *relevant quest list:* the list of stored quests sorted by decreasing similarity score, down to some cutoff level.

For every document that has been judged by a user during any of the relevant quests, the system computes a putative usefulness score, based on the judgments about this document made during the quests on the relevant quest list. If the combined score exceeds a certain threshold (which is a variable parameter determined by the present operating configuration of the Ant World system), the document's URL is included into the *suggestion list* for the present quest Q: the list of URLs of potential interest to the user running quest Q.

During quest Q, whenever the page currently viewed by the user contains a link to any of the documents on the suggestion list, a small red ant appears on the user's screen next to this link.

To reiterate, the system must make two decisions. The first is the determination of similarity between the present quest and stored quests. The second is the setting of a threshold to determine whether ants should appear on the screen. The same threshold is also used to determine whether pages appear in a list of suggested pages, and the judgment of similarity is used to determine the order in which these pages appear.

As stated, the system uses a variant of the vector model. Each document d is represented by a vector over a base consisting of all the unstemmed non-stop-word terms that have appeared in the collection C, which consists of N documents: all documents viewed during all the recorded quests, plus the SD and LD of all quests. For every document d in the collection and for any non-stop-word term t occurring in it, we store $f(t, d)$: the (raw) term frequency, i.e. the number of occurrences of term t in document d.

2.1 Document similarity

In our current computational model (as of May 1999), the similarity of quests is based upon the similarity of the labeled documents associated with the quests, i.e. the quest's SD and LD, and the set of documents judged during the quest). The similarity of documents d_1 and d_2 is computed using Singhal's formula (Singhal, 1997):

$$\mathrm{Sim}\,(d_1, d_2) = \sum_t \phi(t, d_1)\phi(t, d_2)g(t), \tag{2}$$

with summation over terms that occur in both d_1 and d_2 (i.e. $f(t, d_1) > 0$ and $f(t, d_2) > 0$). Here $\phi(t, d)$ is the normalized frequency of term t in document d:

$$\phi(t, d) = \left(\frac{1 + \ln f(t, d)}{1 + \ln \langle f(\cdot, d)\rangle}\right) \cdot \frac{1}{\mu(d)}; \tag{3}$$

$\mu(d)$ is the document length normalization factor:

$$\mu(d) = \left(0.8 + 0.2 \cdot \frac{\mathrm{nDU}\,(d)}{\langle \mathrm{nDU}\,(\cdot)\rangle}\right)^{-1}; \tag{4}$$

$\mathrm{nDU}\,(d)$ is the number of the unique terms in document d

$$\mathrm{nDU}\,(d) = |\{t : f(t, d) > 0\}|\,; \tag{5}$$

$\langle \mathrm{nDU}\,(\cdot)\rangle$ is the average of $\mathrm{nDU}\,(d)$ over all documents d in the collection:

$$\langle \mathrm{nDU}\,(\cdot)\rangle = \sum_{d\in C} \mathrm{nDU}\,(d)/N; \tag{6}$$

$\langle f(\cdot,d)\rangle$, is the average term frequency for document d, i.e.

$$\langle f(\cdot,d)\rangle = \frac{\sum_{t:f(t,d)>0} f(t,d)}{\mathrm{nDU}\,(d)}. \tag{7}$$

The term prevalence factor $g(t)$ in equation (2) is defined so that it is high for rare terms and low for common words:

$$g(t) = \left[\ln\left(\frac{1+N}{\mathrm{df}\,(t)}\right)\right]^2, \tag{8}$$

where $\mathrm{df}\,(t)$ is the document frequency of the term t (the number of documents in the collection that include t):

$$\mathrm{df}\,(t) = |\{d : f(t,d) > 0\}|\,. \tag{9}$$

2.2 Quest similarity

In general, one can define the similarity measure of two quests Q_1 and Q_2 via the similarity of the underlying documents in the following manner:

$$\mathrm{Sim}\,(Q_1,Q_2) = \sum_{d_1\in Q_1}\sum_{d_2\in Q_1} w(J(d_1,Q_1),J(d_2,Q_2))\,\mathrm{Sim}\,(d_1,d_2). \tag{10}$$

Here $J(d,Q)$ is the judgment J — part of the (J,d) pair stored in the profile of the quest Q in the database — that has been given by the user to the document d in the quest Q. The coefficient matrix w can be thought of as a function

$$w : \mathcal{J}\times\mathcal{J}\to\mathcal{R},$$

where J is the set of all possible judgment values, which are either triples of the form (1), or are special values σ_S and σ_L.

The coefficients $w(J_1,J_2)$ should be defined to indicate the importance of the documents involved for the quests. A reasonable model would probably have $w(\sigma_S,\sigma_S)$ as a large positive number, and

$$w(\sigma_S,\sigma_S) \geq w(\sigma_S,\sigma_L) = w(\sigma_L,\sigma_S) \geq w(\sigma_L,\sigma_L) > 0,$$

since the similarity of short descriptions is extremely important, and similarity of long descriptions is quite important too. A smaller positive value could be associated with $w(\sigma_S,J_+)$, or $w(J_+,J'_+)$, where J_+, and J'_+ are some judgment values indicating a user's approval of a page. If J_+ indicates approval and J_- indicates disapproval, then it makes sense to define $w(J_+,J_-)$ as zero or a negative number. We disagree among ourselves as to what value should be assigned to $w(J_-,J_-)$. From one point of view, quests that are not helped by the same page are indeed similar. But the reasons for judging a page negatively may be so diverse that this would be misleading. It is therefore not clear whether this region of the weight matrix should be zero or positive.

In practice, performing fast real-time computations with an arbitrary coefficient matrix w may produce further bottlenecks. For this reason we have so far limited our experiments to a simpler model in which w has a product form ($\mathrm{rank}\,w = 1$):

Judgment label	The converted grade value $\lambda(J)$
"Meets my needs"	1.0
"Adds information"	0.75
"Helps navigation"	0.75
"Not useful"	0.0
"No comment"	0.0

Table 1: Grade conversion table for configuration VM06

$$w(J(d_1, Q_1), J(d_2, Q_2)) = \lambda(J(d_1, Q_1))\lambda(J(d_2, Q_2)). \tag{11}$$

Here $\lambda : \mathcal{J} \to \mathcal{R}$ is the *grade conversion function* that maps users' judgments (which may be symbolic or numeric) to real numbers.

The currently used $\lambda(\cdot)$ is defined as follows: For short and long descriptions, $\lambda(\sigma_S) = 6.0$ and $\lambda(\sigma_L) = 3.0$. For a "page evaluation" judgment $J = (s, v, \tau)$, $\lambda(J)$ is in the range $[0; 1]$, depending on the judgment label s and configuration v, but not (yet) on the judgment time τ. The actual conversion table (from s to J) is created for each configuration v when the configuration is created, along with the names of the judgment buttons, as in Table 2.2.

Note that at present we are making no use of the conceptual distinction between a page that seems useful in a navigational way, and one that is useful in terms of providing "information" per se. We have not yet conducted the human factors experiments needed to tell us whether these should be teated differently. If the are to be treated differently we will most likely have to give up the simple product form for w.

Factorization (11) allows to factorize the quest similarity formula (10) as

$$\mathrm{Sim}\,(Q_1, Q_2) = \sum_t \Psi(t, Q_1)\Psi(t, Q_2)g(t). \tag{12}$$

Here the *quest profile vector* $\Psi(\cdot, Q)$, representing the entire quest Q as if it were a single unified document, is a linear combination of the vectors ϕ for the underlying documents:

$$\Psi(t, Q) = \sum_d \lambda(J(d, Q))\phi(t, d). \tag{13}$$

Our quest matching is *dynamic*. This means that we update the tables that store the quest profile data (Ψ) and term prevalence factor (g) in real time. As the user makes more judgments during his quest Q, the relevant quest list and the suggestion list are continuously updated.

2.3 Problems with measuring quest similarity

A disadvantage of document and quest similarity formulas (2, 12), as compared to cosine models (Salton, 1971), is that there is no natural upper bound, such as 1.0, on the value of this similarity measure. (Since all λ's are non-negative, the $\mathrm{Sim}\,(Q, Q')$ can only increase as new judgments are added to Q or Q'!) When compiling the list of relevant quests Q' for the present quest Q, ordering the quests by the degree of relevance, or scoring a link, we normalize all similarity contributions by $\mathrm{Sim}\,(Q, Q)$ ("the similarity of the quest to itself"). Thus the list of quests relevant to Q is

$$\mathrm{Rel}\,(Q) = \{Q' : \mathrm{Sim}\,(Q', Q) \geq c\,\mathrm{Sim}\,(Q, Q)\} \tag{14}$$

Term t	$s_{11}(t)$	Term t	$s_{22}(t)$	Term t	$s_{12}(t)$
tokugawa	21.6	npa	28.0	brewing	4.2
bakufu	16.6	reactionary	25.8	policy	3.3
modernisation	14.1	fighters	24.6	national	3.2
perry	12.9	offensives	24.4	were	3.2
meiji	12.6	enemy	20.9	monopoly	3.1
edo	11.1	tactical	19.5	had	3.1
edicts	11.1	masses	16.5	there	3.1
anu	9.9	revolutionary	15.1	revolution	2.9
ô	9.8	campaigns	13.4	urban	2.6
1639	9.2	cpp	12.6	production	2.5
1630s	9.2	semicolonial	12.5	their	2.4
û	9.2	monopoly	12.4	occurred	2.3
sakoku	9.2	party	12.0	tactics	2.3
seclusion	9.2	fronts	11.6	been	2.3
japanese	8.1	semifeudal	11.0	was	2.3
japan	7.8	rectification	11.0	carry	2.3
christianity	7.6	guerrilla	10.7	he	2.3
17th	7.6	democratic	10.6	upon	2.2
shizuki	7.1	ndfp	10.5	economic	2.2
kaempfer	7.1	carry	10.5	that	2.2

Table 2: Top 20 contributions to $\mathrm{Sim}\,(d_1, d_1)$, $\mathrm{Sim}\,(d_2, d_2)$, and $\mathrm{Sim}\,(d_1, d_2)$. The contribution of term t to $\mathrm{Sim}\,(d_i, d_j)$ in equation (2) is $s_{ij}(t) \equiv \phi(t, d_i)\phi(t, d_i)g(t)$

(with the cutoff value $c = 0.2$), and the score for a link to document d, in the context of quest Q, is

$$\mathrm{Score}\,(d) = \sum_{Q' \in \mathrm{Rel}\,(Q)} \lambda(J(d, Q'))\,\mathrm{Sim}\,(Q', Q)/\,\mathrm{Sim}\,(Q, Q). \tag{15}$$

The suggestion list consists of all URLs d for which $\mathrm{Score}\,(d) > 0.1$.

This normalization is still far from perfect solution, however. If Q' is a much larger quest than Q (i.e. it includes a much greater number of judged documents), $\mathrm{Sim}\,(Q, Q')/\,\mathrm{Sim}\,(Q, Q)$ may be quite high — theoretically, even greater than 1 – even if the similarity $\mathrm{Sim}\,(d, d')$ of each particular pair $(d \in Q, d' \in Q')$ of underlying documents is low. A possible solution, which we have not tried but which is very much in line with the traditional cosine model, is to normalize $\mathrm{Sim}\,(Q, Q')$ by dividing it by $(\,\mathrm{Sim}\,(Q, Q)\,\mathrm{Sim}\,(Q', Q'))^{1/2}$.

Choosing the values of the coefficients λ's in (11) is a tricky issue. How important should the SD and LD be? In earlier experiments, when $\lambda(\sigma_S)$ and $\lambda(\sigma_L)$ were much higher than the current values of 6 and 3 (on the order of 10–20), the influence of the quests' extended descriptions (judged pages) on the determination of quest similarity was insignificant: basically, we just matched the short and long descriptions of one quest to those of the other. With the current values, $\lambda(\sigma_S) = 6.0$ and $\lambda(\sigma_L) = 3.0$, extended descriptions are important—but the chance of spurious quest similarity are quite high too. The latter phenomenon is due to the fact that even when the topics of two documents d_1 and d_2 are quite different, their similarity value $\mathrm{Sim}\,(d_1, d_2)$ can be quite high because of numerous not-very-common words that they share. With large quests, this may add up quickly.

We illustrate some of the issues surrounding quest matching with the following example. (In this example, all numbers include stopwords.) Document d_1 = `http://coombs.anu.edu.au/SpecialProj/APM/TXT/-low-m-02-96.html` (document ID= -8117), with $\mathrm{nDU}\,(d_1) = 1236$ comes from a quest on Japanese history (quest ID=7856); document d_2 = `http://www.geocities.com/CapitolHill/2078/npa7.htm` (document ID= -958), with $\mathrm{nDU}\,(d_1) = 568$ comes from a quest on Philippine guerilla movements (quest ID=1312).

	Stopwords retained	Stopwords excluded
Sim (d_1, d_1)	1730.6	1557.9
Sim (d_2, d_2)	1821.7	1607.3
Sim (d_1, d_2)	231.7	122.70

Table 3: Using stopword exclusion to suppress spurious document similarity

The self-similarity of quests, using Singhal's formula, Sim $(d_1, d_1) = 1730.6$, and Sim $(d_2, d_2) = 1821.7$, comes to a large extent from words that are indeed specific to the topic. But Sim $(d_1, d_2) = 231.7$ mainly comes from words that can be found in any paper on a political or historic topic. The top 20 terms contributing to each of the three similarity values are shown in Table 2.3. From the first two lists we get a rough idea of what the documents d_1 and d_2 are about; but the third list (that of the top-contributing words the two documents have in common) seem to convey little idea about either document.

We have alleviated the spurious similarity problem by omitting stopwords from all computations, using the popular 429-word stopword list (Fox, 1992). Excluding stopwords helps because their contributions to Sim (d_1, d_2) for unrelated documents d_1 and d_2 is relatively more significant than their contribution to either document's similarity to itself. For the above example, the effect of stopword exclusion is illustrated in Table 2.3. No doubt the situation would be improved much more if we represented texts in terms of two-word and three-word "phrases" based on statistical analysis. This technique has been used (with the initial exclusion of single terms) in the work of Schatz and Chen (Chen et al., 1996).

Another precision problem we observe in an Ant World search arises because the formula (15) is additive. A document need not appear in a highly relevant quests to get a high score; it is enough to appear in a large number of marginally relevant (just above the cutoff for the inclusion into the relevant quest lists) quests.

3 Characteristics of the Ant World Quest Database

The Ant World quest database is stored using a relational database management system (Sybase SQL Server 11). At present (May 21, 1999) the database contains informations on 1,000 non-trivial quests (quests with at least 2 documents viewed), which have been run by the project staff and a few student testers over the previous 11 months. During these quests 8,000 unique URLs have been viewed, and $N = 1650$ of them have been judged. There are 23,000 itinerary and judgment entries (one entry per each document viewed and per each judgment). There are 1.5×10^6 entries in the `Inst` table, which contains the entire text of all N judged documents split into words. (We don't currently use this table, but it is maintained against the time when we consider pairs and triples of consecutive words.) The table `Freq`, which contains the term frequency data $f(t, d)$ contains 462,000 entries, and the table `Psi`, which contains the quest profile vectors $\Psi(t, Q)$ has 305,000 entries. The judged documents contain 65,000 different words; for each word, there is an entry in the `Prev` table, which stores the term prevalence values $g(t)$.

4 Performance of the System

The algorithm we use to compute quest relevance and link scores according to the formulas (3, 4, 8, 12) presented in Section does not scale well as the number of users and quests grows, since every time a user makes a judgment about document d, a large number (nDU (d)) of entries need to be added to the table Ψ describing the current quest profile.

We are running the database server on a Sun Sparc Ultra-1 workstation (aplab.rutgers.edu), with a 167

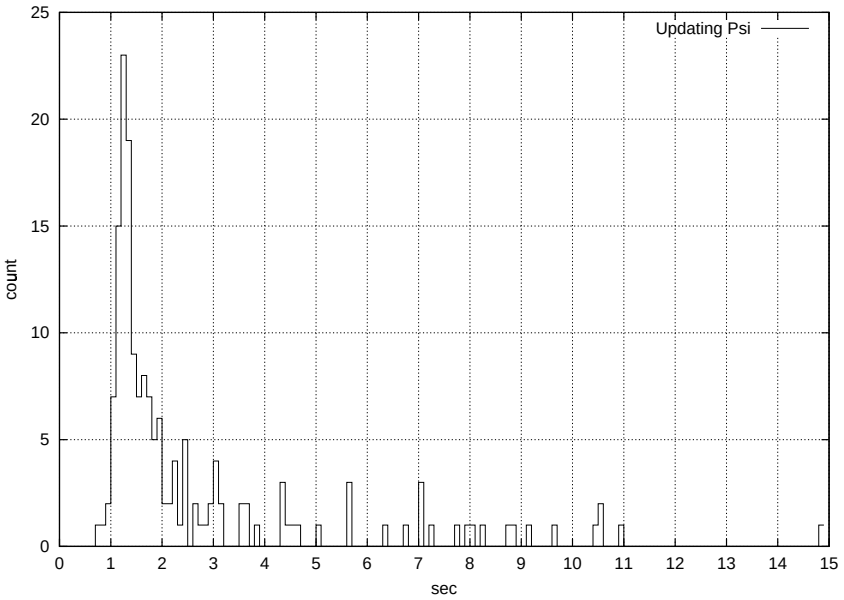

Figure 1: Distribution of the cost of updating Ψ after a document has been added to the quest profile. All 170 data points are within the figure range.

MHz processor (according to `psrinfo`) that performs at about 15 Mflops at simple timing tests, and 246 megabytes of RAM (according to `top`). The web server runs on the same computer.

Depending on the quest and the current load of the computer, it takes 1–10 seconds (wall clock time) to incrementally update the Ψ vector (equation (13) after a new document is added, as shown by the histogram in Figure 1. To compile the relevant quest list as per equation (14) takes 1–5 seconds. It takes 0.1–2 sec to compile the suggested documents list (the list of all d whose scores (15) are above the threshold 0.1; Figure 3).

Acknowledgments

This work is supported by the Defense Advanced Research Projects Agency (DARPA) Contract Number N66001-97-C-8537.

References

Ant World (1999). The Ant World web site, `http://aplab.rutgers.edu/ant/`.

Chen, H., Schatz, B., et al. (1996). A parallel computing approach to creating engineering concept spaces for semantic retrieval: The Illinois Digital Library Initiative Project. *IEEE Trans Pattern Analysis and Machine Intelligence*, 18:771–782.

Direct Hit (1999). The Direct Hit web site, `http://www.directhit.com`.

Fox, C. (1992). Lexical analysis and stop lists. In Frakes, W. and Baeza-Yates, R., editors, *Information Retrieval: Data Structures and Algorithms*, chapter 7. Prentice-Hall. The list of stopwords is available at `ftp://ftp.vt.edu/pub/reuse/IR.code/ir-code/stopper/stop.wrd`.

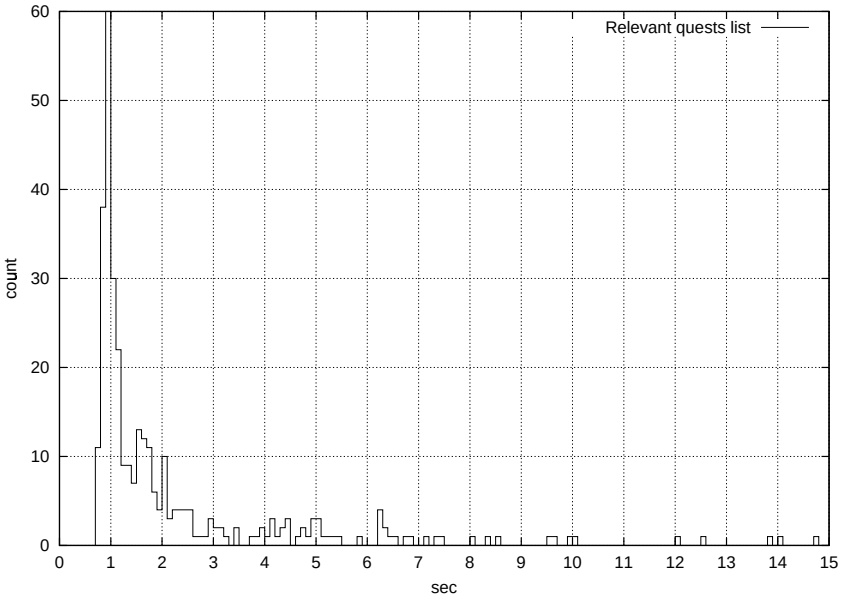

Figure 2: Distribution of the cost of computing the relevant quest list. Out of 334 data points, 6 are outside the figure range (in the range 15 to 70 sec).

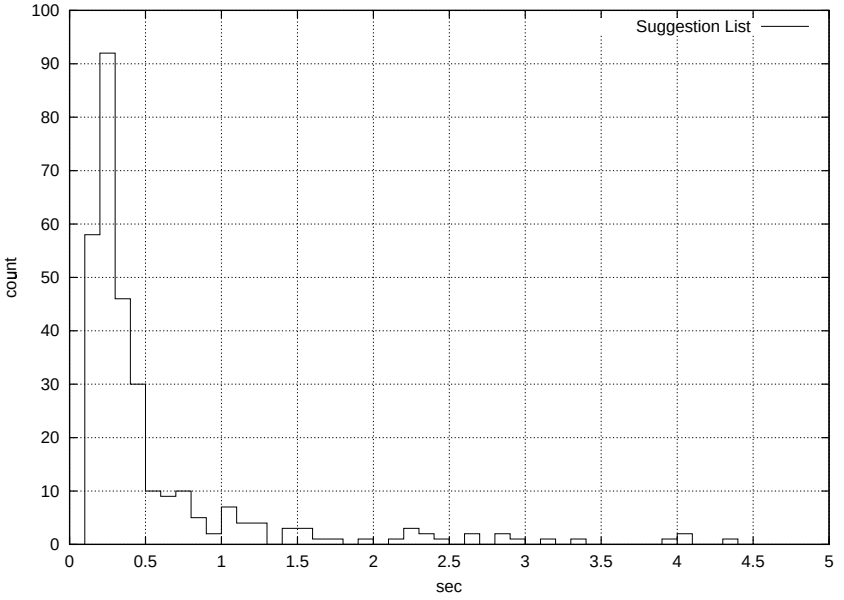

Figure 3: Distribution of the cost of computing the suggestion list. Out of 311 data points, 7 are outside the figure range (in the range 5 to 17 sec).

Kantor, P. B., Melamed, B., Boros, E., Meñkov, V., Neu, D. J., Kim, M.-H., and Shi, Q. (1999). Ant World. In *SIGIR'99 Proceedings*. To appear.

Salton, G. (1971). *The SMART retrieval system; experiments in automatic document processing*. Prentice-Hall, Englewood Cliffs, N.J.

Singhal, A. (1997). AT&T at TREC-6. In *NIST Special Publication 500-240: The Sixth Text REtrieval Conference (TREC 6)*, pages 215–226. NIST. Available at `http://trec.nist.gov/pubs/trec6/papers/att.ps`.