

Citation for published version:

Alessio Malizia, Kai A. Olsen, Tommaso Turchi, and Pierluigi Crescenzi, 'An ant-colony based approach for real-time implicit collaborative information seeking', *Information Processing & Management*, Vol. 53 (3): 608-623, May 2017.

DOI:

<https://doi.org/10.1016/j.ipm.2016.12.005>

Document Version:

This is the Accepted Manuscript version.

The version in the University of Hertfordshire Research Archive may differ from the final published version.

Copyright and Reuse:

© 2016 Elsevier Ltd.

This manuscript version is made available under the terms of the Creative Commons Attribution-NonCommercial-NoDerivatives License CC BY NC-ND 4.0

(<http://creativecommons.org/licenses/by-nc-nd/4.0/>), which permits non-commercial re-use, distribution, and reproduction in any medium, provided the original work is properly cited, and is not altered, transformed, or built upon in any way.

Enquiries

If you believe this document infringes copyright, please contact Research & Scholarly Communications at rsc@herts.ac.uk

An ant-colony based approach for real-time implicit collaborative information seeking

Alessio Malizia^{a,*}, Kai A. Olsen^{b,c}, Tommaso Turchi^a, Pierluigi Crescenzi^d

^a*Human Centred Design Institute, Brunel University London, London, UK*

^b*Faculty of Logistics, Molde University College, Molde, Norway*

^c*Department of Informatics, University of Bergen, Bergen, Norway*

^d*Department of Information Engineering, University of Florence, Florence, Italy*

Abstract

We propose an approach based on Swarm Intelligence — more specifically on Ant Colony Optimization (ACO) — to improve search engines' performance and reduce information overload by exploiting collective users' behavior. We designed and developed three different algorithms that employ an ACO-inspired strategy to provide implicit collaborative-seeking features in real time to search engines. The three different algorithms — NaïveRank, RandomRank, and SessionRank — leverage on different principles of ACO in order to exploit users' interactions and provide them with more relevant results. We designed an evaluation experiment employing two widely used standard datasets of query-click logs issued to two major Web search engines. The results demonstrated how each algorithm is suitable to be employed in ranking results of different types of queries depending on users' intent.

Keywords: Ant Colony Optimization, Cooperative Systems, Evolutionary

*Corresponding author

Email addresses: alessio.malizia@brunel.ac.uk (Alessio Malizia), kai.olsen@himolde.no (Kai A. Olsen), tommaso.turchi@brunel.ac.uk (Tommaso Turchi), pierluigi.crescenzi@unifi.it (Pierluigi Crescenzi)

1 **1. Introduction**

2 Traditionally text retrieval was based on keywords. However, not all
3 documents had been adequately tagged, neither could the keywords describe
4 all aspects of a document. With faster computers, it became possible to
5 perform full-text searches. Then we got the problem of too many hits, i.e.
6 the supplied keywords were found in too many documents. One tried to
7 cope with this by determining relevance as the number of occurrences of
8 each search term in the document, in relation to document size. The first
9 search engines on the Web used this approach.

10 There were several disadvantages to this approach. Looking for informa-
11 tion on a given car, using maker and model as keywords, the search engine
12 did not direct you to any official site. Instead, one was overloaded with car
13 for sale advertisements, as these had a good occurrence to size ratio of the
14 keywords. It was also quite easy to fool the search engines, for example by
15 adding long list of repeated keywords to a Web page, often using a small
16 white font so that it did not clutter the page.

17 Google's PageRank algorithm saved the day. By letting relevance be
18 determined by the number of links to a page, adding up the score if the links
19 also came from pages that had many links to them, Google had captured a
20 semantic understanding of the relevance concept. For example, many Web
21 pages may say something about The White House, and there may be many
22 white houses, but Google puts the official site on top, most probably the page

23 that the user wants. And every time someone makes a link to this page, they
24 increase its relevance.

25 The disadvantage of this approach is that it is static. Pages found im-
26 portant by the PageRank algorithm will probably get more important as
27 they are found by the search engine. That is, important pages will get more
28 important. New pages on similar topics will be hard to find, i.e. placed
29 further down on the search engine result page, and will thus be considered
30 less important. Over time, an algorithm used to determine relevance might
31 be self-fulfilling.

32 Ideally, we would need a search algorithm that were more dynamic, but
33 still gave a good idea of relevance. Our idea is to use data from the actual
34 searches - what we call dynamic trail information. While PageRank uses
35 static information as link structure, we want to collect data from the actual
36 searches performed by other users. For example, you may be interested in
37 renting a boat to go deep-sea fishing outside the Lofoten Islands in Northern
38 Norway. Your keywords may be rent, boat, fishing, Lofoten. The search
39 engine will then return a standard list of relevant pages; however, in addition
40 you will find a list that says: “other users found these pages”. That is, the
41 system have collected data on what other users with similar query terms
42 did. They may have started with the same keywords as you, but may also
43 tried other searches, ending up with a few interesting pages. That is, the
44 effort that other users have put in finding relevant pages can be important
45 information to you.

46 The data needed to offer an “other users found these pages” list can be
47 collected quite easily, but one will need access on the server level, i.e. to

48 collect data from many users. One could strengthen the trail if the user
49 performed some action at the end. This could either be implicit, such as
50 noting that the users stayed on the site for some time, typed in data, printed
51 from the page, bought or booked, etc. Alternatively, it could be explicit,
52 where the users use a “like” button to tell that the page is interesting, e.g.
53 the Google+1 service¹.

54 Such an approach falls into the implicit collaborative information-seeking
55 area in which developing new collaborative search interfaces is still needed,
56 as recently suggested by Hearst [1].

57 According to Golovchinsky et al. [2], a collaborative information search
58 system can be either implicit or explicit, meaning that users can explicitly
59 collaborate on query formulations and review search results or can implicitly
60 take advantage of other users’ search intents. Normally, implicit collaboration
61 systems provide a recommendation and filter the results already explored by
62 previous users, making them available to others with similar information
63 needs.

64 The majority of studies in the implicit area are based on collaborative
65 querying techniques that upgrade information systems with data on past
66 query preferences related to other users. As recently demonstrated [3], such
67 studies primarily tested implicit collaborative information-seeking systems
68 using simulated query formulation instead of employing user analysis involv-
69 ing human participants. In our research, we employed a classic approach by
70 using two existing datasets to simulate queries to evaluate our system in a

¹<https://developers.google.com/+web/+1button/>

71 real setting.

72 Hence, we deal with the problem of improving search engines' perfor-
73 mance by exploiting the actions performed by users. In fact, search engines
74 are tools designed to help people solve their own informational needs and sig-
75 nificant room exists for improvements. Queries submitted to search engines
76 can be clustered into three main categories on the basis of users' aim [4]:

77 **Informational queries** are issued by users willing to acquire information
78 that they assume is present on one or more Web pages;

79 **Navigational queries** are being used to get to a particular Web page be-
80 longing to an organization or an individual; and,

81 **Transactional queries** are issued to perform activities using the Web, such
82 as booking a trip or downloading a file.

83 Ranking results produced through navigational queries can be effectively
84 addressed using existing Link Analysis Ranking (LAR) algorithms, such as
85 PageRank, Hyperlink-Induced Topic Search (HITS), or Stochastic Approach
86 for Link-Structure Analysis (SALSA): a higher number of hyperlinks point-
87 ing toward one particular page results in a higher page relevance (in other
88 words, algorithms assume that this page is the one that the user was look-
89 ing for when she issued the query). Ranking results of informational and
90 transactional queries is another matter: given the high frequency of Web
91 pages' updates and the ever-increasing need to obtain answers in real time,
92 the World Wide Web hyperlinks' configuration is no longer the only effective
93 relevance measure that users assign to Web pages. Thus, devising new rele-
94 vance indicators — to be placed alongside the existing ones (in other words,

95 those based on Link Analysis Ranking) — with the goal of further improv-
96 ing the ranking by considering other relevance measures valued by users is
97 necessary [5].

98 In this paper, we propose to employ the concepts of Swarm Intelligence
99 (SI) in relation to the Ant Colony Optimization (ACO) meta-heuristic to
100 improve search engine performance and to reduce the information overload²
101 by exploiting collective users' behavior in their usage of search engines.

102 **2. Related Work**

103 *2.1. Search Engines*

104 Different studies pointed out users' low degree of satisfaction with search
105 engines. Fox et al. [6] devised a machine learning approach that employs
106 users' actions (for example the time spent on a page, scrolling usage, and
107 page visits) and concluded that users consider 28% of search sessions unsat-
108 isfactory and 30% only partially satisfactory. Xu and Mease [7] measured
109 the average duration of a search session and found that users typically quit
110 a session — even without having satisfied their informational need — after
111 three minutes.

112 The main purpose of search engines is to satisfy users' informational
113 needs, thus they are being used as a starting point of users' Web browsing
114 [8, 4, 9]; nevertheless, the search experience is far from perfect. In fact, a sub-
115 stantial number of searches end up unsatisfied. Many researchers attempted
116 to improve search engines results' relevance by exploiting query-click logs (in

²The inability to make a decision because of the huge quantity of information obtained by the users.

117 other words, logs of all the interactions users carry out with the search en-
118 gine), usually in the form of click-through data. Joachims [10] was the first
119 to exploit these logs as implicit relevance judgments about search engine re-
120 sults and trained a meta-search engine to outperform many other famous
121 ones. After the work of Joachims, using query-click logs to improve search
122 engine performance became a widely popular technique [11].

123 *2.2. Information Foraging on the Web*

124 Many theories attempted to explain users' behavior when searching for
125 information in complex systems (for example, the Web). For the scope of this
126 paper, we refer to two approaches related to the proposed algorithms: the
127 ScentTrails system [12], which continuously allows users to supply keywords
128 and enriches hyperlinks to provide a path that achieves the goal described by
129 them, and the method by Wu and Aberer [13], which operates within a single
130 Website to enrich the information provided by hyperlinks with a technique
131 inspired by ant-foraging behavior (in other words, heavily clicked links are
132 recommended in favor of less visited links).

133 *2.3. Learning to Rank*

134 Learning to Rank aims at automatically learning the right ranking func-
135 tion from a training set, typically a click-through dataset. Joachims [10]
136 outlined three major key points: (1) explicit feedback provided by users can-
137 not be taken for granted and, as a matter of fact, is unnecessary because
138 the information given by the query-click logs are enough; (2) in fact, they
139 can be used as a measure of relevance (with a relative scale); and (3) a ma-
140 chine learning method — Joachims used a Support Vector Machine (SVM)

141 — can be used to obtain a new ranking function that improves search engine
142 performance.

143 In addition to SVMs, other machine learning algorithms may be used,
144 such as RankBoost [14], RankNet [15], QBRank [16], GBRank [17], AdaRank
145 [18], and MCRank [19].

146 However, Learning to Rank approaches exhibit two drawbacks: (1) the
147 training phase is computationally expensive and faster methods are being
148 sought [20]; (2) because most of the outlined machine learning methods are
149 offline, the system must be trained again each time new data become avail-
150 able, which occurs quite frequently because we are dealing with click-through
151 data; thus, online methods are also being investigated [21, 22, 23].

152 In conclusion, it is important to point out that Learning to Rank tech-
153 niques are not the only ones employed in training ranking functions: other
154 studies described alternative soft computing methods, such as genetic pro-
155 gramming [24] and Swarm Intelligence (SI) [25].

156 **3. Ant Colony Ranking**

157 In the introduction, we described how information overload is a major
158 problem affecting Internet users and outlined some useful approaches to ad-
159 dress this issue: software agents, implicit feedback, collaborative filtering,
160 and assisted browsing/searching [26].

161 Given the current wide usage of Web search engines, we focused on all the
162 unsatisfactory searches with the goal of addressing the information overload
163 problem. Some techniques aiming at improving their performance have been
164 summarized (for example, Learning to Rank) to highlight a few key concepts:

165 (1) the relationship between users seeking information and the optimal forag-
166 ing theory; (2) the need for a search engine to adapt itself to users' behavior;
167 and (3) the need to perform such adaptation in real time.

168 Almost none of the aforementioned approaches take into account all three
169 of these aspects, as stated by Wu and Aberer [13] and Olston and Chi [12].
170 Beyond a doubt, a Swarm-based approach can take into account all three key
171 factors and is, nonetheless, a much more elegant and simple method than all
172 of the other "ad-hoc" alternatives.

173 For these reasons, in the next section we introduce a model able to de-
174 scribe ranking algorithms inspired by Swarm Intelligence (SI) that can im-
175 prove the performance of a search engine by adapting themselves to users'
176 behavior.

177 **4. A Model for Ant Colony Ranking Algorithms**

178 Each day ants leave the colony in search of food and building materials;
179 they will exploit the surroundings in all directions in a somewhat random
180 fashion. If an ant finds anything of interest, it will return to the colony
181 depositing pheromone, a chemical substance that the other ants are able
182 to detect. Thus they create trails to signal the path between the colony
183 and the food. The quantity of pheromone deposited, which may depend
184 on the quantity and quality of the food, will guide other ants to the food
185 source. That is, the other ants in the colony may now use the pheromone
186 as a trail marker to reach the food. This marker evaporates over time, so
187 that uninteresting trails disappear. Shorter trails will get a higher level
188 of pheromone, thus shorter trails will endure longer, providing a notion of

189 optimization.

190 Normally, ants from different colonies exhibit aggression toward each
191 other. However, some ants exhibit the phenomenon called unicoloniality.
192 Here worker ants freely mix between different colonies. These species of ants
193 live in populations known as supercolonies that may be used to characterize
194 social behavior on the Web.

195 Let us assume that a set of users all start with the same query, for example
196 “compact camera GPS”. That is, they are all interested in finding Web sites
197 that can offer a good bargain for such a camera (“food”). Our group may
198 start with a Google query, and click on links to explore the results. These
199 click streams will define our “pheromone” or virtual trail. They may for
200 example be implemented by adding score values to each link, or visualized
201 by representing the links by large fonts, stronger color, etc.

202 However, on the Web we can optimize, leaving the trail metaphor and
203 lead subsequent users directly to interesting pages. That is, we let our ants
204 (users) explore the Web, but we let them deposit the pheromone on the
205 most interesting pages. The rest of the colony (i.e. other users with similar
206 interest) can then go directly to these sites.

207 Swarm Intelligence (SI) refers to the emergence of “intelligent” behavior
208 from a group of simple and/or loosely organized agents. Ants are a typi-
209 cal example of SI and their use of stigmergic processes³ inspired the famous
210 family of Ant Colony Optimization (ACO) algorithms [27, 28] and many

³Pierre-Paul Grasse introduced the term in 1950s during his research on termites. It is defined as a method of communication based on individuals modifying their surrounding environment.

211 variants, including Max-Min Ant System (MMAS) [29], Continuous Orghog-
212 onal Ant Colony (COAC) [30], and Rank-based Ant System (ASrank) [31].
213 These classes of algorithms are bio-inspired (Ant Colony) probabilistic meta-
214 heuristics for solving computational problems related to searching for an
215 optimal path in a graph; the probabilistic nature of such techniques — along
216 with some basic rules driving agents towards appropriate solutions — allows
217 for their convergence to an optimal solution, avoiding local optimums.

218 As we have previously stated, we will adapt the strategies employed in
219 food searching by ant colonies in the building of ranking algorithms em-
220 ploying users' behavior; without a doubt, humans are more intelligent and
221 organized than ants. However, some complex phenomena stems from Web
222 surfing, since collective activities like Wikipedia, del.icio.us, or even the entire
223 Web, are indeed stigmergic processes [32, 33].

224 Summarizing, users surfing the Web issue relevance judgments every time
225 they submit a query and select a result among the ones provided by a search
226 engine. Ultimately, a SI-based approach seems a valid idea to make such
227 systems able to exploit users' seeking behavior.

228 It's pretty intuitive to find a parallelism between the way ants forage for
229 food and the way users employ search engines to satisfy their informational
230 needs; yet the latter, unlike ants, don't leave any trace at all, so they can't
231 provide any clues to the next users with their same informational needs, and
232 — since about 30–40% of queries issued to a search engine are already been
233 submitted [34] — that's a pretty common scenario.

234 So, by using a virtual form of pheromone — controlled in the same way as
235 the one used by ants — it's possible to define a ranking algorithm that ranks

236 relevant results based on the pheromone left on each document: the more
237 we'll find on a document the higher its ranking will be, since that document
238 was considered relevant by a large amount of users.

239 4.1. Formalization

240 Here we introduce a brief formalization of the model we just proposed.
241 We will assume that interactions between users and the search engine are
242 available in the form of query-sessions — briefly “sessions”: by the definition
243 of Wen and Zhang [35], a session is formed by the query a user submitted
244 to the search engine — i.e. the text describing what he/she is looking for
245 — together with the visited Web pages consequently to his/her request; an
246 example of a query session can be found in table A.4 in the Appendix.

247 Borrowing the notation of [35], let $D(q)$ be the set of Web pages the
248 search engine presents to the user as results for the query q , selected by
249 filtering only the relevant ones for q through any available retrieval strategy
250 [36]. The page set a user clicked on for a query q may be seen as

$$DC(q) = \{d_{q1}, d_{q2}, \dots, d_{qi}\} \subseteq D(q),$$

251 where d_{qi} represents the i -th document the user clicked among the results
252 of the query q (i.e. d_{q1} being the first selected result — if any — d_{q2} the
253 second — if any — and so on); on the other end, we denote d_q^i the document
254 currently ranked in position i among the results of the query q by the ranking
255 algorithm in use.

256 The pheromone associated to a document d with respect to a query q is
257 denoted by φ_{dq} and is updated every time a user selects the document among

258 $D(q)$ or — carrying on the similarity with ACO — he/she covers the path
259 $q \rightarrow d$; the amount of pheromone deposited each time depends on the specific
260 user session $DC(q)$ and will be detailed in the next section.

261 Considering each document d and any known query q , pheromone evapo-
262 ration follows an exponential decay based both on the current value φ_{dq} and
263 the elapsed time since its last update — denoted with τ_{dq} . In mathematical
264 terms, denoting the new pheromone value by φ'_{dq} , the evaporation rule can
265 be expressed by the equation

$$\varphi'_{dq} = \varphi_{dq} e^{-\lambda \tau_{dq}},$$

266 where λ is the exponential decay constant; this rule can be transformed in a
267 much simpler version by defining a new constant

$$\delta = \frac{\ln(2)}{\lambda},$$

268 which represents the amount of time needed for the pheromone deposited on
269 each document to half its value since its last update for any given query.

270 The evaporation rule becomes then

$$\varphi'_{dq} = \varphi_{dq} 2^{-\frac{\tau_{dq}}{\delta}}.$$

271 Pheromone evaporation is performed periodically and its frequency de-
272 pends on how the relevance of documents changes over time: since evapora-
273 tion is a mechanism that enables the system to forget registered behaviors,
274 the more frequent it gets triggered the more newly registered behaviors will
275 be considered important. To the best of our knowledge, the only similar
276 approach is the one by Koychev and Schwab [37].

277 Finally, the set of documents $D(q)$ — i.e. the results for any query q —
278 are ranked by exploiting the amount of pheromone φ_{dq} , for each $d \in D(q)$.

279 The Ant Colony Ranking strategy can be viewed as an interplay of the
280 three procedures just described, as summarized by Algorithm 1 [38]: the
281 ranking computed using the pheromone deposited over each document $d \in$
282 $D(q)$ is prompted to the user issuing the query q (`ShowAntColonyRanking()`),
283 user’s clicks get processed and partake in the existing pheromone’s configu-
284 ration (`ManageUserActivity()`), and finally the pheromone evaporation is
triggered (`EvaporatePheromone()`).

Algorithm 1 The Ant Colony Ranking strategy in pseudo-code.

```
procedure ANTCOLONYRANKING
  scheduledactivities
    ShowAntColonyRanking()
    ManageUserActivity()
    EvaporatePheromone()
  end scheduledactivities
end procedure
```

285
286 To summarize, we can now define different Ant Colony Ranking algo-
287 rithms by specifying (1) how the amount of pheromone φ_{dq} is updated every
288 time a user selects the document $d \in D(q)$ for any query q (i.e. `ManageUserActivity()`),
289 (2) an evaporation strategy (i.e. `EvaporatePheromone()`), and (3) how it
290 exploits the amount of pheromone φ_{dq} to retain the position of a document

291 $d \in D(q)$ in the final ranking presented to the user (i.e. `ShowAntColonyRanking()`).

292 **5. Three Ant Colony Ranking Algorithms**

293 We focus on unsatisfactory search sessions (approximately 50% of all
294 search sessions, according to [39]) and attempt to improve the entire search
295 users' experience. To do that, we proposed a framework for the definition
296 of ranking algorithms that exploit users' interactions with search engines on
297 the basis of the Ant Colony Optimization (ACO) meta-heuristic by defin-
298 ing a pheromone's update rule, how it evaporates over time and the ranking
299 strategy for the set of pages $D(q)$ presented as results for each query q .

300 In this section, we present three algorithms. The first algorithm is a
301 simple application of the ACO principles to Web pages' ranking, the second
302 algorithm attempts to reinstate the probabilistic nature typical of the ACO
303 meta-heuristic, and the third algorithm is our attempt to leverage on the
304 complete users' search sessions and not just on their single interactions with
305 the search engine.

306 *NaïveRank.* The first algorithm is the simplest and most direct implemen-
307 tation of the principles described so far, and is inspired by the stochastic
308 ranking algorithm by Gayo-Avello and Brenes [40]. We employ the simplest
309 incrementing function, namely the successor; thus, given any user search ses-
310 sion $DC(q) = \{d_{q1}, d_{q2}, \dots, d_{qi}\}$, the pheromone deposited on each document
311 $d_{qi} \in DC(q)$ will be updated with the rule

$$\varphi'_{d_{qi}} = \varphi_{d_{qi}} + 1,$$

312 where $\varphi'_{d_q^i}$ indicated the new value after the update. $D(q)$ is ranked decre-
 313 mentally based on the amount of pheromone deposited on each document
 314 $d \in D(q)$, thus for any given query q and two documents $d_q^i, d_q^j \in D(q)$ with
 315 $i < j$ (recall that d_q^i stands for the document ranked in position i among the
 316 results of the query q) we have

$$\varphi_{d_q^i} \geq \varphi_{d_q^j}.$$

317 Despite the resemblance to the algorithm described in [41], NaïveRank runs
 318 in real time and, more importantly, naturally takes into account the shifts in
 319 users' interests by using the evaporation process.

320 *RandomRank.* The second algorithm uses an alternative approach taken from
 321 the basic principles of ACO [28], through which we considered search engines'
 322 users as agents of a Swarm Intelligence (SI) system. Given our previous
 323 algorithm, the probabilistic nature of the original model — which represents
 324 one of the ACO strengths — fades out: this phenomenon causes neglecting
 325 new paths' discovery once the system reaches convergence. Going back to
 326 Web searching, a gradual empowering of the most popular pages' pheromone
 327 occurs at the expense of the less popular ones. This effect, known as self-
 328 reinforcement, is typical in many techniques of Web ranking [42].

329 Therefore, we want to encourage the discovery of new pages — in other
 330 words, new paths to be explored — by reinstating the probabilistic nature of
 331 ACO algorithms into the ranking mechanism, and keeping the update rule
 332 the same employed by NaïveRank. Thus, we randomly rank each result in
 333 $D(q)$ for any query q with the probabilistic procedure described in Algorithm
 334 2, using a probability distribution based on the quantity of pheromone of each

335 one of them. This way, highly visited pages yield a higher ranking — through
 336 a higher probability of selecting one of them in one of the first positions —
 337 but less relevant documents still have the opportunity of becoming popular
 338 (thanks to the probabilistic nature of the algorithm).

339 Each cycle in the loop is responsible for the ranking of a document in the
 340 set of results: it builds the set $\bar{D}(q)$ of pages that still need to be ranked
 341 and randomly picks a document based on its pheromone configuration. This
 342 random selection is performed every time a user issues a query q , yielding in
 343 a renewed opportunity of discovering new and relevant documents in $D(q)$
 344 and let them gain positions in the ranking.

Algorithm 2 Procedure ranking $D(q)$ used by RandomRank based on [43].

procedure SHOWANTCOLONYRANKING

for $i \leftarrow 1, \#D(q)$ **do** ▷ Rank one result at a time

$\bar{D}(q) \equiv \{d : d \in D(q) \wedge d \neq d_q^j, 1 \leq j < i\}$ ▷ Not yet ranked

 Select $d \in \bar{D}_q$ with probability $\frac{\varphi_{dq}}{\sum_{\bar{d} \in \bar{D}(q)} \varphi_{\bar{d}q}}$ and rank it in position
 i

end for

end procedure

345 *SessionRank*. The last algorithm employs yet another mechanism of the ACO
 346 approach. Indeed, hitherto the increment function has always used a fixed
 347 amount of pheromones regardless of the click's position among the user's

348 search session. Within the ACO meta-heuristic, in order to achieve conver-
349 gence quicker, the pheromone's quantity is set to decline on the basis of the
350 quality of the solution found.

351 On the Web though, users cannot provide a solution to the ranking prob-
352 lem but can assist by providing their own view of relevance. In fact, the first
353 document that a user selects among the results in a given search session is
354 the one that, based on the available clues, is perceived as the most relevant
355 to the user [44]. The most relevant document according to a user is the
356 one that should be in a highly relevant position in the optimal solution to
357 the ranking problem for that given query. The next document, selected in
358 the same session, is considered less relevant since it was selected after the
359 previous document.

360 The SessionRank algorithm employs the relative order of clicks performed
361 by each user during a session and increments the pheromone's quantity ac-
362 cordingly. Therefore, choosing an exponential decay, the update rule be-
363 comes:

$$\varphi'_{dq} = \varphi_{dq} + 2^i,$$

364 where $d \equiv d_{qi}$ and $d_{qi} \in DC(q)$ for any user search session $DC(q)$ yielding
365 $D(q)$.

366 To summarize, the model proposed in the previous section to define rank-
367 ing algorithms on the basis of ACO employs pheromone traces on each doc-
368 ument in relation to any query issued to the search engine; the pheromone
369 increases each time a user selects a page among the results of a query and
370 vaporizes over time taking into account users' gradual loss of interest. There-

371 fore, once a user performs a query already performed by others, the search
372 engine is able to present a new ranking based on the behavior shown by users
373 with the same informational need, de-facto exploiting pheromone traces.

374 We described three different algorithms that, by exploiting the afore-
375 mentioned model, use different ACO-inspired mechanisms to improve the
376 proposed ranking. Thus, since establishing whether improving search engine
377 performance is truly possible by employing this new approach is important,
378 we devised an evaluation of the different algorithms using real query-click
379 logs. In the following section, we present details on measures, setups, and
380 results.

381 **6. Evaluation**

382 *6.1. Search Engine Evaluation*

383 In an ideal situation, an Information Retrieval (IR) system (for example,
384 a search engine) should only provide relevant results for the issued queries.
385 The tendency is to accept that such systems provide the widest set of relevant
386 documents, along with some less relevant results. Normally, evaluating an
387 IR system requires experimental sets containing queries, documents, and
388 relevance judgments; however, building such collections requires a significant
389 amount of work (in other words, data on queries and judgments). Thus, in
390 many recent studies [45, 46, 40, 47, 5], click-through data were employed to
391 evaluate search engines' performance. The concept is simple: employ clicks
392 as relevance judgments, assuming that a user evaluates a result as relevant
393 if it is chosen among the search results related to a query.

394 Consequently, in the following experiments, we employed query-click log

395 datasets provided by two famous search engines — AOL [48] and Yahoo! —
396 to carry out experiments on the proposed algorithms (further details on these
397 datasets can be found in the Appendix); we have validated the new ranking
398 produced by each algorithm using the very same datasets clustered by each
399 user’s search session, applying a simple temporal threshold (30 minutes, as
400 suggested by several previous studies [6, 49]) to decide whether two actions
401 performed by a single user belong to the same search session.

402 Hence, we considered two interactions as belonging to the same search
403 session when they were both (1) issued by the same user, (2) contained the
404 same query, and (3) performed within 30 minutes.

405 After selecting the two datasets, we needed a performance measure to
406 evaluate all the different algorithms we propose.

407 Sakai [50] compared different performance measures that take into ac-
408 count the documents’ positions and recommended the so-called Normalized
409 Discounted Cumulative Gain (NDCG) for its simplicity and robustness [51].

410 The NDCG consists of a parameter and two functions: 1. $k \in \mathbb{N}_0$ is a
411 cutoff parameter defining the number of elements in the results list to be
412 considered; 2. the gain function measures the benefit earned by the user (if
413 a document is only relevant or irrelevant, then the gain is binary), whereas
414 3. the discount function weighs the documents’ relevance on the basis of the
415 position in the results list.

416 Moreover, to obtain an absolute measure — in the real interval $[0,1]$ —
417 we normalized it with respect to the maximum obtainable gain.

418 More formally, let $\vec{y} \in \mathbb{R}^n$ be an array containing relevance values belong-
419 ing to a sequence of n elements (for example, the results of a query) and let

420 $\vec{\pi} \in \mathbb{R}^n$ be a permutation of the same sequence (for example, the ranking
 421 produced by an algorithm). Let $\vec{\pi}(q)$ be the index of the q -th element in
 422 $\vec{\pi}$ and let $\vec{y}_{\vec{\pi}(q)}$ be the value of its relevance. The Discounted Cumulative
 423 Gain (DCG) of the permutation $\vec{\pi}$ is defined as:

$$\text{DCG}@k(\vec{y}, \vec{\pi}) = \sum_{q=1}^k \frac{2^{\vec{y}_{\vec{\pi}(q)}} - 1}{\log_2(2 + q)}.$$

424 In this case, the gain function is a power of 2, whereas the discount function
 425 has logarithmic decay over the permutation length. Thus, the NDCG is
 426 defined as:

$$\text{NDCG}@k(\vec{y}, \vec{\pi}) = \frac{\text{DCG}@k(\vec{y}, \vec{\pi})}{\text{DCG}@k(\vec{y}, \vec{\pi}_{\vec{y}}^*)},$$

427 where $\vec{\pi}_{\vec{y}}^*$ is the permutation corresponding to a perfect ranking w.r.t. the
 428 relevance judgments in \vec{y} or, in other words:

$$\vec{\pi}_{\vec{y}}^* = \underset{\vec{\pi}}{\text{argmax}} \text{DCG}@k(\vec{y}, \vec{\pi}).$$

429 Few publicly available datasets already provide explicit relevance judgements
 430 for each document they refer: the Yahoo! dataset is one of them, but the
 431 AOL one does not; as we state before though, user clicks can be used as a
 432 relative measure of the perceived relevance by each user. In this case, we
 433 then build the array \vec{y} by assigning 1 to each document selected by the user
 434 in the search session we are currently trying to compute the NDCG for, 0 for
 435 the ones that were not selected.

436 To summarise, computing NDCG scores for each search session contained
 437 in the datasets is possible using the available relevance judgments for Yahoo!

438 and by assuming that the visited results are the relevant ones for AOL.
439 Effectively, by doing so we are actually evaluating the current performance
440 of the two search engines, which will be the baseline we are comparing our
441 algorithms against.

442 *6.2. Evaluation of Ant Colony Ranking Algorithms*

443 We described hitherto how search engine performance are evaluated by
444 employing the query-click log containing users' interactions. As we previously
445 stated, the proposed algorithms require queries, clicks, and search sessions to
446 adjust the pheromone deposited on each document in relation to any query
447 submitted to the search engine. The chosen query-click logs contain all this
448 information and, thus, can be employed to simulate our algorithms in a real
449 world scenario.

450 The validation strategy is dictated by the constraints over our context:
451 since we are using these datasets to simulate real users' interactions, we have
452 to obey to time constraints; our test set will then always be consequent to
453 the training set, since the system can only be trained using past interactions,
454 forbidding any kind of cross-validation. According to the most common
455 strategy, we chose then to split our data and use the first two thirds for
456 training and the remaining for testing: moreover, using one third of the
457 available interactions provides a significant amount of data to thoroughly
458 test our algorithms.

459 Therefore, we prepared two partitions for each available query-click log:
460 the AOL partition includes the set of the almost 20 million clicks performed
461 from March to April 2006 and was used for training, whereas the remaining
462 set of approximately 10 million clicks performed in May 2006 was used for

463 the evaluation. The Yahoo! training set contained almost 40 million clicks
464 performed during the first 20 days of July 2010 (excluded), whereas the
465 evaluation set contained the remaining 26 million clicks issued until the end
466 of the same month.

467 After the training phase, we compared the search sessions contained in
468 the test sets with the ranking given by each algorithm and devised a se-
469 quence of potential clicks; this procedure is ruled by a simple and reasonable
470 assumption [40]: during a search session, if a user chooses a result for a given
471 query, we safely assume that the same user in the same search session would
472 have chosen that same result even if it was found in a higher position in the
473 results' list. Finally, we computed the mean of NDCG for each session.

474 Furthermore, because we sought to evaluate how the algorithms param-
475 eters affect performance, we tested three different values of δ and of the
476 evaporation time (one hour, one day, and one week), and three session dura-
477 tion values for the SessionRank training (one, five, and 25 minutes, namely
478 very brief, average — according to [7] — and long-duration search sessions);
479 besides, since RandomRank is a probabilistic algorithm, each experiment
480 were repeated 5 times using different seeds of the random number gener-
481 ator.. Thus, the evaluation involved 162 different experiments. We carried
482 them out using a t2.small Amazon EC2 instance running Amazon Linux AMI
483 and the DEX library to manipulate both datasets [52]; each run took about
484 3 hours to complete.

485 To summarize, our evaluation's aim was to measure and compare the
486 proposed ranking algorithms' performance using data provided by two fa-
487 mous search engines. Because our methods are based on users' interactions

488 to discover the most promising results, the datasets were partitioned into
489 a training set and a test set. In the next sections, we provide the results
490 obtained from using this evaluation method.

491 *6.3. Results*

492 As previously stated, we defined a framework for the evaluation of the
493 three proposed algorithms performance using the interactions from the two
494 different query-click logs. We described (1) the way we selected the training
495 and test sets, (2) how we used the former to simulate real users' behavior, and
496 (3) from the latter we yielded the potential clicks combining new rankings
497 with the original click-through data.

498 In the following, we analyze the results of the performed experiments,
499 reporting NDCG scores for three different cutoff values: 1, 3 and 10 results
500 (NDCG@1, NDCG@3, NDCG@10). We recall that NDCG is a measure in
501 the interval $[0,1]$ where 1 is the ideal ranking. We chose to test our algorithms
502 on three different cutoff values meaning respectively: the result ranked first
503 with NDCG measure representing the ratio between our algorithm result and
504 the first best ranked from the training set; the three highest ranked results
505 and the ten highest results. We can imagine comparing the results within a
506 search engine result page in which only first result is returned, or the first
507 three or the first ten. We report those results for all three proposed algorithms
508 in tables 1 and 2, highlighting the best one obtained by each algorithm in
509 bold. The (δ) factor represents the timeframe set for halving the pheromone
510 (representing evaporation); the wider the timeframe the longer will take to
511 half the pheromone and thus results appearing in the ranking will be more
512 persistent: it, basically, represents the magnitude of the evaporation set to

513 0.5 (delta factor in section 4.1). The results report the NDCG values [0,1]
514 related to the corresponding pheromone upgrading timeframe (upgrades are
515 run hourly, daily and weekly) and to the different NDCG cutoffs. Also, the
516 NDCG values computed respectively on the Yahoo! and AOL datasets by
517 maintaining the default search engine’s ranking represent our benchmark.

518 As expected, NaïveRank performs better on the larger dataset (i.e. Ya-
519 hoo! in table 2); this seems reasonable, since it follows from the Ant Colony
520 Optimization (ACO) approach: the more data is recorded about users’ be-
521 havior the better the algorithm will perform. More surprising are the dif-
522 ferences obtained with the same algorithm using different pheromone evap-
523 oration intervals and by halving (δ) times. Normally, one would expect that
524 depending on the halving delta factor timeframe the algorithm would per-
525 form better in adapting to users’ behavioral changes: in fact, no matter which
526 dataset we used, we obtained the best performance by setting $\delta = 7d$ (a week
527 timeframe); this confirms what implied by Liu et al. [53] about the weekly
528 cycle of the majority of queries which states that users will perceive search
529 engine results as relevant and up to date generally only during one week after
530 which they would expect the results to be updated.

531 Regarding the evaporation time for NaïveRank, we noted an interesting
532 effect: although using non-optimal δ s (as stated above, we found out that
533 seven days was the optimum) doesn’t affect the performance, choosing an
534 optimal δ makes slightly no difference at all.

535 Thus, when implementing the NaïveRank we can safely act on δ to reduce
536 the evaporation frequency, in order to reduce the amount of computations
537 needed. This implies that the algorithm will be more computationally effi-

Cutoff	Evaporation	δ	Algorithm						
			NaiveRank	RandomRank	SessionRank			PageRank	
					t-sess = 1m	t-sess = 5m	t-sess = 25m		
NDCG@1	Hourly	1 hour	0.612796	0.560840	0.587819	0.587811	0.588012		
		1 day	0.665814	0.629042	0.641887	0.642172	0.642872		
		7 days	0.686221	0.634199	0.658954	0.660292	0.661030		
	Daily	1 hour	0.480964	0.397926	0.462242	0.462437	0.462700	0.680486	
		1 day	0.678099	0.648146	0.652467	0.652808	0.654060		
		7 days	0.690504	0.636886	0.663460	0.664649	0.665754		
	Weekly	1 hour	0.438598	0.377019	0.411471	0.411674	0.412434		
		1 day	0.660114	0.612556	0.605444	0.607117	0.608181		
		7 days	0.693982	0.631203	0.665549	0.666693	0.667794		
	NDCG@3	Hourly	1 hour	0.705267	0.683648	0.696969	0.697230	0.697696	
			1 day	0.753037	0.730056	0.741832	0.742368	0.743625	
			7 days	0.768810	0.744888	0.753478	0.754486	0.755795	
Daily		1 hour	0.645189	0.580623	0.632110	0.632354	0.633316	0.770948	
		1 day	0.763790	0.747970	0.750103	0.750927	0.752550		
		7 days	0.772385	0.749670	0.756956	0.758239	0.759788		
Weekly		1 hour	0.647478	0.579306	0.637970	0.638591	0.639621		
		1 day	0.755406	0.735149	0.728631	0.730137	0.732779		
		7 days	0.776140	0.748586	0.758544	0.759937	0.761388		
NDCG@10		Hourly	1 hour	0.748356	0.738564	0.742184	0.742306	0.742555	
			1 day	0.780578	0.765888	0.773331	0.773659	0.774397	
			7 days	0.792010	0.774868	0.783684	0.784458	0.785341	
	Daily	1 hour	0.720219	0.679581	0.710927	0.709515	0.710097	0.798921	
		1 day	0.788710	0.777328	0.780082	0.780651	0.781635		
		7 days	0.793687	0.778011	0.785372	0.786184	0.787441		
	Weekly	1 hour	0.713103	0.671526	0.701405	0.702150	0.702872		
		1 day	0.782900	0.768281	0.765012	0.766283	0.768087		
		7 days	0.795954	0.777030	0.785726	0.787474	0.788685		

Table 1: Experiments results related to the AOL Dataset.

Cutoff	Evaporation	δ	Algorithm						
			NaiveRank	RandomRank	SessionRank			PageRank	
					t-sess = 1m	t-sess = 5m	t-sess = 25m		
NDCG@1	Hourly	1 hour	0.759466	0.770834	0.782178	0.782733	0.783178		
		1 day	0.770370	0.751782	0.790256	0.791962	0.793060		
		7 days	0.771225	0.730059	0.791836	0.793819	0.795348		
	Daily	1 hour	0.438817	0.444115	0.412145	0.413315	0.414003		
		1 day	0.771698	0.747178	0.791750	0.793376	0.795025	0.814651	
		7 days	0.770998	0.729322	0.792389	0.794340	0.796415		
	Weekly	1 hour	0.276672	0.644665	0.250834	0.250987	0.251489		
		1 day	0.721781	0.671024	0.684969	0.688123	0.690983		
		7 days	0.770823	0.727043	0.792595	0.794612	0.796704		
	NDCG@3	Hourly	1 hour	0.876209	0.883413	0.884344	0.884745	0.885044	
			1 day	0.882496	0.882484	0.894532	0.895632	0.896317	
			7 days	0.881654	0.874240	0.897923	0.899525	0.900654	
		Daily	1 hour	0.677897	0.694306	0.662378	0.663326	0.664129	
			1 day	0.882693	0.881387	0.896591	0.897824	0.898901	0.898295
7 days			0.881922	0.875059	0.899163	0.900812	0.902238		
Weekly		1 hour	0.499294	0.830991	0.474018	0.474666	0.475117		
		1 day	0.862717	0.841308	0.839769	0.842197	0.844594		
		7 days	0.881789	0.874471	0.900376	0.902029	0.903527		
NDCG@10		Hourly	1 hour	0.909130	0.917049	0.918082	0.918284	0.918460	
			1 day	0.913118	0.914240	0.923641	0.924251	0.924624	
			7 days	0.912718	0.908590	0.925682	0.926490	0.927077	
		Daily	1 hour	0.793643	0.804557	0.786758	0.787421	0.787798	
			1 day	0.913437	0.913236	0.924918	0.925559	0.926145	0.924707
	7 days		0.912745	0.909546	0.926919	0.927777	0.928532		
	Weekly	1 hour	0.702941	0.881366	0.691655	0.691925	0.692178		
		1 day	0.899319	0.887895	0.889823	0.891161	0.892375		
		7 days	0.912505	0.909222	0.927729	0.928580	0.929364		

Table 2: Experiments results related to the Yahoo! Dataset.

538 cient in real time. Effectively, increasing the evaporation frequency will affect
539 the computational cost of the algorithm since the upgrade denoted by the
540 evaporation rule in section 4 has to be computed less frequently.

541 Consequently, just by observing the first results we can argue that the
542 data size required to get good performances from the algorithm is substantial.
543 While the evaporation time is important to achieve good performance, using
544 a δ set to the weekly cycle of queries allowed us to arbitrarily choose the
545 evaporation time, significantly reducing the computational costs.

546 Considering RandomRank performance, it is interesting to notice how it
547 differentiates from NaïveRank: the variations take place mostly for NDCG@1
548 (i.e. the score related only to the first displayed result), while for NDCG@3
549 and NDCG@10 results are almost identical to NaïveRank. This is due to
550 the probabilistic nature of the random ranking; indeed, probability has on
551 average an higher effect when smaller set of documents are considered due to
552 probability of selecting more than one element of the set for NDCG@3 and
553 NDCG@10.

554 A slightly more interesting result comes from analyzing how such vari-
555 ations actually occur: when tested against the AOL dataset — the smaller
556 one — RandomRank underperforms NaïveRank by about 10%. For Yahoo!
557 Dataset, performances are almost the same for both the algorithms. The
558 exception is the configuration; with $\delta = 1h$ (i.e. the halving factor) and a
559 weekly evaporation cycle. Then RandomRank doubles the score obtained
560 with the same configuration by NaïveRank. This could confirm once more
561 what we stated previously when discussing NaïveRank’s results about the
562 weekly cycle of search queries, as introduced by Liu et al. [53]: the introduc-

563 tion of probability helps users discovering new interesting documents among
564 results. In this particular case the pheromone updates on weekly basis ac-
565 cording to users' perception of documents relevance.

566 Finally, the results related to the SessionRank algorithm show the differ-
567 ent timeframes used by the algorithm to identify a user's search session in
568 its training phase. We recall that based on the session's duration, the algo-
569 rithm distributes differently the quantity of pheromone to be deposited on a
570 document once; by performing multiple experiments, each time with a differ-
571 ent session's duration, the algorithm will consider a sequence of interactions
572 performed by the same user as a single session if they were all done within
573 the allowed timeframe, otherwise it will split them into multiple sessions.
574 We chose to test three configurations for the sessions duration used by the
575 algorithm (as reported in different colors in the figures) namely 1 minute, 5
576 minutes and 25 minutes. We selected these three configurations in order to
577 evaluate the different performance of our algorithms with very brief search
578 sessions, i.e. 1 minute which is shorter than the average search session — 3
579 minutes according to [7]; we also tested our algorithms with 5 minutes ses-
580 sions that can be considered as of average duration and finally we chose to
581 also include 25 minutes sessions to sample longer durations.

582 One can notice straightaway that the training sessions duration is mostly
583 irrelevant; this may be caused by the average short length of search sessions,
584 as demonstrated in [54], since users tend to perform brief sessions, performing
585 different queries; thus search sessions will have only few clicks performed in
586 a short time span and the amount of pheromone to be deposited by the
587 algorithm will be rather fixed, causing no effect.

588 Also, the algorithm performs worse than NaïveRank using the first dataset,
589 while outperforming it with the second one; this is still due to the variation in
590 training set size: a greater number of search sessions used in training causes
591 an improvement due to the greater quantity of pheromone available to be
592 deposited by each user; thus, the pheromone’s modulation — inspired by the
593 ACO metaheuristic — improves the algorithm performance proportionally
594 to the number of interactions recorded during the training phase.

595 Figure 1 recaps our results: the two plots represent only the *best* perfor-
596 mance obtained by the different configurations of our algorithms — showed
597 on the x-axis — with the two datasets used; on the y-axis we show the dif-
598 ferent cutoff points used to compute the related NDCG measure, and the
599 size of the points show the actual NDCG value we obtained, bigger for large
600 values. Finally, the green color is used to show a NDCG measure which im-
601 proves over the baseline ranking algorithm applied by the search engine, red
602 if otherwise.

603 Summarizing these findings, we argue that our first two proposed algo-
604 rithms could be employed in improving results ranking produced by two
605 particular sets of queries: being a simple application of the ACO technique
606 to Web pages ranking, NaïveRank works well for embedding a plain concept
607 of popularity into the ranking measure. Thus it could be very effective in
608 ranking results related to transactional or informative queries whose results
609 do not become obsolete frequently (i.e. it is rare to see a new document con-
610 taining updated information suddenly appear, making the already popular
611 ones out-of-date, e.g. encyclopedia definitions or catalog’s products).

612 By reinstating the probabilistic nature typical of the ACO metaheuristic,

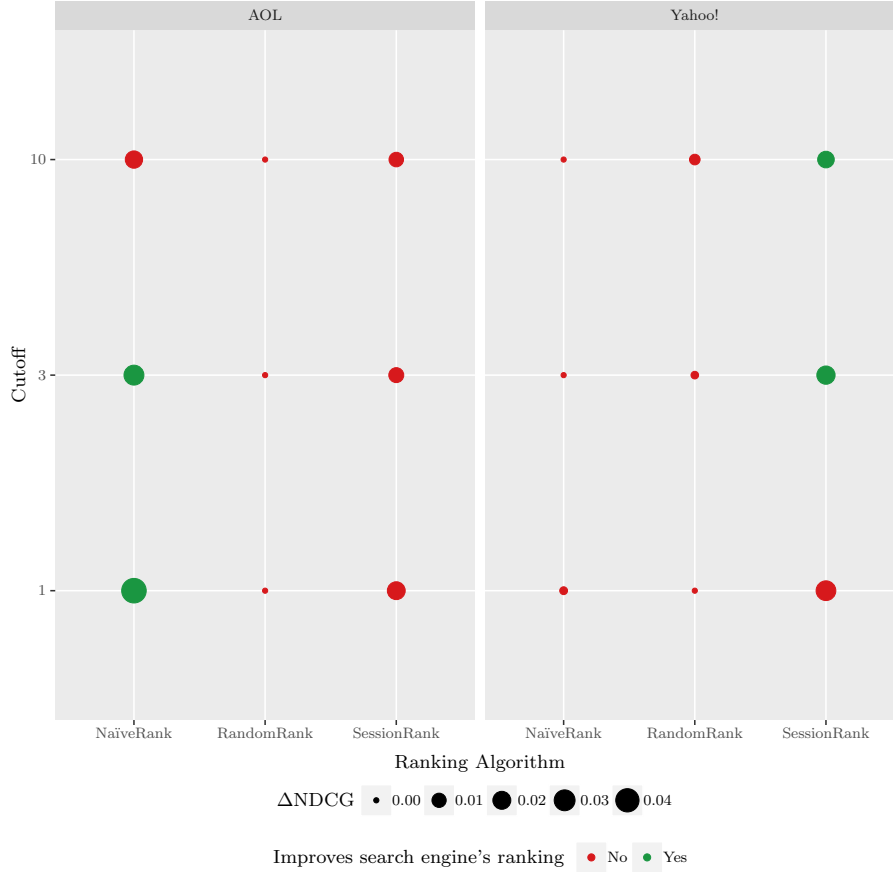


Figure 1: Summary of the three algorithms' best performance: the x-axis shows the algorithm performance related to each dataset we used (on the top x-axis), while the y-axis shows the cutoff point for the corresponding NDCG measure; the size of the points represents the related NDCG's goodness, while the colors indicates whether we achieved an improvements over the default ranking operated by the search engines.

613 RandomRank allows new and bleeding-edge documents to be discovered by
 614 users, thus it could be very effective in ranking results related to breaking
 615 news and current events.

616 Finally, SessionRank shifts on a whole new dimension in terms of the
617 kind of information it exploits and — thus — the search settings that could
618 benefit from its introduction in the ranking mechanism. Albeit the majority
619 of search sessions are brief, composed by just one query and focused only
620 on the first results page, there are some particular sessions that might be
621 longer and could be very frustrating for users. We noticed three types of
622 such problematic sessions in our datasets:

623 **atypical Web search sessions** [55] are being produced by users with atyp-
624 ical information needs, i.e. those outside their regular areas of expertise
625 (often triggered by external events, such as pending medical treatments,
626 financial deadlines or upcoming vacations);

627 **exploring sessions** [56] are those where users are engaged in an open-ended
628 and multi-faceted information-seeking task to foster learning and dis-
629 covery;

630 **struggling sessions** [56] are those where users are experiencing difficulty
631 locating the required information.

632 Examples of both exploring and struggling sessions can be found in figure
633 2.

634 Given that SessionRank exploits not just single interactions with the
635 search engine but whole search sessions, it seems reasonable to argue that
636 the ranking of results related to the aforementioned search sessions could
637 benefit from the introduction of our last ACO-inspired algorithm. Thus it
638 could be useful to test our algorithm against a different dataset containing
639 long-lasting search sessions of this type. Alternatively, one could use some

Query can you use h & r block software for more than one year
Query how do I file 2012 taxes on hr block
Click <http://www.hrblock.com>
Query can you only use h & r block one year
Click http://www.www.consumeaffairs.com/finance/hr_block_free.html
Click <http://financialsoft.about.com/od/taxcut/gr/HR-Block-At-Home-...>
Query do I have to buy new tax software every year
Click http://financialsoft.about.com/od/simpletips/f/upgrade_yearly.htm...
Click <http://askville.amazon.com/buy-version-Tax-Software-year/Answer...>
END OF SESSION

(a) A *struggling* session.

Query career development advice
Click <http://www.sooperarticles.com/business-articles/career-devel...>
Query employment issues articles
Click <http://jobseekeradvice.com/category/employment-issues/...>
Query professional career advice
Click <http://ezinearticles.com/?Career-Advice-and-Professional-Ment...>
Click <http://askville.amazon.com/buy-version-Tax-Software-year/Answer...>
Query what is a resume
Click <http://en.wikipedia.org/wiki/R%C3%A9sum%C3%A9...>
END OF SESSION

(b) An *exploring* session.

Figure 2: Examples of struggling and exploring sessions taken from [56].

640 query-similarity measure with the same datasets we employed, in order to
 641 cluster similar queries belonging to the same session.

642 Our findings are summarized in figure 3: it shows both dimensions that
 643 our algorithms operate on: search sessions' length on the x-axis and docu-
 644 ments update frequency on the y-axis. The horizontal stripes represent the
 645 aforementioned examples of documents sets to be ranked, such as break-
 646 ing news, encyclopedia definitions and catalog's products, while the vertical
 647 ones are the three kinds of search sessions outlined in the previous paragraph
 648 [55, 56]. The dashed blocks indicate which algorithm we think could be the
 649 most effective in ranking results generated for each case. For example — as
 650 we stated previously — RandomRank could be beneficial in ranking results

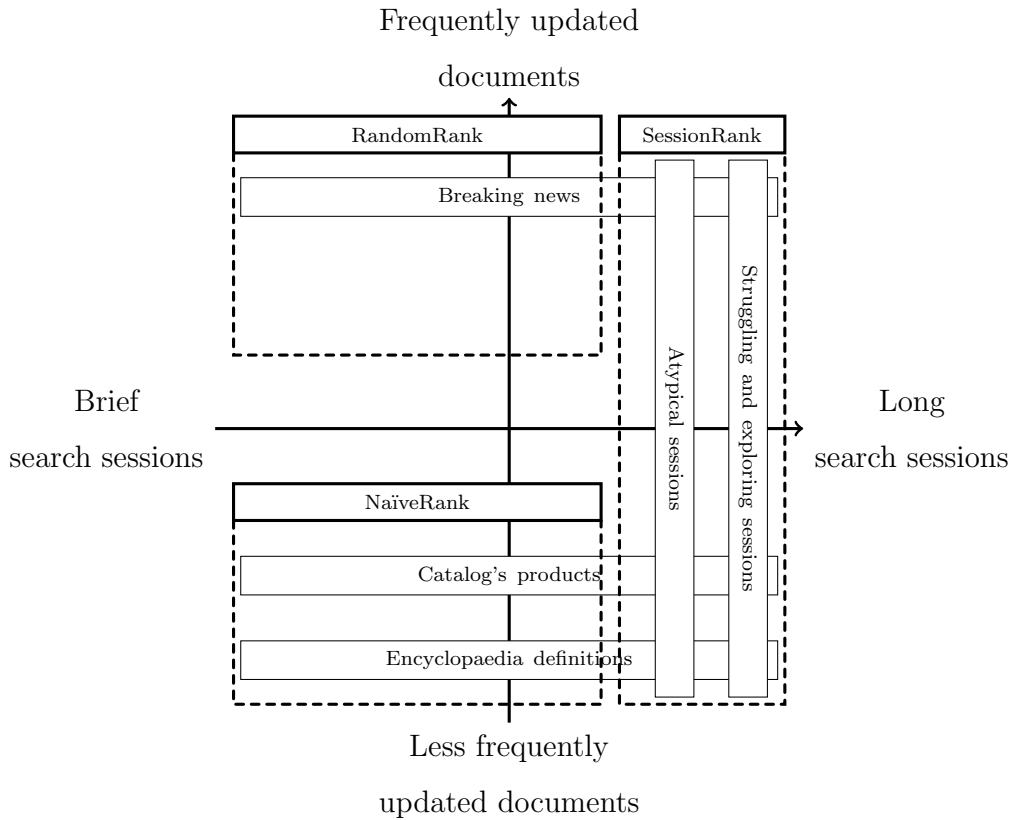


Figure 3: A plot of our findings: on the x-axis we have the search sessions' length, while on y-axis the documents update frequency; vertical and horizontal stripes represent examples of both those settings, while colored blocks indicate the most suitable algorithm to rank results generated by the revealed search setting.

651 among frequently updated documents and does not really take into account
 652 any information about the whole search sessions (focusing only on single in-
 653 teractions), thus it won't work for longer sessions, such as atypical, struggling
 654 or exploring ones for which SessionRank could be more suitable. Ranking
 655 more static collections of documents, such as products inside a catalog for

656 example (e.g. Amazon or Google Shopping) or encyclopedia entries doesn't
657 require a very refined collaborative filtering mechanism — due to the low
658 frequency of updates — thus NaïveRank could fit well with these situations.
659 Indeed, catalogs can be dynamic but vary less frequently than news.

660 **7. Conclusion and Future Work**

661 We presented an approach to developing real time implicit collaborative
662 information-seeking algorithms. Providing implicit collaboration is becom-
663 ing increasingly relevant in search engine research and application areas. Re-
664 cently, Google introduced their Social Search service, declaring that, “with
665 these changes, we want to help you find the most relevant information from
666 the people who matter to you”. In a way, that statement represents our
667 definition of a colony. The mechanism is the Google+1 button, which al-
668 lows users to share interesting pages with their contacts — a way to release
669 pheromones. Bing, Microsoft's new search engine, employs Facebook's social
670 graph for each user to rank search results and to present search history. That
671 is, they define the colony as our own Facebook contacts. Again, we deposit
672 pheromones through a click on the “like” button. This method is viewed as
673 a way to implement pheromone evaporation. However, these stylish inter-
674 actions can be modelled by ants. As ants, we leave “pheromones” to allow
675 others to follow our trails. Additionally, as ants, we use this information to
676 enhance searching.

677 We designed three different algorithms employing an Ant Colony Opti-
678 mization (ACO) strategy to provide implicit collaborative-seeking features
679 in real time to search engines. The three different algorithms — NaïveRank,

680 RandomRank, and SessionRank — all proved to be effective in real time
681 depending on the nature of the queries submitted by users. Real time per-
682 formance is crucial for search engines, particularly when using ACO-inspired
683 algorithms for which a large graph of queries and documents might be cre-
684 ated.

685 The NaïveRank seems particularly interesting for informational queries
686 that seek to retrieve results on relatively static information on the Web,
687 such as looking for products in a catalog or encyclopedia entries. Random-
688 Rank proved effective for the inverse situation, such as breaking news or
689 a sports event. The SessionRank algorithm was suited for struggling and
690 explorative sessions (in other words, open-ended information-seeking tasks
691 fostering users' learning) or atypical query sessions (generated by external
692 events such as specific treatments, deadlines, or upcoming holidays).

693 We evaluated the three algorithms by designing an evaluation, where we
694 compared the performance of the three proposed ranking algorithms with
695 the data provided by two famous search engines: Yahoo! and AOL. Because
696 our methods are based on users' interactions to discover the most promising
697 results, the datasets were partitioned into a training set and a test set.

698 We plan to run an online experiment with a wide sample of participants
699 and test the three algorithms in a real time scenario with users in the future.
700 We hope to prove that in an online environment, real time relevant results
701 can also be obtained by users employing an implicit collaborative approach
702 for information seeking and by selecting the right algorithm depending on
703 the types of queries. We also plan on further investigate how blend some
704 concepts explored by existing ACO extensions in our model, such as limiting

705 the amount of deposited pheromone to avoid deposited as in Max-Min Ant
706 System (MMAS) [29].

707 **Acknowledgment**

708 This work has been partially supported by ANTASTIC — a bio-inspired
709 approach to social search interactions. YGGDRASIL: the Research Council
710 of Norway Grants for highly qualified, younger researchers (2013). Project n.
711 220050/F11. We would like to acknowledge the Research Visibility Award
712 funded by Brunel University London that allowed the main author to es-
713 tablish a research network on collaborative information seeking with the co-
714 authors of this work.

715 **Appendix A. AOL query-click log**

716 This archive, released in August 2006, contains more than 30 million
717 clicks issued by 650000 users, recorded in a timespan going from March to
718 May 2006; each record (table A.3, from left to right) is made up by the user
719 ID, the query, the timestamp, the document’s position among the results,
720 and the URL. The position 0 illustrates that the user issued the query but
721 didn’t click on any result at all.

722 We didn’t employ the whole log in our evaluation, instead we only consid-
723 ered the subset of queries issued on average once a day during the observed
724 period (i.e. March to May 2006): this process allowed us to only employ
725 “significant” interactions with the search engine, ignoring the ones issued
726 less frequently without biasing our later evaluation. By doing so, we ob-
727 tained about 5 million different clicks related to 22000 different queries.

285103	ants	2006-04-01	19:45:23	1	http://www.dna.affrc.go.jp
285103	ants	2006-04-01	19:45:23	3	http://www.uky.edu
285103	ants	2006-04-01	19:50:59	13	http://ohioline.osu.edu
285103	ants	2006-04-01	19:50:59	14	http://ohioline.osu.edu
285103	ants	2006-04-11	21:44:45	7	http://ohioline.osu.edu
889138	ants	2006-03-05	13:22:31	4	http://www.ants.com
889138	ants	2006-03-05	13:22:31	8	http://ohioline.osu.edu
889138	ants	2006-03-05	13:26:14	11	http://www.infowest.com
889138	ants	2006-03-05	13:26:14	19	http://www.greensmiths.com
3519280	ants	2006-03-30	17:14:14	0	
3519280	ants	2006-03-30	17:15:53	1	http://ant.edb.miyakyo-u.ac.jp
3519280	ants	2006-03-30	17:15:53	3	http://www.uky.edu
3519280	ants	2006-03-30	17:15:53	10	http://en.wikipedia.org
3519280	ants	2006-03-30	17:27:46	0	
3519280	ants	2006-04-01	13:55:03	2	http://www.lingolex.com
3519280	ants	2006-04-01	13:55:03	3	http://www.uky.edu
3519280	ants	2006-04-01	14:20:53	0	

Table A.3: AOL Query-click log fragment for the query 'ants'. Horizontal lines separate users by "user ID".

728 After selecting the database to be used in our experiments, we detected
729 the actions' sequence (i.e. clicks) performed by each user during each search
730 session. Therefore, we applied a simple temporal threshold: if two actions
731 were performed within a 30 minutes' timespan then they would belong to
732 the same session.

733 In tables A.3 and A.4 we can observe the sessions' detection process in
734 the original query-click log.

735 After a controversial discussion about the users' privacy following the

736 initial public release, AOL chose to remove the log from its servers and doesn't
 737 offer the download anymore, although it's still available to researchers.

285103	ants	2006-04-01	19:45:23	1	http://www.dna.affrc.go.jp
285103	ants	2006-04-01	19:45:23	3	http://www.uky.edu
285103	ants	2006-04-01	19:50:59	13	http://ohioline.osu.edu
285103	ants	2006-04-01	19:50:59	14	http://ohioline.osu.edu
285103	ants	2006-04-11	21:44:45	7	http://ohioline.osu.edu
889138	ants	2006-03-05	13:22:31	4	http://www.ants.com
889138	ants	2006-03-05	13:22:31	8	http://ohioline.osu.edu
889138	ants	2006-03-05	13:26:14	11	http://www.infowest.com
889138	ants	2006-03-05	13:26:14	19	http://www.greensmiths.com
3519280	ants	2006-03-30	17:14:14	0	
3519280	ants	2006-03-30	17:15:53	1	http://ant.edb.miyakyo-u.ac.jp
3519280	ants	2006-03-30	17:15:53	3	http://www.uky.edu
3519280	ants	2006-03-30	17:15:53	10	http://en.wikipedia.org
3519280	ants	2006-03-30	17:27:46	0	
3519280	ants	2006-04-01	13:55:03	2	http://www.lingolex.com
3519280	ants	2006-04-01	13:55:03	3	http://www.uky.edu
3519280	ants	2006-04-01	14:20:53	0	

Table A.4: The interactions depicted in table A.3 grouped in 30-minutes long sessions.

738 Appendix B. Yahoo! query-click log

739 Yahoo!'s dataset contains only anonymous information due to the same
 740 privacy issues experienced by AOL. It includes 66 million clicks recorded in
 741 July 2010 and relevance judgments of 650 thousand Web pages issued by
 742 experts between 2009 and 2010 related to some of the logged queries are
 743 also available. Each record, contains the interactions related to a single page

744 results of each user, and is made up by `query cookie timestamp url_1`
745 `...url_10 nc et_1 pos_1 ...et_nc pos_nc`, where

746 `query` is the anonymized version of the query,

747 `cookie` is the anonymized version of the user's cookie,

748 `timestamp` is Unix time (the amount of seconds passed since 1 January 1970)
749 of the issued query,

750 `url` is the anonymized version of the URL,

751 `nc` is the number of clicks performed during the entire session,

752 `et` is the time passed between each click and the beginning of the session,

753 `pos` is the position of each click, which could be:

754 `1 ...10` one of the 10 results,

755 `0` above the first result (spelling corrections, header advert, etc.),

756 `11` below the last result (next page, footer advert, etc.),

757 `s` new query,

758 `o` other clicks.

759 As for the previous dataset, we employed in our experiments the subset
760 of records related to the queries performed on average once a day during
761 the reference timespan and for which we have the relevance judgments. This
762 way, we obtained about 65 million different clicks related to almost 44500
763 queries, then grouped in 30 minutes long sessions, as the AOL query log; in
764 table B.5 there is a fragment of the log.

00002efd	1deac14e	1279486689	2722a07f	24f6d649	1b2b5a1c	9ca4edf1				
23045132	84c0d8b5	de33d1de	9f5855b2	477aabf6	e1468bbf	3	10	1	175	
o	215	0								
00002efd	3fef0ac3	12799559361	2722a07f	8f59fce1	de33d1de	a2c8d464				
57a7dd83	a11dbd14	08b5c87e	44a77e61	c21b6dbe	6b0a7915	1	2	0		

Table B.5: Yahoo! query-click log.

765 References

- [1] M. A. Hearst, What’s Missing from Collaborative Search?, *Computer* 47 (2014) 58–61.
- [2] G. Golovchinsky, J. Pickens, M. Back, A Taxonomy of Collaboration in Online Information Seeking, *CoRR* abs/0908.0704 cs.IR (2009).
- [3] Z. Yue, S. Han, D. He, J. Jiang, Influences on Query Reformulation in Collaborative Web Search., *Computer* 47 (2014) 46–53.
- [4] A. Broder, A taxonomy of web search, *ACM SIGIR Forum* 36 (2002) 3.
- [5] A. M. Zareh Bidoki, P. Ghodsnia, N. Yazdani, F. Oroumchian, A3CRank: An adaptive ranking method based on connectivity, content and click-through data, *Information Processing & Management* 46 (2010) 159–169.
- [6] S. Fox, K. Karnawat, M. Mydland, S. Dumais, T. White, Evaluating implicit measures to improve Web search, *ACM Transactions on Information Systems* 23 (2005) 147–168.

- [7] Y. Xu, D. Mease, Evaluating web search using task completion time, in: the 32nd international ACM SIGIR conference, ACM Press, New York, New York, USA, 2009, p. 676.
- [8] S. Lawrence, C. L. Giles, Accessibility of information on the Web, *intelligence* 11 (2000) 32–39.
- [9] D. E. Rose, D. Levinson, Understanding user goals in web search, in: Proceedings of the 13th International Conference on World Wide Web, WWW '04, ACM, New York, NY, USA, 2004, pp. 13–19.
- [10] T. Joachims, Optimizing search engines using clickthrough data, in: the eighth ACM SIGKDD international conference, ACM Press, New York, New York, USA, 2002, p. 133.
- [11] A. M. Zareh Bidoki, N. Yazdani, DistanceRank: An intelligent ranking algorithm for web pages, *Information Processing & Management* 44 (2008) 877–892.
- [12] C. Olston, E. H. Chi, ScentTrails: Integrating browsing and searching on the Web, *ACM Transactions on Computer-Human Interaction* 10 (2003) 177–197.
- [13] J. Wu, K. Aberer, Swarm intelligent surfing in the web, *Web Engineering* (2003).
- [14] Y. Freund, R. Iyer, R. E. Schapire, Y. Singer, An efficient boosting algorithm for combining preferences, *Journal of Machine Learning Research* 4 (2004) 933–969.

- [15] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, G. Hullender, Learning to rank using gradient descent, in: ICML '05: Proceedings of the 22nd international conference on Machine learning, Microsoft, ACM, New York, New York, USA, 2005, pp. 89–96.
- [16] Z. Zheng, K. Chen, G. Sun, H. Zha, A regression framework for learning ranking functions using relative relevance judgments, in: the 30th annual international ACM SIGIR conference, ACM Press, New York, New York, USA, 2007, p. 287.
- [17] Z. Zheng, H. Zha, T. Zhang, O. Chapelle, K. Chen, G. Sun, A General Boosting Method and its Application to Learning Ranking Functions for Web Search., NIPS (2007) 1697–1704.
- [18] J. Xu, H. Li, AdaRank: A boosting algorithm for information retrieval, in: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'07, Microsoft Research Asia, Beijing, China, pp. 391–398.
- [19] P. Li, C. J. C. Burges, Q. Wu, McRank: Learning to rank using multiple classification and gradient boosting, in: Advances in Neural Information Processing Systems 20 - Proceedings of the 2007 Conference, Cornell University, Ithaca, United States.
- [20] J. L. Elsas, V. R. Carvalho, J. G. Carbonell, Fast learning of document ranking functions with the committee perceptron, in: WSDM'08 - Proceedings of the 2008 International Conference on Web Search and

- Data Mining, Carnegie-Mellon University, Pittsburgh, United States, pp. 55–63.
- [21] Z. Chen, X. Meng, B. Zhu, R. H. Fowler, WebSail: from on-line learning to Web search, in: WISE 2000: 1st International Conference on Web Information Systems Engineering, IEEE Comput. Soc, 2000, pp. 206–213.
- [22] F. Radlinski, T. Joachims, Active exploration for learning rankings from clickthrough data, in: the 13th ACM SIGKDD international conference, ACM Press, New York, New York, USA, 2007, p. 570.
- [23] F. Radlinski, R. Kleinberg, T. Joachims, Learning diverse rankings with multi-armed bandits, in: the 25th international conference, ACM Press, New York, New York, USA, 2008, pp. 784–791.
- [24] J. Y. Yeh, J. Y. Lin, H. R. Ke, W. P. Yang, Learning to rank for information retrieval using genetic programming, in: Proceedings of SIGIR 2007: Learning to Rank for Information Retrieval.
- [25] E. Diaz-Aviles, W. Nejdil, L. Schmidt-Thieme, Swarming to rank for information retrieval, in: Proceedings of the 11th Annual conference on Genetic and evolutionary computation - GECCO '09.
- [26] I. Kushchu, Web-Based Evolutionary and Adaptive Information Retrieval, IEEE Transactions on Evolutionary Computation 9 (2005) 117–125.
- [27] M. Dorigo, V. Maniezzo, A. Colorni, The Ant System: Optimization by

- a colony of cooperating agents, *IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics* 26 (1996) 29–41.
- [28] A. Engelbrecht, X. Li, M. Middendorf, L. M. Gambardella, Editorial Special Issue: Swarm Intelligence, *IEEE Transactions on Evolutionary Computation* 13 (2009) 677–680.
- [29] T. Stützle, H. H. Hoos, MAX–MIN ant system, *Future generation computer systems* (2000).
- [30] X.-M. Hu, J. Zhang, Y. Li, Orthogonal Methods Based Ant Colony Search for Solving Continuous Optimization Problems, *Journal of Computer Science and Technology* 23 (2008) 2–18.
- [31] B. Bullnheimer, R. F. Hartl, C. Strauß, A New Rank Based Version of the Ant System - A Computational Study, *Central European Journal for Operations Research and Economics* 7 (1997) 25–38.
- [32] A. Malizia, K. Olsen, Toward a New Search Paradigm-Can We Learn from Ants?, *Computer* 45 (2012) 89–91.
- [33] T. Turchi, A. Malizia, P. Castellucci, K. Olsen, Collaborative information seeking with Ant Colony Ranking in real-time, *Proceedings of the th Italian Research Conference on Digital Libraries* (forthcoming).
- [34] Y. Xie, D. O’Hallaron, Locality in search engine queries and its implications for caching, in: *Proceedings - IEEE INFOCOM*, Carnegie-Mellon University, Pittsburgh, United States, pp. 1238–1247.

- [35] J.-R. Wen, H.-J. Zhang, Query Clustering in the Web Context, in: Clustering and Information Retrieval, Springer US, Boston, MA, 2004, pp. 195–225.
- [36] D. A. Grossman, O. Frieder, Information Retrieval, Springer Netherlands, Dordrecht, 2004.
- [37] I. Koychev, I. Schwab, Adaptation to drifting user’s interests, in: ECML Workshop: Machine Learning in New Information Age, pp. 39–46.
- [38] A. Malizia, K. Olsen, Emerging social search paradigms: can we learn from ants?, in: Proc. of the Italian Workshop on Artificial Life and Evolutionary Computation, pp. 1–4.
- [39] D. Hawking, N. Craswell, P. Bailey, K. Griffiths, Measuring Search Engine Quality, Information Retrieval 4 (2001) 33–59.
- [40] D. Gayo-Avello, D. Brenes, Making the road by searching-A search engine based on Swarm Information Foraging, arXiv.org (2009).
- [41] B. Smyth, E. Balfe, J. Freyne, P. Briggs, M. Coyle, O. Boydell, Exploiting query repetition and regularity in an adaptive community-based Web search engine, User Modelling and User-Adapted Interaction 14 (2004) 383–423.
- [42] S. Chakrabarti, A. Frieze, J. Vera, The influence of search engines on preferential attachment, in: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms, Society for Industrial and Applied Mathematics, pp. 293–300.

- [43] P. S. Efraimidis, P. G. Spirakis, Weighted random sampling with a reservoir, *Information Processing Letters* 97 (2006).
- [44] K. Church, M. T. Keane, B. Smyth, The first click is the deepest: assessing information scent predictions for a personalized search engine, *Proc AH2004 Workshop* (2004).
- [45] T. Joachims, Evaluating Retrieval Performance Using Clickthrough Data., *Text Mining* (2003) 79–96.
- [46] Y. Liu, Y. Fu, M. Zhang, S. Ma, L. Ru, Automatic search engine performance evaluation with click-through data analysis, in: *16th International World Wide Web Conference, WWW2007*, Tsinghua University, Beijing, China, pp. 1133–1134.
- [47] S. Jung, J. L. Herlocker, J. Webster, Click data as implicit relevance feedback in web search, *Information Processing & Management* 43 (2007) 791–807.
- [48] G. Pass, A. Chowdhury, C. Torgeson, A picture of search, in: *ACM International Conference Proceeding Series*, America Online, United States.
- [49] J. Gao, W. Yuan, X. Li, K. Deng, J.-Y. Nie, Smoothing clickthrough data for web search ranking, in: *Proceedings - 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009*, Microsoft Research, Redmond, United States, ACM Press, New York, New York, USA, 2009, pp. 355–362.

- [50] T. Sakai, Alternatives to Bpref, in: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'07, NewsWatch, Inc, Tokyo, Japan, pp. 71–78.
- [51] K. Jarvelin, J. Kekalainen, Cumulated gain-based evaluation of IR techniques, *ACM Transactions on Information Systems* 20 (2002) 422–446.
- [52] N. Martínez-Bazan, V. Muntés-Mulero, S. Gómez-Villamor, J. Nin, M.-A. Sánchez-Martínez, J.-L. Larriba-Pey, Dex: high-performance exploration on large graphs for information retrieval, in: *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CSIC - Instituto de Investigacion en Inteligencia Artificial, ACM, New York, New York, USA, 2007, pp. 573–582.
- [53] B. Liu, R. Jones, K. L. Klinkner, Measuring the meaning in time series clustering of text search queries, in: *International Conference on Information and Knowledge Management, Proceedings*, Virginia Polytechnic Institute and State University, Blacksburg, United States, pp. 836–837.
- [54] C. Silverstein, H. Marais, M. Henzinger, M. Moricz, Analysis of a very large web search engine query log, *ACM SIGIR Forum* 33 (1999) 6–12.
- [55] C. Eickhoff, K. Collins-Thompson, P. N. Bennett, S. Dumais, Personalizing atypical web search sessions, in: *Proceedings of the sixth ACM international conference on Web search and data mining - WSDM '13*.
- [56] A. H. Awadallah, R. W. White, S. T. Dumais, Y.-M. Wang, Struggling

or exploring?: disambiguating long search sessions., WSDM (2014) 53–62.