Question Waves: an algorithm that combines answer relevance with speediness in social search

Albert Trias i Mansilla[1a], Josep Lluís de la Rosa i Esteva[a]

[a] *Agents Research Lab, TECNIO Centre EASY, University of Girona, Campus de Montilivi, E17071 Girona, Catalonia, Spain (EU)*

ABSTRACT

This paper describes Question Waves, an algorithm that can be applied to social search protocols, such as Asknext or Sixearch. In this model, the queries are propagated through the social network, with propagation being faster through more trustable acquaintances. Question Waves uses local information to make decisions and obtain an answer ranking. With Question Waves, the answers that arrive first are likely to be the most relevant, and we computed the correlation of answer relevance with the order of arrival to demonstrate this result. We obtained correlations equivalent to heuristics that use global knowledge, such as profile similarity among users or the expertise value of an agent. Because Question Waves is compatible with the social search protocol Asknext, it is possible to stop a search when enough relevant answers are found; additionally, it is possible to stop the search with a small risk of not obtaining the best answers available. Furthermore, Question Waves does not require a re-ranking algorithm because the results come sorted.

Keywords: multi-agent systems, p2p, query-routing, social networks, social search.

1. Introduction

Query-routing is a key element of social search; where the search process can be reduced in finding the right person to ask [42]. We propose a query-routing algorithm for social search that provides first the answers that are probably the most relevant. The heuristic behind the algorithm consists on asking first the best candidates to satisfy the information need. Although than the usefulness of search engines cannot be questioned, there are information that is not indexable (deep Web [5]) which contains data from online social networks with restricted access. Furthermore, one pending issue in the evolution of search engines is about answering open questions as can be recommendations, which are still ill-served, and social search [10] is a valid answer to them. In this paper, we refer to open questions as those abstract, subjective, and recommendation type questions [39]. For example, questions like: "What would be a good app to find bargains in Barcelona?" cannot be expressed in a way that today's search engines can understand, let alone answer it. In social search, which emerged as a new paradigm of research interest, a good way to get a useful answer for that question would be by posing it to all your closer or distant friends or colleagues. Furthermore, Smyth et al. [33] note that 70% of the time users are searching for information previously found by their friends or colleagues. For these reasons, there are recently developed tools, such as HeyStacks [33] or the +1 button of Google, that contribute to the sharing of interesting results.

Due to these multiple concerns regarding how searches are conducted and the need to turn search into a social issue, several researchers suggest a change of paradigm. Advocates of the Social Search like Yu and Singh [42] expressed that often the most relevant information can be only accessed by asking the right people, and Nardi [28] wrote that "It is not what you know, but who you know", which can be the motto of today's search. Hendler [16] expressed this concept even more clearly by introducing the term of social machines: "Social-networking sites are replacing search engines and news sites as people's favoured homepages and a new generation of applications is centring on large groups of people collaborating over an increasingly distributed network" and "one thing is for sure:

[1] Corresponding author, email address: albert.trias@udg.edu
Email addresses {albert.trias, joseplluis.delarosa}@udg.edu

many of the entrenched models of AI will have to be rethought as computers change from application devices to social machines".

Social search [10] is then a new paradigm, referred to as the village paradigm [17], a type of search that uses social interactions to provide results. It can be classified as Social Feedback Systems (SFS) and Social Answering Systems (SAS) [10]. SFS use social data to sort the answers, while SAS use people's expertise or opinions to question answering. SFS cannot address questions when the information is not available; SAS solve this problem, yet experts might receive a same question many times. What is interesting is that recommender systems [26] can be considered a SFS, ranking within a category (RWC) recommender systems [40]. In an RWC system, users obtain a list of items with a personalized ranking, as in SFS. That is why we consider recommender systems data are valid for testing algorithms of social search, what will be novel and extremely useful for the design and test of social search algorithms. Knowledge exchange portals (Q&A) are SAS that are usually used when users do not obtain satisfactory results from search engines [15], as is the case when they are searching for opinions or customized advice. The most important drawback of these portals is the frequent lack of answers, perhaps due to the fact that the right people are not willing to answer or are not accessible, or the question did not get through to them. Usually, experts receive a burden of questions that overloads their limited bandwidth, and in consequence, the questions might be answered slowly. There are three main options to minimize answering time and provide good results [1]: reusing content [6,19], routing questions to experts on the topic [13,23,45] and encouraging more answers by understanding answering behaviours [13,22]. Furthermore, questioner satisfaction is an interesting problem in SAS; Agichtein and Liu [1] showed that previous interactions are useful predictors of questioner satisfaction and are able to outperform human classifiers.

The drawbacks of social search is that querying all the people you know is very expensive in terms of effort and is thus not a viable solution. As we will see in the paper, there are several query-routing algorithms to help users satisfying their answer need through finding the right person to ask such a question. Instead, there should be an algorithm for getting the most relevant answers, relevant to the questioner, from the right person to answer such a question, with reduced questioning effort. That algorithm that put the question forward a sequence of subsets of acquaintances and later aggregates or selects the right answer is a new type of query-routing algorithm that is referred to as Question Waves (QW). QW uses intelligent agents that works on behalf of people with the aim of automating query-routing and combining the SFS with SAS approaches of social search that takes advantage of the benefits of both of them: when an agent receives a question; if it has enough knowledge then it will answer automatically, otherwise, if it considers that its owner will be interested to answer, then it poses to her the question; the agent also can forward the question to its acquaintances; the agent will send first the question to the acquaintances with a higher probability of providing a relevant answer. After certain time, if the information need is not satisfied, the agent will send the question to the following subset of acquaintances that are best suited to answer; this last process is repeated until their information need is satisfied or there are not acquaintances available.

To the best of our knowledge, there are no such algorithms with a proven track record of getting the most relevant answers. Experiments use measures of relevance (as recall) for comparing the several aspects of the algorithm that contributes to its performance.

QW algorithm is executed at every node of the social network, using local knowledge and contact lists to decide query-routing, social behaviour, and question-answering. Intelligent agents are chosen for QW as a suitable approach for social search because they share the reactive, social, proactive, and autonomous features of the actors, people, in the social networks; furthermore, they are a good match for P2P systems [24] that are at the cradle of the query-routing algorithms. Agents decide the actions to perform after receiving a question, are capable of asking questions to other agents, and can take the initiative of requesting knowledge maintenance from their users. Agents can improve the answering time by appropriately routing questions to experts and reusing previous answers.

We advocate the usage of P2P systems for this scenario due the following reasons: the model of a general search engine for everything might not be scalable with the size, variability and heterogeneity of the Web and its users [41]; the entities that create these centralized systems can have conflict of interest with the results [40]; or can act as a "big brother" [8]; social networks are inherently peer-to-peer (P2P) [8].

P2P systems require query-routing protocols to find the right information source. Query-routing for decentralized search is an important research problem with applications in social search and file-sharing environments [3].

In all, QW can be described as a query-routing algorithm for social search using agents that operate on the same (P2P) social network of people. We will see that QW provides first the answers of the highest probability of being relevant before providing other, less relevant answers. This study demonstrates the mechanisms behind such results with every detail. Section 2 contains a brief literature review of query routing in unstructured P2P networks. In Section 3, the proposal of QW is presented in detail. In Section 4, simulations of the algorithm are described, and their results are discussed. Finally, in Section 5, the conclusions and future work are presented.

2. Literature review

Considering social search, query-routing has been applied to several domains, such as Internet browsers [41], question answering [3,17,38] and recommender systems [40]. The classification of the query-routing algorithms used in those domains is based on if the algorithms are unicast or multicast. In unicast query-routing algorithms, a query is only sent to one acquaintance or user [3,25,34], while multicast algorithms are ones in which a query is sent to many of them [38,40,41].

To better understand query-routing, it is important to know Milgram's experiments. In the 1960s, Milgram carried out a set of experiments to study the small-world problem. [34] explains the experiments: in brief, they selected 2 sets of individuals. The individuals in the first set received a document explaining the experiment, defining who should be the target person (belonging to the second set) of the document and containing some information about her. If the people of the first set knew the target person (TP) personally, then she would have sent the document to the TP; otherwise, she would have sent it to an acquaintance more likely to know the TP, and these acquaintances would be requested to repeat the process. The authors sent the document to 296 initial users, of which 73% followed the instructions, and the other 27% ignored it. Each time the folder was forwarded, there was probability of it being forgotten (dropped). From these experiments, we can see that unicast query-routing is feasible, but its drop rate considerably reduces the success rate; in the work of Travers and Milgram, only 64 folders reached their destination, while the others were dropped at some point.

More recently, but in a similar vein, Banerjee and Basu proposed the Social Query Model (SQM) [3], which is a probabilistic model that measures the probability of obtaining an answer P_i (Eq. 1). In the SQM, the nodes that receive a query can ignore, answer or forward it. The model considers the following components:

- The expertise $e_i \in [0,1]$ indicates the probability that the node a_i answers a query, with a probability 1- e_i that the node a_i forwards the query.
- The correctness $w_i \in [0,1]$ indicates the probability that the answer provided by the node a_i is correct.
- The response rate $r_{i,j} \in [0,1]$ indicates the probability that a node a_j accepts a query from node a_i. The probability that a node a_j drops the query is then 1- $r_{i,j}$.
- The policy π_j of a node a_i is a probability distribution that indicates the probability of selecting neighbours of a_i (N_i) when a_i decides to forward a query; concretely, each π_j^i indicates the probability that a_i forwards the query to a_j.

$$P_i = w_i e_i + (1 - e_i) \sum_{j \in N_i}^{n} \pi_j^i \, r_{i,j} P_j \tag{1}$$

The SQM is based on unicast query-routing in which the authors propose an algorithm to compute the optimal policy, although they also suggest finding the next-best policies in the case of a multicast application. In their experiments [3], they set the expertise by random or based on degree, making nodes with higher in-degree to have more expertise. The response rate can also be random or degree-based; the degree-based rate is directly proportional to

the expertise of the asking node and inversely to the node requested because the authors assumed that "people tend not to ignore requests from experts" and "experts tend to be often too busy to answer".

[25] presents SEMANT, an algorithm for distributed searching in P2P networks. In this algorithm, a forward ant that has the responsibility of answering the query is generated for each one; when it reaches a node with some content that might satisfy the query as an answer, the forward ant finishes its travel, and a new backward ant is generated and follows the inverse path while dropping pheromones. The algorithm uses two strategies for query-routing: in the exploiting strategy, the ant follows a link to the best, yet unvisited, neighbour; in the exploring strategy, the forward ant tries to find new paths. For every unvisited peer, it is randomly decided whether the peer should be visited on the basis of its goodness value. If more than one peer is selected, the forward ant is duplicated; each peer stores the information of the ants that visited it, allowing termination of the copies of the forward ant if they reach the peer more than once. In SEMANT, the exploiting strategy is clearly unicast, while the exploring strategy is multicast.

Multicast query-routing in unstructured P2P networks is usually based on the breadth first search (BFS) algorithm, with the following main idea:

- one node, that starts a search, asks a subset of its neighbours by sending a query/question message;
- when a node receives a question, the node may answer or forward it;
- when a node has an answer, an answer message is sent following the inverse path.

These algorithms usually use the *time to live* (TTL) to reduce the number of question messages. The TTL determines how many hops can go between the questioner and the answerers. The TTL is initialized, and every time it is forwarded, it is reduced by one; when the TTL reaches 0, the message cannot be forwarded anymore.

Sixearch [41] is a P2P search engine based on BFS that uses TTL. Peers store their neighbours' profiles with the goal of increasing the probability of choosing an appropriate neighbour to receive a question. Peers store information in these profiles about the terms in which their acquaintances have expertise. Initially, a profile is requested from an acquaintance, and the profile is subsequently updated with the interactions of the node with its acquaintances.

Walter et al. [40] proposed a recommender system for P2P social networks in which query-routing is also based on BFS. In their model, each answer has a trust value, which is the trust associated to the path followed to find the answer. All possible answers are collected, and finally, an answer is selected randomly out of the set, with answers that come from more trusted paths having a higher probability of being chosen. The authors concluded that this model shows self-organisation and sub-optimal performance.

More recently, the Asknext protocol [35,38] was introduced, resulting in 20-fold fewer messages and higher scalability than Sixearch or related social search BFS query-routing based protocols. Asknext is also based on BFS with the novelty of stop messages, which stop questions from being sent when a relevant answer is found. The use of *stop messages* requires that *answer* and *stop* messages travel faster than *question* messages. It is possible to use a dynamic speed for *question* messages, making their propagation slower, and doing so allows the question propagation to be stopped at the same hops distance that a relevant answer is found (Eq. 2). T_F and T_R are the periods during which questions are forwarded and answers are sent, respectively. The forward speed can be set using Eq. 2. For example, with a $T_R = 1$ for nodes at distance (r) 1, T_F should be ≥ 3; for nodes at $r = 2$, T_F should be ≥ 5, and for nodes at $r = 3$, T_F should be ≥ 7. In practice, using different periods for T_F and T_R does not mean that the question messages travel slowly but rather that a node retains a question message for longer before forwarding it.

$$r < \left\lfloor \frac{T_F}{2T_R} \right\rfloor = \left\lfloor \frac{v_R}{2v_F} \right\rfloor \tag{2}$$

In the case of multi-agent referral system (MARS) [42,43,44], the authors follow a different approach. Yu and Singh proposed a P2P application in which agents assist users in obtaining and following referrals to find experts who might answer their questions. When a user formulates a question, it is processed by her agent, who determines which contacts should be the question recipients. When an agent receives a question, it decides whether to show the question

to its owner or answer by providing referrals. When an agent receives an answer, the user's feedback is used to evaluate the expertise of the answerer.

While in the unicast query-routing there is no threat to the system scalability and costs, there is a lower probability of finding an answer. In the end, using multicast query-routing threats the system scalability, yet there is higher probability of satisfying the information need.

In the following Section, we will explain our contribution: each node consists on an intelligent agent that represents a user; these agents conduct several attempts to obtain answers in multicast query-routing and relax the problem of threatening system scalability to increase the probability of finding an answer. Initially, a question is sent to the best candidates. After a given time, if the information need is not satisfied (because it is not answered or the answer is not satisfactory) then the question is sent to the next best candidates. The process is repeated at each agent until the question is answered or there are no more candidates.

3. Question Waves

As has been stated in Section 2, unicast query-routing can be used to satisfy information needs while bothering only a few people in the network, but the probability of satisfying the information need is low; in contrast, using multicast query-routing, so that a question is propagated to all of the acquaintances, threats the system scalability. Finding the best number of acquaintances to which to forward a query is not straightforward. We consider that the best behaviour is requesting as few users as possible and obtaining the greatest or enough number of correct, relevant answers available. However, going rather far in reducing the number of recipients might avoid obtaining some relevant answers; otherwise, requesting all available peers can overload the system.

We consider that one option to deal with this problem is to first ask the candidates who are estimated to give the best answers and only then, if after some time the initial candidates are not able to provide a suitable answer, requesting the next in the contact list. Heuristically, a unicast query-routing can be used when the requester is near to an expert willing to answer and highly available. In the worst scenario, the system will work as a BFS multicast query-routing in which the majority of the nodes are requested only when no answer is yet available. The number of candidates that are requested will be adapted on the basis of the availability of answers. Subsection 4.3 provides details about the number of messages generated and the number of relevant answers provided by the different algorithms considered.

The work described in this paper is based on an agent P2P social network; in which each user is represented by an intelligent agent (we will refer them as agents), we also will refer users as owners of an agent. Each user has a knowledge base that is managed and can be accessed by her agent. Those agents can send questions to a subset (Sa) of their acquaintances, and when they receive a question they can forward the question to Sa as well or can answer. The answers follow the reverse path [25,38,40,41].

For each question, the agents can play three roles:

- **Questioner**: The questioner is the agent who started the question.
- **Answerer**: An answerer is an agent who answers a question with the agent's own knowledge or its user's knowledge.
- **Mediator**: A mediator is an agent who receives a question and forwards it to others. They also may forward an answer that they receive from another mediator or an answerer.

In the following Subsection, we will explain briefly the probability of obtaining an answer in a multicast query-routing, starting from the unicast social query model [3]. Then, with a better understanding of multicast query-routing, we will explain our main contribution, Question Waves, including the main points and the analogy from physical waves. In Subsection 3.3, we will explain some aspects that can be taken into account to set up Question Waves.

3.1 Introduction to multicast query-routing

In Section 2, we explained briefly the SQM proposed by Banerjee and Basu [3]. Their model is based on unicast query-routing and in that case, the probability of obtaining an answer asking others (expressed by the sum operator) contains a mutually exclusive probability. In the case of multicast query-routing, the probability of obtaining an answer that satisfies the question is not mutually exclusive, as several candidates can answer it correctly. We consider that it is the only aspect needed to be considered with the aim to apply their model in multicast query-routing.

In this Subsection we explain the probability of satisfying a query in multicast query-routing, in function of the distance. Given that:

- At a distance d, $\mathrm{P}_{i,j}(d)$ is the probability that the question q_j of questioner agent a_i is satisfied with the answers obtained from answerers at a maximum distance d.
- At a distance d, $\partial\mathrm{P}_{i,j}(d)$ is the probability that the question q_j of questioner agent a_i is satisfied with the answers obtained from answerers at a distance d exactly.
- $k_{i,j}$ is the probability that an agent a_i satisfies the question need with an own answer the question q_j.
- At a distance $d = 0$, the probability of satisfying the question need depends only on the agent a_i.

Then:

- $P_{i,j}(0) = k_{i,j}$
- $P_{i,j}(d+1) = P_{i,j}(d)\ or\ \partial P_{i,j}(d+1) = P_{i,j}(d) + \left(1 - P_{i,j}(d)\right) * \partial P_{i,j}(d+1)$

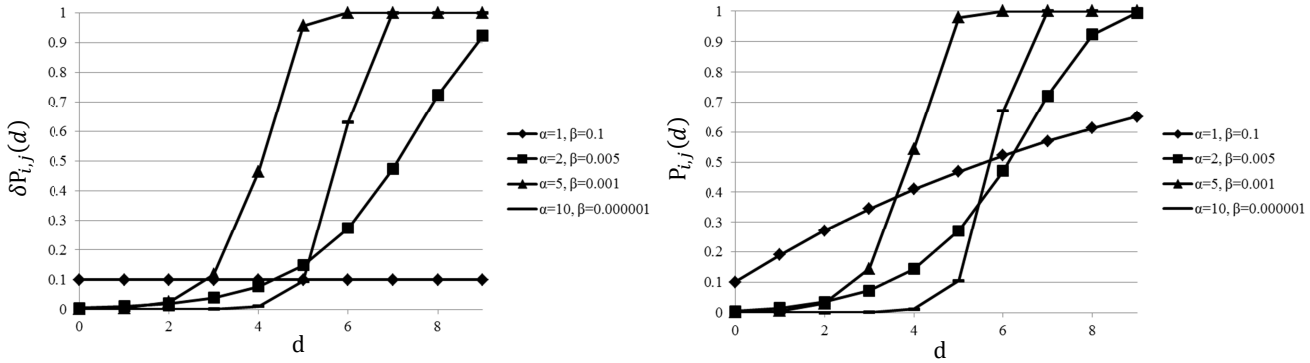
As a simplification, we assume the following major constraints:

- $\forall_i\ k_{i,j}=\beta.\ \beta\epsilon[0,1]$.
- Agents never drop a question.
- The social network topology is an infinite tree of degree α (α∈ℕ), so when the nodes forward a question, it is forwarded to α nodes that did not previously receive it.

Now:

- $\delta\mathrm{P}_{i,j}(\mathrm{d}) = 1 - (1-\beta)^{\alpha\mathrm{d}}$
- Given a $\beta < 1$ and a $\alpha > 1$, then $\lim_{d\to\infty} 1 - (1-\beta)^{\alpha\mathrm{d}} = 1$

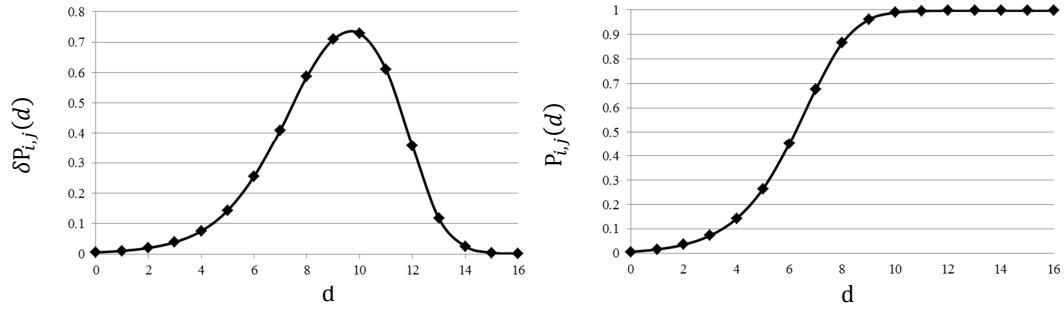
Example for several α and β are shown on Fig. 1.



(a) (b)

Fig. 1. (a) $\delta P_{i,j}$ in function of distance (b) and $P_{i,j}$ in function of distance. For several β and α.

Because social networks do not follow a tree topology, it is not always possible that each node forwards the question to α nodes that did not previously receive the question. Furthermore, social network topologies are finite. If the network topology is random, the probability that a neighbour already received the question will depend on the number of nodes that have been forwarded to before. In this case, the probability depends on the current distance. However, all nodes do not have the same amount of contacts. For example, for a topology of 1,000 agents with an average nodal degree of 3, we can obtain a situation similar to that in Fig. 2.



(a) (b)

Fig. 2. (a) $\delta P_{i,j}$ in function of distance, and (b) and $P_{i,j}$ in function of distance. For $\beta = .005$ in a finite network of 1000 agents with an average nodal degree of 3.

We can see that $\delta P_{i,j}(d)$ starts growing until a certain distance, after which it starts to decrease. The maximum point will depend on the shortest path from the questioner to the other nodes the point at which $\delta P_{i,j}(d)$ starts to decrease is the average shortest path length. When using these simplifications, $P_{i,j}$ does not depend on the network topology as long as all nodes are requested. With different k_i but accessing the nodes in the same way, we obtain the same kind of results as in Fig. 2.

If we consider a case in which all of the agents do not put equal effort toward trying to obtain an answer—for example, not all of the agents forward the question to the same number of contacts and the probability that agents can satisfy the information need (p) depends on the contacts' expertise (k), correctness and response rate—and if we assume that all of the agents are able to identify the nodes that are more likely to answer and first ask the nodes with greater p then ask other nodes with the next highest p in different attempts (or *waves*), we expect a higher probability of satisfying the information need with answers from earlier waves.

Now instead of evaluating $\delta P_{i,j}(d)$, we are evaluating $\delta P_{i,j}(t)$ where t represents the time in simulation steps. We simulated an environment in which the questioner agent sends a request to 5 nodes, sorted from greater expertise to less, at timestamps {0, 1, 2, 3, 4}. There is a higher $\delta P_{i,j}(t)$ from the initial waves (Fig. 3 (a) and (b)).

Furthermore, if the questioner agent delays the waves, for example, initiating them at timestamps {0, 10, 20, 30, 40}, we obtain the probabilities in Fig. 3 (c) and (d). In the first attempt (first wave), $\delta P_{i,j}(t)$ is higher than in the other waves; after an initial increase, $\delta P_{i,j}(t)$ decays over time. In this Section, we will try to obtain a system that generates the kind of probability distribution from Fig. 3 (c) when all of the agents have the same behaviour.

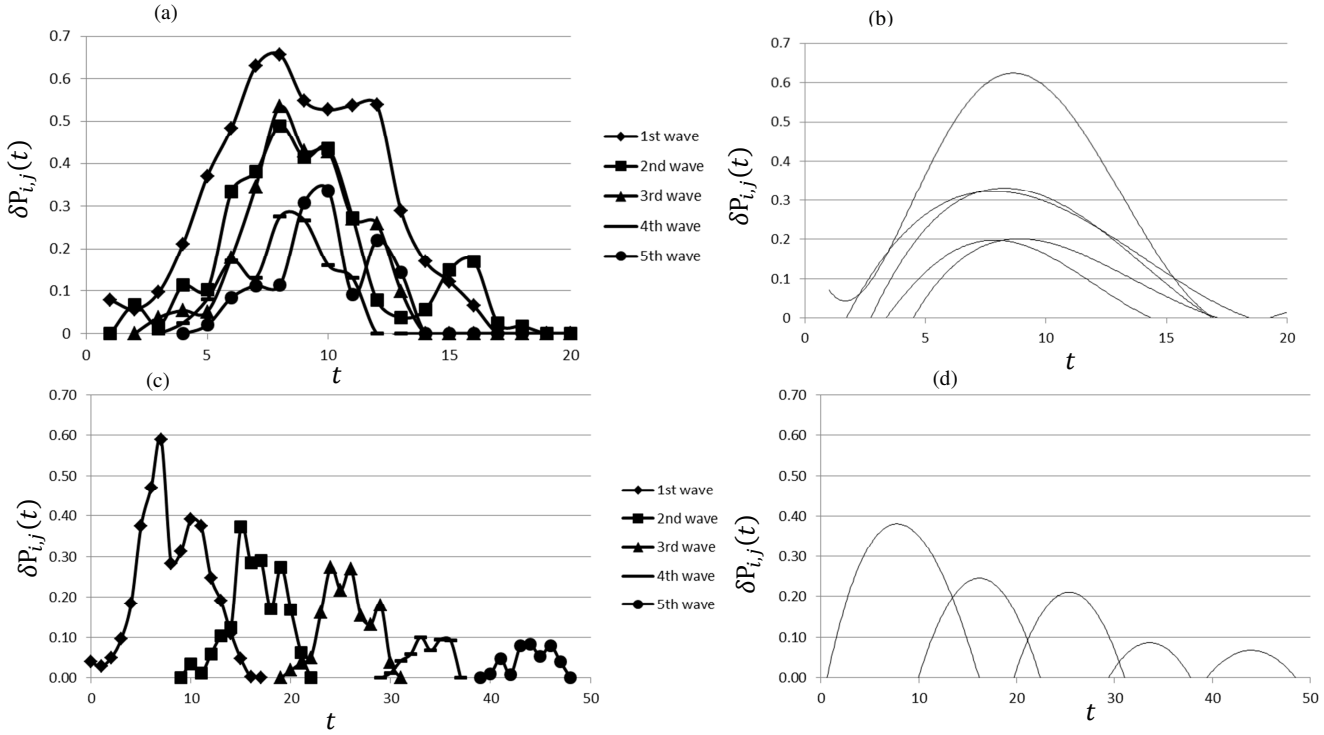


Fig 3. $\delta P_{i,j}(t)$ for each attempt or wave. (a) and (c) show the values obtained through simulation, while (b) and (d) show its tendency lines. (a) and (b) are the results when all of the agents execute the same behaviour, and (c) and (d) are the results when a questioner agent has applied higher delay to the waves. y axis represents the probability in that time step, while x axis represents the time in simulation steps.

In the area of delay tolerant networks, Bulut et al. proposed and algorithm, that also uses some attempts to distribute copies of a message [9]. The problem they deal with consists on delivering a message before a deadline using the minimum number of message copies. Their algorithm first send a small number of copies (this is an attempt), and if after some time periods it is not delivered to the destiny, then more messages are sent to increase the delivery probability (further attempts). The main idea is that if the message is delivered in the initial attempts then the number of message copies needed to deliver the message has been reduced. They also use messages to stop the propagation of the copies once the message has been delivered (they call it acknowledgment of delivery). Although its domain of application is different, the usage of several attempts follows a similar idea of QW. One of the most important differences dealing with different attempts in social search and delay tolerant networks, is that in the first one it is important to provide relevant answers, while in the second one it is only important to deliver the message. That is why

in this paper we compute several measures in our simulation to evaluate that the most relevant answers are received early.

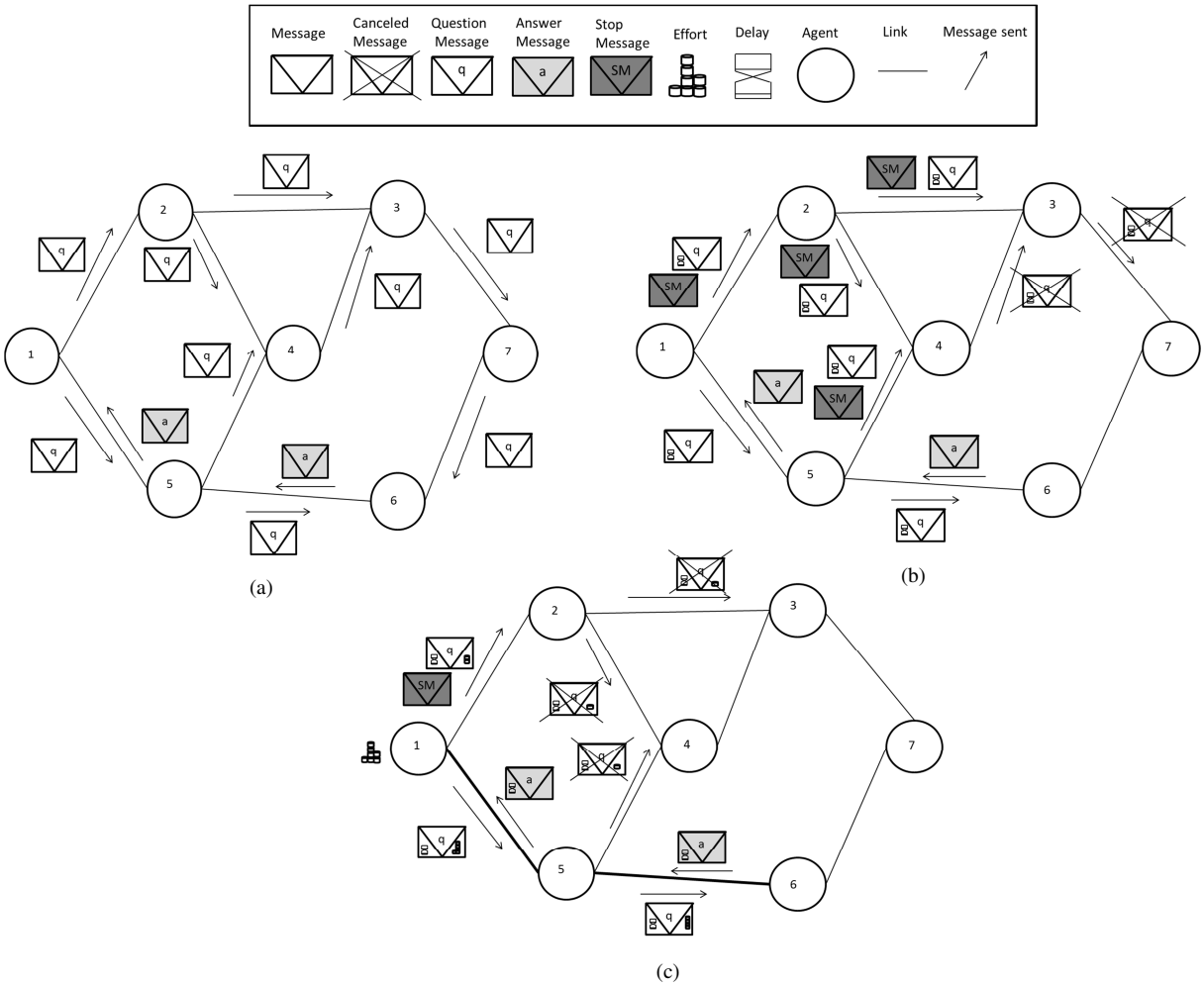


Fig. 4. Example of BFS (a), Asknext (b), and QW (c) with the same social network, where a_1 is the questioner and a_6 the answerer.

Fig. 4. shows some of the main differences among BFS, Asknext and QW search algorithms. In this example, agent a_1 is the questioner and a_6 is the only possible answerer, because one answer would be enough for the questioner and mediators. We will explain how it may work with the different algorithms:

- BFS is the most basic algorithm, where only there are question messages (q) and answer messages (a). These example corresponds to a TTL of 4 or greater:
  - Step 0: a_1 is the questioner and sends the question to $\{a_2,a_5\}$.
  - Step 1: a_2 and a_5 forward the question to $\{a_3,a_4\}$ and $\{a_4,a_6\}$ respectively.
  - Step 2: a_6 answers to a_5, at the same time a_4 forward the question to a_3, and a_3, who received the message previously from a_2, forwards it to a_7.
  - Step 3: a_5 forwards the answer to a_1, and a_7 forwards the question to a_6.

- Asknext introduces the usage of stop messages (SM), and it needs a delay on the question messages that depends on the distance, according Eq. 2.
    - Step 0: a_1 is the questioner and sends the question to $\{a_2,a_5\}$.
    - Step 1: a_2 and a_5 receive the question, they do not have an answer, so they will forward the question in step 3.
    - Step 2: a_2 and a_5 wait until step 3.
    - Step 3: a_2 and a_5 forward the question to $\{a_3,a_4\}$ and $\{a_4,a_6\}$ respectively.
    - Step 4: a_6 answers to a_5. a_3 and a_4 will forward the question at step 8.
    - Step 5: a_5 answers to a_1. a_5 and sends a SM to a_4.
    - Step 6: a_1 sends a SM to a_2. a_4 cancels forwarding the question.
    - Step 7: a_2 sends a SM to a_3.
    - Step 8: a_3 cancels forwarding the question.
- QW adds the concepts of effort, the question delay is based on trust on the contacts, not in the distance from the questioner as Asknext, and the answers also can contain a delay. In Fig. 4. (c) thicker links represents trust, and the question message propagate earlier for those links.
    - Step 0: a_1 has 7 units of effort to try to obtain and answer. a_1 decide to send first the question to a_5 with an effort of 5 units, and it will send to a_2 the question with an effort of 2 units at step 3 if no answer is found yet.
    - Step 1: a_5 sends the question immediately to a_6 with an effort of 4 units, and a_5 will forward the question to a_4 at step 4 if no answer is found yet.
    - Step 2: a_6 has confidence in his answer and sends it immediately to a_5.
    - Step 3: a_5 forward the answer to a1, and cancels forwarding the question to a_4. a_1 has no received an answer yet and forwards the question to a_2 (it is the second wave).
    - Step 4: a_1 receives the answer, and sends a SM to a_2. a_2 decides to forward the question to a_3 and a_4 at step 6 as it has not enough trust on that agents.
    - Step 5: a_2 receives the SM and cancels forwarding the question to a_2 and a_4.

3.2 Modelling multicast query routing as waves of questions or question waves

As we have seen in Subsection 3.1, it is possible to first ask the acquaintances with the highest probability of providing a relevant answer. We were distantly inspired by physical waves for modelling this process, which is why we call it *Question Waves* (QW), however the aim of QW is not obtaining physical waves properties and the resulting algorithm does not guaranties any of them, physical waves have been only an inspiration source.

A *question wave* is an attempt to find an answer to a question. In every attempt, the same question is sent to a subset of acquaintances. The probability of finding appropriate answers decays after every attempt. The question propagates through the network of agents, which amplify or attenuate the effort invested in answering it, as a wave is attenuated over distance in an aquatic environment. QW tries to solve the following problems:

a) In P2P there are scalability issues when you ask all peers.
b) If you ask only one peer (or a small subset), it is very likely you will not find the right answer.
c) If you ask several sequentially it is very likely you get stuck if any in the sequence not answers you.

In any case (b, c) choosing the right peers to ask is not straightforward. The solution that we propose to address the above problem is to use several attempts (waves), elapsed in time. In each attempt, the question sender (questioner or mediator) selects the most reliable acquaintances who have not been previously selected for that question. Adding new recipients implies that the agent is not sure that it will receive an answer from the current recipients and is trying to obtain an answer from less-trusted recipients. This approach can be used when useful answers are received, enabling the user to read them in real time (when the search process is not complete) as a heuristic from the most to

least relevant. The advantage of QW is that multiple agents are committed to finding the answers with diversified options to find them, resulting in more relevant[2], faster[3], and more robust[4] answers.

In physics, "a wave may be defined as a periodic disturbance in a medium that carries energy from one point to another" [14]. In our model, when an agent has a question, the agent is a wave source and generates a disturbance on the system with an amount of energy (E_0). This energy indicates the amount of effort that the agent devotes to satisfy its question need. The probability of finding a relevant answer depends on the amount of effort invested in finding, and the agents can transmit part of this effort to others, delegating a part of the task (finding relevant answers). The propagation media are the links of the social network.

For physical waves, the Huygens principle states that each region reached by a wave generated by a disturbance generates another disturbance in other regions. In our model, when an agent receives a question, it may act as the next mediator that transmits the disturbance to other agents forwarding the question.

A physical wave speed depends on the medium properties. In our model, the medium is the link, and the speed of the question message will depend on the trust as the key property of the link. When a physical wave is propagated to several directions, the energy is spread over a large area, reducing the intensity at each point, although the total amount of energy in the system remains equal. In our model, the effort is also distributed among a subset of acquaintances; on average, each agent will receive less effort as more acquaintances are requested. In our model of question waves, effort does not need to be propagated uniformly to all of the acquaintances; agents can decide how they distribute the effort among their acquaintances.

In physics, waves may lose energy due to attenuation [14]. The loss of energy is due to the viscosity and friction of non-perfectly elastic media and is described by the absorption law. As in question waves we modelled energy as effort it can be lost due to communication costs. Some effort will be dissipated due to friction in the sense that not all of the effort transmitted to an agent is converted to effort to answer a question because these agents may be less motivated to help a concrete acquaintance or the acquaintance does not want to contribute to this question. The principle of the superposition of physical waves explains that when several waves are coincident in the same space, their amplitudes are added. This interference can be constructive, i.e., resulting in a bigger amplitude, or destructive, resulting in a lower amplitude. In our model, it is possible that one agent receives a same question more than once from different sources or as a result of several question waves. In this case, it is possible to use more effort than that contained in each request; for example, the agent may add the efforts requested from each request. Finally, an answer can act as a destructive interference, e.g., when a relevant answer is found, further attempts are not needed. This destructive interference can be propagated in the form of answers and stop messages.

To set up an environment to experiment with Question Waves (QW), it is necessary to consider the effort introduced with disturbances, attenuation, friction, speed of links to acquaintances, effort distribution, and constructive and destructive interferences. In the following Subsection, we will explain some ideas to build a QW environment.

3.3 Setting up a question wave behaviour

Previously, we have seen that to build a QW environment, we must consider the effort introduced with disturbances, attenuation or effort costs, medium properties (such as speed and the amount of effort transmitted to the recipients), and interferences (constructive and destructive). But in QW, the properties of the media are not physically different; the agents decide which acquaintances to send the question to first and which amount of effort the recipients would receive. A trust model is needed for this aim to evaluate the behaviour of the acquaintances, requiring some measures to evaluate the answers that they provide. In this Subsection, we will start with the answer evaluation, we will explain what the trust model must consider, and then, we will explain how to set the effort costs, medium speeds, effort distribution, and interferences.

---

[2] Relevant in the sense that the answers come ranked by trust.
[3] Faster in the sense of reducing the burden of questions, and the agents are less overwhelmed.
[4] Robust in the sense of finding answers persistently.

3.3.1 Answer relevance and trust model

The relevance of one answer might depend on several factors, such as the expertise (*k*) of the person who generated it, the grade of implication (I) put on answering (the amount of effort), and the rewards used to motivate the agent, although motivation might also stem from the task itself or who asked for the task. Answer relevance also depends on other factors (R), such as the personal situation, the amount of time available to answer, the answerer mood, etc. [36,37]. The first aspect that we must consider before setting up a QW environment is if the answers will be generated to satisfy the requested information (i.e. requesting an answer from the user), if the system will reuse the answers available in the community, or both. It is also important to consider if the answer relevance is determined subjectively or objectively. In Section 4, we simulated a case in which the answers were objective and generated another one in which the answers were reused and subjective.

The trust model is a key aspect of the QW behaviour because trust is used to decide the order in which to ask agents. Trust can be used as a predictor of answer relevance as well as it can be used to distribute the energy among acquaintances. In environments like this network, trust models tell us what is expected from an acquaintance.

Agents' behaviour may be based on *reciprocity*. [18] explains that social exchange theory assumes that people try to have balanced relationships (reciprocity); people prefer relationships in which they give and receive a similar amount of support. When there is a discrepancy between giving and receiving, the continuation of the relationship is threatened. There are some exceptions, such the cases of family or close friends. In close relationships, people feel responsible for each other's well-being and do not consider the balance. For these reasons, we think that a Q&A agent will have contacts classified into two groups: one group for whom the agent feels responsible and does not consider the balance in the relationship and another group for whom the agent expects the relationship to be balanced. We call the members of the first group **close acquaintances** and those of the second group **convenient acquaintances** [36].
In QW, when we consider convenient acquaintances, a trust model can be used to estimate the viscosity as a kind of *reciprocity*. If an agent does not receive any help from an acquaintance, the acquaintance's trust value should decrease. Trust models also should detect the presence of malicious agents, who, for example, answer fast with spam.

3.3.2 Question delay

The QW behaviour is based on initially requesting sources with a higher likelihood of returning a suitable answer. For this proposal, it is necessary to implement a function that indicates the time needed to wait before submitting the question. This function can be implemented as a continuous or a piecewise function. When an agent sends a question to another recipient, the recipient's ability as an answerer but also as a mediator should be considered.

3.3.3 Delaying an answer

When an agent receives a question and has an answer (or a set of them) available, it is able to answer immediately. In contrast, if the agent thinks that it is possible to obtain a better answer from an acquaintance, then it can delay the answer to allow a better one to be sent. In the case that the agent does not know of an available answer but can generate one, then it can delay the answer generation if the agent is aware of the quality of the answer it can generate. The heuristics used are sending any available answer when the agent considers that the probability of obtaining a better answer is low and delaying answering otherwise.

3.3.4 Effort

When an agent has to initiate a question, the agent puts an amount of effort E_0 to answer it. This effort is put directly into answering or obtaining an answer from acquaintances or into the effort that the agent will delegate to acquaintances.

Some of the effort can be lost in the search of answers because there are some communication costs that include processing the messages, sending them, and planning how to satisfy the requests. Most of these costs do not depend on the amount of effort requested, and we will refer them to as *constant costs*[5]. In contrast, the delegated agents may give less effort than requested due to the implication relation they have to the task, and the imbalance can be seen as some kind of commission or the benefit that the delegated agents get in the case that the system uses social currencies; we will refer to these as *friction costs*.

Effort is also used to obtain answers. In the functioning of the environment, answer generation can require different amounts of effort (one clear example is numerical analysis, in which more effort can be understood as more iterations) or can have a constant cost, such as querying a database.

The effort that an agent uses in solving a question can be used to set a threshold τ after which the agent will stop trying to obtain new answers. This threshold can indicate the number of answers sent, the probability of providing a relevant answer, etc.

Although the model considers that the effort will be attenuated with the question distribution, agents are autonomous, and it is possible that one agent might use more effort to obtain an answer than requested. This excess can happen in the case that one agent has its own interest in obtaining an answer for the question or it would like to improve the relationship with the requester (which can be seen as a kind of investment).

3.3.5 Effort distribution

When an agent delegates a question to its acquaintances, the agent can decide how the effort can be distributed. The most basic configuration would be an isotropic distribution, in which all of the acquaintances would receive the same amount of effort. However, it is also possible to use other configurations, which distribute effort as a function of trust, for example. The distribution may depend on the question, on aspects as the domain or question context, or on aspects of the self-confidence of providing a relevant answer. An agent can also take into account not continuously annoying the same agent, with the intention of not threatening their relationship.

3.3.6 Interferences

Mainly, the model considers two kinds of interference, constructive and destructive. Constructive interference is produced when an agent receives the same petition from different sources. As a function of how the model is designed, the amount of effort taken into account to try to answer a question can be the minimum effort transmitted in one of the requests or the sum of the requests.

Destructive interferences are transmitted by answers and stop messages. For answers, when an agent receives enough answers of a certain quality that cross the threshold σ, the agent will stop trying to find more answers, and it can also ask to stop the search process by sending a stop message to the agents it posed the question. The objective threshold σ will depend on the effort put toward answering the question.

4. Simulations

The objective of this Section is to show how the QW model can be implemented and the benefits that it can provide. Optimizing the results will be a next step out of the scope of this paper because it would depend on every context. In our simulations, a set of agents $A=\{a_0, a_1, \ldots, a_i\}$ perform the algorithm in Fig. 5 at every simulation step.

[5] By *constant*, we refer to the fact that the costs do not depend on the amount of effort, but the costs can be different for each agent or can change over time.

This Section contains two Subsections: in the first Subsection, we will use randomly generated data models to show that the answer relevance is correlated with the answer speed and that the distribution of ∂P in practice is similar to that expected from Fig. 3 (c). The second Subsection shows how QW is used as a collaborative filtering system.

```
Method Step
For each Received Answer
  If Own Question, Update result and Trust
  Else forward answer
       If first answer from sender, Update Trust
       If question objective achieved
              Deprogram messages
              If use stop messages, program stop messages
If I have a stop message
   Deprogram messages
   Forward stop message to my recipients
If I have a new Own question
     Select contacts in contact waves; Program question messages
For each received question
   If I received it before, ignore it
   Else
     If I can answer, Program answer message
     Select contacts in contact waves; Program question messages
If I have a programmed answer
  Send answer
  If question objective achieved
     Deprogram messages
     If use stop messages, program stop messages
Send programmed messages
```

Fig. 5. Agents step QW algorithm.

4.1 Knowledge exchanges dataset

In the knowledge exchanges (KE) dataset simulations, the data has been randomly generated. We configured the simulator with the following requirements:

- The expertise vector k of each agent has dimension 5. The value of $k_j \in [0,1]$ of an expertise vector $k = \{k_1, k_2, k_3, k_4, k_5\}$ means the expertise level in a domain j. In the initial conditions, k values are set randomly.
- At each step, each agent has a question probability of having its own question (Ψ). In our simulations, $\Psi = .05$.
- The interest vector $It=\{i_1, i_2, i_3, i_4, i_5\}$ has the same dimension as vector k. It denotes the probability that a question will belong to a domain j. $\sum_j i_j = 1$.
- Each execution consists of 2,000 simulation steps with 100 agents.
- The social network is represented by an undirected graph, although each connection has two different trust values. We used the social networks represented in Fig. 6. SN-a has 495 links, a clustering coefficient of .521, an assortativity of .0763, a diameter of 5, and an average shortest path length of 2.84. SN-b has 395 links, a clustering coefficient of .386, an assortativity of .0568, a diameter of 5, and an average shortest path length of 2.86.
- The initial trust from each agent to its acquaintances is .75 for each domain.
- We executed each configuration 20 times.
- The questioning agents rate the answers received at their ϑ value.
- Each query belongs only to one domain.

- In all three of the cases, we consider that the probability that an answer satisfies a question is p(ϑ)= $\vartheta/100$.

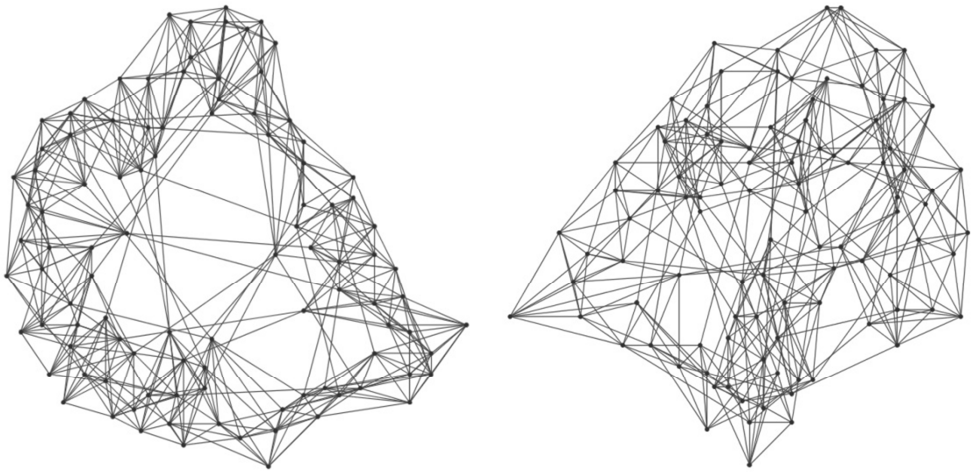


Fig. 6. Social networks used. The network on the left is SN-a, and the network on the right is SN-b.

4.1.1 Answer relevance and trust model

In these simulations, the answer quality (ϑ) is computed in three different ways:

- KM0: Use the knowledge model used by [36,37], expressed in Eq. 4, where I represents the implication, k is the expertise, and R is the external factors represented randomly, with the settings $w_i = .2$, $w_k = .7$, and $w_R = .1$
- KM1: Use the above knowledge model with the settings $w_i = .4$, $w_k = .5$, and $w_R = .1$

$$\vartheta = w_i I + w_k\, k + w_R R \tag{4}$$

- KM2: Use a knowledge model in which an agent with an expertise of e will produce a ϑ of e when the energy used to generate an answer is ∞, with $E=10$, $\vartheta = \frac{k}{2}$, with $E=50$, $\vartheta = .9k$ and with $E=0$, $\vartheta = 0$ (Eq. 5). The model is inspired by the ART-Testbed [12], in which the authors focus on the standard deviation produced instead of ϑ.

$$\vartheta = k\frac{E^2}{E^2+\frac{40}{9}E+\frac{500}{9}} \tag{5}$$

Agents evaluate their acquaintances for each domain considering the mean ϑ of the first answer provided. When agents have to determine the amount of effort that they will allocate to obtaining answers or generating an answer, the agents use the highest trust value.

4.1.2 Question delay and delaying own answer

Although there are many ways to delay questions and answers based on estimations of their relevance, we considered that there are two important points. The first one is that the answers that are more likely to be relevant should arrive quickly, and the time that users have to wait should be short. The second one is that answers that are

more likely to be not relevant have to be sent only when there are no other answers available, so their delay should increase considerably. With this purpose, in our simulations, we used Eq. 6, and the delay distribution based on trust can be seen in Fig. 7. With these function, with a $T_{i,j}$ of {.9,.8,.6} $t_{i,j}$ are ≃{2,5,15} respectively.

$$t_{i,j} = \frac{1}{3}\left(-150\ {T_{i,j}}^3 + 550\ {T_{i,j}}^2 - 700\ T_{i,j} + 300\right) \tag{6}$$

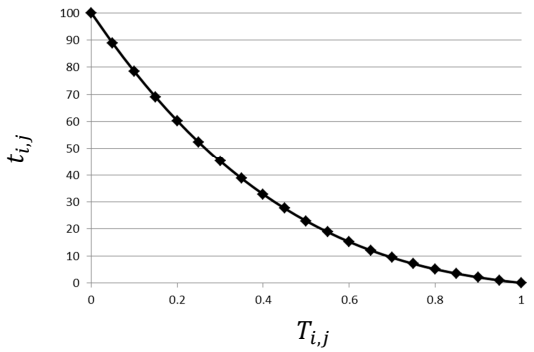


Fig.7. Question and answering delay ($t_{i,j}$) in simulation steps in function of trust ($T_{i,j}$).

4.1.3 Effort settings

In these simulations, we used 3 effort settings:

- ES0: The more trusted acquaintances receive more effort. An acquaintance a_j receives an amount of effort $E_{i,j}$ from the agent a_i based on the trust that a_i has in a_j $T_{i,j}$, the effort that a_i can transmit E_i and in comparison to the trust that a_i has of the other acquaintances. $E_{i,j} = E_i\ {T_{i,j}^2}\Big/{\sum_k T_{i,k}^2}$.
- ES1: The effort is distributed equally among the acquaintances.
- ES2: The effort used to try to answer a question is infinite.

The initial effort (E_0) in ES0 and ES1 is 10^5, and the effort costs are 2 for planning how to answer a question, including checking if would be possible to answer the question without forwarding it, selecting how to distribute the effort and sending messages. The costs do not include the cost of generating an answer, for which the maximum value will be 90 in ES0 and ES1.The effort loss due to *friction* is expressed in Eq. 7, were E_j is the effort that agent a_j will use to try to solve the question while $E_{i,j}$ is the effort transmitted by a_i to a_j (so, before applying friction). We used $\mu = .1$ because we assumed that the maximum loss due to friction for these simulations may be 10%, which will happen when the trust value is 0.

$$E_j = E_{i,j}\left(1 - \mu\ (1 - T_{i,j})\right) \tag{7}$$

4.1.4 Interferences

In the following simulations, we only considered destructive interferences. As a function of the effort used to obtain answers, each agent will continue seeking answers until the answers reach an estimated probability of answering the question correctly (σ). When an agent reaches σ, it will not put more effort into trying to obtain answers to the question, and it may send a stop message to the agents to which it delegated the question. When an agent receives a stop message, the agent will not put more effort into trying to answer the question. However, in both cases,

agents may continue forwarding answers received. We consider that σ must tend to 1 when E tends to ∞ and that σ must be 0 when $E = 0$. Furthermore, considering that $E_0 = 10^5$, we designed σ to be .9 when $E = E_0$, and σ when $E = E_0/2$. An approximated function is Eq. 8. In this equation, we used E' instead of E, where $E' = \varphi E$. The idea is to use φ to modify agents sensibility. $\varphi \in [0,\infty]$, values of φ lower than 1 suggest that the agents have a lower σ, and they decide to stop their contribution faster. In contrast, values of φ greater than 1 suggest that agents are more conservative, and they will need more answers to end their contribution.

$$\sigma = \frac{E'^2}{E'^2 - 2.78\ 10^4 E' + 3.89\ 10^9} \tag{8}$$

4.1.5 Evaluation and results

With the aim of evaluating the correlation of answer speediness with ϑ (- Delay), we added other correlations that would not be as easy to use in a real system and would be needed to re-rank the results:

- the distance to the answering agent (D)
- the reputation of the acquaintance that sends the answer (R)
- the transitive reputation of the path (TR). We computed this factor as the product of the reputation values of the path
- the reputation of the answerer (RA)
- the expertise of the answerer (Ex), directly based on the expertise value with which the answers are generated.

To compute these correlations, we used simulations without waves. The results are shown in Table 1. As expected, Ex has the best correlation with ϑ in the simulations with KM0, as Ex has a weight of .7 to ϑ. RA has results close to Ex, and the correlation with answer speed is high, but D and R provide no correlation in this group. There are no significant differences in the ES and networks for this KM.

Considering KM1, the results are similar to KM0, but there are weaker correlations for TR, Ex, RA and -Delay. Also, the results seem to indicate that there is a slightly higher correlation with Time and TR for the second network than the first.

Considering KM2, with finite effort (ES0 and ES1), D has some correlation, TR has the best correlation for network SN-a, while -Delay has the best correlation for network SN-b. In the case of infinite effort, the simulation has a similar behaviour as KM0 and KM1.

Additionally, Fig. 8. shows $\partial P(t)$ for each ES and KM; we can observe that it follows a similar pattern than the presented in Subsection 3.1 (Fig. 3. (c)), where after a setup time, the probability of finding a relevant answer at time instant t is inversely proportional to t.

Table 1.
QW results.

KM	E. S.	Network	D	R	TR	Ex	RA	-Delay
0	0	SN-a	-.028	.090	.446	.985	.984	.935
0	0	SN-b	-.010	.065	.447	.985	.984	.937
0	1	SN-a	.003	.098	.441	.985	.985	.935
0	1	SN-b	.001	.076	.445	.985	.985	.936
0	2	SN-a	.002	.098	.435	.985	.985	.936
0	2	SN-a	.001	.076	.442	.985	.985	.937
1	0	SN-a	-.010	.092	.341	.954	.955	.909
1	0	SN-b	.0	.078	.352	.953	.955	.912
1	1	SN-a	.003	.101	.346	.954	.956	.909
1	1	SN-b	.002	.080	.353	.953	.955	.912
1	2	SN-a	.003	.100	.341	.954	.956	.908
1	2	SN-b	.002	.080	.350	.953	.955	.912
2	0	SN-a	.540	.119	.776	.573	.599	.759
2	0	SN-b	.448	.133	.749	.670	.684	.811
2	1	SN-a	.627	.055	.668	.586	.617	.612
2	1	SN-b	.514	.056	.632	.699	.711	.736
2	2	SN-a	.002	.095	.488	1	.999	.953
2	2	SN-b	.001	.073	.488	1	.999	.953

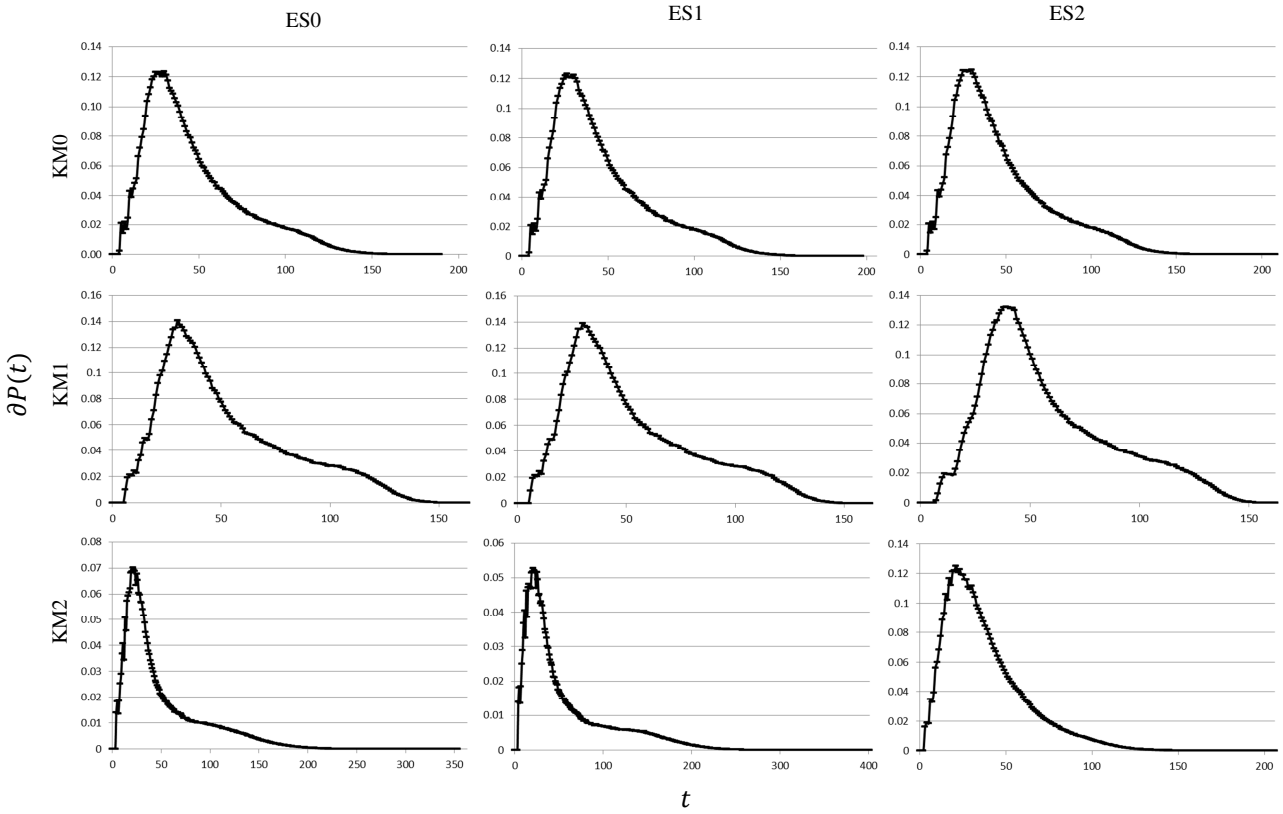


Fig. 8. Probability of finding a relevant answer at time t, $\partial P(t)$, as a function of time. The x axis shows the time (in simulation steps), and the y axis shows the probability.

Table 2.
Results using interferences.

	φ	-Delay	$\bar{\vartheta}$	m (10^6)	*a/q*
Without Interferences	No	.935	.569	4.77	97.35
With Interferences	4	.838	.671	3.304	42.958
	2	.804	.687	2.677	35.34
	1	.757	.714	2.586	30.731
	.5	.544	.792	1.726	17.19
	.25	.488	.804	1.43	14.34

To show the behaviour of interferences, we used KM0, SN-a and stop messages that will be sent when the probability of having a good answer is higher than τ (Eq. 8). The results can be seen in Table 2, where we can observe that when we increase φ, there are more answers per question (*a/q*), but the mean quality of the answers ($\bar{\vartheta}$) is decreased. In contrast, decreasing φ makes less (*a/q*), but $\bar{\vartheta}$ is increased. The correlation with -Delay is reduced when *a/q* is reduced. We see several explanations for this pattern. The first one, as can be seen in Fig. 7, is that the difference of the time delays for higher trust values is really small; a second one is that the R factor takes more importance when the expertise rankings are similar. The third explanation is that the distance to reach agents with higher expertise also affects the arrival time and has more importance when we only receive a few answers. This loss of correlation is not really important because the system provides the best answers, as can be seen for the increase of $\bar{\vartheta}$.

4.2 Experiments with real data extracted from a recommender systems dataset

It is not straightforward to use real data with questions answered and rated by users because on Q&A sites the same question is often repeated, and it is difficult to determine which questions are equivalent.

Some of the protocols referred to in this paper [40,41,44] used generated data especially suited for their experiments. For example, [40,44] set up a few categories and generated random profiles that were used for one [44] or more agents [40], while the queries were restricted to one category only. The answer suitability was based on user expertise evaluated at random [40] or on the similarity among user profiles [40]. In Sixearch [41], the authors based the content on real data, categories and answers (that consisted of real webpages), although they modelled the users randomly. In all of the cases, they generated random test users.

Using a real social network with real data for this kind of simulation is not straightforward because there is no available real data pool that contains all of the following points:

- A real social network in which the links among users indicate relationships (friends, colleagues, or family).
- A set of questions sent by the users of that social network and a set of similar or equivalent questions sent to one another, based on the idea that more than one user asks the same question.
- A set of answers for different questions provided by one user, which should be rated by a considerable number of users, not only a few.

Some exceptions might be Twitter and Quora. In the case of Twitter, and compared to other social networks, it might be easier to extract information, being it done through hoses (accessible through API), and several datasets have been published in the very recent years (e.g. [21]). In Twitter, there are less question answering interactions compared to other social networks yet there is vital exchange of comments and opinions among their users that are following each other. As said, Twitter is not a platform for question answering, though its open contact lists are very advantageous for our research, as we can create a replica of the social network behind and do experimentation. In the case of Quora, users also follow others and its usage is well focused on Q&A, and as an additional strong feature, Quora does suggest similar questions to guide the question preparation, yet it does not require the existence of any path in the social network from the questioner to the answerer (that is, potential answerers are accessible directly), and

finally we did not find any dataset with our required data (contact lists, questions and answers). In all, Quora did not fit our experimentation needs for our approach of social search research, yet we are open to extend our results to that platform soon in the future. Due to the lack of a true social question-answering environment useful for experimentation, we reused existing datasets used for information retrieval or recommender systems. Two of the most popular datasets are Text REtrieval Conference (TREC[6]) and movielens[7]. The first dataset contains a set of textual questions with a set of documents that may contain the answer. However, TREC datasets have no user ratings and no users who would encourage the reconstruction of a possible social network on it. In the case of movielens, the identified users rate a set of items. In this case, we can consider an item as a type of question and a rating as a type of answer, so there are many answers for each question. It is also possible to use recommendation techniques to evaluate answer suitability for a user, which equivalent to having the answers rated. The main item missing is a true social network, which could be generated randomly or using similarity functions [26]. For these experiments, we consider it better to generate the social network randomly, although considering that the network has social network properties. The main motivation is that in QW, we consider that people should have a network in which most of their contacts are people that they know as friends, family, colleagues, etc. An additional reason is that to identify similar profiles from unknown users, a centralized approach or some method to construct this similarity network that is out of the scope of this paper would be required.

Social search engines and recommendation systems (collaborative filtering or opinion-based filtering) are converging, as we consider that recommendation systems must be classified as SFS according to the classification of social search systems by [10], more concretely when we consider RWC recommender systems [40], which provide a personalized list of items. In SFS and RWC, users receive a list of items sorted with the aim to fit the user's taste. Therefore, using a dataset originally designed as a recommender-system dataset for social search simulations is appropriate for experimenting with social search because recommender systems can be seen as a subtype of social search (like an SFS, according to Chi [10]). Ratings gathered in these databases can be considered subjective answers to open questions like "What do you think of Gone With the Wind?", to which one can answer "I love it", "I hate it" or "It is all right, but not that much".

We decided to use the movielens dataset, consisting of 100,000 ratings for 1,682 movies produced by 943 users. The ratings are split into two sets: a training or base set with 80% of the ratings and a test set with the remaining 20%. With the movielens dataset, the only aspect missing is the social network, which is generated at random.

In the following Subsection, we will explain how we prepared the data for the simulations and, briefly, the social networks used. In Subsection 4.3.2, we will briefly explain collaborative filtering recommender systems, then we will explain how we used Question Waves with these data, and finally, we will show our results.

4.2.1 Simulation data

We used the data that we prepared to use with Asknext [38], it was as follows:

- We have removed the users who had fewer than 10 questions in the test set because many users had no questions.
- We have a queue of questions generated by distributing them between 1 and 1,000 steps of simulation.
- All of the simulations end at step 2,000.

As a result, the number of questions is 19,463, and there are 387 users. We generated 2 social networks, SN-1 and SN-2 (Fig. 9). SN-1 has 3,089 links, a clustering coefficient of .53, an assortativity of .0401, an average path length of 3.02 and a diameter of 5. SN-2 has 1,933 links, a clustering coefficient of .515, an assortativity of .0277 and average path length of 3.905 and a diameter of 7.

---

[6] http://trec.nist.gov/data/qa.html

[7] http://www.grouplens.org/node/73

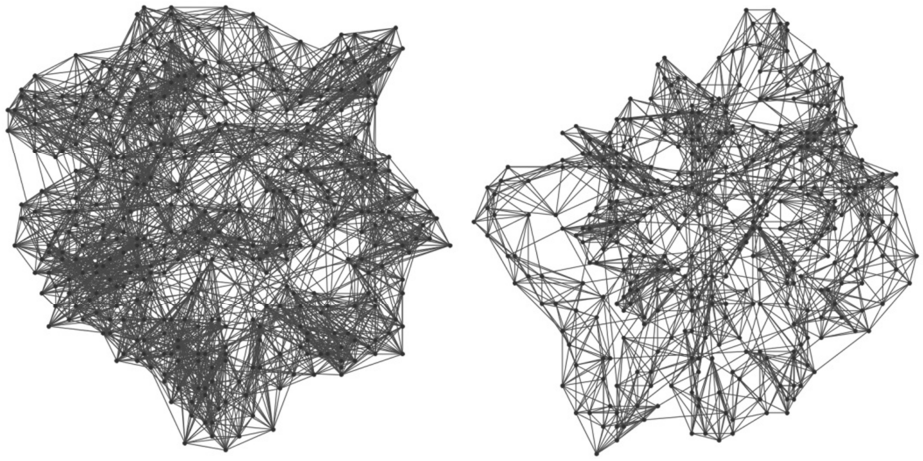


Fig. 9. Networks used in the simulations with movielens data. The network on the left is SN-1, and the network on the right is SN-2.

4.2.2 Collaborative filtering recommender systems

Collaborative filtering (CF) is used to recommend items that similar users have liked. CF is based on the social practice of exchange opinions. In the 1990s, GroupLens worked with this family of algorithms [20,30], leading a research line on recommender systems that expanded globally because of growing interest in the Internet. Usually, these systems rate items between 1 and 5. The most popular measures to estimate the similarity of pairs of users are the cosine [7,31] (Eq. 9) and Pearson's correlation coefficient (Eq. 10) [30,32].

$$u(t,v) = \frac{\sum_{i \in Im}(r_{t,i})(r_{v,i})}{\sqrt{\sum_{i \in Im}(r_{t,i})^2 \sum_{i \in I}(r_{v,i})^2}} \tag{9}$$

$$u(t,v) = \frac{\sum_{i \in Im}(r_{t,i} - \overline{r_t})(r_{v,i} - \overline{r_v})}{\sqrt{\sum_{i \in Im}(r_{t,i} - \overline{r_t})^2 \sum_{i \in I}(r_{v,i} - \overline{r_v})^2}} \tag{10}$$

Where:

- $u(t,v)$ is the similarity of owners of the agents a_t and a_v.
- $r_{t,i}$ is the rating given by owner of the agent a_t to item im_i.
- *Im* is the set of items.
- $\overline{r_t}$ is the mean of the ratings of owner of the agent a_t.

There are other measures, for example, the PIP proposed by Ahn [2] that is composed of three factors: proximity, impact, and popularity. A PIP score is computed for each co-rated item, and the similarity between two users is the sum of these PIP scores. This measure takes into account the distance between the co-rated items (proximity), gives more significance to ratings in the extremes of the rating range (impact) and also increases the importance of similar ratings that are far from the mean rating of the items (popularity).

The owner of the agent a_i rating item im_j ($r_{i,j}$) can be predicted with Eq. 11. In this equation, the constant *K* is used to match the prediction within the range of values.

$$r_{i,j} = \overline{r_i} + K \sum_{r_{k,j \mid k \neq i}} u_{i,k}\left(r_{k,j} - \overline{r_k}\right) \quad (11)$$

The main benefits of this method compared to another dominant recommendation technique, content-based filtering (CBF), are that items can be recommended without knowing their features, and users with hard-to-describe taste can still be classified with accurate recommendations. The drawbacks are that this method needs a lot of ratings and suffers from problems of cold starts and scattering data.

As alternative evaluation measures to CBF, the mean absolute error (MAE), mean squared error (MSE) or rooted MSE (RMSE) are well suited for simulated experiments and are good alternatives for measuring the suitability of answers or recommendations. The MAE, MSE, and RMSE show the mean error of the predictions; to achieve relevance on this kind of SFS, it is important to obtain low errors.

Once we have introduced the different metrics and concepts related to CF, in the following Subsection, we will explain how we simulated the network and show the results that were obtained.

4.2.3 Simulating a P2P Recommender System with Question Waves

In the literature, there are some examples of P2P recommender systems. The first ones, proposed by Walter et al. [40] and Montaner [26], were mentioned previously. Other ones based on BFS are [4,11,29]. The objective of this Subsection is to show the benefits of using QW in a more real scenario where it can be applied. To study the answer suitability offered by Asknext, we use an algorithm similar to traditional CF. How this proposal works is beyond the scope of this paper, except that because these CF systems are BFS based, Question Waves can be used with them.

To use Question Waves as a P2P recommender system, we determined the following:

- The question message includes the questioner profile.
- In the answer field of the answer message, we added two fields:
  - Accumulated vector: $\sum_{r_{k,j \mid k \neq i}} u_{i,k}\left(r_{k,j} - \overline{r_k}\right)$
  - Similarity vector: $\sum_{r_{k,j \mid k \neq i}} u_{i,k}$
- The prediction of $r_{i,j}$ is performed with Eq. 11 using $K = 1/(\, a^{1/2} \sum_{r_{k,j \mid k \neq i}} u_{i,k})$, where a is the number of answers.

When we applied CF, we obtained an MAE of .765 and an MSE of .958. It was possible to predict 19,210 items. We use these values as a reference target for our implementation of QW.

We modelled the QW as follows:

- In the simulations, we used Pearson's similarity measure between questioners and answerers to measure answer relevance.
- We added a domain for each possible questioner, and agents determine how good a mediator is at providing answers for a profile similar to a concrete questioner.
- The question time and the delay of the own answer are determined with a piecewise function (Eq. 12)
- The initial effort used by the questioners is $E_0 = 10^5$. The effort is distributed equally to all of the agents, and there is no cost of answering, but we maintained the same friction cost of Eq. 7, as in Subsection 4.1.3.
- When a mediator sends back an answer, the mediator will check if it has achieved its objective for that task; if so, it will send stop messages and will stop forwarding the questions related to that task. The objective for the task (σ) depends on the effort applied to obtain an answer ($E' = \varphi E_0$), computed with Eq. 13. φ has the same meaning as in Subsection 4.1. We used $\varphi = \{.25, .5625, 1, 1.5625, 2.25, 3.0625, 4\}$.

$$t_{i,j} = \begin{cases} 1\ if\ T_{i,j} > .9 \\ 2\ if\ .9 \geq T_{i,j} > .8 \\ 3\ if\ .8 \geq T_{i,j} > .7 \\ 4\ if\ .7 \geq T_{i,j} > .6 \\ 5\ if\ .6 \geq T_{i,j} > .5 \\ 10\ if\ .5 \geq T_{i,j} > .3 \\ 15\ if\ .3 \geq T_{i,j} > .1 \\ 20\ if\ .1 \geq T_{i,j} > -.3 \\ 25\ if\ -.3 \geq T_{i,j} > -.5 \\ 40\ if\ -.5 \geq T_{i,j} \end{cases} \quad (12)$$

$$\sigma = \frac{\sqrt{E'}}{\sqrt{E_0}} \quad (13)$$

4.2.4 Results

In Fig. 10, one can see the results of applying QW as a collaborative filtering system. Fig. 10 (a) and (b) show the MAE and MSE, respectively; the system provides slightly fewer errors using QW, and the results are getting closer of the reference values of CF (MAE of .765 and MSE of .958) along with the effort applied. Fig. 10 (c) shows the number of messages sent, where one can see that the configurations without waves (A1 and A2) have a higher slope. A2 starts using fewer messages than QW2, but due to the higher slope, the simulations have a similar number of messages at $\varphi = 1$.

Fig. 10 (d) shows the number of questions being answered. The variation between A and QW is not significant; in some cases, A does more predictions and vice versa.

Table 3.
Correlations of error prediction with time and similarity.

	-Delay$_{SN-1}$	-Delay$_{SN-2}$	Similarity
Mean correlation	-.5759	-.62	-.5934
St. dev.	.075	.08	0

We computed the correlation of the error produced by each answer message and the answer delay, and we obtained a correlation of 57.59% for SN-1 with a standard deviation of .075 and 62% for SN-2 with a standard deviation of .08, as can be checked in Table 3. We also computed the correlation of the error produced and users' similarity (similarity has been computed with Eq. 10) and we obtained -59.34%. We can claim that the correlation obtained between the answer relevance and answer speed (that is the inverse of the answer delay) is equivalent to the correlation of users' similarity and answer relevance. The similarity is the standard reference in a centralised CF recommender system. Our results of QW using partial information are comparable to a CF system that has information full access.

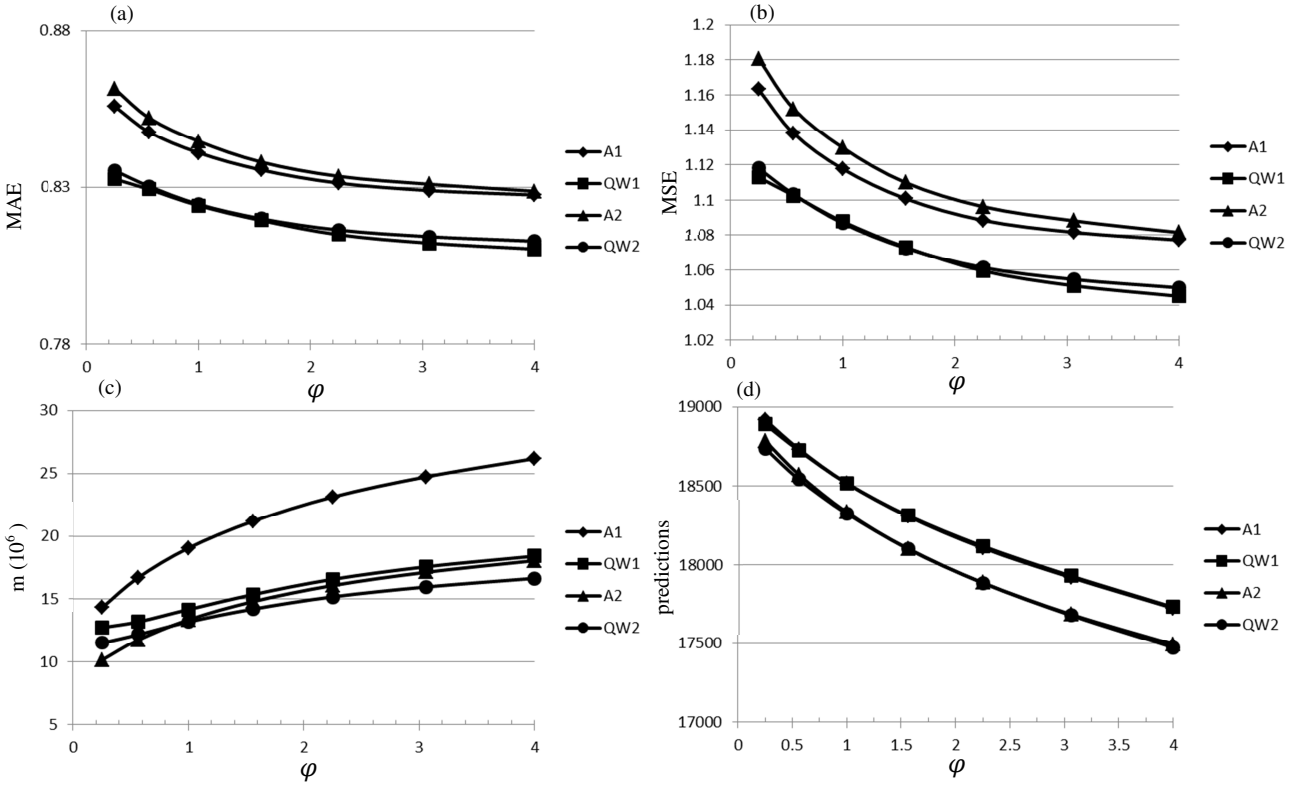


Fig. 10. Results of applying QW as a CF, comparing the results of using waves (QW1 and QW2) and not using waves (A1 and A2). (a) shows the MAE, (b) the MSE, (c) the number of messages and (d) the number of predictions done, all as a function of φ.

4.3 Practical Considerations

At the beginning of Section 3, we explained that the best behaviour of query-routing would be requesting to the minimum number of agents and obtaining the maximum or enough number of relevant answers. We perform now a set of experiments to measure the cost of obtaining relevant answers regarding the effort used and the number of generated messages, to test what configuration of QW is behaving best, and whether they are in all behaving better than the other algorithms of social search.

We compare 4 configurations:

- The first configuration (BFSe), is BFS with the improvement of effort to limit the question propagation.
- A second configuration (SMe) uses stop messages, as Asknext does; and it uses effort to limit the question propagation.
- A third configuration (QW) introduces the effects of answer and question messages delay but does not use stop messages.
- The fourth configuration (QW-SM) uses answer and question delay messages and uses stop messages.

The only input for every simulation is the initial effort per question, while the outputs are the following:

- Number of messages generated (m). In the introduction, we talked about how many agents are requested. Now, more precisely, the number of messages represents how many times agents are requested.
- Proportion of relevant answers obtained from relevant answers available (recall). Answers would be considered as relevant when their ϑ is above a threshold χ.
- Proportion of relevant answers obtained in order of arrival within a window of size n (precision). We considered size n=3.

The thresholds χ that we used are {.95, .85, .5} for the KE dataset, and {.95, .75, .45} for the movielens dataset creating the scenarios of {very high, high, and medium χ} for every configuration and dataset. The networks used are SN-a and SN-1. Furthermore, we ensured that all compared models had a same number of relevant answers available: in the case of the KE dataset, we computed the answer relevance only with the expertise, which is equivalent to Eq. 4 with $w_I = 0,\ w_k = 1,$ and $w_R = 0$.

Fig. 11 (a) and (b) shows the number of messages in function of the effort for KE dataset and movielens dataset respectively. In the KE dataset, we can see that QW and QW-SM start with higher number of messages than BFSe and SMe, but their convergence is slower; they converge to the highest number of messages because answers might follow longest paths than in BFSe and SMe. In the KE dataset, stop messages do not reduce the number of messages. In the case of the movielens dataset, the usage of stop messages shows some benefits, although in the case of QW-SM there are some effort values where the number of messages goes over the steady value of the number of messages. In this scenario, BFSe shows faster convergence, while SMe shows the slowest convergence.

Fig. 12 (a), (b) and (c), shows the number of relevant answers for the KE dataset with χ = {.95, .85, .5} respectively. We can observe than BFSe and SMe follow a same behaviour. In the cases of QW and QW-SM, in the initial values, QW-SM obtains few more answers, but with χ = {.90, .85} QW shows faster convergence.

Fig. 12 (d), (e) and (f), shows the number of relevant answers for the movielens dataset, with {.95, .75, .45} respectively. In this scenario, we can see that the behaviour of QW is equivalent to QW-SM, while BFSe is equivalent to SMe. Also, it is possible to observe that the QW and QW-SM converge faster, yet the convergence gap of QW (and QW-SM) and BFSe and SMe get narrower when χ is being decreased (from very high to medium values) so that in Fig. 12 (f) (for medium χ) the convergence behaviours are nearly equivalent for all configurations.

In Fig. 11 (b) and Fig. 12 (d, e, f) corresponding to the movielens dataset, we can clearly appreciate that the recall and the number of messages belong to a saturated growth model.

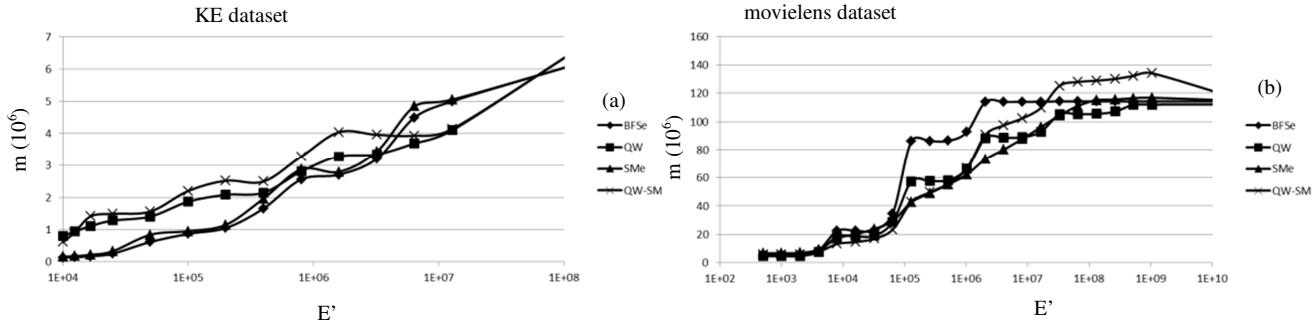


Fig 11. Number of messages in function of effort. For the KE dataset (a) and movielens dataset (b).

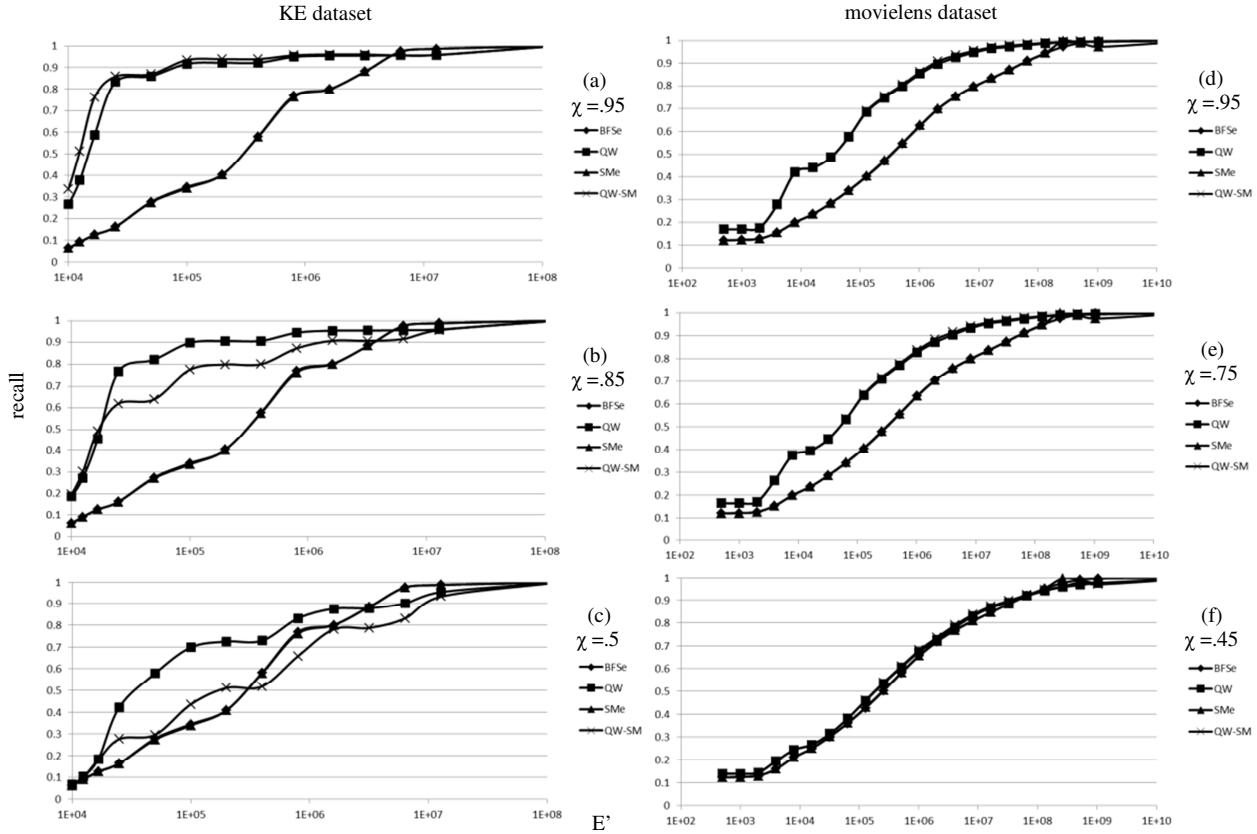


Fig. 12. Recall (y-axis) in function of effort (x-axis) for the KE dataset (a) (b) and (c) with χ of .95, .85 and .5 respectively, and for the movielens dataset (d), (e) and (f) with χ of .95, .75 and .45 respectively.

Considering that we want to obtain a good recall, let us consider it good a recall of at least 75%; the following configurations hit that figure of recalls:

- KE dataset: BFSe and SMe obtain it (the good recall) with an effort of $8*10^5$. QW-SM obtains it at efforts of $\{1.66*10^4, 10^5, 1.6*10^6\}$ with a χ of .95, .85 and .5 respectively, and QW obtains it with efforts of $2.5*10^4$ with a χ of .95 and .85, while it needs an effort of $8*10^5$ to achieve it with a χ of .5.
- Movielens dataset: BFSe and SMe obtain it with an effort of $4.1*10^6$ for the 3 values of χ, while QW and QW-SM depend of χ, they obtain it with efforts of $2.56*10^5$, $5.12*10^5$ and $4.1*10^6$ with a χ of .95, .75 and .45 respectively.

For better explaining the relation between the number of relevant answers found in comparison to the number of messages with the purpose of comparing the configurations, we approximate the relation between the recall and the number of messages as a saturated growth model (Eq. 14), where τ is the time constant and A the saturated (or final) value of y in function of t.

$$y\,(t) = A(1 - \mathrm{e}^{-\mathrm{t}/\tau}) \qquad (14)$$

This approximation works well for the recall in function of the number of messages, except for the tail and head of the function; the most relevant behaviour is approximated with Eq. 15, inspired by [27].

$$r\,(m) = 1 - \mathrm{e}^{-\mathrm{m}/\tau} \qquad (15)$$

Where:

- m is the number of messages.
- r is the *recall* estimated model. It must converge to 100%.
- τ is the constant that represents the recall model convergence, and it will be referred to as the *number of messages constant.* It has the property that $r = (1 - e^{-1}) \simeq .632$ at $m = \tau$ that indicates that with τ messages we obtain a recall of .632. Dividing the τ of the several configurations we can obtain how many times more messages a configuration needs to obtain a same recall r compared to the other configurations.

We used the method of Newton-Rapshon to obtain the τ of the regression model for Eq. 15 that minimizes the sum or squared error (S) of the real and the modelled recalls. The results of the τ of the 4 configurations (BFSe, QW, SMe, QW-SM) at the 3 scenarios (very high, high, and medium χ) and 2 datasets (KE and movielens) are shown in Table 4.

Table 4.
τ and sum of squared error (S) for each configuration in function of χ and dataset.

		KE dataset			movielens dataset		
		χ =.95	χ =.85	χ =.5	χ =.95	χ =.75	χ =.45
BFSe	$\tau\ (x10^6)$	1.84	1.85	1.83	111.11	109.26	101.59
	S	.00621	.00680	.00637	.0521	.0516	.0538
QW	$\tau\ (x10^6)$	1.17	1.35	2.22	35.25	40.12	66.16
	S	.222	.305	.282	.0306	.0316	.0317
SMe	$\tau\ (x10^6)$	2.11	2.13	2.11	71.46	65.9	70.4
	S	.0107	.0118	.0111	.00612	.00654	.00625
QW-SM	$\tau\ (x10^6)$	1.03	1.76	3.8	29.3	33.56	56.73
	S	.0483	.0423	.0686	.0246	.0224	.0199

The models corresponding τ of Table 4 are shown on Fig. 13.

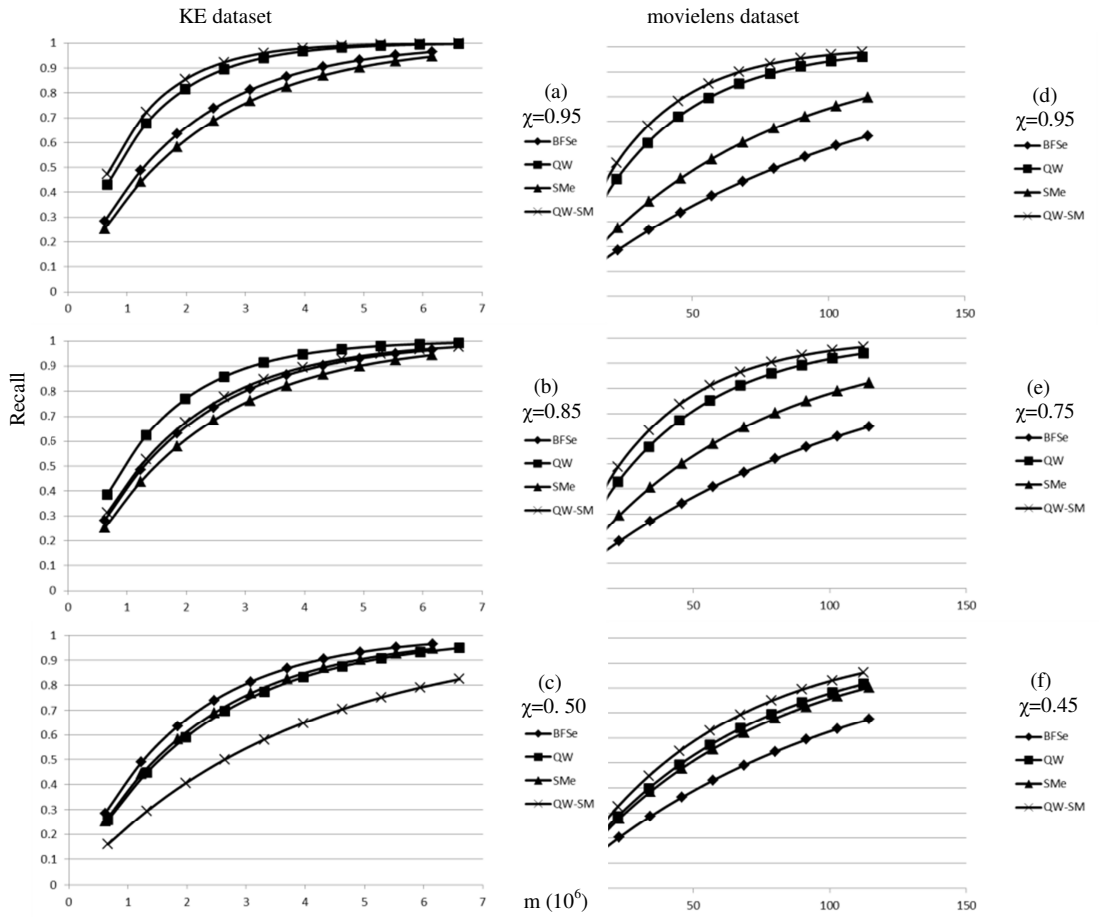


Fig. 13. Modelled recall (y-axis) in function of number of messages (x-axis). For KE dataset ((a), (b) and (c)) and movielens dataset ((c), (d) and (e)), with different χ values.

In Table 5 we show the quotient q of the τ of pairs of configurations. Quotient $q = \tau_i/\tau_j$ means that configuration i hits a same recall in q times the number of messages of configuration j, thus the lower q the lower number of messages are necessary for configuration i to converge compared to configuration j. We can observe that, in the KE dataset scenarios of χ={.95, .85}, QW needs the least number of messages to provide a same number of relevant answers (that is it has the lowest τ and the lowest quotient q), and that SMe is the configuration that needs to use more messages to provide the same number of relevant answers; in the other hand with χ=.85 BFS needs fewer messages to provide the same number of relevant answers.

Observing the outputs of the movielens dataset, we can observe that with all χ values the quotient of τ (QW-SM) divided by the τ of the other configurations is the lowest, thus QW-SM requires lesser messages to provide a same number of relevant answers than the other configurations. However, this advantage diminishes when the χ is decreased; this is due to the effect of σ, because in QW the answers arrive sorted by relevance and the agents decide that they do not need more answers sooner that in consequence the agents are satisfied with lower recall.

Table 5.
Quotient between the τ of the different configurations in function of the χ and dataset.

KE dataset						movielens dataset					
	quotient	**BFSe**	**QW**	**SMe**	**QW-SM**		***quotient***	**BFSe**	**QW**	**SMe**	**QW-SM**
	BFSe	1	1.57	.871	1.79		**BFSe**	1	3.15	1.56	3.79
χ =.95	**QW**	.637	1	.555	1.14	χ =.95	**QW**	.317	1	.493	1.2
	SMe	1.15	1.8	1	2.06		**SMe**	.643	2.03	1	2.44
	QW-SM	.56	.877	.487	1		**QW-SM**	.264	.831	.410	1
	quotient	**BFSe**	**QW**	**SMe**	**QW-SM**		***quotient***	**BFSe**	**QW**	**SMe**	**QW-SM**
	BFSe	1	1.37	.87	1.05		**BFSe**	1	2.72	1.55	3.26
χ =.85	**QW**	.732	1	.637	.771	χ =.75	**QW**	.367	1	.57	1.2
	SMe	1.15	1.57	1	1.21		**SMe**	.644	1.76	1	2.1
	QW-SM	.949	1.3	.825	1		**QW-SM**	.307	.837	.477	1
	quotient	**BFSe**	**QW**	**SMe**	**QW-SM**		***quotient***	**BFSe**	**QW**	**SMe**	**QW-SM**
	BFSe	1	.827	.869	.483		**BFSe**	1	1.54	1.54	1.79
χ =.5	**QW**	1.21	1	1.05	.584	χ =.45	**QW**	.651	1	1	1.17
	SMe	1.15	0.951	1	.556		**SMe**	.649	1	1	1.16
	QW-SM	2.07	1.712	1.8	1		**QW-SM**	.558	.857	.861	1

A new measure has to be analysed for engineering reasons: Precision. It is easier and more practical to be measured than recall. Recall is in fact only measurable in simulation. Precision is measured as the fulfilment out of a window of answers. We took a window size of 3 answers. The results obtained shown in Fig. 14; (a) and (b) are the proportion for the KE dataset with χ={.95, .85} respectively while (c) and (d) are the values for the movielens dataset with χ={.95,.45} respectively. In all cases, QW and QW-SM show lower quotient than BFSe and SMe; the explanation is that with BFSe and SMe the answers do not come sorted by relevance, as it does happen with QW and QW-SM. In the KE dataset, we can see that stop messages help to increase this proportion while using QW, although that the convergence value is the same.

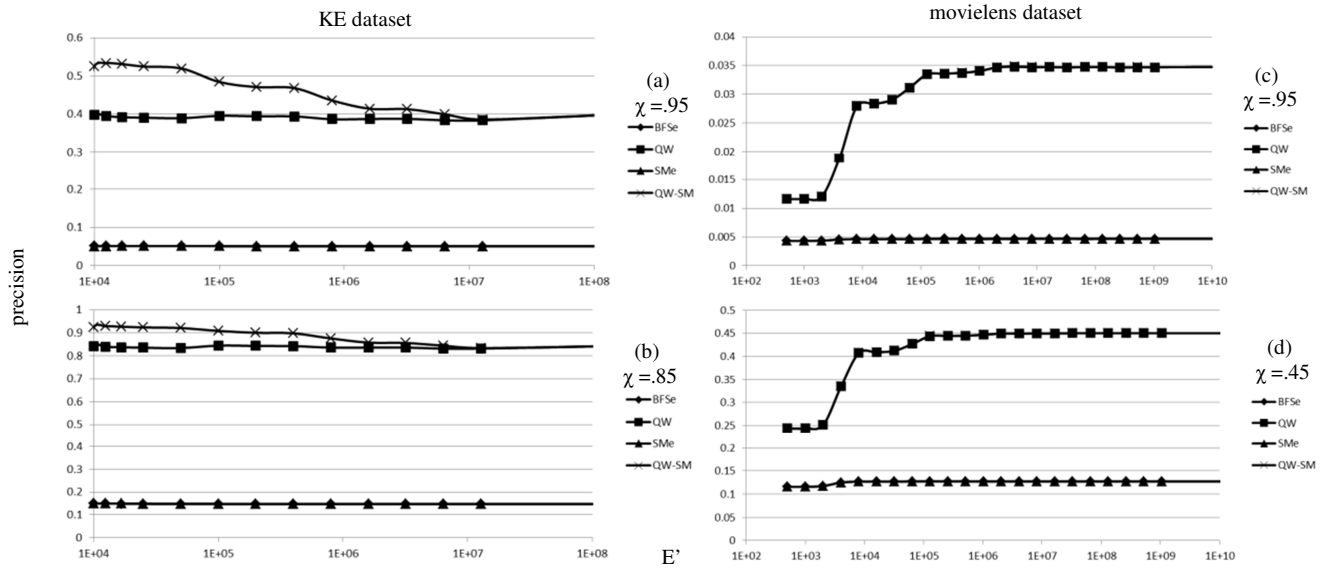


Fig. 14. Precision (in a window size of n=3) in function of effort for KE dataset ((a) and (b), with χ=.95 and χ=.85 respectively) and movielens dataset ((c) and (d) with a χ=.95 and χ=.45 respectively).

Considering Fig. 14, the best performance of KE dataset is achieved by QW-SM with an effort of 10^4, while in the movielens dataset it is achieved by QW and QW-SM with an effort of 10^5 when χ is .45 and with and effort of $4*10^6$ when χ is .95.

QW and QW-SM results can be modelled using Eq. 14 resulting into a τ (which is the *effort constant*) of $2.77*10^3$ when χ is .45 and a τ of $5.35*10^3$ for χ is .95. While the precision obtained by algorithms as BFSe and SMe does not depend on E', QW and QW-SM increase the precision obtained when the effort is increased until saturation of the precision is gotten. Furthermore, it is important to consider that for χ=.95 there are only 4,006 relevant answers in total available for the 19,463 questions; in case of χ=.45 there are 105,514 relevant answers available in total.

5. Conclusions and Future Work

Although Travers and Milgram [34] showed that it is feasible to reach a node using unicast query-routing algorithms, there is high probability that a message may be dropped before the message reaches its destination. Banerjee and Basu [3] developed a probability model that takes into account the probability that a message is answered using unicast query routing, considering that each node has enough expertise to answer the question but also some probability of dropping the question. Several P2P systems use multicast query routing, in which a question is sent to many contacts instead of sending it to one acquaintance at a time. Although the probability of obtaining an answer is increased with multicast query routing, there are more nodes requested, and as a consequence, system scalability can be threatened.

Several state-of-the-art techniques have been proposed to reduce the number of messages generated by multicast query-routing algorithms. For example, the usage of time to live (TTL) limits the number of hops for a given question. Recently, in [38], we proposed Asknext, an agent protocol for social search that uses stop messages. Using stop messages allows flooding to be stopped at the same number of hops at which the answer that satisfies the question is found. With these two techniques (TTL and stop messages), the best answer that is present somewhere in the network is not found due to being out of the distance or time range set in the search process.

In this paper, we presented Question Waves (QW). QW generates several attempts (waves) to find the information needed. Initially, agents send the question to their acquaintances with the highest probability of providing a good answer, i.e., a subset of their contact list. If, after some time, there is no satisfying answer, then the agents send the request to the next best contacts on the list. With QW, the answer relevance is correlated with the answer arrival order. This feature is key for stopping the search process with a lower probability of losing answers that are more relevant than the received ones. Furthermore, the process can save user's time because users can read the answers as they come, and the first answers are the most likely to be the relevant ones. Finally, the number of messages generated by QW is fewer than those generated by other methods, such as Asknext, yet even better relevance is achieved.

With QW inspired from physical waves, we used the ideas of *effort* to answer a question, effort distribution among acquaintances, effort that is dissipated when it is propagated to other agents due to friction, different speeds to send question messages to several acquaintances, and constructive and destructive interferences, which modify the effort that an agent uses to search for answers.

These ideas can be implemented with continuous or piecewise functions. We implemented and used them in two types of simulations. The first type of simulation consisted of using randomly generated knowledge bases (KE dataset) in which all of the agents were able to answer all of the questions but with different relevance due to their expertise, as a type of social question answering. In this set of simulations, we used 2 social networks with 100 agents, 495 and 395 links, .521 and .386 clustering coefficients, and .0736 and .0568 assortativity coefficients, respectively. The objective of this type of simulation was to show that it is possible to obtain answers ranked by their arrival time by using only local information (as in a P2P environment). We obtained as a best correlation .953, which is a high value for using local information; this value is clearly higher than the correlation of answer quality with the reputation of the mediator or the transitive reputation, which were the highest reputations of .133 and .776, respectively. A second objective was to determine the answer relevance when the search process was stopped. When a search is stopped with fewer answers, the correlation with time is lower, but the mean answer quality is higher. We consider the trust model we

used in this experiment as the main reason for this decrease of correlation coefficients; there is not enough discrimination between answerers of high expertise and the usage of random values to compute the answer relevance, which may affect when they have similar expertise values. This set of simulations can be seen as the application of QW in a social answering system. In summary, the highest correlation is achieved by using QW with long answering sets, although even in small answering sets, the correlation is high enough to consider that the first answers are the most relevant.

The second type of simulation was in the context of recommender systems and consisted of using data from movielens, a highly reputable and widely used movie-recommender dataset. The contribution of this type of simulation is a step forward with real data in the experimentation with the aim of showing that the method can be applicable in a real domain: recommender systems can be seen as a kind of social search system, concretely a social feedback system [10]. The movielens dataset does not include a social network; thus, with the aim to simulate QW, we created 2 social networks of 387 agents out of the movielens dataset, with 3,089 and 1,933 links each one and the following features of every network: .53 and .515 clustering coefficients and .0401 and .0277 assortativity coefficients, respectively. The objective of these simulations was to show that the QW method can generate a smaller number of messages than other algorithms applied in P2P social networks, and the answer quality is even higher. Because we are using data from a recommender system, we understand that the answers of higher quality are those with lower prediction errors (MAE and MSE). The results demonstrate that the system uses fewer messages than Asknext, and the answer quality is higher. We also compared the correlation of the answer quality with the arrival time, and it was comparable with the correlation of the answer quality with answerer similarity (approximately .6), on which the agents' behaviour is based. This set of simulations can be considered an appropriate application of QW in a social feedback system.

Finally, in Subsection 4.3 we studied the recall and the precision in a window of the 3 first answers per question, in function of the initial effort. The simulation results show that to achieve a target recall, QW needs less effort than BFSe (BFS with usage of effort) and SMe (using SM with effort); this behaviour is accentuated for more strict classifications of relevant items. We modelled the relation of recall and number of messages as a saturated growth model, obtaining that QW-SM needs approximately 3 times less messages to obtain the same recall than BFSe and approximately 2 times less messages than SMe (for χ =.75 or higher with the movielens dataset). About precision, considering the first three answers per question, in movielens dataset, considering QW and QW-SM reach 3.5 times higher precision than BFSe and SMe; and at the initial values ($E'=10^3$) the precision is at least 2 times higher with QW.

In future efforts, we think that it would be interesting to study the effects of small-world network topology characteristics in the scalability and correlation (or correctness) of answers because these factors can have a considerable impact and may affect the design of functions. Extracting which network features contribute to higher scalability, speediness of relevant answers or higher correlation with answer arrival and answer relevance can help to create local methods to modify the network dynamically to obtain these objectives. We also believe that allowing agents to modify their contact list would improve the correlations because the agents will be able to correct some shortcomings of the arrival order of the answers. The trust model is the base of QW, and studying the effect of different trust models can be interesting. A better understanding of the effects of trust models may help to implement an adaptable delay of the question time. Because the agents are autonomous, each one can have different implementations of the functions of the model and can adapt them based on interactions, and an interesting future work would be to study the behaviour of agents with heterogeneous behaviours. Also, studying question-message propagation in an epidemic model could be interesting. Additionally, once the benefits of the algorithm have been shown, it is needed to use a more complete and real dataset that include natural language processing, a dataset like that can be extracted from Quora, Twitter, etc. Finally, it would be interesting compare QW results with traditional search systems as search engines and Q&A portals.

6. Acknowledgements

This research is funded by the EU project Num. 238887 (iSAC6+), the IPT-430000-2010-13 project (SAKE), the EU DURAFILE num. 605356, FP7-SME-2013, BSG-SME (Research for SMEs) Innovative Digital Preservation using Social Search in Agent Environments, TIN2010-17903 *Comparative approaches to the implementation of intelligent agents in digital preservation from a perspective of the automation of social networks*, the Universitat de Girona research grant BR09/10 awarded to Albert Trias, and the AGAUR grant for the CSI-ref.2009SGR-1202.

References

[1] E. Agichtein and Y. Liu, Modeling Information-Seeker Satisfaction in Community Question Answering. ACM Transactions on Knowledge Discovery from Data 3(2) (2009). doi:10.1145/1514888.1514893

[2] H.J. Ahn, A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem, Information Sciences 178 (2008) 37-51. doi:10.1016/j.ins.2007.07.024

[3] A. Banerjee, S. Basu, A Social Query Model for Descentralized Search, In: Proc. 2nd ACM Workshop on Social Network Mining and Analysis, SNAKDD-08, Las Vegas, 2008.

[4] R. Baraglia, M. Mordacchini, P. Dazzi, L. Ricci, A P2P REcommender System based on Gossip Overlays (PREGO), In: Proc. 10th Int. Conf. Computer and Information Technology, CIT, 2010, pp. 83-90. doi:10.1109/CIT.2010.502

[5] M.K. Bergman, White Paper: The deep web: surfacing hidden value, The journal of electronic publishing 7(1) 2001 doi:10.3998/3336451.0007.104

[6] J. Bian, Y. Liu, E. Agichtein, H. Zha, Finding the right facts in the crowd: factoid question answering over social media, In: Proc. Int. Conf World Wide Web, WWW'08, New York, 2008, pp. 467-476.

[7] J.S. Breese, D. Heckerman, C. Kadie, Empirical analysis of predictive algorithms for collaborative filtering, In: Proc. 14th Conf. Uncertainty in Artificial Intelligence, Madison, WI, 1998, pp. 43-52.

[8] S. Buchegger, A. Datta, A case for P2P infrastructure for social networks - opportunities & challenges, In: Proc. 6th Int. Conf. Wireless On-Demand Network Systems and Services, Utah, 2009, pp. 161–168.

[9] E. Bulut, J. Wang, B.K Szymanski. Cost Effective Multi-Period Spraying for Routing in Delay Tolerant Networks. IEEE/ACM Transactions on Networking, 18(5):1530-43. 2010

[10] E.H. Chi, Information seeking can be social, Computer 42 (3) (2009) 42–46.

[11] J. Golbeck, Generating predictive movie recommendations from trust in social networks, In: K. Stølen, W.H. Winsboroughs, F. Martinelli, F. Massacci (Eds.), Proc. of the 4th Int. Conf. on Trust Management, iTrust 2006, LNCS 3986, 2006, pp. 93–104.

[12] M. Gomez, J. Sabater-Mir, J. Carbo, G. Muller, Improving the ART-Testbed, thoughts and reflections, In: Proc. Workshop on Competitive agents in: Agent Reputation and Trust Testbed, Salamanca, 2008, pp. 1-15.

[13] L. Guo. E. Tan, S. Chen, X. Zhang, Y.E. Zhao, Analyzing patterns of user content generation in online social networks, In: Proc. 15th ACM Int. Conf. Knowledge Discovery and Data Mining, KDD'09, 2009, pp. 369-378.

[14] M. Hackworth. Waves- Transverse and Longitudinal Waves, http://www.physics.isu.edu/~hackmart/waves100.PDF

[15] F.M. Harper, D. Moy, J.A. Konstan, Facts or friends?: distinguishing informational and conversational questions in social Q&A sites, in: D.R. Olsen. Jr., R.B. Arthur, K. Hinckley, M.R. Morris, S.E. Hudson, S. Greenberg (Eds.), Proc. 27th Int. Conf. Human Factors in computing systems, CHI'09, New York, 2009, pp. 759–768.

[16] J. Hendler, Avoiding another AI winter, IEEE Intelligent Systems 23(2) (2008) 2–4.

[17] D. Horowitz, S.D. Kamvar, The anatomy of a large-scale social search engine, In: Proc. 19th Int. Conf. World Wide Web, New York, 2010, pp. 431–440.

[18] K.K. Ikkink, T. van Tilburg, Broken ties: reciprocity and other factors affecting the termination of older adults' relationships, Social Networks 21 (1999) 131-146.

[19] P. Jurczyk, E. Agichtein, Discovering authorities in question answer communities by using link analysis, In: Proc. 16th ACM Conf Information and Knowledge Management, CIKM'07, New York, 2007, pp. 919-922. doi:10.1145/1321440.1321575

[20] J.A. Konstan, N. Kapoor, S.M. McNee, J.T. Butler, Techlens: Exploring the use of recommenders to support users of digital libraries. Task Force Meeting, Phoenix, AZ. 2005. Available at <http://www.grouplens.org/papers/pdf/CNI-TechLens-Final.pdf>.

[21] H. Kwak, C. Lee, H. Park, S. Moon. What is Twitter a social network or a new media? In: Proc. 19th International Conference on World Wide Web. pp. 591-600. 2010 Doi:10.1145/1772690.1772751

[22] Q. Liu, E. Agichtein, Modeling Answering Behavior in Collaborative Question Answering Systems, In: P. Clough et al. (Eds.), Proc. 33rd European Conf. Advances in Information Retrieval, ECIR 2011, LNCS 6611, 2011, pp. 67-79.

[23] X. Liu, W.B. Croft, M. Koll, Finding experts in community-based question-answering services, In: Proc. 14th ACM Int. Conf. Information and Knowledge Management, CIKM'05, New York, 2005, pp. 315-316. doi:10.1145/1099554.1099644

[24] M. Luck, P. McBurney, O. Shehory, S. Willmott, Agent technology: computing as interaction roadmap for agent-based computing. A roadmap for agent based computing, AgentLink 2005, ISBN 085432 845 9.
[25] E. Michlmayr, A. Pany, G. Kappel, Using Taxonomies for Content-based Routing with Ants, Journal of Computer Networks 51 (2007) 4514-4528.
[26] M. Montaner, Collaborative Recommender Agents. State-of-the-art and New Techniques. Lambert Academic Publishing. Saarbrücken, 2011. ISBN 978-3-8383-5426-2.
[27] A. Moreno, J. Ll. de la Rosa, B.K. Szymanski, J.M. Bárcenas. Reward System for Completing FAQ, Frontiers in Artificial Intelligence and Applications – AI Research & Development ISSN 0922-6389, Vol: 202, pp: 361-370, Nov 2009, IOS Press
[28] B.A. Nardi, S. Whittaker, H. Schwarz, It's not what you know, it's who you know: Work in the information age, First Monday 5 (5) (2000).
[29] G. Pitsilis, L. Marshall, A Trust-enabled P2P Recommender System, In: Proc. 15th Int. Workshop Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE '06, 2006, pp. 59-64, doi: 10.1109/WETICE.2006.12
[30] P. Resnick, N. Iakovou, M. Sushak, P. Bergstrom, J. Riedl. GroupLens: An open architecture for collaborative filtering of netnews, In: Proc. Conf. Computer Supported Cooperative Work, CSCW'94, 1994, pp.175-186.
[31] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based Collaborative Filtering Recommendation Algorithms, In: Proc. 10th Int. Conf. World Wide Web, WWW'01, 2001, pp. 286-295.
[32] U. Shardanand, P. Maes. Social information filtering: Algorithms for automating 'word of mouth', In: Proc. Conf. Human Factors in Computing Systems. CHI'95, 1995, pp. 210-127.
[33] B. Smyth, P. Briggs, M. Coyle, M.P. O'Mahony, Google shared a case-study in social search, In: G.J Houben et al. (Eds.), Proc. 17th Int. Conf. User Modeling, Adaptation, and Personalization, UMAP'09, LNCS 5535, 2009, pp. 283-294.
[34] J. Travers, S. Milgram, An Experimental Study of the Small World Problem. Sociometry 32 (4) (1969) 425-443.
[35] A. Trias, J. L. de la Rosa, B. Galitsky, G. Drobocsi, Automation of social networks with QA agents (extended abstract), In: van der Hoek, Kaminka, Lespérance, Luck and Sen (Eds.), Proc. 9th Int. Conf. Autonomous Agents and Multi-Agent Systems, AAMAS'10, Toronto, 2010, pp. 1437-1438.
[36] A. Trias, J.L. de la Rosa, Propagation of Question Waves by Means of Trust in a Social Network, In: H. Christiansen et al. (Eds.): Proc. 9th Int. Conf. Flexible Question Answering Systems, FQAS'11, 2011, LNAI 7022, 2011, pp. 186–197.
[37] A. Trias, Relevance and Speed of Answers: How Can MAS Answering Systems Deal With That?, Online Proc. 13th European Agent Systems Summer School, EASSS 2011, Girona, 2011, <http://eia.udg.edu/easss2011/resources/docs/paper9.pdf>.
[38] A. Trias i Mansilla, J.L. de la Rosa i Esteva, Asknext: An Agent Protocol for Social Search, Information Sciences 190 (2012) 144–161.
[39] T. Vu, A. Baid. Ask, Don't Search: A Social Help Engine for Online Social Network Mobile Users. In: Proc. 35th IEEE Sarnoff Symposium 2012.
[40] F.E. Walter, S. Battiston, F. Schweitzer, A model of a trust-based recommendation system on a social network, Autonomous Agents and Multi-Agent Systems 16 (1) (2008) 57–74.
[41] L.S. Wu, R. Akavipat, A. Maguitman, F. Menczer, Adaptive peer to peer social networks for distributed content based web search, in: Social Information Retrieval Systems: Emergent Technologies and Applications for Searching the Web Effectively, IGI Global, 2007, pp. 155-178.
[42] B. Yu, M.P. Singh, An agent-based approach to knowledge management, In: C. Nicholas, D. Grossman, K. Kalpakis, S. Qureshi, H. van Dissel, L. Seligman (Eds.), Proc. 11th Int. Conf. Information and Knowledge Management, CIKM'02, New York, 2002, pp. 642–644.
[43] B. Yu, M.P. Singh, Searching social networks, In: Proc. 2nd Int. joint Conf. Autonomous Agents and Multi-Agent Systems, AAMAS'03, Melbourne, 2003, pp. 65–72.
[44] B. Yu, M.P Singh, K. Sycara, Developing Trust in Large-Scale Peer-to-Peer Systems, In: Proc. 1st IEEE Symposium Multi-Agent Security and Survivability, 2004, pp. 1-10.
[45] Y. Zhou, G. Cong, B. Cui, C.S. Jensen, J. Yao: Routing questions to the right users in online communities, In: Proc. IEEE Int. Conf. Data Engineering, ICDE, 2009, pp. 700-711. doi:10.1109/ICDE.2009.44