Modeling and Performance Analysis of Service Discovery Strategies in Ad Hoc Networks
(Extended Abstract)

Michel Barbeau Evangelos Kranakis
School of Computer Science
Carleton University
Ottawa, Canada
Email: {barbeau,kranakis}@scs.carleton.ca

***Abstract*—We define post-query strategies: these are time dependent post-query protocols that are being executed in rounds. We show how existing service discovery protocols, like Jini, Bluetooth, and SLP fit within this framework. We propose several new post-query strategies that can be used for locating a service in an ad-hoc network. These include: non-adaptive, deterministic, randomized, memoryless, incremental, and greedy. We analyze and evaluate several of these strategies and present results of our simulations. Our methodology for evaluating the performance and efficiency of such service discovery protocols can be useful in the context of ad-hoc networks.**

Keywords: Ad Hoc Network, Service Location Protocol, Service Discovery Protocol, Post-Query Strategy.

I. Introduction

A network is said to be ad hoc when it is consisting of set of nodes with wireless network interfaces, it is self organizing (i.e., its topology is established and maintained automatically), it is short lived and does not involve routing across the Internet (i.e., the availability of a network infrastructure is not required) [8]. Service discovery can be defined as the problem of locating servers that fulfill requirements of clients [5]. A service discovery protocol is a set of message formats and rules that can be used by clients to find servers over a network. Most of the service discovery protocols use a request-reply communication model. They can, however, be used in several different ways that we term strategies.

In the context of service discovery in ad hoc networks, this paper addresses the following important question: how can the different service discovery strategies be characterized? In particular, how is it possible to obtain a characterization with respect to tradeoffs between expected cost and probability that a given client succeeds in finding a server?

In this paper, we introduce a novel framework which core is a probabilistic model in which service discovery strategies can be modeled and evaluated in a uniform manner. We show how strategies for discovering services in ad hoc network can be represented, analyzed and evaluated through simulations in this model.

Service discovery in ad hoc networks is reviewed in Section II. Our strategy representation model is introduced in Section III. Strategies represented in this mode are discussed in Section IV. Simulation results are presented in Section V. We conclude with Section VI.

II. Service Discovery in Ad Hoc Networks

Existing standards for dynamic configuration of nodes (see for example Guttman *et al* [4]) are inadequate for ad hoc networks (see Perkins [9]). A natural question posed by Perkins [8] is: *where do services reside?* The difficulty of answering this problem is compounded by the fact that in addition to the topological changes affecting ad hoc networks over time, it may not be possible either for service discovery to be inserted into a network layer (due to vertical design obstacles) or for a node to specify the exact service it wants in a network layer (see Perkins [9]).

Bluetooth is an ad hoc network technology for nodes that are very close to each other [1]. Bluetooth networks are called piconets or scatternets. A piconet is a set of two to seven interconnected Bluetooth devices. A scatternet is a set of interconnected piconets. Bluetooth defines the Service Discovery Application Profile [2] for finding services over Bluetooth networks. To find a service, a client sends a *ServiceSearchRequest* message to every device on the Bluetooth network, one after the other using unicast. They each respond with the *ServiceSearchResponse* message, which can be either positive of negative.

For the purposes of service discovery, Koodli and Perkins define extensions to ad hoc network routing protocols [6]. Their extensions together with a routing protocol can find services and routes to services at the same time. Service information is available at the same time as route information. The extensions can be combined with both proactive and reactive routing protocols. With proactive protocols, service information is included in packets about the network topology that are exchanged between routers. With reactive protocols, service discovery uses the same messages as the one defined for route discovery with extensions.

In light of the above service discovery protocols for ad hoc networks, we make the following assumption in the sequel of this paper: routes to services are discovered before or at the same time of services.

III. Strategy Model

A network is modeled as a graph $G = (N, L)$ where $N = \{1, 2, \ldots, n\}$ is a set of n nodes and $L \subseteq N \times N$ is a set of directed links. Whenever $(i, j) \in L$, then there is a communication path between the nodes i and j and i can talk to j. Given a set of nodes N, an ad hoc network is modeled as a discrete sequence of graphs $G_0, G_1, G_2, \ldots$. At time t, $G_t = (N_t, L_t)$, with $N_t \subseteq N$, modeling the set of active nodes, and $L_t \subseteq N_t \times N_t$, modeling the set of active links. Whenever $(i, j) \in L_t$, then there is a communication path between the nodes i and j and i can talk to j at time t. This model is similar to the one defined in [3]. To model the uncertainty in ad hoc networks, we consider the probability p, with $0 \leq p \leq 1$, that there is a communication path between two given nodes. Let $1, 2, \ldots, k$ be the k different types of services offered in the network and let $K = \{1, 2, \ldots, k\}$.

Service discovery protocols are abstracted as post-query protocols, which are time dependent protocols executed in rounds modeling request-reply interactions between client and servers on a network. The post-query model described by Mullender and Vitányi [7] and Kranakis and Vitányi [10] refers only to traditional networks. In this paper, we adapt and extend their post-query model to the context of ad hoc networks.

At any given time a client wants to locate a service that has been posted by a server.[1] Servers post services and clients query nodes in order to locate the desired services. *Post-query protocols* are algorithms for posting and querying services. A client may request services of different types.

Definition 1: A *post-query protocol* is a pair of functions (P, Q). $P : N \to 2^{N \times K} : s \to P(s)$ is the *posting protocol*, and $Q : N \to 2^{N \times K} : c \to Q(c)$ is the *querying protocol*.[2]
Each server s posts (respectively, client c queries for) services in nodes of the set $P(s)$ (respectively, $Q(c)$) according to the following rules. Server s posts service i to node u if and only if $(u, i) \in P(s)$. Similarly, client c queries node u for service i if and only if $(u, i) \in Q(s)$. In the sequel, it will be useful to adopt the following notations. $N_P(s) = \{u \in N : (\exists i \in K)((u, i) \in P(s))\}$ be the set of nodes to which services are posted by server s. Let $K_P(u, s) = \{i \in K : (u, i) \in P(s)\}$ be the set of services posted to u by server s and

$$K_P(s) = \bigcup_{u \in N} K_P(u, s)$$

the set of all services posted by server s. We define in a similar manner the set $N_Q(c)$ of nodes queried by client c and the sets $K_Q(v, c)$ (respectively, $K_Q(c)$) of services requested by client c when querying node v (respectively, nodes of the network).

Several post-query protocols fall under the category of our model. Here we illustrate with some examples of simple protocols. A server s (respectively, client c) may not post (respectively, query) at all, i.e., $P(s) = \emptyset$ (respectively, $Q(c) = \emptyset$). A server s (respectively, client c) may post (respectively, query) all its services everywhere, i.e., $P(s) = N \times K$ (respectively, $Q(c) = N \times K$).

The cost of the posting and querying protocols is the number of postings and queryings made by all nodes, i.e.,

$$C(P) := \sum_{s=1}^{n} |N_P(s)| \text{ and } C(Q) := \sum_{c=1}^{n} |N_Q(c)|, \quad (1)$$

respectively. The cost of the post-query protocol (P, Q) is the sum of the costs of the posting and querying protocols, i.e.,

$$C(P, Q) = \frac{1}{n} \sum_{s,c=1}^{n} (|N_P(s)| + |N_Q(c)|). \quad (2)$$

Definition 2: A client succeeds in finding all services it wants if every service it wants has been posted by some server to a node that it queries.
In view of the above notation, $K_Q(c)$ is the set of services requested by client c. Therefore, success is assured if for all services $i \in K_Q(c)$ there exists a server s and a node u such that $(u, i) \in P(s)$ (i.e., service i has been posted at node u) and $(u, i) \in P(s)$ (i.e., client c requests service i from node u).

We extend this model to capture the unknown environment of an ad hoc network. We modify Definition 1 so that P, Q are random variables.

[1] Although in the application model we have in mind all nodes are treated as computationally identical, we will still find it convenient to use the client/server terminology in order to distinguish between nodes querying/posting for services, respectively.

[2] In this paper, we use 2^S to denote the set of all subsets of a set S.

Definition 3: A *post-query protocol* is a pair (P, Q) of functions. $P : N \to 2^{N\times K} : s \to P(s)$ is the *posting protocol*, and $Q : N \to 2^{N\times K} : c \to Q(c)$ is the *querying protocol* such that $P(s)$ and $Q(c)$ are random variables.
The *expected cost* of the posting and querying protocols is defined by

$$E[P] := \sum_{s=1}^{n} E[P(s)], \text{ and } E[Q] := \sum_{c=1}^{n} E[Q(c)]. \quad (3)$$

respectively, where

$$\begin{aligned} E[P(s)] &:= \sum_{i=1}^{n} i \cdot \Pr[|N_P(s)| = i], \text{ and} \\ E[Q(c)] &:= \sum_{c=1}^{n} i \cdot \Pr[|N_Q(c)| = i]. \end{aligned}$$

The *expected cost* of the post-query protocol (P, Q) is the sum of the expected costs of the posting and querying protocols, i.e.,

$$E(P, Q) := E(P)+E(Q) = \frac{1}{n} \sum_{s,c=1}^{n} (E[P(s)]+E[Q(c)]). \quad (4)$$

As given in Definition 3, we are interested in maximizing the probability that a given client $c \in N$ *succeeds* in finding a service. This can be expressed compactly as the probability

$$\Pr[\forall s (N_Q(c) \cap N_P(s) \neq \emptyset)]. \quad (5)$$

A lower bound on the expected cost $E(P, Q)$ can be derived easily as in the main result of [10]. Namely, if we define the random variables U_i as *the number of occurrences of the node i in an intersection $N_P(s) \cap N_Q(c)$* then $E(P, Q) \geq \frac{2}{n} \sum_{i=1}^{n} E[\sqrt{U_i}]$. Protocols matching this lower bound exist, e.g. based on the projective model (for more details see [10], [**?**]) however this is not very useful in the unknown environment of ad hoc networks.

Remark 1: In the context of ad hoc networks it is reasonable to assume that the post and query protocols $P(s), Q(c),\ s, c = 1, 2, \ldots, n$, are idependent and identically distributed random variables. A similar remark applies to the case of post-query strategies which are post-query protocols executed in rounds, although in the latter case a post-query protocol at a given round may depend on the protocol of the previous round.

In general, in the model with k types of services we are interested in maximizing the probability that a given client succeeds in obtaining all services, namely $p_{(P,Q)}(c) :=$ which is defined by

$$\Pr[\forall i \in K \exists s \in N (i \text{ is available in } N_Q(c) \cap N_P(s))]. \quad (6)$$

The dynamic changes affecting an ad hoc network indicate that a post-query protocol is not by itself sufficient to locate a service efficiently. For example, during the execution of a posting (respectively, querying) protocol nodes may not be available either because they are operating in a non-active mode or in an active mode but at a different frequency. To overcome this problem we consider post-query strategies. *Post-query strategies* are post-query protocols that are executed in a sequence of rounds, each round consisting of a post-query protocol.

Definition 4: A *post-query strategy* is a sequence $(P_1, Q_1), \ldots, (P_r, Q_r), \ldots, (P_R, Q_R)$ of post-query protocols executed in R rounds.
At each round r, the nodes of the system first post services they have available to other nodes of the system according to the *posting protocol* P_r and then query other nodes of the system according to a *querying protocol* Q_r. Thus, post-query strategies comprise a sequence of post-query protocols that adapt to changes in the network. Execution of the strategy is in R rounds, so that during each round $r \leq R$ a post-query protocol is executed. Rounds are necessary in order to adapt to topological changes over time in an ad hoc network.

For simplicity, from now on and for the rest of this paper we use the notation (P, Q) to denote a post-query strategy consisting of a sequence $(P_1, Q_1), (P_2, Q_2), \ldots, (P_R, Q_R)$ of post-query protocols. The quantity R is a parameter indicating an upper bound on the number of rounds within which clients and servers need to terminate execution of their respective strategies. We can extend the definition in Equation 4 to define the expected cost of the post-query strategy when executed in R rounds as the sum over all the rounds of the expected costs of its constituent post-query protocols, namely

$$E(P, Q) := \sum_{r=1}^{R} .E(P_r, Q_r). \quad (7)$$

In general, a given post-query strategy (P, Q) is executed by node s (s can be wither a client or a server) in at most R rounds or until a desired service service is located. The specific algorithm in detail is as follows.

Algorithm 1—**Post-Query Strategy (P,Q)** (s)**:**

1. **for** $r := 1$ **to** R or until service is located (which ever comes first) **do**
2. for all nodes u,
3. s posts service i at node u if and only if $(u, i) \in P_r(s)$
4. for all nodes u,
5. s queries for service j at node u if and only if $(u, j) \in Q_r(s)$

In round r, node s first posts services according to the posting set $P_r(s)$ and subsequently queries nodes for services according to the query set $Q_r(s)$. Each posting is followed by querying. The algorithm terminates either when it succeeds in finding the desired service or when $r := R$, which ever comes first.

In the model of Definition 3, we are interested in maximizing the probability that in the post-query protocol (P, Q) a given client $c \in N$ *succeeds* in finding a service (see Formula 5). In the case of a post-query strategy (P, Q), we are interested in minimizing the waiting time, i.e., the time until c succeeds in finding a service according to the post-query startegy (P, Q). This is given by the formula

$$W_{(P,Q)}(c) := \sum_{r=1}^{\infty} r \cdot p'_{(P_r,Q_r)}(c), \tag{8}$$

where $p'_{(P_r,Q_r)}(c)$, is the probability that client c succeeds in finding a service at exactly the rth round. The *maximum waiting time* is the maximum taken over all clients in order to acquire a service and is given by the formula

$$W_{(P,Q)} := \max_{c \in N} W_{(P,Q)}(c). \tag{9}$$

IV. Strategies

In this section we introduce the post-to-all, query-to-all strategy, uniform memoryless strategy, and incremental post-query strategy.

The *post-to-all, query-to-all* is a greedy strategy where all nodes post to all nodes, and all nodes query all nodes of the network. $N_P(s) = N_Q(c) = N$, for all $s, c \in N$. This strategy is non-adaptive and does not change with each round.

The (l, l')-post-query protocol works by following the post-to-l and query-l' rule, where $l, l' \leq n$ are positive integers. In detail, this means the following.

1) **Posting:** each server posts to a random set of l nodes all the services it has to offer.
2) **Querying:** each client queries a random set of l' nodes.

The (l, l')-post-query strategy consists of rounds of uniform and memoryless repetition of the (l, l')-post-query protocol. Analysis of this strategy is given in the appendix. The expected cost of the post-query protocol is $(l + l')n$ since each server posts to exactly l nodes and each client queries exactly l' nodes.

In such a protocol, it is easy to calculate the waiting time from the probability of success because it obeys the geometric distribution. Hence, the waiting time until client c succeeds in finding all services in this post-query protocol is equal to $1/p_{(P,Q)}(c)$.

The *incrmental post-query strategy* starts by posting and querying a small number of nodes in the first round and gradually increase the number of nodes they post to and query from. The advantage of this is to conserve valuable resources in an ad hoc network, e.g. power consumption.

The *incrmental* post-query strategy (P, Q) is a sequence $(P_r, Q_r), r = 1, 2, \ldots, R$ of post-query protocols such that (P_r, Q_r) is defined to be the (r, r)-post-query strategy, for all $r \leq R$. Two variants of this strategy are the *post-incremental* and *query-incremental* strategies. In the former, only the posting set is incremented, i.e., (P_r, Q_r) is defined to be the $(r, 1)$-post-query strategy, for all $r \leq R$, while in the latter case, only the querying set is incremented, i.e., (P_r, Q_r) is defined to be the $(1, r)$-post-query strategy, for all $r \leq R$.

In all the incremental strategies above, the probability of success in the r-th round is calculated as in the memoryless post-to-all, query-to-all strategy. However, the waiting times $W_{(P,Q)}(c)$ and $W_{(P,Q)}$ are calculated by using the formulas in Identities 8 and 9.

V. Simulation Results

Figure 1 shows the result of a simulation of a post-to-all query-to-all strategy. The figure depicts the results of a number of simulated sequences. For each sequence, the post-to-all strategy is applied by node 1 until a service, randomly chosen at the begining of the sequence, is found or a maximum number of attempts is reached (set to $R = 10$). Note that a strategy can run forever because the service queried by node v_1 is not posted by any server node. The number of nodes is set to $n = 25$, the number of services to $k = 10$ and 1000 sequences are generated.

The left part of Figure 1 depicts the distribution of successful sequences as a function of the number of required rounds, for four different values of the probability of a communication path p, i.e. $p = 0.1, 0.25, 0.5$ and 0.75. As expected, the probability to succeed within a low number of rounds increases as the number of communication paths and reachable nodes increase.

The right part of Figure 1 shows the distribution of all sequences (either successful or unsuccessful) as a function of their cost (defined in Equation 1).

Similar results can be obtained as well if we vary the number of nodes. That is, the probability to succeed within a low number of steps increases with the number of nodes because the number of offers increases as well.

The post-to-all query-to-all strategy represents an optimum in terms of maximizing the probability of succeding within a number of rounds. It is also the most costly in terms of using network resources. Hence, it will serves as a reference to compare other strategies.

Figure 2 shows the result of a simulation of the (l, l')-post-query strategy. In this simulation, the node v_1 applies a query-l' strategy. In other words, among the reachable nodes in $V = \{v_2, \ldots, v_n\}$, node v_1 queries a random subset of V of size at most l'. Each node in V is selected with uniform probability. We call the strategy post-to-all, query-l'. Table I shows that it is superior to the query-all, post-to-all strategy when $p \geq 0.5$.

Figure 3 shows the result of a simulation of post-to-all query-incremental strategy. Among the reachable nodes in $V = \{v_2, \ldots, v_n\}$, node v_1 queries a random subset of at most r nodes of V, with $r = 1, \ldots, 10$. Each node in V is selected with uniform probability.

A comparative summary of the results of our simulations is presented in Table I. From left to right, columns are probability of a communication path p, strategy, an estimator of the success rate, an estimator of the waiting time $\hat{W}_{(P,Q)}(c)$, and an estimator of the cost $\hat{E}(P,Q)$. In particular, it appears that post-query strategies are more helpful in dense ad hoc networks when the average degree of a node tends to be relatively high.

VI. Conclusion

In this paper we have defined post-query strategies as time dependent post-query protocols that are executed in rounds. We have created a general methodology for evaluating the performance and efficiency of service discovery protocols in ad hoc networks. Several interesting problems remain for further investigation. We believe it is an interesting mathematical challenge to refine our methodologies and propose optimal strategies. Our heuristics are under the assumption that all clients are employing the same probability distribution. What are appropriate heuristics when the number of network nodes is unknown and clients do not necessarily employ identical demand probability distributions? A more difficult problem is what strategies to consider when the demand distribution used is unknown.

VII. Acknowledgments

Research supported in part by NSERC (Natural Sciences and Engineering Research Council of Canada) and MITACS (Mathematics of Information Technology and Complex Systems) grants.

References

[1] Bluetooth, Specification of the Bluetooth System, Specification Volume 1, 2001.

[2] Bluetooth, Specification of the Bluetooth System, Specification Volume 2, 2001.

[3] A. Farago, V.R. Syrotiuk, MERIT: A Unified Framework for Routing Protocol Assessment in Mobile Ad Hoc Networks, ACM SIGMOBILE 7, pp. 53-60, 2001.

[4] E. Guttman, C. Perkins, J. Veizades, M. Dago, Service Location Protocol, Version 2, RFC 2608, IETF, 2000.

[5] J. Kempf and P. St. Pierre, Service Location Protocol for Enterprise Networks, Wiley, 1999.

[6] R. Koodli abd C.E. Perkins, Service Doscovery in On-demand Ad Hoc Networks, Internet-Draft, 2002

[7] S Mullender, P. Vitanyi, Distributed Match-Making, Algorithmica, Vol. 3, pp. 367-391, 1988.

[8] C. E. Perkins, editor, Ad Hoc Networking, Addison Wesley, 2001.

[9] C. E. Perkins, Is the Client Server Model Viable?, pages 353-354, 2001. Entry in [8].

[10] E. Kranakis, P. Vitanyi, A Note on Weighted Distributed Match-Making, in Mathematical Systems Theory, Vol 25, 123-140, 1992.

VIII. Appendix: Tables and Figures

p	Strategy	Success rate	Waiting time $\hat{W}_{(P,Q)}(c)$	Cost $\hat{E}(P,Q)$
0.1	Post-to-all Query-all	66 %	4.1	2.3
	Post-to-all Query-l'	40 %	4.6	2.3
	Post-to-all Query-incremental	61 %	4.9	2.3
0.25	Post-to-all Query-all	99.6 %	1.8	2.8
	Post-to-all Query-l'	97 %	3	2.6
	Post-to-all Query-incremental	99.6 %	3.2	2.5
0.5	Post-to-all Query-all	100 %	1.1	6.2
	Post-to-all Query-l'	100 %	1.3	3.9
	Post-to-all Query-incremental	100 %	2.7	2.7
0.75	Post-to-all Query-all	100 %	1.1	13.5
	Post-to-all Query-l'	100 %	1	6.9
	Post-to-all Query-incremental	100 %	2.7	2.6

TABLE I

Performance summary of the proposed protocols with $p = 0.1, 0.25, 0, 5$ and 0.75.

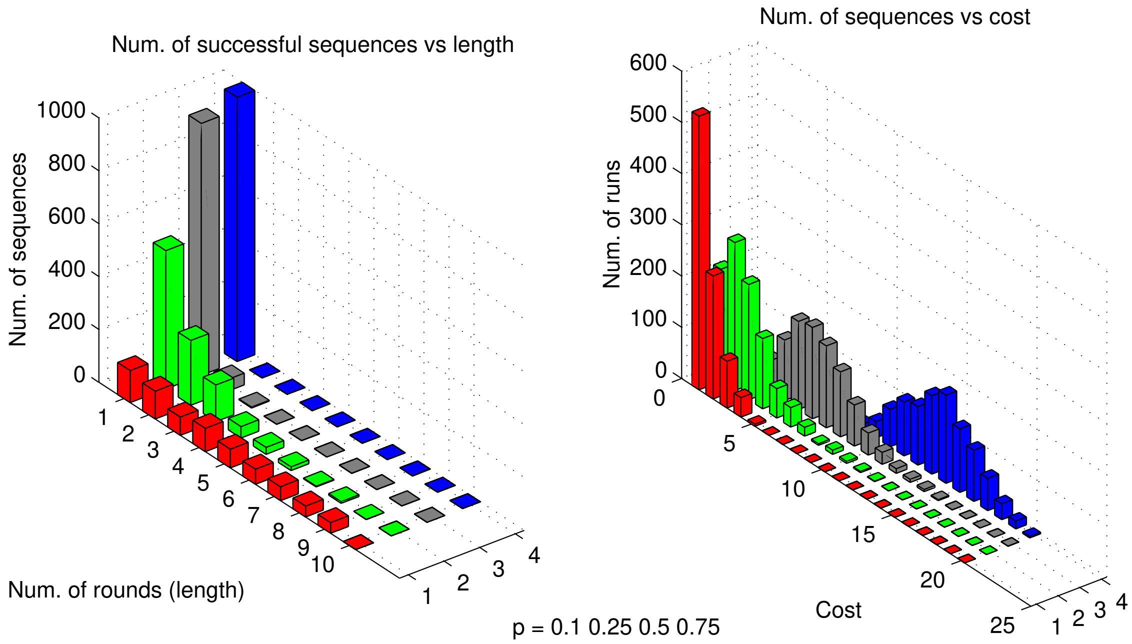

Fig. 1. Post-to-all query-to-all strategy with $p = 0.1, 0.25, 0.5, 0.75$.

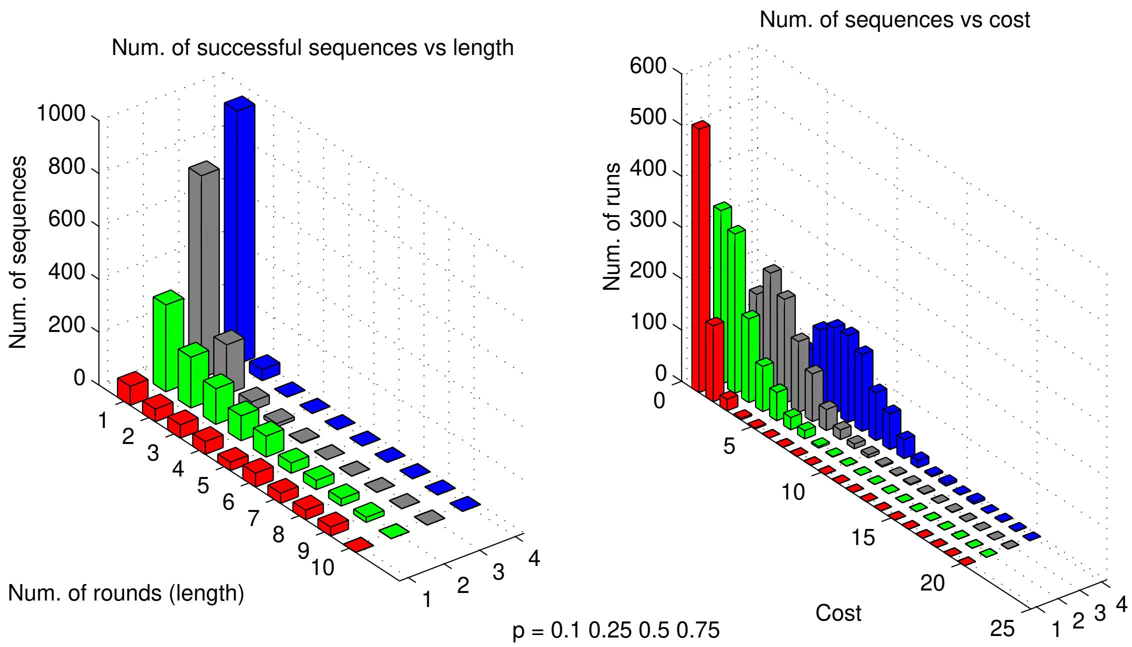

Fig. 2. Post-to-all query-l' strategy with $p = 0.1, 0.25, 0.5, 0.75$.

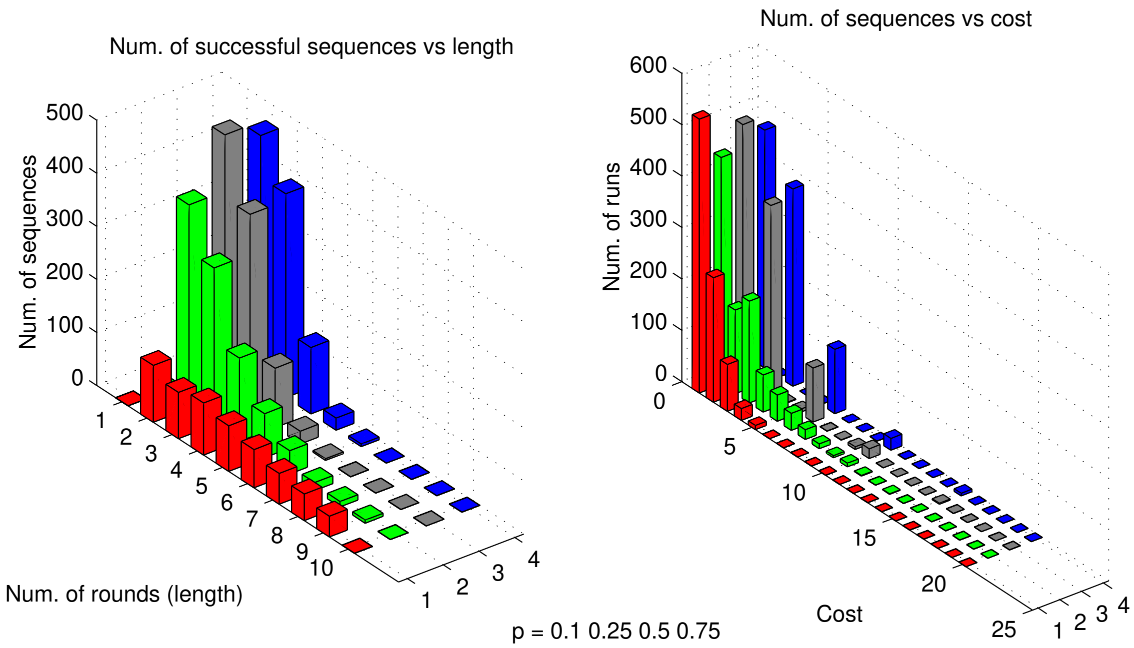

Fig. 3. Query-incremental strategy with $p = 0.1, 0.25, 0.5, 0.75$.