Using Generalized Query Tree to cope with the Capture Effect in RFID Singulation

Victor K. Y. Wu
Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign, Illinois
Email: vwu3@illinois.edu

Roy H. Campbell
Department of Computer Science
University of Illinois at Urbana-Champaign, Illinois
Email: rhc@illinois.edu

***Abstract*—The Query Tree Protocol (QT) in [1] is an efficient RFID tag singulation algorithm that is guaranteed to read all the tags in the broadcast range of a reader. However, QT ignores the *capture effect*. That is, after the reader broadcasts a bit string query prefix, it is assumed that it can distinguish one of three responses, namely {no response, one response, collision}. If the capture effect is modeled, QT would no longer be guaranteed to singulate all the tags in the reader's range, since "capturing" a tag ID in the midst of a collision would leave all the other tags in that collision unsingulated. In this paper, we introduce two modifications to QT that always singulate all the tags even when the capture effect is considered. We call these the *Generalized Query Tree Protocols* (GQT1, GQT2). We provide analytical bounds and simulation results of the singulation times of these new protocols in relation to QT.**

I. Introduction

Radio frequency identification (RFID) passive tag singulation is the process where a single reader collects the identifiers (IDs) of all the passive tags in its broadcast range. Since the tags share the same wireless medium for backscattering, singulation algorithms are collision resolution protocols. There are two main classes of such protocols. In probabilistic approaches, a mechanism similar to Aloha is used, where the reader broadcasts a query, telling each tag to randomly choose a time slot in which to reply [2]. A collision occurs when two or more tags choose the same time slot. In contrast, the Query Tree Protocol (QT) in [1] is a deterministic approach. In this case, the reader successively sends longer query prefixes. If a tag's ID matches the prefix, the tag responds with it's entire ID. A collision occurs when two or more tags' IDs match the same prefix. Since the query prefixes become longer, eventually only one tag will respond. [3], [4], and [5] provide improvements to QT, but they are all based on the same underlying principle.

In [1], after the reader broadcasts a bit string query prefix, it is assumed that it can distinguish one of three responses, namely {no response, one response, collision}. In other words, the capture effect, which is a receiver decoding a signal even in the presence of other interfering signals, is ignored. If the capture effect is modeled, QT would no longer be guaranteed to singulate all the tags in the reader's range, since "capturing" a tag ID in the midst of a collision would leave all the other tags in that collision unsingulated. In this paper, we introduce two modifications to QT that always singulate all the tags even when the capture effect is considered. We call these the *Generalized Query Tree Protocols* (GQT1, GQT2). An important performance metric for these protocols is the time required to singulate all the tags. We provide analytical bounds and simulation results on the singulation times of GQT1 and GQT2, and compare them to QT.

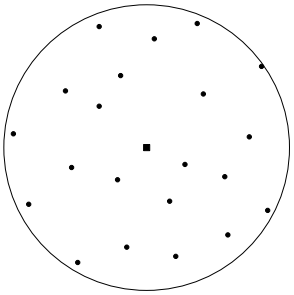


Fig. 1. Tag singulation on a circular disk of unit radius. The tags are indicated by the small circles. The reader is indicated by the square, which is located at the center of the disk.

The rest of the paper is organized as follows. In Section II, we provide the system model for tag singulation. In Section III, we re-introduce QT as explained in [1], and explain its shortcomings in the context of the capture effect. In Section IV, we introduce our new protocols, GQT1, and GQT2. In Section V, we provide simulation results. Section VI concludes the paper.

II. System Model

We introduce the system model for tag singulation. A single reader is located at the center of a circular disk of unit radius, which represents the transmit range of the reader. That is, any tag lying in the disk can decode the reader's transmissions. n tags are placed in the disk, with their locations following a uniform distribution. This is shown in Fig. 1. Each tag has a unique binary string identifier (ID) of length k bits. Therefore, $n \leq 2^k$. The n IDs are chosen uniformly.

Using a wireless transmission protocol, the reader and the tags work together, in order for the reader to collect all the tag IDs. The reader has a single antenna, and transmits signals in an omni-directional fashion. The reader also has no a priori knowledge of n. The tags are passive. That is, they transmit by backscattering energy received by the reader's transmissions.

978-1-4244-2309-5/09/$25.00 ©2009 IEEE

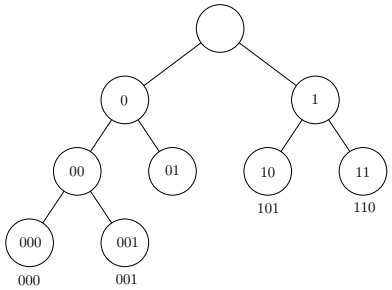


Fig. 2. QT with tag IDs {000, 001, 101, 110}. The bit strings inside each node are query prefixes. In a query prefix that results in one tag responding, the associated node is a leaf, and the bit string below that leaf is the decoded tag ID. Note that the query prefix 01 is also associated with a leaf, but since it results in no response from the tags, there is no ID below it.

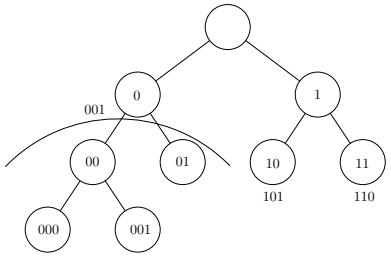


Fig. 3. Capture effect in 2-state QT with tag IDs {000, 001, 101, 110}. The capture effect enables the reader to decode 001 when the prefix 0 is queried. This causes the entire segment of the tree below the arc to be eliminated. As a result, the tag with ID 000 is not singulated.

III. Query Tree Protocol

In this section, we consider the Query Tree Protocol (QT), as explained in [1]. The algorithm works with the reader repeatedly sending a query, and then waiting for a response from the tags. In each query, the reader asks the tags if their IDs contain a certain prefix. All the tags that do, respond by sending their tag IDs. If more than one tag responds, the reader detects a collision. It then appends a 0 or 1 to the prefix, and makes another query with the new longer prefix. When only one tag responds, its ID can be decoded by the reader. Each query prefix is associated with a node in a full binary tree (the query tree). If a node has prefix xxx, then its two children have prefixes xxx0 and xxx1. Fig. 2, reproduced from [1], illustrates QT. Note that [1] does not take into account of the locations of the tags, as we model in Section II. That is, the backscattered signals of the responding tags for a particular query are assumed to be received simultaneously at the reader. The algorithm is shown below for completeness and comparison. At the end of the algorithm, the IDs of all the tags are stored in M.

QT Protocol

Reader

1) $Q := \langle 0, 1\rangle$, and $M := \langle\rangle$.
2) Suppose $Q = \langle q_1, \ldots, q_l\rangle$, and $M = \langle t_1, \ldots, t_m\rangle$.
3) Broadcast query q_1 to tags. $Q := \langle q_2, \ldots, q_l\rangle$.
4) Listen to responses from tags.
   - If no response, then do nothing.
   - Else if the response is tag ID t, then $M := \langle t_1, \ldots, t_m, t\rangle$.
   - Else, a collision is detected, then $Q := \langle q_2, \ldots, q_l, q_1 0, q_1 1\rangle$.
5) If Q is nonempty, go to 2).

Tag

1) Listen to query from reader.
   - If prefix matches, then send tag ID.
2) Go to 1).

A. Singulation Time Upper Bound

[1] defines the identification time of QT as the number of queries required for the reader to singulate all n tags. For comparison purposes, we define the *singulation time*, $T_{\mathrm{P}}(n)$, as the total running time of singulating n tags, using protocol P, where P $\in$ {QT, GQT1, GQT2}. (GQT1 and GQT2 are explained in Section IV.) We further assume that both a reader query and a tag response each take half a time unit. [1] shows that the worst case time complexity (that is, an upper bound on $T_{\mathrm{QT}}(n)$) is given by

$$T_{\mathrm{QT}}(n) \leq n\left(k + 2 - \log_2 n\right). \tag{1}$$

B. Capture Effect

In [1], QT relies on the assumption that the tags' response to a reader query falls into one of three choices. These are {no response, one response, collision}. More precisely, the reader receiver is assumed to be able to distinguish between the three cases, and decode a tag ID only for the case of a single tag backscattering. However, this assumption is rather strong, and ignores the capture effect. The capture effect describes a situation where a receiver may be able to decode a signal even in the presence of other interfering signals. In the context of QT, the reader receiver can only distinguish between two cases, namely {no response, response}. If there is no response, the reader can be certain that no tags have replied. If there is a response, the reader only knows that at least one tag has replied. It may or may not be able to decode a tag ID from the response, but this would still not allow it to know how many tags had replied.

We call 3-state QT as the situation where no capture effect occurs. Note that $T_{\mathrm{QT}}(n)$ applies to 3-state QT. 2-state QT is when there is capture effect. 2-state QT is faulty in the sense that the reader does not necessarily singulate all n tags in its range. That is, if the capture effect causes a reader to decode a tag ID from multiple signals due to multiple tags responding to a common prefix, then all but one of those tags would not be singulated, since that segment of the query tree with that common prefix would no longer grow. This is illustrated in Fig. 3. In Section V, we show that these losses are significant. In Section IV, we present two simple modifications of QT that can singulate all n tags, even when the capture effect is considered.

In this paper, we model the capture effect using a parameterized signal-to-interference (SIR) threshold, γ, of the reader receiver. We assume a free space, path loss radio propagation model. Therefore, a backscattered signal arriving at the reader has power attenuated by a factor of d^4, compared to the

original reader transmission, where d is distance between the reader and the backscattering tag. All tags are assumed to be physically identical, and have the same effective area for absorbing incoming radio energy from the reader. Suppose a reader query results in l tags responding, where $l \in \{1, \ldots, n\}$. Then, if the i^{th} tag's backscatter is considered the signal, its SIR is

$$\text{SIR}_i = \frac{1/d_i^4}{\sum_{j \neq i, j=1}^{l} 1/d_j^4}, \tag{2}$$

where d_i is the distance between the reader and the i^{th} tag, and $i \in \{1, \ldots, l\}$. If

$$\max_{i=1}^{l} \text{SIR}_i \geq \gamma, \tag{3}$$

then the capture effect occurs, and the reader decodes the ID of the k^{th} tag, where $k = \arg\max_{i=1}^{l} \text{SIR}_i$. Therefore, γ is a parameter that determines how effective the reader receiver is at extracting a tag ID in the midst of multiple signals.

IV. Generalized Query Tree Protocols

We introduce two modifications to QT. Since these are natural generalizations of QT that can handle the capture effect, we call them the *Generalized Query Tree Protocols* (GQT1 and GQT2). We also provide bounds on the singulation time of these two protocols, in relation to $T_{\text{QT}}(n)$. In Section V, we provide simulations to verify these results.

A. GQT1

There are two major changes that GQT1 makes to QT. In 3-state QT, when a query prefix results in no response or a single response, we can be sure that there are no more IDs with that prefix. In other words, this is the stopping condition of lengthening a prefix (by appending a 0 or 1). However, if the capture effect is taken into consideration, a stopping condition that guarantees singulation of all tags is that there is no response. Therefore, the first change to QT is that in GQT1, the prefix is always lengthened if there is a response. (The one exception is if the prefix length is already k bits.) The second change is that after an ID has been decoded by the reader, it sends an ACK signal to that specific tag, to tell it to silence itself. That is, the reader lets the tag know it has already singulated it, and the tag need not send any more backscatter responses for the rest of the singulation process. This is important, since when the prefix is lengthened, and queried again, it may still match the ID of the previously singulated tag. For example, suppose there are three tags with respective IDs $\{10000, 10111, 10110\}$, and the reader broadcasts a query with prefix 10. All three tags will respond. Suppose the capture effect allows the reader to decode 10000 from the three responses. If the reader does not send an ACK signal to the first tag, and it broadcasts the next query with prefix 100, the first tag will respond again, which is unnecessary. This second change is also a result of the the differing stopping conditions of 3-state QT and GQT1. GQT1 is shown below.

GQT1 Protocol

Reader

1) $Q := \langle 0, 1 \rangle$, and $M := \langle \rangle$.
2) Suppose $Q = \langle q_1, \ldots, q_l \rangle$, and $M = \langle t_1, \ldots, t_m \rangle$.
3) Broadcast query q_1 to tags. $Q := \langle q_2, \ldots, q_l \rangle$.
4) Listen to responses from tags.
   - If no response, then do nothing.
   - Else, try to decode a tag ID.
     - If able to decode tag ID t, then $Q := \langle q_2, \ldots, q_l, q_1 0, q_1 1 \rangle$, and $M := \langle t_1, \ldots, t_m, t \rangle$. Send ACK to tag with ID t.
     - Else unable to decode. $Q := \langle q_2, \ldots, q_l, q_1 0, q_1 1 \rangle$.
5) If Q is nonempty, go to 2).

Tag

1) Listen to query, or ACK from reader.
   - If query prefix matches, then send tag ID.
   - Else if ACK is intended for myself, silence myself.
2) If not silenced, then go to 1).

GQT1 leverages the capture effect to prevent query prefixes from becoming very long, and thus, reduces the size of the query tree. As the SIR threshold, γ, of the reader receiver increases, less "captures" occur, and the singulation time increases. An upper bound of $T_{\text{GQT1}}(n)$ is thus when γ is sufficiently large, and therefore, the capture effect will never take place. In this case, the algorithm runs exactly the same as QT, with the exception that there are two extra queries for each ID, since the stopping condition for GQT1 is that there is no response. This results in an additional time of $2n$. Recall that both a reader query and a tag response each take half a time unit. Suppose that ACKing a tag also takes half a time unit. Therefore, the total ACKing time is $n/2$. We thus have the following upper bound for $n \geq 2$.

$$T_{\text{GQT1}}(n) \leq T_{\text{QT}}(n) + \frac{5}{2}n$$

A lower bound of $T_{\text{GQT1}}(n)$ is when γ is sufficiently small, and the capture effect always occurs if more than one tag responds. In this case, the query tree will consist of a root node (which is just a place holder), n other internal nodes representing the n queries that each result in a decoded tag ID, and the leaves of the tree. This is illustrated in Fig. 4. Since the tree is a full binary tree, we have $n+1 = r-1$, where r is the number of leaves. Therefore, the number of nodes in the tree, excluding the root, is $n + r = 2n + 2$. Including the total ACKing time, we have the following lower bound for $n \geq 2$.

$$T_{\text{GQT1}}(n) \geq 2 + \frac{5}{2}n$$

Combining the two bounds, we have,

$$T_{\text{GQT1}}(n) \in \left[2 + \frac{5}{2}n, T_{\text{QT}}(n) + \frac{5}{2}n\right]. \tag{4}$$

Using the upper bound for $T_{\text{QT}}(n)$ in (1), we have

$$T_{\text{GQT1}}(n) \in \left[2 + \frac{5}{2}n, n\left(k + \frac{9}{2} - \log_2 n\right)\right]. \tag{5}$$

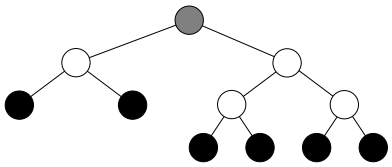


Fig. 4. Lower bound on $T_{\text{GQT1}}(n)$. The four white internal nodes represent queries that each result in a decoded tag ID. The six black leaf nodes represent queries that each result in no response. We thus have, $n = 4$ and $r = 6$.

B. GQT2

There is only one slight change that GQT2 makes to GQT1. To further take advantage of the capture effect, GQT2 does not append a 0 or 1 to the prefix when the reader decodes a tag ID. Rather, the reader re-broadcasts the same prefix. Since the ACK silences the tag that was just singulated in the previous query, another tag can be "captured" using the same prefix. Only when no ID can be decoded from a response does the reader append 0 or 1 to the prefix. GQT2 is shown below.

GQT2 Protocol

Reader

1) $Q := \langle 0, 1\rangle$, and $M := \langle\rangle$.
2) Suppose $Q = \langle q_1, \ldots, q_l\rangle$, and $M = \langle t_1, \ldots, t_m\rangle$.
3) Broadcast query q_1 to tags. $Q := \langle q_2, \ldots, q_l\rangle$.
4) Listen to responses from tags.
   - If no response, then do nothing.
   - Else, try to decode a tag ID.
     - If able to decode tag ID t, then $Q := \langle q_2, \ldots, q_l, q_1\rangle$, and $M := \langle t_1, \ldots, t_m, t\rangle$. Send ACK to tag with ID t.
     - Else unable to decode. $Q := \langle q_2, \ldots, q_l, q_1 0, q_1 1\rangle$.
5) If Q is nonempty, go to 2).

Tag

1) Listen to query, or ACK from reader.
   - If query prefix matches, then send tag ID.
   - Else if ACK is intended for myself, silence myself.
2) If not silenced, then go to 1).

An upper bound of $T_{\text{GQT2}}(n)$ follows the same reasoning of the upper bound of $T_{\text{GQT1}}(n)$. The only difference is that there is only one extra query for each ID, since after decoding a tag ID from a prefix, it only has to re-broadcast the same prefix to detect that there is no response, which is the stopping condition. The upper bound for $n \geq 2$ is thus,

$$T_{\text{GQT2}}(n) \leq T_{\text{QT}}(n) + \frac{3}{2}n$$

A lower bound of $T_{\text{GQT1}}(n)$ is when γ is sufficiently small, and the capture effect always occurs if more than one tag responds. In this case, the initial prefixes $\{0, 1\}$ are sufficient to decode all the IDs, one-by-one with the capture effect. This contributes a time of n. There are also two extra queries, one for each prefix that results in the stopping condition. Including the total ACKing time, the lower bound for $n \geq 2$ is thus,

$$T_{\text{GQT2}}(n) \geq 2 + \frac{3}{2}n$$

Combining the two bounds, we have,

$$T_{\text{GQT2}}(n) \in \left[2 + \frac{3}{2}n, T_{\text{QT}}(n) + \frac{3}{2}n\right]. \quad (6)$$

Using the upper bound for $T_{\text{QT}}(n)$ in (1), we have

$$T_{\text{GQT2}}(n) \in \left[2 + \frac{3}{2}n, n\left(k + \frac{7}{2} - \log_2 n\right)\right]. \quad (7)$$

V. Simulation Results

We present Monte Carlo simulation results of the three protocols described in this paper. All results are averages. In all cases, $k = 12$ bits is chosen to be the length of a tag ID. $n = 10, 20, \ldots, 100$ is the number of tags within the range of the reader. The tag IDs are unique and chosen from a uniform distribution. For the 2-state QT, GQT1 and GQT2 cases, the locations of the tags in the unit disk are also chosen from a uniform distribution. The tag distances are used to calculate when the capture effect occurs using the parameterized SIR threshold, γ.

A. QT

Fig. 5(a) shows the average singulation time of 3-state QT, i.e. $T_{\text{QT}}(n)$. [1] says that $T_{\text{QT}}(n)$ is $O(n)$ with high probability. The straight line confirms this. The upper bound in (1) is also shown, which appears very loose. Fig. 5(b) shows the number of tags that 2-state QT fails to singulate, due to the capture effect. The plot shows that the percentage of missed tags stays fairly stable as n is increased. As γ is decreased, the capture effect becomes more pronounced (i.e. more "captures" occur), and thus, more tags are not singulated. For $\gamma = 5$ dB, and $n = 100, 78$ tags are not singulated.

B. GQT1, GQT2

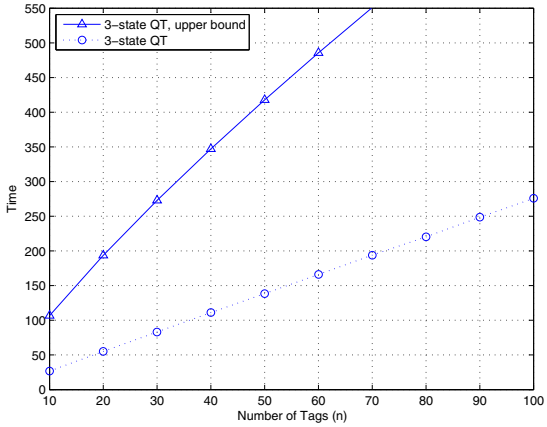
Fig. 6(a) and Fig. 6(b) show the average singulation times of GQT1 and GQT2, i.e. $T_{\text{GQT1}}(n)$ and $T_{\text{GQT2}}(n)$, respectively. The upper and lower bounds in (4) and (6) are also shown respectively, in the two plots. As expected, as γ increases, there are less "captures", and the average singulation time increases. In both protocols, $\gamma = 100$ dB is sufficient for the average singulation time to approximate the upper bound, and $\gamma = -5$ dB for the lower bound. Fig. 7 compares the average singulation times of 3-state QT, GQT1 and GQT2. 3-state gives the best performance (since the capture effect is ignored in this case). GQT2 performs better than GQT1 at the same γ. At $\gamma = 5$ dB and $n = 100$, GQT2 requires approximately 66 additional seconds compared to 3-state QT.

VI. Conclusion

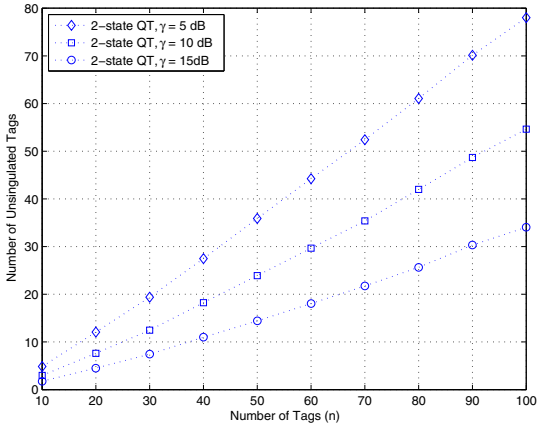
QT is an efficient RFID tag singulation algorithm. However, it ignores the capture effect. In this paper, we provide GQT1 and GQT2, two natural generalizations of QT that retain the tree-based spirit of QT, but also cope with the capture effect. Analytic bounds, and simulation results of the singulation times of these protocols are also provided.

References

[1] C. Law, K. Lee, and K.-Y. Siu, "Efficient memoryless protocol for tag identification," in *Proceedings of the 4th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, Boston, MA, Aug. 2000, pp. 75–84.

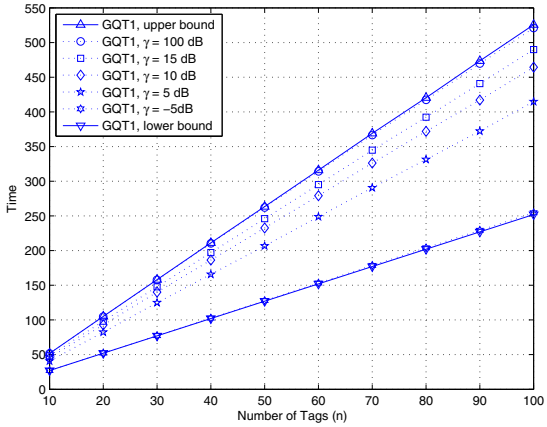


(a) Average singulation time of 3-state QT.

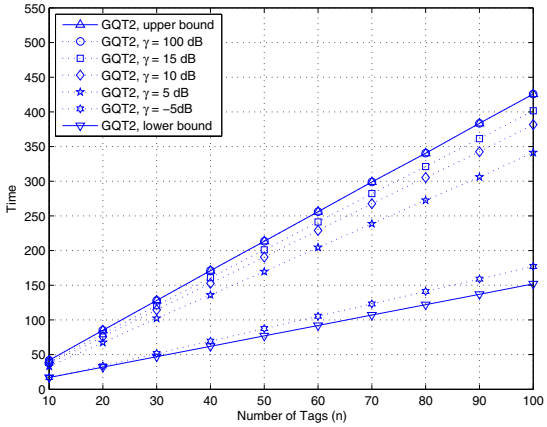


(b) Average number of unsingulated tags in 2-state QT.

Fig. 5. QT simulation results.



(a) Average singulation time of GQT1.



(b) Average singulation time of GQT2.

Fig. 6. GQT1, GQT2 simulation results.

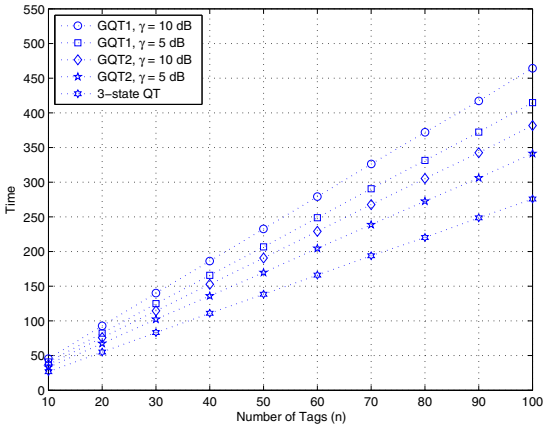


Fig. 7. Average singulation time comparison of 3-state QT, GQT1 and GQT2.

[2] H. Vogt, "Multiple object identification with passive RFID tags," in *IEEE International Conference on Systems, Man and Cybernetics*, Hammamet, Tunisia, Oct. 2002.

[3] K. W. Chiang, C. Hua, and T.-S. P. Yum, "Prefix-randomized query-tree protocol for RFID systems," in *IEEE International Conference on Communications*, Istanbul, Turkey, Jun. 2006, pp. 1653–1657.

[4] N. Bhandari, A. Sahoo, and S. Iyer, "Intelligent query tree (IQT) protocol to improve RFID tag read efficiency," in *International Conference on Information Technology*, Bhubaneswar, India, Dec. 2006, pp. 46–51.

[5] J. Myung, W. Lee, and T. Shih, "An adaptive memoryless protocol for RFID tag collision arbitration," *IEEE Trans. Multimedia*, vol. 8, no. 5, pp. 1096– 1101, Oct. 2006.