Reliable Peer-to-Peer Semantic Knowledge Sharing System

Abdul-Rahman Mawlood-Yunis

School of Computer Science, Carleton University
1125 Colonel By Drive, Ottawa, Ontario, K1S5B6, Canada
armyunis@scs.carleton.ca

Abstract. Reliability issues arise in Peer-to-Peer Semantic Knowledge Sharing (P2PSKS) systems. P2PSKS systems are prone to disconnection failures. Disconnection failures deprive the network peers from having access to complete knowledge in the PSPSKS system. Disconnection failures arise when the P2PSKS systems employ the adaptive query routing methods in order to avoid flooding the networks with redundant search messages or queries during information exchange. P2PSKS systems need to utilize fault-tolerant query result evaluation function in order to be reliable. The current publication will focus on two objectives in P2PSKS research: 1. reliability problem identification and 2. reliability problem solutions. The P2PSKS system reliability problem will be identified through the use of simulation modeling techniques. The P2PSKS system reliability problem solutions will be advanced through adaptations of the generous tit-for-tat method which was originally developed in evolutionary game theory. The current research will demonstrate more reliable solutions to address P2PSKS system reliability issues by advancing the Fault-Tolerant Adaptive Query Routing (FTAQR) algorithm.

1 Introduction

Current research directions in Peer-to-Peer Semantic Knowledge Sharing (P2PSKS) systems are evolving to combine two complementary technologies: 1. Peer-to-Peer (P2P) networks and 2. formally-structured information (ontology). P2PSKS systems represent the next step in the evolution of P2P networks because P2PSKS systems incorporate several additional features not present in P2P networks. The most important additional feature found in P2PSKS systems is the point-to-point mapping. The point-to-point mapping is used as a translational capability to forward queries between peers under the conditions when the peers possess different data schema or knowledge representations. P2P networks which employ ontologies include four examples: 1. P2P knowledge management systems [5, 9, 15, 16, 8], 2. P2P Semantic Web [10, 18, 19], 3. P2P Web Services [4] and 4. P2P systems which require cooperation among distributed and autonomous peers with heterogeneous information sources, i.e. local ontologies [1, 6]

Ontologies are advantageous in P2P networks because they provide the possibility for improving search and content retrieval. The incorporation of ontologies in P2P networks has been previously reported in the scientific literature in three

research precedents: 1. creation of semantic networks on existing P2P networks which has been referred to as semantic overlay networks, 2. semantic-based query routing and 3. adaptive query routing. These three methods are used to avoid flooding the P2PSKS systems with redundant search messages or queries during information exchange. Flood avoidance refers to selectively sending queries to only those peers which are highly knowledgeable or expert about query content.

The current research will focus on P2PSKS systems reliability issues. The P2PSKS systems reliability refers to the systems resiliency under the conditions for handling temporary faults. P2PSKS systems need to utilize fault-tolerant query result evaluation function in order to be reliable. That is, proper query result evolution function is absolute necessary to guide query forwarding or risk network connectivity deterioration.

In order to address the P2PSKS systems reliability issues, we have invented the Fault-Tolerant Adaptive Query Routing (FTAQR) algorithm. The FTAQR algorithm is capable of handling temporary faults under three conditions: 1. systems unavailability, 2. peer misbehavior, and 3. mapping corruption. The further details of the three conditions will be provided below. The FTAQR algorithm is based on adaptations of the generous tit-for-tat method which was originally developed in evolutionary game theory (see e.g. [2] for information on iterative prisoner's dilemma).

The importance of this study goes beyond the P2PSKS systems. Fault-tolerant adaptive query routing is essential in other research directions as well. These include, for example, automatic consensus building or bottom-up construction of ontology, and adequate resource utilization in large and open information system. This is because these researches also depend on the correct decision about query routing adaptation.

The remainder of this publication is organized into the following sections: Section 2, discusses faults due to resource unavailability and peers misbehavior. Section 3, presents the FTAQR Algorithm and its related design decisions. Section 4, discusses simulation building blocks and simulation setup. Section 5, presents initial result evaluation and reliability metrics . Section 6 reviews related work, and finally Section 7, concludes the paper and identifies directions for future work.

2 Resource unavailability and peers misbehavior

Formal fault definition and fault classification along temporal dimension as well the effects of mapping corruption on bottom-up construction of ontology have been provided in [13, 14]. Hence, we avoid repeating this issues here. However, for the sake of self-containment, we provide a short description of the reasons for temporary faults due to resource or service unavailability and peers misbehavior in this section:

a) It has been pointed out by Gal [7] that the design of the conceptual schema for information services possesses special properties. These include (i) a rapid change of data sources and metadata; and (ii) instability, since there is no

control over the information sources. The *availability* of information sources is solely dependent upon information source providers. A possible scenario is the temporary unavailability of information when such information is needed. This possibility is particularly acute during query execution.

b) System operation in P2PSKS systems depends on the honest conduct of peers. A peer could be dishonest or biased in its interaction with other peers during query forwarding for reasons such as selfishness or greed. There are various ways through which a peer could influence query forwarding. These include (i) not forwarding a query to other peers during transitive query process (ii) not forwarding answers to the other peers; or (iii) altering or delaying queries (results) before forwarding them to other peers.

In both above cases, the querying peer will receive incorrect query results, i.e. faults occur. These faults could be permanent or temporary. Hence, it is desirable for P2PSKS systems to be able to differentiate between different types of faults (permanent, intermittent and transient), and tolerate the non-permanent ones (transient and intermittent). This is because P2PSKS systems employing adaptive query routing method risk, unnecessarily, partial or total disconnection, based on the network topology at the time, if they do not tolerate temporary faults. More detailed discussion about this observation is provided in [12]

3 FTAQR Algorithm Description and Design Decisions

In this section we are going to describe algorithm steps and procedure. The algorithm is simple in concept, easy to implement and highly effective. Several design decision have been made during algorithm development. These include:

A) Use of local knowledge. Peers have only knowledge about their immediate neighbors. Peer's knowledge is about their belief in the reliability or ability of their neighboring peers on providing correct answers to their queries. The reliability value ≥ 1 refers to total reliability, and reliability value ≤ 0 refers to totally unreliability . Peers disconnected from each other when reliability values reaches ≤ 0.

B) Normalization is not applied. Sending query on an outgoing link could result in several query answers. The number of query answers depends on the number of cycles in the network starting from the querying peer. All answers are treated equally. That is, no extra weight is given to any particular answer or querying path.

C) Use of average value. The average value of query answers is used for evaluating the reliability of the outgoing links.

These decisions are made to make the algorithm simple to be understood. Future revision of these decisions is possible. The algorithm is made up of three essential functions: 1. initialization, 2. result evaluation, and 3. an update function, and proceeds along the following steps:

1. At the startup of network, peers start *connection*; connected peers set their trust value in each other to 1, and system parameters for query result evaluation are initialized.
2. query result evaluation checks for the ($\equiv$, $\supset$, $\subset$, $*$, $\perp$) relations between concepts in query answer and concepts in querying peer's local data set. These relations imply that [100%, 75%, 75%, 25%, 0%] of concepts returned by query answer are understood by querying peer respectively. That is, the relation between query concepts and peers' local concepts are exact same, related, and totally not related.
3. based on query result evaluation relation, i.e. step 2, peers update their confidence in the reliability of their outgoing links. The numerical values used for updating are [0.2, 0.1, 0.1, 0.05, -0.2]. The update values correspond to the semantic relation between query answer concepts and peer's local concepts.

We decrease peer's confidence in its outgoing link by 0.2 every time it receives incorrect query answers. This is in order to initially tolerate up to 4 faults in sequence. Hence, the generosity part of algorithm to tolerate temporary faults. After that, peers will be treated by the tit-for-tat rule. That is, peers will be punished for returning any incorrect query answer, and rewarded for their correct query answers. Snapshot pseudocodes corresponding to the three described steps are as follow:

```
Initialize () { ChangeInStrength [ ] ← {0.2, 0.1, 0.1, 0.05, −0.2}
    While(aPeer.haveMoreLinks())
            aPeer.setOutGoingMappingLink ←1
    End While
}

Result_Evaluation() {
    result ← query.getResultContent()
    newStrength ← 0
    IF (Relation_is_synonyms (result))
        newStrength ← anEdge.getStrength() + ChangeInStrength[0]
    Else If (Relation_is_hypernyms (result) )
        newStrength ← anEdge.getStrength() + ChangeInStrength[1]
    Else If (Relation_is_hyponyms (result))
        newStrength ← anEdge.getStrength() + ChangeInStrength[2]
    Else If (Related (result))
        newStrength ← anEdge.getStrength() + ChangeInStrength[3]
    Else
        newStrength ← anEdge.getStrength() + ChangeInStrength[4]
    End IF
}

Update (int newStrength)  {
    anEdge.setStrength(newStrength)
    removeUnusedLinks (anEdge)
```

}

4 Simulation Building Blocks and Setup

In this section, the simulation building blocks are described. These include a short description of peers, resources, semantic neighborhood and query formulator abstracts.

4.1 Peers

A peer represents an instance of a participant in the network. Each peer has a unique id, peerID, data schema, and schema_description or profile. The latter is used for forming semantic neighborhood.

4.2 Resource

To simulate the generic characteristics of a P2PSKS system and keep distance to any particular realization, we abstract in our simulation from the underlining data model and consider data to be set of concepts. A file which contains Laptop Ontology developed by Standford University[1] constitute peer's data/knowledge. Laptop ontology is made of 40 concepts, each peer stores a percentage of the total ontology concepts (file content). Network peers are divided into four groups automatically. The number of peers in each group is N/4, where N is the total number of peers. The size of the dataset each peer holds is based on the group type it belongs to. Group1, Group2, Group3 and Group4 hold $15\%, 30\%, 50\%$ and 70% of total concepts respectively. Peers are assigned to the groups randomly.

4.3 Semantic Neighborhood

In our simulation, we employ the discovery method. Peer schema descriptions (profile file) are simple XML files. On the network startup, peers exchange their profile files and a tree comparison algorithm is used to determine the similarity relation between schemas. When a new peer joins the network, it will advertise its schema description, and peers with similar schema will reply to the advertisement and connection is established. Beside the similarity function (sim) which has to be greater than a predefined threshold (e.g. sim $\geq \delta$) in order for peers to be able to connect, the number of connections each peer could have is also another restriction. The in/out connection degree(I, O), and δ are user defined variables and their values are defined on the simulation startup. The following snapshot pseudocode represents the described steps.

```
/*compare, a tree comparison function call */
Set sourceAndTarget ← new HashSet()
For( int i ← 0 To i< numberOfNodes - 1)
```

[1] http://www.ksl.stanford.edu/DAML/laptops.owl

```
  For ( int j← i To j< numberOfNodes - 1)
      /* SchemaCollection contains all schemas */
      compare (sourceAndTarget, SchemaCollection[i], SchemaCollection[j +1 ], i, j+1)
      j← j+1
  End For
  i ← i+1
End For
/*set of function calls to rank neighboring peers and create semantic neighborhood created */
connectionMap (sourceAndTarget )
sortConnectionMap (connections, rows, columns)
sortByStrength (sortedMapById )
semanticNeighborhood (getGroups ())
```

4.4 Query Formulator

In our P2P-SKS system simulation, queries are made up of 4 concepts and SELECT query operator is used. Query concepts are chosen from local data set, and any peer could initiate a query. Peers that initiate queries as well as query concepts are selected randomly. At each run a new query is created and a new initiator is selected. Chances for the new query concepts and query initiators to be different than prior concepts and initiators are high. The probability of a peer to be a query initiator or a querier at each run of our automatic query formulator is $1/N$, where N is the number of peers of the simulation. The probability of the query to be exactly the same query as the previous one is $(1/D)^{\|f\|}$, where D is the size of data set each peer possesses, and $\|f\|$ is number of concepts that comprise the query content. The probability of the same peer to pose the the same query in two different system runs is then the multiplication of both above terms, i.e., $(1/N)(1/D)^{\|f\|}$. The following snapshot pseudocode represents the described steps.

```
/* Queries variable is peers local data set */
ArrayList queries ← FileInput.read()
Vector queryConcepts
Query q ← null
numberOfConceptsInQuery ← 4
For(int i ←0 To i < numberOfConceptsInQuery) {
    int QueryIndex ← Random.uniform.nextIntFromTo(0, queries.size() -1)
    queryConcepts.add(queries.get(QueryIndex).toString())
    i ← i+1
End For
/* create query object with its content */
q ← new Query (queryConcepts)
```

The reader may notice that, two other essential components, mapping and router, are not described here. The former is not used because in our current simulation

version, although different in sizes, all peers use same resource, and the latter is simulated only partially through simple query forwarding and connection dropping. Peers use the semantic neighborhood initially created for search and content retrieval, and drop relations with related neighbors when their strength level, i.e., ability to provide correct query answers, reach zero. Further improvement regarding the mapping and router issue will be considered in subsequent works.

5 Experimental results and reliability metrics

In this section we describe network **reliability metric**, **fault simulation** and **experimental results**. Connectivity is used as a metric for evaluating network reliability. That is, we observe the network deterioration speed (slop function), and the network connectivity degree of the system setups running for a constant period of time. The experiments run under two different settings: with and without the capability to tolerate faults. The variation in the network deterioration trend and the connectivity degree between the two system settings is reported as the difference in the reliability that a FTAQR algorithm guarantee.

Faults are generated using the knowledge about peers' different data sizes, query formulation and routing strategy. That is, when a query is created with a set of peer's local concepts and that query is passed through other peers which have only a subset of query concept constituents, then the end query result will contain only the minimum common denominator of concepts. Hence the fault is generated.

Figures (1, 3) and(2, 4) represent initial results for two different experimental settings. The system components, i.e. data sets, query formulator, query router and number of peers (15), are same for both settings. The only difference between the two system settings is the difference in query evaluation function, system fault-tolerance capability. The results clearly demonstrate that building P2PSKS systems with built-in fault-tolerance capability increases system reliability. The network deterioration trend for the second system setting, Figure 4, is less sharp than the network deterioration trend in the Figure 3. Similarly, the difference between network connection states demonstrate that a P2PSKS system with fault-tolerant capability is more reliable than one without such a capability. Figures 1 and 2 show that after running system simulation for the period of $4*10^2$ ms, the network disconnection rate for the first system setting was $> 93\%$, and only 33% for the second one. Table 5 summarize these results.

Its worth to mention that, in both system settings the network could reach to the state of total disconnection. This is because, in the current version of our simulation, we do not add new peers to the system during system operation. That is, while peers lose connection because of incorrect query results, they do not make new connections. Thus, the possibility of total disconnection arises. Similarly, the disconnected peers are not considered for reconnection. Further, in situations where peers are about to be totally disconnected, e.g. when peers have only one outgoing link, a different action from the query result evaluation

component might be necessary. This includes, for example, changing the way peers are punished for the incorrect query answers. These are issues which we will consider in the subsequent simulation versions.

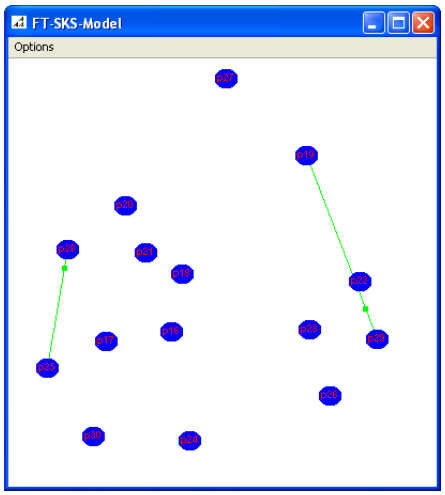

Fig. 1. Network disconnection when FTAQR is Not used

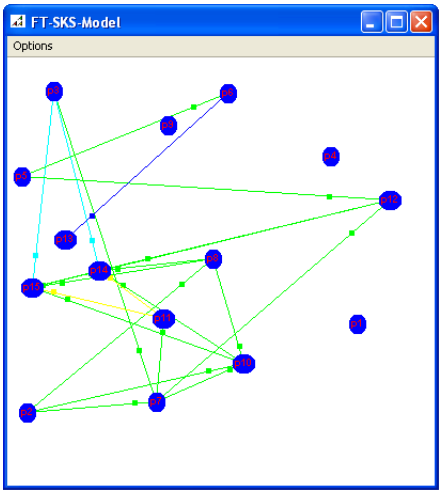

Fig. 2. Network disconnection when FTAQR is used

	P2PSKS without FTAQR	P2P with FTAQR
Network Disconnection	93%	33%
Function Slop	3	1.25

Table 1. FTAQR's Effect on Network Connection and Network Deterioration Trend

6 Related Work

Loser et al. [11] suggest that the information evaluation strategy, among others, is a criterion for distinguishing adaptive query routing in semantic overly networks from each other. While, we concur with Loser, we also presented a new algorithm for query result evaluation. Acknowledging the fact that P2PSKS networks could change during query propagation, i.e., a peer may become temporarily unavailable or a new peer with relevant information source joins the network, Zaihrayeu [20] highlights three different scenarios which have the potential for generating faults. Zaihrayeu, however, tries to transform (avoid) the identified problems through a set of assumptions. Our approach is resilient to temporally unavailability problem highlight by Zaihrayeu. More generally, our approach is resilient to the update problems associated with P2P networks. Other relevant contributions such as [9] and [15] do not consider fault-tolerance. Hence, our FTAQR algorithm complements these contributions. Checking on recurrent incorrect query answers by semantically relevant peers in [8], and probabilistic

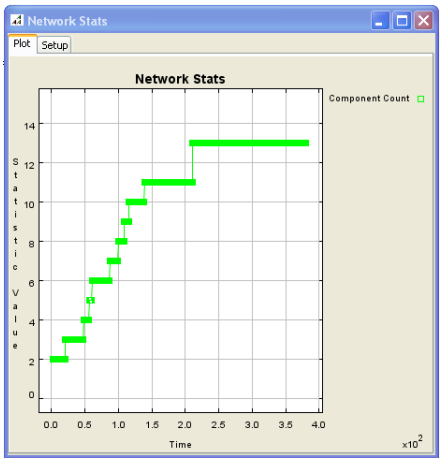

Fig. 3. Network determination when FTAQR is Not used

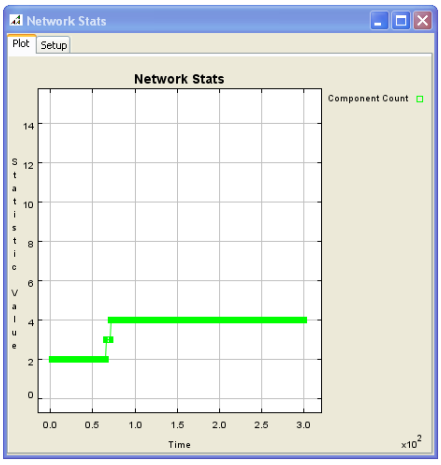

Fig. 4. Network determination when FTAQR is used

based fault-tolerance approach employed by [1] could be further improved using our FTAQR algorithm through tolerating non-permanent fault.

7 Conclusion and Future Work

In this work, we identified that reliability is an issue in P2PSKS systems. P2PSKS system referred to combining two different technology worlds: standard P2P networks and Ontologies. The identification of the problem was demonstrated through the possibility of total or partial network failure due to the existence of faults in the query evaluation function. The occurrence of faults was presented through two different scenarios: temporally unavailability of peers and peers misbehavior. In addition to reliability identification problem in P2PSKS systems, a solution to the problem was proposed as well. The proposed solution, FTARQ algorithm, did not require time, information or component redundancy. FTARQ algorithm is an adaptation of generosity tit-for-tat technique used in game theory. The effectiveness of the FTARQ algorithm demonstrated through system simulation. The description of simulation components, and test results were provided as well.

As a future work, we are working on extending our current FTAQR algorithm as follow: Instead of allowing all query results to equally influence the confidence peers have in their out-going links, we could use various majority voting techniques to derive new algorithm. The voting algorithm would be compared to the current FTAQR algorithm to further study system reliability improvement. We are also working on building a more complete P2PSKS system simulation than the current one. A completed system simulation would be used for studying various aspects and issues of P2PSKS systems. This includes further examination of the influence that an FTAQR algorithm could have on systems such as Chatty Web [1], KEx [5], Piazza [9], P2PSLN [8].

References

1. K. Aberer and P. Cudre-Mauroux and M. Hauswirth. Start making sense: The Chatty Web approach for global semantic agreements. In *Journal of Web Semantics*, 1(1): 89-114, 2003.
2. R. Axelrod. The Complexity of Cooperation. Princeton University Press, 1997.
3. M. R. Lyu. Software Fault Tolerance. *Wiley Publishing*, 1995.
4. D. Bianchini, V. De Antonellis, M. Melchiori, D. Salvi and D. Bianchini. Peer-to-peer semantic-based web service discovery: state of the art, Technical Report, Dipartimento di Elettronica per lAutomazione Universit di, 2006.
5. Bonifacio, M., Bouquet, P., Mameli, G. and Nori. Peer-mediated distributed knowledge management. In *International Symposium Agent-Mediated Knowledge Management, AMKM 2003*. Revised and Invited Papers (LNCS 2926), p 31-47, 2004
6. P. Fergus, A. Mingkhwan, M. Merabti,and M. Hanneghan . Distributed emergent semantics in P2P networks. In *Proc. of the Second IASTED International Conference on Information and Knowledge Sharing*, P: 75-82, 2003.
7. A. Gal. Semantic Interoperability in Information Services: Experiencing with CoopWARE. In *SIGMOD Record*, 28(1):68-75, 1999.
8. Z. Hai, L. Jie et al. Query Routing in a Peer-to-Peer Semantic Link Network. In *Computational Intelligence*, Vol 21(2): 197-216, 2005
9. A. Halevy, Z. Ives, P. Mork, and I. Tatarinov. Piazza: Mediation and integration infrastructure for semantic web data. In *proceedings of the International World-Wide Web Conference WWW-03*, 2003.
10. P. Haase, J. Broekstra et al. Bibster – A Semantics-based Bibliographic Peer-to-Peer System. In *Third Intl. Semantic Web Conf. (ISWC)*, 122-136, 2004.
11. A. Loser, S. Staab, C. Tempich. Semantic social overlay networks. IEEE Journal on Selected Areas in Communications 25(1): 5-14, 2007.
12. A-R. Mawlood-Yunis, M. Weiss and N. Santoro. Issues for Robust Consensus Building in P2P Networks. In *Intl. Workshop on Ontology Content and Evaluation in Enterprise (OnToContent)*, LNCS 4278, 1020-1028, 2006.
13. A-R. Mawlood-Yunis, M. Weiss and N. Santoro. Fault Classification in P2P Semantic Mapping. *Workshop on Semantic Web for Collaborative Knowledge Acquisition (SWeCKa)* at *Intl. Conf. on Artificial Intelligence (IJCAI)*, 2007.
14. A-R. Mawlood-Yunis, M. Weiss and N. Santoro. Fault-tolerant Emergent Semantics in P2P Networks. Book Chapter in *Semantic Web Engineering in the Knowledge Society*, Idea Group, 2007 (to appear).
15. E. Mena, A. Illarramendi et al. OBSERVER: an approach for query processing in global information systems based on interpretation across pre-existing ontologies. In *Distributed and Parallel Databases*, 8(2):223-71, 2000.
16. A. Kementsietsidis, M. Arenas et al. Managing Data Mappings in the Hyperion Project. In *the 19th Intl. Conf. on Data Engineering (ICDE)*,732-734, 2003.
17. D. K. Paradhan. Fault-Tolerant Computing System Design. *Prentice-Hall PTR* publication, 1996.
18. M. Rousset, P. Chatalic et al. Somewhere in the Semantic Web. In *Intl. Workshop on Principles and Practice of Semantic Web Reasoning*, 84-99, 2006.
19. S. Staab and S. Stuckenschmidt. Semantic Web and Peer-to-Peer. Springer Publishing, 2006.
20. I. Zaihrayeu Towards Peer-to-Peer Information Management Systems. PhD Dissertation, International Doctorate School in Information and Communication Technologies, DIT - University of Trento, 2006.