Searching for PI resources on MANETs using JXTA

Rodolfo Oliveira, Luis Bernardo, Nelson Ruivo,
and Paulo Pinto
Faculdade de Ciências e Tecnologia
Universidade Nova de Lisboa, Portugal
{rado,lflb,pfp}@uninova.pt
nelsonruivo@gmail.com

Abstract

This paper presents an evaluation of partial and intermittent (PI) resource support on JXTA and proposes a new MANET rendezvous service, optimized for handling PI resources. JXTA is a peer-to-peer modular open platform optimized for developing Internet applications, which already provides basic support for MANETs. JXTA rendezvous service on adhoc mode already supports synchronous searches. However it can not handle PI resources and it may produce a broadcast storm on dense MANETs. This paper also describes a prototype implementation of the proposed MANET rendezvous service, and presents some performance measurements.

1. Introduction

JXTA 2 [1] is a peer-to-peer (p2p) modular open platform optimized for developing Internet applications. It supports the creation of several self-organized virtual overlay networks (called peer groups) on top of multiple fixed physical networks. P2p systems are dynamic entities that have more or less members over the time. They already have a flexible structure that adapts to these changes. Mobile and Adhoc Networks (MANETs), on the other hand, are dynamic right from the bottom – at node level. The utilization of p2p system over MANETs will enlarge the kind of applications that can be deployed over this type of networks. JXTA has a particularly interesting feature that makes it suitable to be used over MANETs – all its structure is based on a low-level search service.

MANETs' conditions differ considerably from fixed networks because connectivity is intermittent. The probability of a node on the path moving out of range or failing is high for dynamic MANETs. This problem is normally solved by the routing algorithm but before it can run, nodes have to know their neighbors. Common MANET routing algorithms (e.g. AODV, DSR) [2] react on demand: they flood an address query on the network when someone tries to send a packet to a new address, or when a link fails due to node movement or failure. The flooding procedure may fail if the network is temporarily partitioned, originating temporary unavailability of a set of resources.

JXTA was not designed to use MANETs and consequently partial and intermittent (PI) resources at node level which can change at a faster rate than the normal annexation (and de-annexation) of nodes in a fixed network p2p system. Therefore, there are some modifications at JXTA core that need to be performed before the system can be used with success. The focus of this paper is on the lower core level modifications to create a JXTA version able to support higher level services dedicated to MANETs.

The next section presents an overview of the JXTA architecture, identifying the key components that must be enhanced to support PI resources on MANETs. Section 3 describes the proposed MANET rendezvous protocol (MANET-RVP) that contains these enhancements, and the following section presents the status of a JXTA prototype and a testing application, together with a set of measurements attesting their performance. Finally, section 5 draws some conclusions.

2. JXTA improvements for PI resources

JXTA defines a set of core p2p protocols [3], on top of which applications can be implemented. These protocols provide the basic functionality for peer and resource discovery, communication and organization. JXTA also includes a reference implementation for

JXTA core on Java, developed under the JXTA platform project.

All entities and resources in JXTA (peers, groups, pipes, endpoints, services, etc.) are represented by advertisements. Advertisements are XML documents having unique IDs. JXTA core p2p services manipulate advertisements caching them and using searching procedures when caching fails. Caching is not effective on MANETs, because information tends to be outdated too fast. Therefore, advertisements must be handled preferentially by searching on MANETs. The most basic search service is the Rendezvous service and higher level searching services are then defined on top of it (e.g. Discovery service, Endpoint Routing service). Applications run on a JXTA virtual network, which is a set of peers grouped on peer groups created by exchanging advertisements.

There are basically three problems to address: first, JXTA was designed for fixed links and the search service has to take node mobility in account; second, PI resources might not be available by the exact time of the search and some relaxation on the search semantics can make the system include the PI characteristics of resources; third, JXTA assumes wired links and MANETs are usually mounted on wireless links. There is a considerable difference in error rates that must be taken into consideration. Before looking at each problem at a time, there is a characteristic at application level that once the three problems are solved makes JXTA suitable for MANET applications. As components in the system move around their network addresses can change. Fortunately inter-peer communication in JXTA is usually implemented by exchanging messages through pipes. Pipes use logical identification at system level that is bound and resolved to physical addresses at run-time, and when connection is lost. Therefore, failures are fixed automatically by JXTA protocols. One of these protocols are based on the search service which means that once the search service is modified this part of the system poses no problems.

JXTA 2 implements a universal searching service which is the base of most of the system features and can use, according to [4], one of two approaches: a loosely-consistent global index is created using a distributed hash table (DHT) algorithm, and when it fails, a random walk approach. In fact, there is a third approach based on pure flooding on the physical network: the adhoc mode of RendezVous Protocol (RVP). For this third approach each peer receives the query and multicasts it over all its interfaces if it is new, until the query reaches the specified maximum number of hops.

Searching is supported by rendezvous peers. Any normal peer can be a rendezvous peer by static configuration, or by invoking the function *net.jxta.rendezvous.startRendezVous*. JXTA 2 searching service has two levels [4]: first, ordinary end peers send advertisement indices containing local resources' hash values to their rendezvous peers; then, rendezvous peers use this information to create the resource index of all end-peer resources under their responsibility. This index is used to respond to queries. The loosely-consistent global index is created amongst rendezvous peers using the Rendezvous Peer View (RPV) service. For a MANET the global index cannot be used because it does not pay the overhead of creating and maintaining it. We are then left with the other two approaches: a rendezvous peer may forward the query message designing a random walk over virtual links (when configured in peer or rendezvous modes) or may flood the query message using multicast packets to the next hops (when configured in adhoc or rendezvous modes).

JXTA peers usually start discovering other peers by using either IP multicast or getting cached advertisements from other peers already present on the network. Then they create a virtual overlay network (VON) composed by a set of peers connected by TCP links, which supports peer communication and the random walk searching. If the MANET topology changes after the VON creation, a random walk search may produce several MANET routing floods in order to restore the TCP links that failed. For instance, if the VON is created when the MANET topology is A-B-C-D and later changes to D-B-C-A, then a search from A to a resource in D over the VON will produce two floods at the routing layer and will follow a long physical path (A-C-B-C-B-D). Therefore, the random walk approach is not a suitable solution either.

JXTA pure flooding approach is the most appropriate for dynamic MANETs because it does not use any MANET routing services to locate resources - it just floods a single time the search query message on the specified range (now measured in multicast hops) locating the nearest resource available. A JXTA message includes the sequence of rendezvous peers visited during the flooding. This information is used to route answers back to the searching peer. MANET routing tables are updated when an answer message is sent back from the peer with the resource. The cost of this update is low because every hop represents a direct connection between peers. Afterwards, the application runs on top of this VON (created on demand during the flood) in a JXTA typical way.

Since the neighborhood keeps changing on a MANET, the relations between rendezvous peers and

their ordinary end peers can change. A straightforward way of guaranteeing that all possible resources are scanned is to have all peers operating as rendezvous-peers in adhoc or rendezvous mode. However, this may cause a broadcast storm problem [5]. An alternative is to still make all peers as rendezvous-peers but run a clustering algorithm for reducing the number of multicast messages flooded [6], reducing the problem. This clustering algorithm is described in the following section.

A second new facility is needed in JXTA for handling PI resources: the support of asynchronous flooding of query messages, to wait for the appearance of a resource. Query messages must include a validity field, which indicates for how long the query is valid. We propose the caching of query message at rendezvous peers. When a new query message is received, and after checking local resources and forwarding it, the rendezvous peer must store the query message on a fixed length buffer. Like in Gnutella [7], the total caching overhead can be controlled by the size of the query buffer. The clustering algorithm, proposed on the next section, can be used to trigger the re-sending of the queries stored in cache. Since the RVP already detects and eliminates duplicated messages, this strategy produces an asynchronous mechanism adapted for discovering PI resources.

A third problem relates to the 802.11 network characteristics. 802.11 networks are much more error prone than the Ethernet reducing TCP protocol efficiency. Current JXTA reference implementation restricts application pipes to TCP transport protocol. TCP has the advantage of creating a channel transparent to transitory network errors. However, it has a bad performance for wireless links if congestion control algorithms are not turn off, and JXTA does not allow any tuning there because it hides the transport layer used from the applications. TCP is also too expensive for short lived connections, for applications with jitter sensible semantics, and does not support one-to-many communication efficiently on a broadcast medium. So, an UDP transport protocol is needed for improving application communication.

A JXTA extension prototype was implemented with the improvements proposed on this paper for the three problems above.

3. MANET rendezvous protocol

Our MANET RendezVous Protocol (MANET-RVP) was developed as an evolution of the standard JXTA adhoc mode of RendezVous Protocol (RVP). It is composed by a clustering protocol and by a searching protocol.

3.1. The Rendezvous Clustering Protocol

The Rendezvous Clustering Protocol (RCP) was introduced to group peers into broadcast groups (BG), and to select a BG leader (BGL) that will be responsible for broadcasting packets into the BG. BGLs define geographic reference points that the searching protocol uses to restrict flooding. RCP was designed to have small maintenance costs: BGs are limited to neighbor peers (within radio range), RCP messages are short (restricted to 1-hop information); and BGL selection is local (no election specific messages are used). RCP was also designed to reduce the clustering churn that results from mixing unstable and stable peers. For instance, a group of executives on a moving van form a MANET with only stable peers, because they are all stable in relation to each other. However, they are moving in relation to peers external to the van. Classical clustering approaches [6][8] usually group peers using their ID or peer degree (the number of neighbors), originating frequent BGL changes. RCP uses a clustering algorithm based on beacon stability, a concept inspired in ABR [9].

Peers periodically broadcast short beacon messages, and keep a table of link stability counters (η) with the sum of consecutive beacons received from each neighbor. Link stability is set to null when more than one beacon is lost, producing a slight delay on link failure detection, but tolerating one beacon loss due to packet collision. A link is considered stable when η is above `stability_threshold` and a peer is stable when it has at least one stable link. In each beacon message a peer sends its ID, its BGL's ID, and the higher link stability value contained in its table, which is represented by μ. Each peer selects autonomously its BGL from the set of stable neighbor peers (including itself), before sending a new beacon. A stable peer applies the following rules:

a) When no neighbor BGLs exist, the peer with the lowest ID from the group with the highest value for η is selected as BGL;

b) When the peer was selected as BGL by a neighbor and all neighbor BGLs have higher IDs, the peer selects itself as its BGL;

c) When neighbor BGLs exist, the BGL with the lowest ID is selected.

Rule a) selects a first set of BGLs using local stability criteria, which may produce overlapped BGs. Rules b) and c) reduce the set of selected BGLs for overlapped BGs, maintaining the stability properties of

the initial selection. Notice that peers do not self-select as BGL as long as someone selects them as BGL, avoiding the appearance of a single peer BG on the periphery of existing BGs. All overlapped peers converge to the same BGL, because they all receive the same beacons and apply the same rules.

Unstable nodes are not members of a BG. However, when they receive a μ value on a neighbor's beacon above `stability_threshold` they select the peer with the lowest ID for BGL from the group with the highest μ value. Otherwise, they identify themselves as isolated, sending a null BGL on their beacons. This information is used for positioning the unstable peer on the neighborhood of a stable node. A receive knows that the peer is unstable because the μ value is above the stability threshold.

Figure 1 illustrates a MANET with three BG, six stable peers (dashed lines represent stable links) and one unstable peer (17) moving towards peer 3. The BG structure resulted from having peer 1 selected on the initial formation of the cluster. Rule a) led to the formation of a big root BGL (1) with the set of peers (1, 6, 5, 3). Peers 2 and 4 are outside the range of BGL 1, so when their links to peers 6 and 5 get stable, they select 6 and 5 as their BGL, creating two BGs overlapped with BG 1. When unstable peer 17 enters into the coverage region of peers 2, and 3, it selects one of them (3) as its BGL. Its beacon positions it in the neighborhood of peer 3. This selection does not change anything in the BG structure. However, if peer 17 remains within the same region, new links become stable, and due to rule a), peer 3 becomes a BGL. Due to rule c), peer 3 is selected as BGL by peer 2 and 17, and peer 6 stop acting as BGLs. A second possible scenario results when peer 17 is stable in relation to an external BG, with an external BGL. On this case, the original BG would not change (if 17's BGL is outside the range of peers 2 and 3) and the two BGs would be connected by two non-BGL peers (17 and 3). In both cases, RCP originates the appearance of a kind of backbone composed by BGLs, which can be used to flood the network. BGLs can be directly connected, as presented in figure 1, or at most two non-BGLs plus a variable number of unstable peers can exists between them.

Notice that maintaining the clustering information updated on an unstable network can be too costly: it requires a high frequency beacon to track rapid changes. This extra load increases the probability of packet loss due to collisions (including beacon loss), and may worsen the broadcast storm problem. Therefore, an alternative option was taken: to use low frequency beacon rates to detect peer stability relations, possibly introducing some errors on the peer neighborhood information. These errors are partially compensated by a redundant searching protocol.

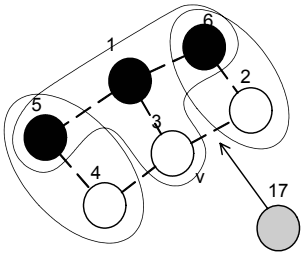

Figure 1. Illustration of a MANET with 3 BGs. Peers 1, 5 and 6 are BGLs. Peer 17 is unstable.

3.2. Searching protocol

As stated before, MANET-RVP is defined as an extension of RVP. It uses the basic flooding control features provided by RVP including: the query search duplicate elimination; the hop counting control; the answer routing using the path recorded on the query message; and the multicast flooding supported by the transport module. MANET-RVP uses the RCP beacon information to improve RVP. RCP classifies a peer as BGL (if a beacon selects it as BGL), non-BGL, or isolated (if it sends empty BGL). The proposed searching algorithm modifies RVP in three ways:

a) BGL and isolated peers always broadcast queries one time, although isolated peers delay message transmission. A non-BGL delays the sending of a query and hears the neighbor's transmissions, listing the senders ID and senders' BGL ID. It cancels the query sending if all neighbors' BGLs have already received the query;

b) Answer messages use flooding when the reverse path fails;

c) Rendezvous peers cache query messages and resend them when a new neighbor is detected.

Modification (a) reduces the number of query transmissions by non-BGLs. A peer has information about its neighbor's BGLs via beacon messages. When a non-BGL peer receives a new query, it delays the message sending for a fixed delay plus a jitter interval, and lists the visited BGL on a local variable. While the timer is active, the peer continues to receive replicas of the query message resent by neighbors. It just extracts the query path list, and updates the visited BGL list with the peer's ID and the peer's BGL ID. When the timer goes off, the peer checks to see if all its neighbors' BGLs and his own BGL are already listed. If they are not, then it resends the message to cover the missing BGLs. Otherwise, it drops the message. Since

BGLs do not delay the message and isolated peers do, search path goes preferentially over BGL peers. For BG borders defined by non-BGLs, the timer's jitter limits the number of retransmissions that occur on dense networks. The faster non-BGL on an area transmits the query to the destination BGL (or non-BGL for BGLs separated by two non-BGLs), which retransmits it. The BGL is added to the visited BGL list of other non-BGLs on the same area suspending their transmissions. This modification results from adding clustering support to SBA algorithm [10]. It trades-off searching delay and lower tolerance to packet transmission errors for lower bandwidth usage. When a message transmission fails, if other paths are available, it retransmits it over all available paths until a first echo is received.

The second modification (b) introduces a flooding mechanism for recovering from the failure of a peer on the reverse path. When the route defined by the source fails, JXTA usually looks for a relay super-peer, for locating an alternative route. This procedure may lead to successive floods, if the discovered path is also unstable. In our solution, the MANET-RVP sends the answer message piggybacked on a query message searching for the OR of JXTA IDs on the reverse path. This procedure skips failed peers, and continues to follow the reverse path once an intermediate peer is found. Experiments with simulations [11] showed that source routing fails for fast moving peers, and that flooding is more reliable for these extreme scenarios.

The third modification (c) supports the discovery of PI resources over MANETs with partitions. MANET-RVP can set a RCP callback, which is called whenever the RCP receives a beacon from a previously unknown peer. The values for the BGL ID and μ are passed to the callback to help in the decision of the worthiness of sending the cached query message again. The default behavior is to send the message as long as a minimum interval as passed since the last sending, but other behaviors can be configured. This way, MANET-RVP is capable of passing queries and answers over intermittent connections. It is also capable of handling unstable peers in the neighborhood that did not transmit a beacon before the initial flood, compensating RCP errors.

4. MANET-RVP prototype

A MANET-RVP prototype is being implemented, extending the JXTA core reference implementation on Java. Presently, it provides only partial support for the protocol presented above: it implements RCP and the query message flooding control algorithm. Answer message propagation and Query caching is still in development.

MANET-RVP was implemented on the class *ManetRvp*. In order to have access to up link and down link JXTA messages, a *ManetRvp* private object was placed in the abstract class *net.jxta.impl.rendezvous.RendezVousServiceProvider*. The methods of JXTA service initialization were changed to initialize and start it.

ManetRvp uses the endpoint service to implement RCP. Beacon messages are sent using the EndpointService's method *propagate*. Beacon messages are received using an endpoint listener registered in the EndpointService. The clustering service operation is independent of the message transport used by the EndpointService. It is only required the ability to send multicast messages, available on TCP and UDP transport modules. UDP module was also developed for this prototype. A thread is used to control the periodic beacon sending (affected by a uniform random jitter to reduce the probability of beacon transmission collisions), and to perform the BGL selection.

For implementing the searching algorithm, *ManetRvp* introduced two callbacks on the RVP implementation that intercept the reception and the transmission of messages. In order to control the propagation of query messages, the first callback was placed in the method *sendToNetwork*, called by RVP to send a query. Therefore, *ManetRvp* object is able to cancel or delay the transmission of the query messages by buffering them. The second callback was placed on the *checkPropHeader* method to access messages not processed by RVP because of repeated message ID. This callback receives all query messages resent by the neighbor peers. All messages are composed of several message elements originated by several services, including the RendezVousPropagateMessage element. This element has a single ID, time to live, and a path to control the propagation of the message in the JXTA network. The path is used to obtain the peer's ID that had propagated the message, and to fill in the visited BGL list. Non-BGL peers capture messages in the up-link and in the down-link callbacks, and implement the algorithm described in the previous section. BGL peers do not intercept messages and run the original RVP protocol. Isolated peers just delay messages on the down-link callback.

A simplified file sharing transfer application was implemented to test the MANET-RVP prototype. The communication between peers is done through three pipes: one server pipe and one client pipe to receive and send file requests in unicast; and one propagate pipe to search for resources. This propagate pipe is

created within the group and is loaded by all peers when they start, to ensure that all peers are listening to the same pipe.

On the current version, peers use the discovery service to obtain groups, peers, rendezvous and pipe advertisements. They invoke the discovery service periodically, which calls MANET-RVP to flood the network. Advertisements are cached at the discovery service, but are tested during the opening of new pipes, and cleaned when it fails. The prototype performance was evaluated measuring the number of messages exchanged during an interval of time using the message trace log created by each peer, and measuring the number of bytes multicasted to the network using Ethereal. Table 1 shows the measurements for a MANET with three peers within a distance of one hop, using RVP and MANET-RVP, with a beacon period of 5 seconds, during an interval of five minutes, and with a discovery service refreshment period of 30 seconds.

Table. 1. Measurements obtained with the prototype with RVP and MANET-RVP

MANET-RVP			
Beacons	Queries	Messages	Total
179 msg 139KB	81	133 msgs 225KB	312 msgs 364KB
RVP			
Queries		Messages	
79		245 msgs 414KB	

The results show that the total number of RVP messages propagated per query was reduced by 47%, and the total number of messages (including RVP and beacons) increased 24%. However, the total number of bytes propagated to the network per query reduced 14%. The objective is achieved because the total load was significantly reduced during the critical moment of the message query flood, reducing the probability of occurring a broadcast storm. The beacon load occurs outside the searching load peak. MANET-RVP algorithm was previously simulated in ns-2 for MANETs with hundreds of peers, and the results showed that it scales to a medium size dynamic network with two hundred peers [11]. A future application version will avoid the periodic searching using asynchronous queries (query caching), further reducing the network load for PI resources.

5. Conclusions and further work

This paper proposes a new protocol for supporting generic searching on JXTA, and presents its prototype implementation. Results showed that it can contribute to the reduction of the bandwidth usage and to the reduction of the number of messages sent during the critical flooding period, where a broadcast storm could occur. MANET-RVP can also contribute to the performance improvement of JXTA applications that use PI resources on unstable MANETs, because it can be used by all JXTA services (Endpoint Routing Protocol, etc.) and applications that use RVP.

The MANET-RVP implementation proceeds, with the addition of the two MANET-RVP protocol components missing. Further work is also being done on the protocol development. We are working on adaptive beacon strategies (to reduce its overhead), and on searching algorithms adapted to meshed networks and to less unstable MANETs, where DHT approaches can be used.

10. References

[1] JXTA home page. Retrieved from http://www.jxta.org

[2] C.E. Perkins, *Ad Hoc Networking*, AddisonWesley, 2001.

[3] Project JXTA: JXTA v2.0 Protocols Specification, 2004. Retrieved from http://spec.jxta.org/nonav/v1.0/docbook/JXTAProtocols.html

[4] B. Traversat, A. Arora, M. Abdelaziz, M. Duigou, C. Haywood, J.-C. Hugly, E. Pouyoul, and B. Yeager, "Project JXTA 2.0 Super-Peer Virtual Network", 2003. From http://www.jxta.org/project/www/docs/JXTA2.0protocols1.pdf

[5] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, "The Broadcast Storm Problem in a Mobile Ad Hoc Network", Wireless Networks 8(2/3), Mar-May 2002, pp. 153-167.

[6] F. Dai, and J. Wu, "Performance Analysis of Broadcast Protocol in Ad Hoc Networks Based on Self-Pruning", Proc. IEEE WCNC'2004, 2004, pp. 802-807

[7] Clip2, "The Gnutella Protocol Specification v0.4 Rev. 2.3", 2002. Retrieved from http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf

[8] J. Wu, and W. Lou, "Forward-node-set-based broadcast in clustered mobile ad hoc networks", Wireless Communications and Mobile Computing, N.3, 2003, pp. 155-173.

[9] C.-K. Toh, "Associativity-Based Routing for Ad-hoc Mobile Networks", Journal of Wireless Personal Communications, vol. 4, 1997, pp. 103-139.

[10] W. Peng, and X. Lu, "On the Reduction of Broadcast Redundancy in Mobile Ad Hoc Networks", Proc. MobiHoc'00, 2000, pp. 129-130.

[11] R. Oliveira, L. Bernardo, and P. Pinto, "Flooding Techniques for Resource Discovery on High Mobility MANETs", International Workshop on Wireless Ad-hoc Networks 2005 (IWWAN'05), May 2005.