Routing Different Traffic in Deep Space Networks

Chao Chen* and Zesheng Chen†
*Department of Engineering, Indiana University - Purdue University Fort Wayne, Fort Wayne, IN 46805, USA
†Department of Electrical and Computer Engineering, Florida International University, Miami, FL 33174, USA

***Abstract*—The unique characteristics of deep space networks lead to different research approaches from those in terrestrial networks. In this paper, two different routing mechanisms are proposed to addresses the delivery of remote control messages and scientific data in deep space networks. The Location-Predicted Directional Broadcast (LPDB) is proposed for the fast delivery of remote control messages and automatic data delivery. For controlled data delivery, the Receiver-Initiated On-demand Routing (RIOR) is proposed. In RIOR, the route discovery is initiated on-demand by the receiver, and routing tables are maintained in soft state at intermediate nodes. The simulation results show that LPDB and RIOR address the service requirements of different types of traffic, and are efficient in both message delivery and power consumption.**

I. Introduction

The developments in space technologies are enabling the realization of deep-space scientific missions such as Mars exploration. These missions produce a significant amount of scientific data to be collected from remote space exploration sites and delivered to the Earth. Reliable control is also required to ensure the success of the delivery. An example deep space network architecture shown in Fig. 1 helps to build a general space network architecture that combines differently challenged parts [1].

The main challenges that affect routing in deep space networks include [1]: long and variable propagation delays, intermittent connectivity, high bit error rates, power constraints, and link asymmetry. Most of these characteristics are unique to the space communication paradigm and thus lead to different research approaches from those in the terrestrial networks. A deep space network is also a special type of Delay-Tolerant Networks (DTNs) [2], where the continuous end-to-end connectivity cannot be assumed. When the connectivity patterns of the network are known, Jain et al. formulate the DTN routing problem based on different knowledge about the network topology [3]. The proposed algorithms require error-free communications, and no effective solutions are given when unpredictable link failures occur. Furthermore, the mechanisms for gathering the forwarding information through the network are not discussed.

In this paper, two different routing protocols are proposed for deep space networks to address the delivery of remote control messages and scientific data. The *Location-Predicted Directional Broadcast (LPDB)* is proposed in Section III for the fast and reliable delivery of remote control messages and automatic data reports. Paths to the destination are calculated en route based on the predictable node locations and reachability information. These paths are used to direct and limit the forwarding area of the message broadcast. For controlled data delivery that contains large amounts of scientific data from

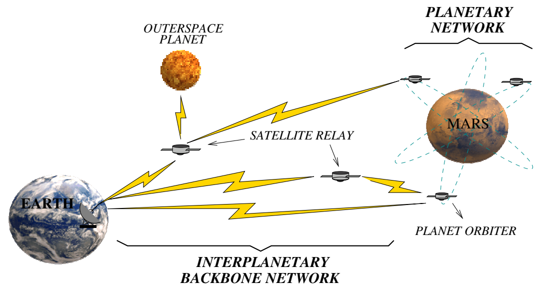


Fig. 1. An example deep space network architecture [1].

remote exploration sites back to the Earth and requires high reliability, a combination of reactive and proactive approaches is utilized in the *Receiver-Initiated On-demand Routing (RIOR)* in Section IV. The route discovery is initiated on-demand by the receiver, and routing tables are maintained in soft state at nodes along the forwarding area. No end-to-end path is recorded for the data delivery. Link state exchanges during the data transmission process provide the nodes with up-to-date path information. We demonstrate by simulation in Section V that the proposed protocols are efficient in both message delivery and power consumption, and are suitable for different traffic types in deep space networks with respect to their specific requirements.

II. Network Description

A deep space network is composed of multiple *autonomous regions* (ARs). An AR contains communication entities that are located close (i.e., much shorter than the interplanetary distance) to each other. These regions are called "autonomous" since the local nodes can communicate among themselves using a single common protocol family without intervention from other regions. The Mars planetary network in Fig. 1 is an example of AR. Here, we abstract a deep space network by an AR topology that is composed of AR nodes and AR links.

An *AR node* is an abstraction of one AR. The location of an AR node can be represented by a position within the AR. Let the position of an AR node u at time t be represented by a vector $\vec{r}_u(t)$ originating from the sun. An AR node v is *reachable* by node u and called an *AR neighbor* of node u at time t, if there exists some $\Delta_{uv}(t) > 0$ that satisfies the following condition:

$$||\vec{r}_v(t + \Delta_{uv}(t)) - \vec{r}_u(t)|| = C \cdot \Delta_{uv}(t) < L^{uv}(t), \quad (1)$$

where C is the speed of light and $L^{uv}(t)$ is the reachable range limit between node u and v. If the transmission delay can be omitted, Eqn. (1) states that a signal transmitted at time t from node u can be received by node v at time $t + \Delta_{uv}(t)$.

The reachability from u to v at time t is denoted by an *AR link* $l_{uv}(t)$.

The main traffic types through deep space networks are:

- *Remote control:* Command and control messages sent from the Earth to remote devices at exploration sites. The delivery of remote control messages is time-sensitive and requires high reliability.
- *Data delivery:* Scientific data delivered from exploration sites back to the Earth. We further classify the data delivery into two types with respect to the initiator and the service requirements:
  - *Automatic data delivery:* This type of data delivery is initiated by the mission devices at remote exploration sites, reporting mission status and some environmental data typically via repetitive transmissions. Automatic data delivery is time-sensitive and does not have strict reliability requirements.
  - *Controlled data delivery:* The Earth control center actively queries the mission devices for important scientific data delivery. In this application, the Earth center is aware of where to retrieve the scientific data from and initiates the data delivery. This type of data delivery requires a higher level of reliability.

For the above different traffic types, we propose

- Location-Predicted Directional Broadcast (LPDB) for remote control and automatic data delivery, and
- Receiver-Initiated On-demand Routing (RIOR) for controlled data delivery.

III. Location-Predicted Directional Broadcast

Although the locations of AR nodes are predictable, there exist unpredictable factors in the AR topology caused by different contact schedules at AR nodes, interferences, and power variation. Flooding is the most reliable method for the fast delivery of control messages and automatic data delivery, but at the expense of high network resource and power consumption. Therefore, we limit the broadcast area in space and time.

A control message or an automatic data delivery message contains fields of {`destAR, expireAt`}, where `destAR` identifies the destination AR node and `expireAt` indicates the time constraint set by the application. LPDB is done independently at each AR node and has two parts: *reference AR path computation* and *directional forwarding*.

A. Reference AR path computation

A reference AR path is computed according to the predictable AR topology and locally available information at the source AR. Specifically, given

- a time-varying AR topology $G(V, E(t), t)$, which composes of AR nodes V and a set of AR links $E(t)$,
- source s, destination d, and message arrival time t_s,
- the expected waiting time ω_{sv} at s to its AR neighbor v, $\forall v \in \mathcal{N}^s$, where $\mathcal{N}^s$ is a set of possible AR neighbors of AR s,

compute a fastest traversal AR path, which is a concatenation of possibly time-disjoint AR links $P_{s\to d} = (l_{sv_1}(\tau_0), l_{v_1v_2}(\tau_1), ..., l_{v_{m-1}d}(\tau_{m-1}))$, where $l_{v_{i-1}v_i}(\tau_{i-1})$ is an AR link at time τ_{i-1}. Then $\pi = (s, v_1, ..., v_{m-1}, d)$ is the topological AR path, and $\tau = (\tau_0, \tau_1, ..., \tau_{m-1})$ shows the departure times at the AR nodes on the path and is computed by $\tau_0 = t_s + \omega_{sv_1}, \tau_i = \tau_{i-1} + \Delta_{v_{i-1}v_i}(\tau_{i-1}) + \epsilon_i, 0 < i \le m-1$, where ϵ_i is the message buffering time at node v_i. The above problem can be solved using an extension of Dijkstra's algorithm in time-dependent networks [4], where the fixed link cost is replaced by the sum of message buffering time and the time-dependent link propagation delay ($\Delta_{uv}(t)$). In the first hop, this buffering time is the locally available expected waiting time (ω_{sv}). With the queuing delay omitted, the message buffering time (ϵ_i) in later hops is approximated by the waiting time for a link to be reachable, which can be calculated by the predictable location information.

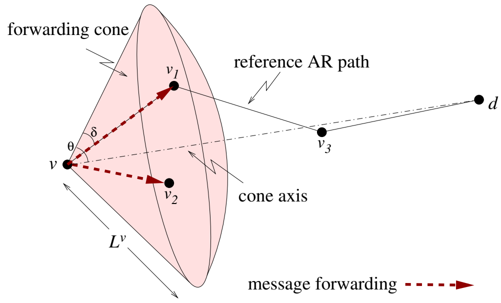


Fig. 2. Location-predicted directional broadcast.

The AR path computed in this way only represents the shortest-delay path under the condition that the queuing delay can be omitted at the computed departure time ($\tau_i, 0 < i \le m-1$) and the intermediate nodes are ready to pick up the messages at the reception time. When scheduling or retransmission delays the messages, however, the computed AR path may not be optimal or exist any more. The computed AR path is used just as a reference to direct message forwarding and is thus called the "*reference AR path*". As the actual message delivery can deviate from the pre-calculated timeline, intermediate AR nodes update the reference AR path as messages traverse deep space networks.

B. Directional forwarding

Message forwarding is limited in *space* and *time*. Specifically, suppose that an AR node v receives a message at time t_0, the topological AR path from v to the destination d is computed as $\pi = (v, v_1, ..., v_{m-1}, d)$, and the departure timeline is $\tau = (\tau_0, \tau_1, ..., \tau_{m-1})$. Then, only AR neighbors that lie within the *forwarding cone* within time interval $[t_0, \tau_0 + T_{thresh}]$ can receive a copy of the message, where T_{thresh} is a parameter set by the application or AR nodes.

As shown in Fig. 2, the forwarding cone contains the space that is within node v's transmission power range L^v and limited within cone angle θ around the axis from v to d. The *cone angle* is calculated by:

$$\theta(\pi, t) = \max_{v_i \in \pi \setminus \{v\}} \{\angle v_i v d + \delta\}, \tag{2}$$

where δ is a parameter that controls the width of the forwarding cone. In order to adjust the forwarding cone to the movement of AR nodes, $\angle v_i v d$ is computed by the predicted location of node v_i on the reference AR path, e.g., the location of v_i on the path is represented by $\vec{r}_{v_i}(\tau_{i-1}+\Delta_{v_{i-1}v_i}(\tau_{i-1}))$.

Remarks: The difference between LPDB and traditional location-aware routing protocols like LAR [5] and DREAM [6] is that no network-wide flooding is needed in LPDB to obtain nodes' location information, which can be calculated according to the orbital mechanics. Furthermore, the network connectivity intermittency caused by predictable reasons is addressed by allowing message buffering at the AR nodes. Directional forwarding provides multipath routing near the reference AR path to handle link unreliability and speed up end-to-end delivery.

IV. Receiver-Initiated On-demand Routing

The Earth-controlled data delivery carries scientific data that are usually unprocessed and very large in volume. Therefore, flooding and the directional broadcast in LPDB would consume very high network resources. This type of traffic also requires high reliability, which is difficult to achieve in the deep-space environment without redundant delivery or maintenance of up-to-date routing information. Since the Earth control center knows *when* and *where* this type of data needs to be gathered, we propose the use of on-demand route discovery initiated by the receiver, i.e., the Earth control center. Routing tables at the intermediate AR nodes that are possibly on the data delivery path are built on-demand and maintained in soft state by exploring the link status and the load distribution. The proposed routing protocol is called *Receiver-Initiated On-demand Routing (RIOR)*.

A. Route Discovery

For convenience, we call the Earth control center as the "*sink*" in this application scenario. The route discovery contains two parts: `RREQ` notification and `KeepAlive` exchanges, and routing table maintenance.

1) `RREQ` *notification and* `KeepAlive` *exchange*

At some time long enough (considering the propagation delay between the sink and the data source) before the data delivery will start, the sink initiates route discovery by sending out an `RREQ` control message to the data source AR periodically with interval T_{RREQ}. The delivery of `RREQ` messages follows the LPDB scheme as in Section III.

The reception of the `RREQ` message initiates periodical `KeepAlive` requests from the receiving AR node to the message sender, and `KeepAlive` replies in the reverse direction. The `KeepAlive` exchange interval T_{KA} is much smaller than T_{RREQ}. The `KeepAlive` exchanges serve for two purposes: To measure the delay and monitor the quality of AR links, and to build route entries to the sink AR.

Due to the constant movement of AR nodes, new rounds of `RREQ` message transmission are initiated periodically until the expected data arrives the sink or until a specified timeout limit is reached. Later `RREQ` messages may follow different reference AR paths to the data source. AR nodes on the new forwarding paths exchange `KeepAlive` messages with their neighbors and maintain route entries to the sink. This is to adapt the area of the control message exchange to node movement and AR link condition changes.

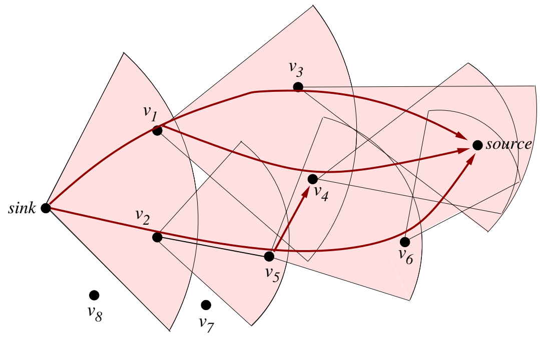


Fig. 3. `RREQ` message forwarding.

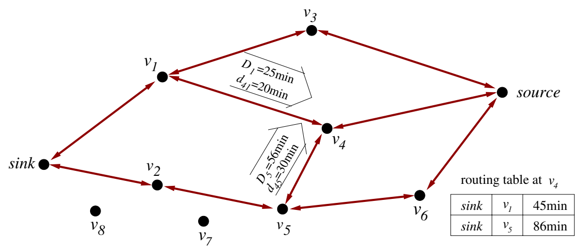


Fig. 4. `KeepAlive` message exchange.

2) *Routing table maintenance*

Routing tables at the intermediate AR nodes are built upon the reception of the `KeepAlive` reply messages. A route entry has three fields: (`sink`, `nh`, `delayToSink`), where `sink` is the destination AR; `nh` specifies one AR neighbor to reach `sink`; and `delayToSink` denotes the delay from the local AR to the `sink` by way of `nh`. There may be multiple route entries for the same `sink`, enabling multipath routing and providing alternate paths when one path fails.

A new route entry is built as follows:

- Upon receiving the `KeepAlive` request, an AR node (u) records the directional link delay (d_{vu}) from the sending node (v). d_{vu} is the time elapsed from the transmission of the request at v to its reception at u.[1]
- Node u selects the minimum `delayToSink` value (D_u) in all entries associated with the same sink.
- A `KeepAlive` reply is sent in the reverse direction (from u to v), containing the link delay (d_{vu}) and the minimum `delayToSink` value (D_u).
- Node v then retrieves the information (d_{vu} and D_u) from the `KeepAlive` reply, and builds a new route entry with parameters of (`sink`, `nh`=u, `delayToSink`=$d_{vu}+D_u$).

Once a route entry is built, the `delayToSink` is updated as the value contained in the latest `KeepAlive` reply to capture the current link property. If an AR node has not received any `RREQ` message from one AR neighbor for a long time, it stops `KeepAlive` message exchanges with this neighbor, and the corresponding route entry is removed as well.

The data delivery follows the information in local routing tables. The `nh` with the minimum `delayToSink` value is chosen as the next-hop AR (data can also be forwarded to mul-

[1] It is assumed that the AR nodes are time-synchronized. The difference in time synchronization between ARs is omittable, compared to the propagation delay on AR links.

tiple next-hops). As the delayToSink value is augmented with the propagation of the RREQ message from the sink to the source, the delay of the previous part of the path that the RREQ message traverses may be outdated. The correctness of this delayToSink value therefore decreases as the distance from an AR node to the sink grows. When the RREQ message first reaches the source AR, the minimum delay path seen from the source may not be optimal. Nevertheless, KeepAlive exchanges continue updating the route state during the data delivery and refining the remaining path toward the sink. As the data message traverses closer to the sink, delayToSink approaches its actual value.

RREQ message forwarding and KeepAlive message exchanges in an example AR topology are depicted in Fig.s 3 and 4, respectively. The forwarding of the RREQ message from the sink initiates KeepAlive exchanges between AR neighbors. The AR nodes in the forwarding paths, i.e., nodes v_1 to v_6, build and update route entries to the sink. The routing table at node v_4 is given as an example. Node v_4's routing entries to the sink are built according to the KeepAlive replies from v_1 and v_5.

B. Route Repair

As a data message traverses the network, if an intermediate AR node finds that the nh with the minimum delayToSink value is not reachable or that it cannot receive an acknowledgment from the nh after a certain number (K) of consecutive retransmission attempts, a link failure to this nh is detected. In this case, it reroutes the data message to an alternate, possibly longer delay path in the routing table. If no alternate path is available, an RREQ message is initiated and sent to the sink periodically with an interval T_{RREQ} using the LPDB scheme. KeepAlive exchanges are also initiated between AR neighbors along the forwarding path to the sink.

Remarks: RIOR executes reactively to the application requests without network-wide topology propagation. It does not look for a specific route used for the data delivery session as other on-demand routing protocols such as DSR [7] and AODV [8], since this route may be obsolete after the long route discovery phase. Upon the detection of link failures, DSR and AODV notify the sender node, which then restarts the route discovery process. In RIOR protocol, on the other hand, KeepAlive messages are utilized to obtain the up-to-date link property, so that the updated route entries reflect more recent delay metrics. The maintenance of multiple route entries provides alternate routing options. New route discovery can be issued at any intermediate node that encounters link failure. These mechanisms help RIOR adapt fast to the changes in deep space networks with long and variable delays. Moreover, RIOR does not assume link symmetry, and all delay measurements are directional.

Directed diffusion [9] proposed for wireless sensor networks is also a receiver-initiated protocol. The feedback-based adjustment and the end-to-end negotiation in directed diffusion, however, are not applicable in deep space networks with long delays.

V. Performance Evaluation

An event-driven simulator on C++ is developed to evaluate the performance of proposed routing protocols. The AR nodes in the network model are 9 planet ARs and 18 planetary Lagrangian ARs. To simplify the model, all planets move around the sun in circular orbits in the same plane. The orbit radii of the planets are from 1 to 9 AU (1 AU $\approx$ 149,600,000 km), with 1 AU in between. The planet orbiting period $\mathcal{T}$ (in year) is decided by the Kepler's third law: $\mathcal{T}^2 = a^3$, where a is the orbit radius in AU. The Lagrangian ARs are on the same orbits, 60^o ahead or behind of the associated planets. The reachable range limits of all AR pairs are set as 5 AU.

In each simulation round, the initial positions of planet AR nodes are random with central angles uniformly distributed in $[0^o, 360^o)$. Control/Data messages are sent in 1 hour interval. All results are the averages of 20 simulation rounds, each of 1 day long. Control/Data message delivery performances are evaluated under different link failure probabilities (p). Each link between AR neighbors is prone to failure, and the failure is independent across different links. Three message delivery metrics are measured to evaluate the routing performance: the *delivery ratio* and the *average delay* of successfully delivered messages; and the *transmission cost*, i.e., the average number of transmissions for each successful message delivery.

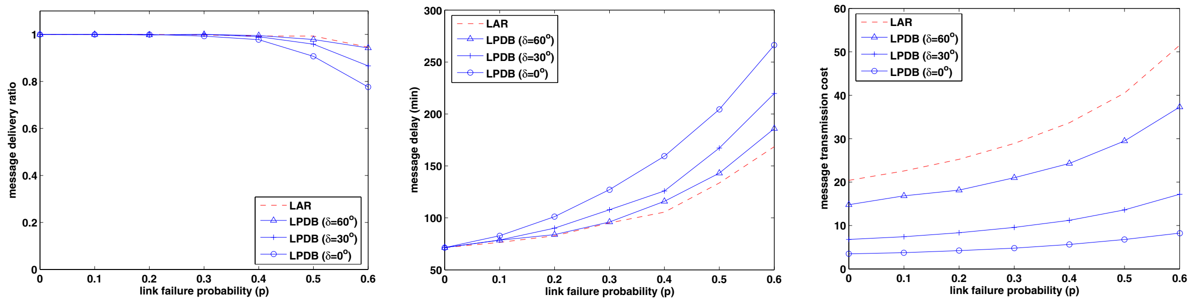
A. LPDB for Remote Control and Automatic Data Delivery

For remote control and automatic data delivery, we compare LPDB with *Location-Aided Routing (LAR)*, which is the LAR scheme 2 in [5] with $\alpha = 1$ and $\beta = 0$, i.e., a message is forwarded from the current AR node to all neighbors that are closer to the destination. The computation of distance considers the movement of the destination node. To measure the effect of the forwarding cone angle on the performance of the LPDB scheme, three different values of δ are chosen in the simulation. The case $\delta = 0^o$ provides the performance bound for LPDB.

Messages are sent from node 0 (orbit radius = 1 AU) to node 7 (orbit radius = 8 AU) in a store-and-forward manner, with a maximum life time as 6 hours. Each AR node stores a copy of a message and should make sure that every next-hop AR gets a correct copy of the message via per-hop acknowledgment before it removes its local copy.

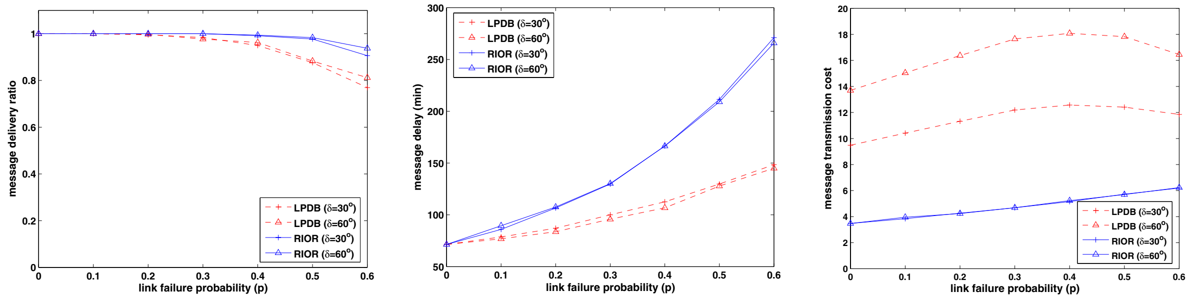
Fig. 5(a) shows the message delivery ratio of these two schemes under different link failure probabilities (p). For LPDB, the delivery ratio is always higher than 90% when $p \leq 0.5$. As the forwarding cone angle increases, more messages are successfully delivered to the destination. LAR provides a high degree of redundancy for the message delivery. Thus, it achieves low end-to-end delay but with high transmission cost, as can be seen in Fig.s 5(b) and 5(c), respectively. As the forwarding cone angle increases, LPDB's delay and ratio performance gets closer to LAR. Even in the case of $\delta = 60^o$, LPDB can achieve the same delivery ratio but with less transmission cost than LAR at $p = 0.6$.

In summary, LPDB balances between reliability and redundancy, as well as between the delay and the transmission cost. AR nodes can change the forwarding cone angle by adjusting δ, which in turn controls the message delivery performance.



(a) Message delivery ratio. (b) Message delay. (c) Message transmission cost.

Fig. 5. Performance comparison between LPDB and LAR.



(a) Message delivery ratio. (b) Message delay. (c) Message transmission cost.

Fig. 6. Performance comparison between RIOR and LPDB.

B. RIOR for Controlled Data Delivery

We compare our RIOR scheme with LPDB, where the parameter δ in LPDB controls the width of the forwarding cone, whereas δ in RIOR controls the forwarding cone of RREQ messages. The transmission cost refers to the average number of data message transmission for each successful delivery. We set $T_{RREQ} = 1$ hour, $T_{KA} = 10$ minutes, and $K = 3$. Data messages are sent from node 7 to node 0. The maximum life time for data messages is 10 hours. No multipath forwarding is utilized in RIOR in our simulation.

Fig. 6(a) shows that RIOR always keeps the delivery ratio higher than 90%, even when p is as high as 0.6. Compared with RIOR, LPDB is less reliable especially when p is high. This is because LPDB does not provide any route repair under link failures. As a benefit of the multipath transfer, LPDB results in a lower message delay than RIOR as in Fig. 6(b), thus is more suitable for messages that require fast delivery. When the transmission cost is concerned, as seen in Fig. 6(c), RIOR costs much less data overhead, which is around 30% to 50% of LPDB with $\delta = 60^o$ and 30^o, respectively. Note that the transmission cost of LPDB does not show consistent growth as p increases. When p grows beyond 0.4, the message delivery ratio starts to decrease fast, more messages get dropped or lost early on the delivery path, and only "lucky" ones reach the destination. As the result, the average transmission cost for the successful end-to-end deliveries is lower compared to the cases when p is smaller.

VI. Conclusions

To address the challenges in the deep-space communication environment and meet the application requirements of control and data delivery, Location-Predicted Directional Broadcast (LPDB) and Receiver-Initiated On-demand Routing (RIOR) are proposed. LPDB provides the fast delivery of messages with considerably smaller sizes and lower reliability requirements; whereas RIOR achieves higher data delivery reliability when there is no strict time constraint. LPDB and RIOR can be utilized to deliver different types of traffic in deep space networks, addressing specific requirements in each application scenario.

References

[1] I. F. Akyildiz *et al.*, "InterPlaNetary Internet: state-of-the-art and research challenges," *Computer Networks Journal (Elsevier Science)*, vol. 43, no. 2, pp. 75-112, Oct. 2003.

[2] S. Burleigh *et al.*, "Delay-tolerant networking: an approach to Interplanetary Internet," *IEEE Communications Magazine*, vol. 41, no. 6, pp. 128-136, June 2003.

[3] S. Jain, K. Fall, and R. Patra, "Routing in a delay tolerant network," in *Proc. of ACM SIGCOMM*, pp. 145-158, Aug.-Sept., 2004, Portland, USA.

[4] A. Orda and R. Rom, "Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length," *Journal of the ACM*, vol. 37, no. 3, pp. 607-625, July 1990.

[5] Y-B Ko and N. H. Vaidya, "Location-aided routing (LAR) in mobile ad hoc networks," *ACM/Kluwer Wireless Networks*, vol. 6, no. 4, pp. 307-321, 2000.

[6] S. Basagni, I. Chlamtac, R. Syrotiuk, B. Woodward, "A distance routing effect algorithm for mobility (DREAM)," in *Proc. of IEEE/ACM MOBICOM '98*, Dallas, USA, Oct. 1998.

[7] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad-hoc wireless networks," *Mobile Computing*, edited by T. Imielinski and H. Korth, Kluwer Adademic Publishers, pp. 153-181, 1996.

[8] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Proc. of IEEE Workshop on Mobile Computing Systems and Applications*, pp. 90-100, Feb., 1999.

[9] C. Intanagonwiwat *et al.*, "Directed diffusion for wireless sensor networking," *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, pp. 2-16, Feb. 2003.