Partially Known Routing for Satellite IP Networks

Yu Zhang, Feng Liu
School of Computer and Information Science, Southwest University, Chongqing, China
Email: zhangyu, liuf@swu.edu.cn

***Abstract*—Satellite IP network (SIPN), which includes satellites whose orbit information is known or unknown is a kind of Delay- and Disruption-Tolerant Networking. This paper introduces a new Partially Known Routing (PKR) for SIPN since there is no efficient routing method for this partially known network. Available orbit information is used by PKR to estimate next meeting time between satellites. The meeting time is mapped to the edge weight of a graph. Then a satellite set which includes all satellites in the shortest path is outputted by PKR algorithm, which is a based on extended Dijkstra algorithm. Finally, PKR routes packets based on the satellite set. If the satellite set is not null, PKR forwards just one copy of a packet to only one satellite in the satellite. Otherwise, PKR will send multi-copies of a packet to other satellites.**

The simulation result shows that PKR has better performance on packet loss rate and transferring throughput under all traffic loads; the average end-to-end delays of PKR is a little higher than Epidemic routing under lower traffic loads, but the delays is close to and even better than Epidemic routing under higher traffic loads. PKR fits for the routing where higher loss rate and throughput are required. It also routes packets timely when the traffic load is high.

***Index Terms*—Partially Known Routing; satellite IP networks; Epidemic routing; DTN**

I. INTRODUCTION

A. Requirement

Satellite IP networks (SIPN) is a part of interplanetary Internet[1]. Satellite is the communication node of the network. As shown in Fig. 1, each satellite runs along its orbit. One may encounter another in a communication region, in which the two satellites can talk or exchange packets. The satellites may move around Earth, Moon, Mars, etc. There are chances for them to communicate when they encountered in their communication regions.

SIPN may be huge and complex. Sometimes, some satellites know orbit information of others. For example, the satellites in a same constellation around Earth always know the orbit information of each other. We found this information is very useful for routing in SIPN. Because the motion pattern can be learned from the known orbit information, and the encounter time can be estimated. This time information can be used to calculate the routing path.

At the same time, some satellites' orbit information is totally unknown to others. For example, 1) a satellite of Earth may know nothing about another one of Moon; 2) a satellite has no knowledge about another one which launched later; 3) a satellite loses the information about another one whose orbit is changed due to different mission. Normally a SIPN consists of the satellites with known orbit information and the satellites without this information. This kind of network is called partially known SIPN in this paper.

The right side of Fig.1 also shows an example of partially known SIPN with four satellites. Each satellite has its own orbit. Only when two satellites get close enough, they can communicate with each other. The range, in which one satellite can communicate with another one, is defined as communication intersection. $ab1$ and $ab2$ are the communication intersections for satellite S_a and S_b. Only in the intersection, the two satellites have the possibility to communicate. In such a case that satellite S_a holds the orbit information of S_b and S_c, but knows nothing about S_d. So S_a can use this information to estimate the encounter time with S_b in communication intersections $ab1$ or $ab2$, and the encounter time with S_c in $ac1$ or $ac2$. However, satellite S_a will not get the encounter time with S_d by calculation. So the four satellites form a partially known SIPN in which one satellite only knows part knowledge of others.

Partially known SIPN is a special kind of IP networks. To the best of our knowledge, there is no efficient routing method designed for partially known SIPN. So PRK is presented in this paper.

B. Related Work and Motivation

SIPN is also a kind of Delay- and Disruption-Tolerant Networking (DTN)[2]. DTN provides communication in highly stressed environments such as variable delays, discontinuous connectivity and high bit error rate[3].

For DTNs, there exists huge diversity among the several different application scenarios. [4] raises a question: *"Can one protocol stack deal with all potential DTN application scenarios?"* The answer to the problem is clearly negative. As in DTNs, the tasks are different in each application; the construction and purpose of DTNs are also different and may change with time. Furthermore,

doi:10.4304/jsw.8.8.1851-1858

DTNs have the characteristic, such as the mobilization model is heterogeneous; the end-to-end connectivity may not exist; the computing capabilities are different; the buffers and other resources are different. Therefore, different solutions should be presented for kinds of application scenarios.

Traditional routing method for Internet does not meet the requirement of DTN. Open Shortest Path First (OSPF)[5], Intermediate System to Intermediate System (IS-IS)[6] and Optimized Link State Routing (OLSR)[7] are used in traditional Internet. They flood topology changes to all nodes in a routing area to maintain link states and to avoid routing loops. But in DTNs, maintaining link states is something practically impossible. There is no known end-to-end route in DTNs. So Distance Vector protocols such as Border Gateway Protocol (BGP) and Enhanced Interior Routing Gateway Protocol (EIRGP), which have built-in routing loop detection mechanisms, can not be applied in DTNs either [8].

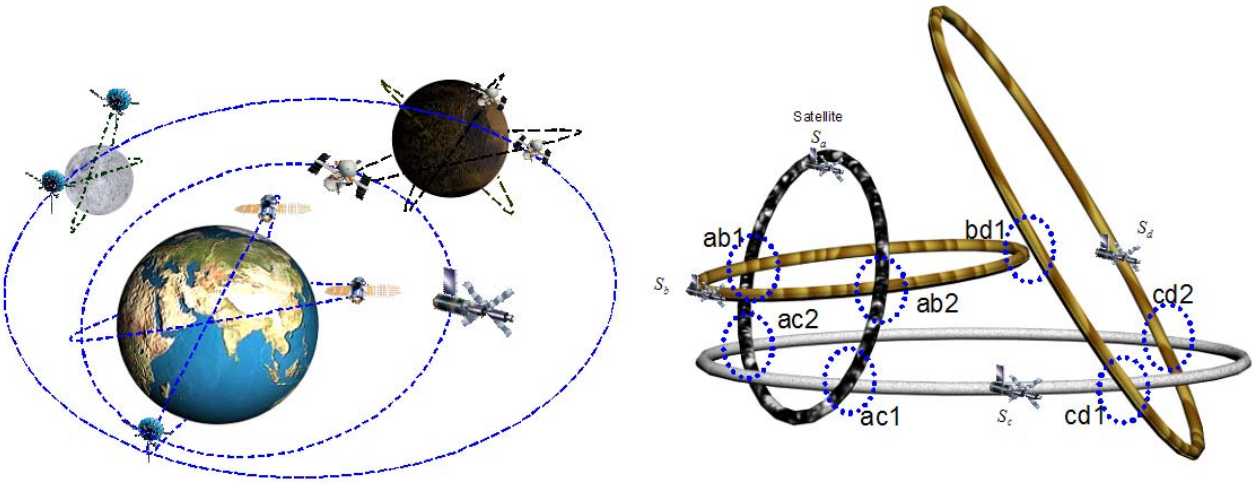

Figure 1. Partially known SIPN

Routing in DTNs is very challenging. Many researches carry out research on DTNs routing. In [9], the developments in this area is addressed into three different sub-categories: Message Ferrying Approach, Inter-Region Routing and Multicast Routing. Multicast routing methods show higher performance in [10-12]. It may be used in Satellite IP networks routing.

Epidemic routing [10] is wildly used. It relies upon the transitive distribution of messages through ad hoc networks, with messages eventually reaching their destination. Each host maintains a buffer consisting of messages that it has originated as well as messages that it is buffering on behalf of other hosts. When two hosts come into communication range of one another, the host with the smaller identifier initiates an anti-entropy session. During anti-entropy, the two hosts exchange their summary vectors to determine which messages stored remotely have not been seen by the local host. In turn, each host then requests copies of messages that it has not yet seen.

When adopting Epidemic as the routing method for Satellite IP network, we found that the packet loss rate may be high and the throughput may be low. The buffers in satellite are very limited. Epidemic may produce more copies of packet. At the time of all the buffers are used, the coming new packets will overwrite the packet waiting in queue. This may lead to the loss of the packet. We also found that some orbits of satellites are known and the orbits keep constant. This information can be used to find the best path for a packet. So we introduce a new routing method, named Partially Known Routing (PKR), for satellite IP networks which parts of the orbit information are known. PKR takes advantages of the known satellite orbit, and uses flooding method for unknown nodes.

C. Paper Construction

The paper is organized as follows. The estimation of encounter time is based on satellite orbit. The introduction to satellite orbit is presented in Section II. The PKR routing method is given in Section III in detail. The performance evaluation and analysis of PKR and Epidemic routing is presented in Section IV. Finally, Section V concludes the paper.

II. SATELLITE ORBIT

The motion pattern of a satellite is decided by its orbit. Fig. 2 illustrates the Keplerian orbital elements and a satellite position[13]. The satellite's position in an ideal, non-perturbed orbit can be represented by:

- size and shape of the ellipse : semi-major axis a and eccentricity e;
- orientation of the orbital plane relative to the Earth : orbit inclination I and longitude of the ascending node Ω;
- orientation of the ellipse in the orbital plane : argument of perigee ω;
- satellite position in the ellipse : true anomaly U; and
- a reference time when passes the perigee : t_p.

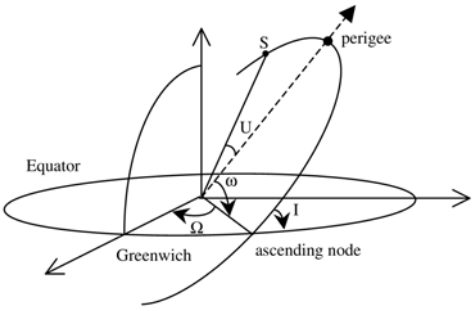

Figure 2. Keplerian orbital

The "natural" satellite orbital plane system is defined by the origin located at one focus of the elliptical orbit, which corresponds to the position of the mass centre of the Earth. The x-axis and y-axis are coincident with the major and minor axes of the orbital ellipse respectively. Fig. 3 illustrates the natural orbital plane system where the positive x-axis passes through perigee. A satellite position in the natural orbital plane system can be expressed by Eq.(1), where a is the semi-major axis of the satellite orbit ; e is the eccentricity of the orbit ; E is the orbital eccentric anomaly ; r is the instantaneous distance between the satellite and the centre of the Earth ; and U is the true anomaly.

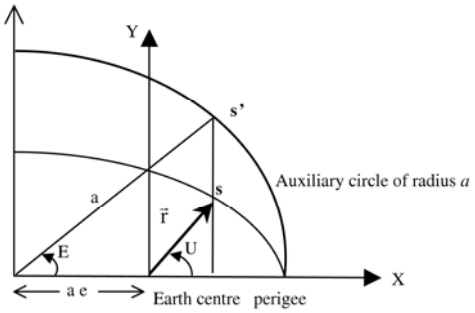

Figure 3. the natural orbital plane system

$$\vec{r} = \begin{pmatrix} x \\ y \\ 0 \end{pmatrix} = \begin{pmatrix} a\cos E - a \cdot e \\ a\sqrt{1-e^2}\sin E \\ 0 \end{pmatrix} = \begin{pmatrix} r\cos U \\ r\sin U \\ 0 \end{pmatrix} \qquad (1)$$

Using the coordinates x and y, which denote the satellite's position in the orbital plane with respect the center of Earth, one may express the real velocity $h = |\mathbf{h}|$ as a function of

E : $h = x \cdot \dot{y} - y \cdot \dot{x} = a^2\sqrt{1-e^2}\dot{E}(1-e\cos(E))$ [14] .

This equation may further be simplified using $h = \sqrt{GM_{\oplus}a(1-e^2)}$ to give the following differential equation for the eccentric anomaly:

$$(1-e\cos E)\dot{E} = n .$$

Here the mean motion $n = \sqrt{GM_{\oplus}/a^3}$ has been introduced to simplify the notation. $GM_{\oplus}$ is the gravitational coefficient, i.e. the product of the gravitational constant and the Earth's mass. It has been determined with considerable precision from the analysis of laser distance measurements of artificial Earth satellites:

$$GM_{\oplus} = 398600.4405 \pm 0.001 km^3 s^{-2} .$$

Integrating with respect to time finally yields Kepler's Equation $E(t) - e\sin E(t) = n(t - t_p)$, where t_p denotes the time of perigee passage at which the eccentric anomaly vanishes. The right side $M = n(t - t_p)$ is called the mean anomaly. It changes by 360° during one revolution, but in contrast to the true and eccentric anomalies, increases uniformly with time[14]. The orbital period is proportional to the inverse of the mean motion n and is given by

$$T = 2\pi / n = 2\pi\sqrt{a^3 / GM_{\oplus}} .$$

From Fig. 2, it can be seen that the transformation of the Cartesian components of a satellite position vector $\vec{r}$ from the orbital plane coordinate system to the ECEF system (Earth-Centered, Earth-Fixed coordinate system[15]) may be carried out by three rotations in the following order:

- a first rotation by the argument of the perigee ω;
- a second rotation by the angle of inclination I ; and
- finally a rotation by the angle of the longitude of ascending node Ω .

The corresponding transform equation is calculated by Eq.(2), where $R_n(\theta)$ is the rotation matrix, the subscript $n = 1,3$ corresponding to the rotation axes of x, z respectively.

$$\vec{r}_{ECEF} = R_3(-\Omega) \cdot R_1(-I) \cdot R_3(-\omega)\vec{r} \qquad (2)$$

The rotation matrixes $R_3(\theta)$ is expressed in the forms of

$$R_3(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The $R_1(\theta)$ is

$$R_1(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix}$$

III. PARTIALLY KNOWN ROUTING

In satellite IP networks, some satellites' orbits information is known to others. For example, typically the satellites have knowledge about orbits of others in the same constellation. With this information, a satellite can get other's position by computation. The satellite can also know whether other satellite locates in its communication regions. Then it can estimate the time when it encounters other one. This encounter time information is useful for finding the appropriate routing path.

PKR is introduced for partially known SIPN. Encounter time information is used in PKR to find best routing path. However, PKR does not require the satellite to know all the orbits information. It allows satellites runs in the network, which are totally unknown to others.

As shown in Fig. 4, PKR has four steps: 1) calculating encounter time, 2) mapping to a graph, 3) finding node sets and 4) forwarding packets.

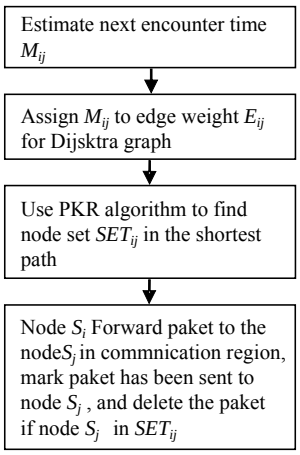

Figure 4. Phase I Routing

A. Calculating Encounter Time

If satellite S_i knows the orbit of S_j , then S_i may estimate the encounter meeting time with S_j through computation. For calculating the encounter time between any two satellites, an iteration procedure can accomplish this job. The iteration may begin with a small value of reference time t_p . With a given t_p , the satellites' position can be obtained from Eq.(1) and Eq.(2). Then the two satellites can know whether they locate in the communication region.

- For satellites known orbit information of others

If they are just entering the region, time t_p will be marked as the next encounter time. The encounter time from Satellite S_i to S_j is denoted with M_{ij} .

If not, iteration will begin with a little greater t_p . The iteration will terminate when all M_{ij} s are obtained or a given time has elapsed.

Under this case, Algorithm 1 can be used to calculate the encounter time. The increment variable *Delta_time* will decide the computational accuracy. With a small *Delta_time*, the computation of the algorithm will be huge, but it outputs result with higher accuracy. Otherwise, with a great *Delta_time*, the algorithm will output lower accuracy result, but runs fast and saves time.

Algorithm 1 Calculating encounter time

```
// _time: the given time t_p
// Delta_time: the increment variable
while (_time < End time) do
   satellites' position with the time _time;
   calculate the distance between satellites;
   check whether the satellites encountered at time _time;
   if encountered, mark and save the encounter time;
   if all the encounter time are got, then exit;
   // Increase the time variable time
   _ time:= _ time + Delta_time;
endwhile;
```

- For satellites known orbit information of others

If the orbit of satellite S_j is unknown to satellite S_i , then M_{ij} is set to ∞ .

B. Mapping to A Graph

The satellites are mapped to nodes of a graph. After the estimation of all M_{ij} s are finished, M_{ij} is assigned to E_{ij} . E_{ij} is the weight of edge from node i to j . Finally, a bidirectional graph with known edge weigh is formed, as shown in Fig. 5.

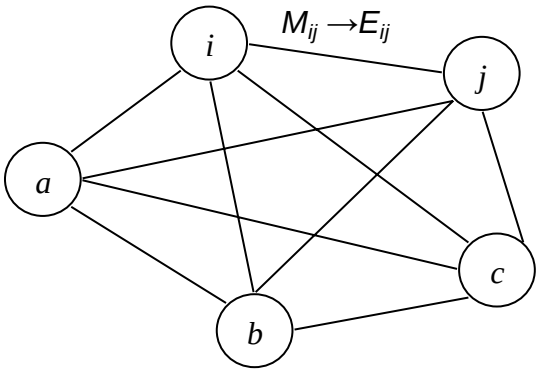

Figure 5. Phase I Routing

Based on the bidirectional graph, shortest path algorithm, like Dijkstra[16], Bellman-Ford[17], Floyd-Warshall[18], can be used to find the shortest path. A modified Dijkstra is used in PKR.

C. Finding Node Sets

PKR algorithm is based on Dijkstra shortest path algorithm[16] with some modifications. There are two main differences from Dijkstra algorithm.

- PKR algorithm outputs a set of nodes

Dijkstra algorithm will output a shortest path between any two nodes. For satellite S_j and S_j , the path may looks like $S_i \rightarrow S_2 \rightarrow S_3 \rightarrow S_j$. While PKR algorithm outputs a set like $SET_{ij} = [2, 3, j]$.

- PKR algorithm outputs a null set, i.e. $SET_{ij} = [\,]$, if the total weight of the path $\sum E_{ij} = \infty$.

For example, if Dijkstra algorithm outputs a path $S_i \rightarrow S_2 \rightarrow S_3 \rightarrow S_j$, and the total

weight $E_{i2} + E_{23} + E_{3j} = \infty$, then PKR will output a null set.

The algorithm is illustrated in Algorithm 2.

Algorithm 2 PKR algorithm

```
// SolvedSet, UnSolvedSet: the sets of points which have been handled and not handled by PKR algorithm
// PkrSet: the set of points outputted by PKR algorithm
// EdgeWeight: the weight of edge in the graph
//DynamicWeight: the weight from the source point to current point
// PointsNum: the number of points in the graph
// StartPoint: the point specified by user
//initialize
Put StartPoint into SolvedSet;
Delete StartPoint from UnSolvedSet;
Put StartPoint into PkrSet;
Set PkrPathWeight[StartPoint] to 0;
NextPoint := 0;
//while
while UnSolvedSet is not null do
   //find a point(NextPoint) in UnSolvedSet which has the smallest weight
   MinWeight := +∞;  // MinWeight: a temperate real variable
   for i := 0 to PointsNum - 1 do
      if i is in UnSolvedSet then
         if DynamicWeight[i] <= MinWeight then
           MinWeight := DynamicWeight[i];
           NextPoint := i;
         endif;
      endif;
   endfor;
  // Update the weight
   for i := 0 to PointsNum - 1 do
      if i is in UnSolvedSet then
         if (DynamicWeight[NextPoint] + EdgeWeight[NextPoint, i]) < DynamicWeight[i] then
           DynamicWeight[i] := DynamicWeight[NextPoint] + EdgeWeight[NextPoint, i];
         endif;
         PkrPathWeight[i] := DynamicWeight[i];
      endif;
   endfor;
   // find the set
   for i := 0 to PointsNum - 1 do
      if i  is in SolvedSet then
        if Round(PkrPathWeight[i] + EdgeWeight[NextPoint, i]) = Round(PkrPathWeight[NextPoint]) then
             PkrSet[NextPoint] := PkrSet[i] + [NextPoint];
        endif;
      endif;
   endfor;
   //update SolvedSet and UnSolvedSet
   Put NextPoint into SolvedSet;
   Delete NextPoint from UnSolvedSet;
 endwhile;
```

D. Forwarding Packets

Each Satellite will keep on checking whether another one is in the communication region. When two satellites meet in the communication region, they use following procedure to route packets, as shown in Fig. 6.

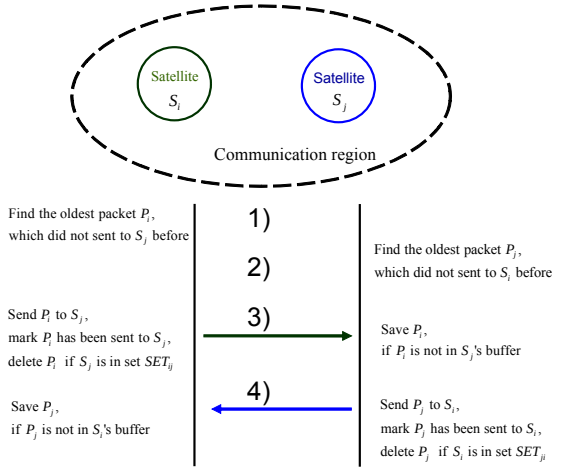

Figure 6. PKR Routing

Satellite S_i finds the oldest packet P_i which did not send to Satellite S_j before;

Satellite S_j finds the oldest packet P_j which did not send to Satellite S_i before;

As soon as the link from S_i to S_j is available, S_i sends P_i to S_j and marks P_i has been sent to S_j; when Satellite S_j received P_i, it checks whether it has been save in the buffer. If yes, desert P_i; if no, save P_i into the buffer;

When the link from S_j to S_i is available, Satellite S_j sends P_j to S_i and marks P_j has been sent to S_i; When Satellite S_i received P_j, it checks whether it has been save in the buffer. If yes, desert P_j; if no, save P_j into the buffer.

IV. SIMULATION

PKR simulator is developed with Delphi programming language. In the simulation, Satellite 1, 2, 3 and 4 forms Group1. Satellites in Group 1 are assumed to know the orbits information each other. Group 2 includes Satellite 5 and 6. But satellite 5 and 6 do not know the orbits each other. Satellites in Group 1 have not the orbits information of satellites in Group 2, and vice versa. It can be got that, in this network only partial orbits information is known. PKR is simulated on this partially known routing scenario. The parameters of the orbits information is shown in TABLE I. .

TABLE I.
SATELLITE ORBIT PARAMETERS

Satellite orbit parameters	Group 1				Group 2	
	1	2	3	4	5	6
semi-major axis a (km)	12000	13000	14000	15000	16000	17000
eccentricity e	0.6	0.1	0.09	0.6	0.31	0.2
orbit inclination I ($^\circ$)	0.1667	0.0278	0.1977	0.1861	0.1194	0.2222
longitude of the ascending node Ω ($^\circ$)	0.0833	0.0833	0.0833	0.0833	0.0833	0.1056
argument of perigee ω($^\circ$)	0.1139	0.1722	0.2222	0.1139	0.0306	0.2083
reference time t_p	36	2	10	36	3	26

First in first out (FIFO) queue is used in the simulation. When the buffer is fully used, the coming of a new packet will overwrite the oldest packet in the buffer. If all copies of the overwritten packet have not reached the destination, the packet will be marketed as a loss.

The simulation is carried out under different traffic load n, $n = 1,2,...,12$. Where n is the number of packet(s) from one to others satellite in each 10 minutes. For example, for traffic load 5, there are 5 packets generated from S_i to S_i, $j \neq i$, $j \in S_{all}$ every 10 minutes, where S_{all} is the universal set of satellites.

The simulation time is set to 48 hours or two days.

A. Simulation Result - Loss Rate

The packet loss rate of PKR is lower than Epidemic routing, as shown in Fig. 7. A packet will be regarded as loss when it does not reach to the destination and all the copies have been deleted in satellites' buffer. PKR uses the knowledge of orbit to predicate the next encounter time of satellites which know the orbit information of each other. In the next step, the next encounter time information is used to find the shortest path. Finally, PKR outputs a satellite set which includes all the satellites in the shortest path.

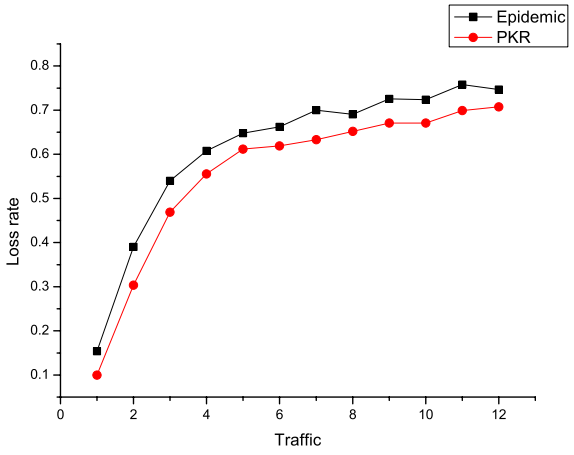

Figure 7. Loss rate

When forwarding the packets, PKR will check whether the satellite set is null. If yes, PKR forwards the packets to all satellites met. If not, PKR only forwards the packets to the satellites in satellite set. By this mean, fewer packet copies are generated and reduce the possibility to overwrite other packets. So packets can get more time and chances to reach its destination. It can be seen easily in Fig. 7, PKR has lower loss rate than Epidemic routing under each traffic load. Comparing to Epidemic routing, it can be drawn that PKR should be used if lower loss rate is required.

B. Simulation Result - Throughput

From Fig. 8, it is known that PKR has higher throughput than Epidemic routing. Under each traffic load simulated, the number of reached packets of PKR is bigger than Epidemic routing.

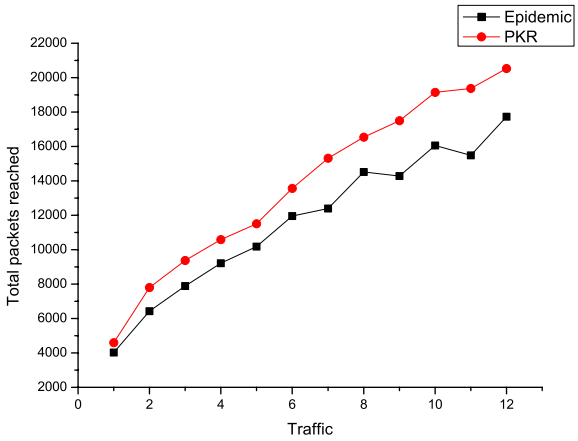

Figure 8. Throughput

As analyzed before, PKR generates fewer packet copies than Epidemic routing. The chance for each packet to reach its destination is higher. PKR uses orbit information partially known to decide the routing path. For these satellites with known orbit information, PKR can route their packets efficiently with fewer copies and lower loss. For satellites without known orbit information, PKR works as normally flooding routing does. The performance is close to Epidemic routing. So the overall throughput performance of PKR is better.

C. Simulation result - Average Delay

As shown in Fig. 9, under lower traffic load, the average end-to-end delay of PKR is a little higher than Epidemic routing. Under higher traffic load, the average end-to-end delay performance of PKR is close to Epidemic routing.

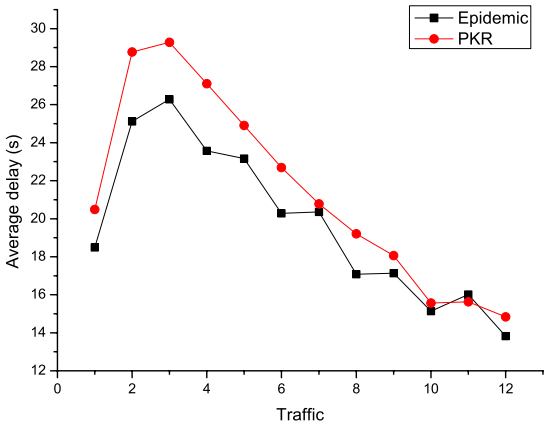

Figure 9. Average delay

Epidemic routing does not use any orbit information to decide the routing path. It forwards all packets to the satellites it met. So the number of packet copies is increasing scientifically with the number of nodes in the network. Under lower traffic load, more copies of a packet help to reach its destination timely. For computing easily, our simulator estimates the next encounter time roughly. So PKR gets higher average end-to-end delay than Epidemic routing. If the simulator computes with higher accuracy, the performance of PKR will be more close to Epidemic routing.

Under higher traffic load, PKR performs similarly as Epidemic routing, as the right part of the curve in Fig. 9 showed. The buffer of the satellites is fully filled because the high traffic loads lead to generate more packets in the queues. Packets forwarded by PKR or Epidemic routing are suffered high average end-to-end delays. Comparing with Epidemic routing, the higher traffic loads go, the better PKR performs

V. CONCLUSION

Routing method is a key part of satellite IP networks of routing. It is found in this paper that, as a kind of Delay Torrent Network, satellite's motion pattern may be estimated by using its orbit information. But in a satellite IP network, not all orbits can be known for every satellite. Normally, a satellite only knows part of them. This partially known information is also valuable for routing. So the paper presented a new routing method for partially known satellite IP networks (PKR).

PKR uses orbit information of part satellites to estimate next meeting time between satellites. Then the meeting time is mapped to the edge weight of a graph. The satellite set which includes all satellites in the shortest path is outputted by PKR algorithm. Finally, PKR forwards packets based on the satellite set. If the satellite set is not null, PKR will forward one copy of a packet to only one satellite in the satellite. Otherwise, packets will be flooded to other satellites.

From the simulation result, PKR has better performance on packet loss rate and transferring

throughput under all traffic loads; the average end-to-end delays of PKR is a little higher than Epidemic routing under lower traffic loads, but the delays is close to and even better than Epidemic routing under higher traffic loads. It can be drawn that PKR can be used when higher loss rate and throughput are required, or when routing with high traffic load.

The source code of our simulator is open and uploaded to the Internet. It can be downloaded at http://www.zuotiwang.com/simulator/pkr.zip. Through this way, other researchers may check our simulator, reproduce the simulating, and reuse the source code for further research in this field.

ACKNOWLEDGMENT

This work is supported in part by Chongqing Engineering Research Center for Instrument and Control Equipment.

REFERENCE

[1] Cerf, V., et al., *Interplanetary Internet (IPN): Architectural Definition*. 2001.
[2] Fall, K. and S. Farrell, *DTN: an architectural retrospective.* Selected Areas in Communications, IEEE Journal on, 2008. 26(5): p. 828-836.
[3] Yanggratoke, R., et al., *Delay Tolerant Network on Android Phones: Implementation Issues and Performance Measurements.* Journal of Communications, 2011. 6(6): p. 477-484.
[4] Psaras, *I*., L. Wood, and R. Tafazolli, *Delay-/Disruption-Tolerant Networking: State of the Art and Future Challenges*. 2010, Technical Report, University of Surrey, UK.
[5] Moy, J., *OSPF version 2*. 1997.
[6] Smit, H. and T. Li, *Intermediate system to intermediate system (IS-IS) extensions for traffic engineering (TE).* 2004.
[7] Jacquet, P., *Optimized link state routing protocol (OLSR).* 2003.
[8] Khabbaz, M., C. Assi, and W. Fawaz, *Disruption-Tolerant Networking: A Comprehensive Survey on Recent Developments and Persisting Challenges.* Communications Surveys & Tutorials, IEEE, 2011(99): p. 1-34.
[9] Zhang, Z. and Q. Zhang, *Delay/disruption tolerant mobile ad hoc networks: latest developments.* Wireless Communications and Mobile Computing, 2007. 7(10): p. 1219-1232.
[10] Vahdat, A. and D. Becker, *Epidemic routing for partially connected ad hoc networks*. 2000.
[11] Wang, Y., X. Li, and J. Wu, *Delegation Forwarding in Delay Tolerant Networks Multicasting.* Journal of Communications, 2011. 6(5): p. 384-392.
[12] Zhao, W., M. Ammar, and E. Zegura. *Multicasting in delay tolerant networks: semantic models and routing algorithms*. in *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*. 2005: ACM.
[13] Zhang, J., et al., *GPS satellite velocity and acceleration determination using the broadcast ephemeris.* Journal of Navigation, 2006. 59(2): p. 293-306.
[14] Montenbruck, O. and E. Gill, *Satellite orbits: models, methods and applications*. 2005: Springer.
[15] Leick, A., *GPS satellite surveying*. 2003: Wiley.
[16] Xie, X., *Computer networks (the 5 version)*. 2008: Electronics Industry Press.
[17] Cheng, C., et al., *A loop-free extended Bellman-Ford routing protocol without bouncing effect.* ACM SIGCOMM Computer Communication Review, 1989. 19(4): p. 224-236.
[18] Floyd, R.W., *Algorithm 97: shortest path.* Communications of the ACM, 1962. 5(6): p. 345.

Zhang Yu born in 1979, he is currently an associate professor in School of Computer and Information Science, Southwest University, China. His main research interests include communication networks, fieldbus device integration, space monitoring and communication, etc.

Liu Feng born in 1957, he is currently a professor in School of Computer and Information Science, Southwest University, China. His main research interests include automation systems, communication networks, fieldbus device integration, etc.