International Journal of Advance Research in Science and Engineering
Volume No.06, Special Issue No.(01), December 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

Conditional Shortest Path Routing in Delay Tolerant Networks

Mr. Darshan Dinkarrao Nikam[1], Mr. Harsh Pratap Singh[2], Mr. Rishi Kushwah[3]

[1]*M. Tech Scholar, Computer Science & Engineering, SSSUTMS Sehore, Madhya Pradesh, (India)*

[2,3]*Asst. Prof. Computer Science & Engineering, SSSUTMS Sehore, Madhya Pradesh, (India)*

ABSTRACT

In this article studies Delay tolerant networks (DTNs) where each node knows the probabilistic distribution of contacts with new nodes. Delay tolerant networks are characterized by the sporadic connectivity between the nodes and therefore the lack of unchanging end-to-end paths from source to destination. Since the future node connections are mostly unknown in that networks, opportunistic forwarding is used to deliver messages. Based on the observations about human flexibility traces and the findings of previous work, we familiarize a new metric called conditional intermeeting time. We propose Conditional Shortest Path Routing protocol that route the communications over conditional shortest paths in which the cost of relations between nodes is defined by conditional intermeeting times rather than the conventional intermeeting times. When a node receives a message from one of its links, it stores the message in its buffer and carries the message until it meetings another node which is at least as useful as itself. Through trace-driven simulations, we determine that CSPR reaches higher delivery rate and lower end-to-end delay compared to the shortest path based routing protocols that use the conventional intermeeting time as the relation metric.

Keyword: - Delay Tolerant Network, Network Model, Routing protocols, Shortest Path.

I.INTRODUCTION

Routing in delay tolerant networks (DTN) is a stimulating problem because at any given time instance, the probability that there is an endwise path from a source to a destination is low. Since the routing algorithms for conventional networks assume that the relations between nodes are stable most of the time and do not fail frequently, they do not generally work in DTN's. Therefore, the routing problem is quiet an dynamic research area in DTN's. Routing algorithms in DTN's utilize a model called store-carry-and-forward. When a node receives a message from one of its contacts, it stores the message in its buffer and carries the communication until it meets another node which is at least as useful as itself. Then the message is forwarded to it. Based on this model, several routing algorithms with different objectives and different routing techniques have been proposed recently.

However, some of these algorithms used improbable assumptions, such as the existence of revelations which provide future contact times of nodes. Recent educations on routing problem in DTN's have focused on the

analysis of real flexibility traces Different traces from various DTN atmospheres are analyzed and the removed characteristics of the mobile objects are used on the design of routing algorithms for DTN's. From the analysis of these traces achieved in previous work, we have made two main observations. First, rather than being memory less, the pair wise intermeeting times between the nodes usually follow a log-normal delivery Therefore, future contacts of nodes become reliant on on the previous contacts. Second, the flexibility of many real objects is non-deterministic but cyclic. Hence, in a cyclic Mobi Space, if two nodes were often in contact at a specific time in previous cycles, then they will most likely be in contact at around the similar time in the next cycle. To show the benefits of the projected metric, we accepted it for the shortest path based routing algorithms designed for DTN's.

1.1 PROBLEM STATEMENT

We recommend conditional shortest path routing (CSPR) protocol in which normal conditional intermeeting times are used as link costs rather than normal intermeeting times and the messages are routed over conditional shortest paths (CSP). We equate CSPR protocol with the existing shortest path (SP) based routing protocol through actual trace-driven simulations. The results establish that CSPR achieves higher delivery rate and lower end-to-end delay equated to the shortest path based routing protocols. This demonstrations how well the conditional intermeeting time characterizes internodes link costs and helps making active forwarding decisions while routing a message.

We familiar a new metric called conditional intermeeting time motivated by the results of the recent studies showing that nodule intermeeting times are not memory less and that signal patterns of mobile nodes are frequently tedious. Then, we observed at the effects of this metric on shortest path based routing in DTN's. For this purpose, we efficient the shortest path based routing algorithms spending conditional intermeeting times and planned to route the communications over conditional shortest paths. Finally, we can simulations to assess the planned algorithm and confirmed the advantage of CSPR protocol.

1.2 OBJECTIVES

1. The program can correctly send the file in minimum time.
2. The file is send using shortest path.
3. The program is easy to use with graphical user interfaces.

II. LITERATURE SURVEY

That provides a brief review of the extensive literature that exists in the area of network optimization problems. In the area of network routing:

2.1 THE TIME-DEPENDENT SHORTEST PATH PROBLEM

There are many shortest path problems. One such a problem called the time-dependent shortest path (TDSP) problems has studied for the case of determining a single shortest path. Such a problem was briefly treated by Dreyfus and Ling et al.

The model and solution methods for shortest paths on time-dependent networks (SPTDN) are used in many areas. In the transportation management, the dynamic traffic assignment needs to calculate SPTDN repeatedly as a sub-procedure. In telecommunication and logistics management, the computation of the shortest paths accounting for time-dependency of networks is needed.

Many existing models and algorithms finding SPTDN are based on the time-dependent network where link lengths or link travel times depend on the time interval. The name of this model is the link travel time model (LTM). In the LTM, the shortest paths do not satisfy the non-passing property (NPP), often referred to as first-in first-out rule. In the real transportation networks, the NPP implies that two vehicles travelling on the same link will arrive at the end of the link in the same order as they start, even when some congestion occurs during the travel. Because the model does not satisfy the NPP, the shortest path is very unstable to the change of the time interval length. In other words, the optimum shortest path is apt to shift when the time interval length is changed, though the traffic conditions of the network remain unchanged. So it is still an unsolved problem until now to develop a more useful model satisfying the NPP and stable to the change of the time interval length.

Here system present a new model, named the flow speed model (FSM), of SPTDN where the stream speed of each relation depends on the time interval. In the FSM, the stream speed of each link and not the travel time changes as the time interval changes. It also suggests a solution algorithm improved from Dijkstra's label-setting algorithm. The optimal solution of the model is reliable to the NPP and stable to the modification of the time interval length. The design of the optimum SPTDN in the FSM is easier than that in the LTM.

Cons:

1. The shortest paths do not satisfy the non-passing property (NPP), often referred to as first-in first-out rule.
2. The shortest path is very unstable to the change of the time interval length.
3. The waiting restrictions at intermediate nodes do not really impose limitations on minimal delay nor do they increase the complexity of computing these delays.
4. For a wide class of delay functions, it is shown that an efficient solution exists if we allow waiting just at the source node.

2.2 Orda and Rom [1988]

On the other hand they studied the expounded use of node-disjoint paths to route duplicate packets in computer networks in order to achieve a more balanced utilization of network resources. They also prescribed several algorithms that can be used to generate such sets of disjoint paths. An algorithm that solves the SPDP problem for the all sources - single sink case and they studied various types of waiting-at-nodes scenarios, and proposed algorithms for these different cases. They showed that if waiting is allowed at nodes, then the consistency assumption is not required, and they prescribed an algorithm for identifying optimal waiting times at the source node if waiting is not allowed elsewhere in the network. Furthermore, they demonstrated that for the forbidden

waiting case, the paths obtained without the consistency assumption may not be simple, and showed that the continuous-time version of the problem is NP-Hard (1989). Orda and Rom had prescribed an algorithm that myopically determines the location sequence over a given horizon by solving 1-median problems on a network having time varying edge weights, given a starting optimal location.

Cons:

1. It does not solve the SPDP problem.
2. The paths obtained without the consistency assumption.

2.3 Ariel Orda and Raphael Rom [1990]

Due to the limitation of time-dependent shortest path problems which uses the shortest path problem in networks where inter-nodal time requirements are included. Shortest paths algorithms have been the subject of extensive research for many years resulting in a large number of algorithms for various conditions and constraints. The vast majority of these deal with fixed graphs i.e., fixed topology and fixed link weights.

In this it allow arbitrary functions for link delays this is the broadest generalization. Which address only limited cases. The most direct treatment to date was done by Halpern where arbitrary waiting times are also considered. In this work an algorithm is proposed for various waiting constraints, but this algorithm cannot be bounded by network topology (i.e., the number of operations cannot be bounded by a function of the number of nodes or edges) nor are the properties of the resulting path investigated (e.g., whether it is a simple path). All the above works avoid the treatment of functions by addressing the problem for a single instance of time and not for time ranges. These algorithms for finding the shortest-path and minimum-delay for all instances of time and under various waiting constraints and investigate properties of the derived path. If messages can be arbitrarily delayed at the source node then a shortest path can be found that is simple and achieves a delay as short as the most unrestricted path, without having to wait enroute.

Cons:

1. This algorithm cannot be bounded by network topology (i.e., the number of operations cannot be bounded by a function of the number of nodes or edges) nor are the properties of the resulting path investigated (e.g., whether it is a simple path).
2. It is just for a wide class of delay functions, it is shown that an efficient solution exists if we allow waiting just at the source node.
3. It should be pointed out that delaying the departure at the source node is common place in practical cases. Moreover, it is proved that performance in this case, from the minimum delay standpoint, is equivalent to that of the most unrestricted case.

2.4 Ziliaskopoulos and Mahmassani [1993]

They provided an efficient solution approach to the problem by discrediting time into t time periods and developed a pseudo-polynomial time algorithm for the all source - single sink case having a complexity of O (*m*3t2), for a network having *m* source nodes. This algorithm can identify least cost shortest paths in networks

having general link travel costs that do not necessarily satisfy the FIFO rule, for a given set of starting times (multi-states).

The authors have also developed an efficient implementation procedure for Intelligent Transportation Systems applications. Mahmassani and Ziliaskopoulos noted that turning movements of vehicles in congested urban networks contribute significantly to the travel time. The authors have prescribed an efficient label-correcting procedure that uses an extended forward-star structure to represent the network including intersection movements and movement prohibitions.

Cons:

1. The least cost shortest paths in networks do not necessarily satisfy the FIFO rule, for a given set of starting times (multi-states).

2.5 Chen and Tang [1998]

They have analyzed a shortest path problem on a mixed-schedule network, subject to side constraints. This network has a subset of nodes that admit waiting and a discrete, finite set of feasible arrival times. It states that such a problem can be transformed to the TDSP case (subject to side constraints) by suitably redefining the link delays for the network based on the restricted set of arrival times on the mixed schedule network.

The time-constrained shortest path problem is an important generalization of the shortest path problem and has attracted much research interest in a new time constraint, namely time-schedule constraint, is introduced. This time constraint assumes that every node in the network has a list of pre-specified departure times and requires that departure from a node take place only at one of these departure times. Therefore, when a time-schedule constraint is considered, the total time in a network includes traveling time and waiting time a network consisting of two types of nodes in terms of their time constraints. The first types of nodes are subject to time-schedule constraints, but the second type is not. For such a network, a set of minimum time (shortest) path problems is studied, including minimization of total time, minimization of total time subject to a total traveling time constraint, minimization of total traveling time subject to a total time constraint and minimization of a weighted sum of total time and total traveling time.

Cons:

1. It only considers time-varying costs and knapsack-like constraints.
2. No deterministic TDSP algorithm has been reported in the literature that prescribes an effective implementation for cases other than the single shortest path problem.

III.SYSTEM DESIGN

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. One could see it as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering.

International Journal of Advance Research in Science and Engineering
Volume No.06, Special Issue No.(01), December 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

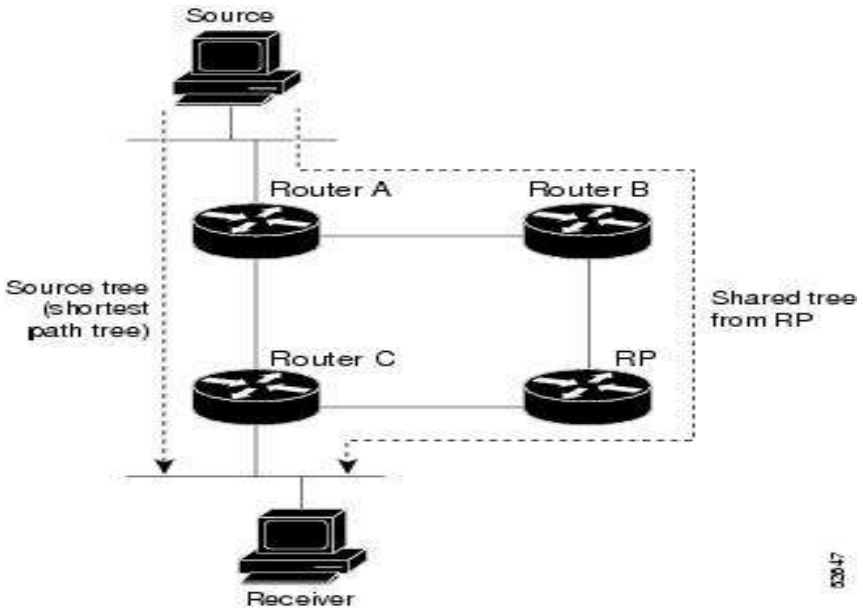

Fig -1 System Architecture

We propose conditional shortest path routing (CSPR) protocol in which average conditional intermeeting times are used as link costs rather than standard2 intermeeting times and the messages are routed over conditional shortest paths (CSP). We compare CSPR protocol with the existing shortest path (SP) based routing protocol through real trace- driven simulations. The results demonstrate that CSPR achieves higher delivery rate and lower end-to-end delay compared to the shortest path based routing protocols. This shows how well the conditional intermeeting time represents internode link costs (in the context of routing) and helps making effective forwarding decisions while routing a message.

3.1 TECHNOLOGY DETAILS USED IN THE PROJECT

C# syntax is highly expressive, yet with less than 90 keywords, it is also simple and easy to learn. The curly-brace syntax of C# will be instantly recognizable to anyone familiar with C, C++ or Java. Developers who know any of these languages are typically able to begin working productively in C# within a very short time. C# syntax simplifies many of the complexities of C++ while providing powerful features such as nullable value types, enumerations, delegates, anonymous methods and direct memory access, which are not found in Java. C# also supports generic methods and types, which provide increased type safety and performance, and iterators, which enable implementers of collection classes to define custom iteration behaviors that are simple to use by client code.

As an object-oriented language, C# supports the concepts of encapsulation, inheritance and polymorphism. All variables and methods, including the Main method, the application's entry point, are encapsulated within class definitions. A class may inherit directly from one parent class, but it may implement any number of interfaces. Methods that override virtual methods in a parent class require the override keyword as a way to avoid

accidental redefinition. In C#, a struct is like a lightweight class; it is a stack-allocated type that can implement interfaces but does not support inheritance.

In addition to these basic object-oriented principles, C# facilitates the development of software components through several innovative language constructs, including:

- Encapsulated method signatures called delegates, which enable type-safe event notifications.
- Properties, which serve as assessors for private member variables.
- Attributes, which provide declarative metadata about types at run time.
- Inline XML documentation comments.
- It provides automatic garbage collection.
- It provides robust security model.
- It provides decimal data type for financial application.
- It provides modern approach for debugging.
- It provides a rich intrinsic model for error handling.

IV.SOFTWARE IMPLEMENTATION

4.1 INTRODUCTION

The projected system is divided into four modules:

1. Networking Module

Client-server computing or networking is a distributed application architecture that partitions tasks or workloads between service providers and service requesters, called clients. Often clients and servers work over a computer network on separate hardware. A server machine is a high-performance host that is running one or more server programs which share its resources with clients. A client also shares any of its resources; Clients therefore initiate communiqué sessions with servers which await arriving requests.

2. Shortest Path Module

In networks packets are transferred through routes that could be collected of multiple transmit nodes between sources and destinations. In many networks, shortest path routing is often used for its simplicity and scalability, and this is closely estimated by straight line routing for large networks. Thus we will focus on straight line routing for sending packets from sources to destinations.

3. Straight Line Routing Module

Both simulations and exploration show that the relay load over the network, imposed by straight line routing, depends on the model of the traffic design. Even if the system settings are identical and straight line routing is commonly accepted, the relay load induced by “arbitrary” traffic could be distributed differently over the network. Thus, in contrast to traditional acceptance, there are many scenarios in which straight line routing itself can balance the load over the network, and in such cases explicit load-balanced routing may not help moderate the communicating load.

4. Simulations result

To calculate the performance of our algorithm, we have built a discrete event simulator in Java. In this section, we describe the details of our simulations through which we compare the proposed Conditional Shortest Path Routing (CSPR) algorithm with standard Shortest Path Routing (SPR). To collect some routing statistics, we have generated traffic on the traces of these two data groups. We assume that the nodes have enough buffer space to store every message they receive, the bandwidth is high and the contact periods of nodes are long enough to allow the exchange of all messages between nodes.

4.2 Conditional intermeeting time:

An analysis of real mobility traces has been done in different environments with different objects and with variable number of attendants and led to significant results about the aggregate and pair wise mobility characteristics of real objects. Recent analysis on real mobility traces have demonstrated that models assuming the exponential distribution of intermeeting times between pairs of nodes do not match real data well. Instead up to 99% of intermeeting times in many datasets is log-normal distribution. This makes the pair wise contacts between nodes depend on their pasts. Such a finding invalidates a common assumption that the pair wise intermeeting times are exponentially distributed and memory less.

To take advantage of such knowledge, we suggest a new metric called conditional intermeeting time that measures the intermeeting time between two nodes relative to a meeting with a third node using only the local knowledge of the past contacts. Such measure is particularly beneficial if the nodes move in a cyclic so-called MobiSpace in which if two nodes contact frequently at particular time in previous cycles,

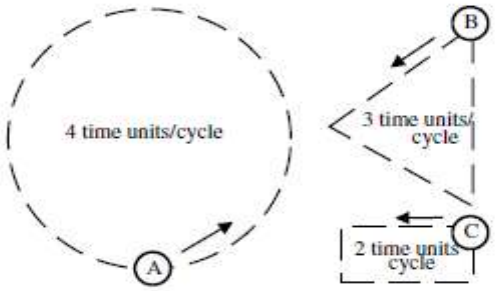

Fig-2: A physical cyclic MobiSpace with a common motion cycle of 12 time units.

They will probably be in contact around the same time in the next cycle. Consider the sample cyclic MobiSpace with three objects illustrated in Figure 1. The common motion cycle is 12 time units, so the discrete probabilistic contacts between *A* and *B* happen in every 12 time units (1, 13, 25,) and between *B* and *C* in every 6 time units (2, 8, 14,). The average intermeeting time between nodes *B* and *C* indicates that node *B* can forward its message to node *C* in 6 time units. However, the conditional intermeeting time of *B* with *C* relative to prior meeting of node *A* indicates that the message can be forwarded to node *C* within one time unit. In a DTN, each node can

compute the average of its standard and conditional intermeeting times with other nodes using its contact history.

In Algorithm we show how a node, *s*, can compute these metrics from its previous meetings. The notations we use in this algorithm (and also throughout the paper) are listed below with their meanings:

• τA (B): Average time that elapses between two consecutive meetings of nodes A and B. Obviously when the node connections are bidirectional, τA (B) = τB(A).

• τA ($B|C$): Average time it takes for node A to meet node B after it meets node C. Note that, τA($B|C$) and τB($A|C$) are not necessarily equal.

• S: $N \times N$ matrix where S (i, j) shows the sum of all samples of conditional intermeeting times with node j relative to the meeting with node i. Here, N is the neighbor count of current node (i.e. N(s) for node s).

• C: $N \times N$ matrix where C (i, j) shows the total number of conditional intermeeting time samples with node j relative to its meeting with node i.

• βi: Total meeting count with node i.

In Algorithm each node first adds up times expired between repeating meetings of one neighbor and the meeting of another neighbor. Then it divides this total by the number of times it has met the first neighbor prior to meeting the second one. For example, if node A has two neighbors (B and C), to find the conditional intermeeting time of τA ($B|C$), each time node A meets node C, it starts a different timer. When it meets node B, it sums up the values of these timers and divides the results by the number of active timers before deleting them. This computation is repeated again each time node B is encountered. Then, the total of times collected from each timer is divided by the total count of timers used, to find the value of τA ($B|C$).

4.3 MATHEMATICAL MODEL

4.3.1 System

S= {I, O, P, C}

Where, I=Input

O=Output

P=Process

C=Constraint

4.3.2 Input

I= {N, F, S, D}

Where, N= Number of nodes

F=Select file to traversing

S1= Source node

D=Destination node

4.3.3 Output

O= {F1, R, T}

Where, F1=File

R=File is receive

T=Sent successfully

4.3.4 Process

Algorithm updates (node m, time t)

1: if m is seen first time then

2: firstTimeAt[m] ← t

3: else

4: increment βm by 1

5: lastTimeAt[m] ← t

6: end if

7: for each neighbor j ∈ N and j _= m do

8: start a timer tmj

9: end for

10: for each neighbour j ∈ N and j _= m do

11: for each timer tjm running do

12: S[j][m] += time on tjm

13: increment C[j][m] by 1

14: end for

15: delete all timers tjm

16: end for

17: for each neighbour i ∈ N do

18: for each neighbour j ∈ N and j _= i do

19: if S[j][i] _= 0 then

20: τs (i|j) ← S[j][i] / C[j][i]

21: end if

22: end for

23: τs (i) ← (lastTimeAt[i] − firstTimeAt[i]) / βi

24: end for

While computing standard and conditional intermeeting times, we ignore the edge effects by including intermeeting times of atypical meetings. That means that we include the values of τA(B) for the first and last meetings of node B with node A. Likewise, we include the values of τA(B|C) for the first meeting of node A with node C and the last meeting of that node with node B. Although this may change the results, this change will be negligible if long enough time passed to collect many other meeting data.

The drawback of this computation can also be minimized either by updating the computation by including the edge effects as in or by keeping an appropriate size of window of the past contacts. Consider the sample meeting times of a node A with its neighbors B and C in Figure 2. Node A meets with node B at times {5, 16, 25, 30} and with node C at times {11, 14, 23, 34}. Following the procedure described above, we find that τA (B|C) = (5 + 2 + 2)/3 = 3 time units and τA (C|B) = (6 + 7 + 4 + 9)/4 = 6.5 time units.

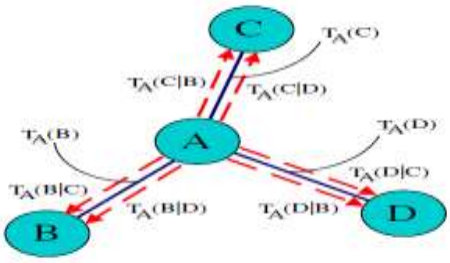

Fig-3: The graph of a sample DTN with four nodes and nine edges in total.

International Journal of Advance Research in Science and Engineering
Volume No.06, Special Issue No.(01), December 2017
www.ijarse.com
IJARSE
ISSN: 2319-8354

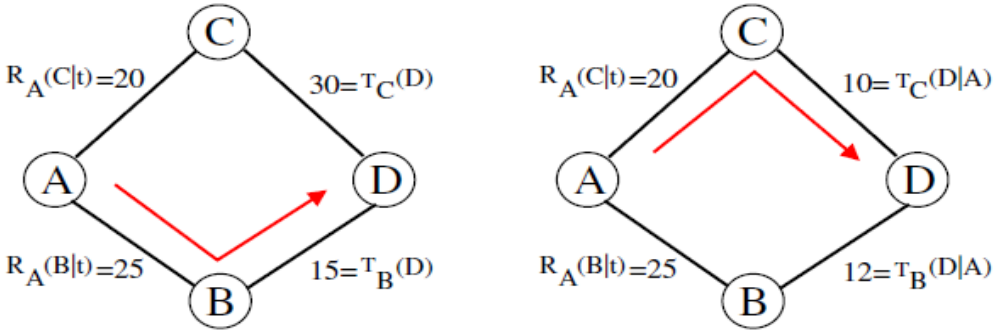

Fig-4: The shortest path from a source to destination node can be different when conditional intermeeting times are used as the weights of links in the network graph.

4.4 Conditional Shortest Path Routing:

Our algorithm basically finds conditional shortest paths (CSP) for each source-destination pair and routes the messages over these paths.

1. CSP from a node *n0* to a node *nd* as follows:

$$CSP(n_0, n_d) = \{n_0, n_1. n_{d-1}, n_d \mid R_{n0}(n_1|t) + \sum_{i=1}^{d-1} T_{ni}(n_{i+1}|n_{i-1}) \text{ is minimized.}\}$$

Here, t represents the time that has passed since the last meeting of node n_0 with n_1 and $R_{n0}(n_1|t)$ is the expected residual time for node n_0 to meet with node n_1 given that they have not met in the last t time units. $R_{n0}(n_1|t)$ can be computed as in with parameters of distribution representing the intermeeting time between n_0 and n_1. It can also be computed in a discrete manner from the contact history of n_0 and n_1.

2. Assume that node i observed d intermeeting times with node j in its past. Let $\tau_i^1(j), \tau_i^2(j) \ldots \tau_i^d(j)$ denote these values. Then:

$$R_i(j|t) = \left(\sum f^k_i(j) / |\{T^k_i(j \geq t\}|\right) \text{ where,}$$

$$f^k_i(j) = \begin{cases} T^k_i(j)-t & \text{if } T^k_i(j) \geq t \\ 0 & \text{otherwise} \end{cases}$$

Here, if none of the d observed intermeeting times is bigger than t (this case occurs less likely as the contact history grows), a good approximation can be to assume R_i (j|t) = 0.We will next provide an example to show the benefit of CSP over SP.

3. Consider the DTN illustrated in Figure 4.3 the weights of edges (A, C) and (A, B) show the expected residual time of node A with nodes C and B respectively in both graphs. But the weights of edges (C, D) and (B, D) are different in both graphs. While in the left graph, they show the average intermeeting times of nodes C and B with D respectively, in the right graph, they show the average conditional intermeeting times of the same nodes with D relative to their meeting with node A. From the left graph, we conclude that SP (A, D) follows (A, B, D). Hence, it is expected that on average a message from node A will be delivered to node D in 40 time units. However this may not be the actual shortest delay path. As the weight of edge (C, D) states in the right graph, node C can have a smaller conditional intermeeting time (than the standard intermeeting time) with node D assuming that it has met node A. This provides node C with a faster transfer of the message to node D after meeting node A. Hence, in the right graph, CSP (A, D) is (A, C, D) with the path cost of 30 time units.

4. Each node forms the aforementioned network model and collects the standard and conditional intermeeting times of other nodes between each other through epidemic link state protocol. However, once the weights are known, it is not as easy to find CSP's as it is to find SP's. Where the CSP (A, E) follows path 2 and CSP (A, D) follows (A, B, D). This situation is likely to happen in a DTN, if $\tau D (E|B) \geq \tau D (E|C)$ is satisfied. Running Dijkstra's or Bellman-ford algorithm on the current graph structure cannot detect such cases and concludes that CSP (A, E) is (A, B, D, and E).

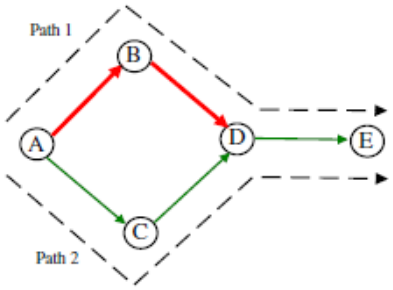

Fig-5: Path 2 may have smaller conditional delay than path 1 even though the CSP from node *A* to node *D* is through node *B*.

V.ADVANTAGES

- Exposure of secure data is very negligible.
- No Data Loss.
- Lower end-to-end delay compared to the shortest path based routing protocols that use the conventional intermeeting time as the link metric.

- New metric called conditional intermeeting time that measures the intermeeting time between two nodes relative to a meeting with a third node using only the local knowledge of the past contacts.
- We updated the shortest path based routing algorithms using conditional intermeeting times and proposed to route the messages over conditional shortest paths.

VI.DISADVANTAGES

- It sends text and audio file only not a video file.
- Our system is unidirectional.

VII. APPLICATION

1. Fast delivery of files in less time.
2. It finds the shortest path between two destinations such that the sum of the weights of its constituent edges is minimized.

VIII.CONCLUSION

The existing system which we use is insufficient information on faulty components .The paths chosen by this scheme may not always be the shortest. In our system it guarantees all messages to be routed via shortest paths, the system which we propose to equip every node with more information than that on its own links. The effect of that extra information on routing efficiency is analyzed, and the additional information to be placed at every node for the shortest path routing is determined in short time of period.

REFERENCES

Journal Papers:

[1] T. Spyropoulos, K. Psounis, C. S. Raghavendra, *Efficient routing in intermittently connected mobile networks: The single-copy case*, IEEE/ACM Transactions on Networking, vol. 16, no. 1, Feb. 2008.

[2] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, *MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks*, In Proc. IEEE Infocom, April 2006.

[3] S. Srinivasa and S. Krishnamurthy, *CREST: An Opportunistic Forwarding Protocol Based on Conditional Residual Time*, in Proceedings of IEEE SECON, 2009.

[4] T. Spyropoulos, K. Psounis, and C. Raghavendra, *Spray and Focus: Efficient Mobility-Assisted Routing for Heterogeneous and Correlated Mobility*, In Proceedings of IEEE PerCom, 2007.

[5] E. Bulut, Z. Wang, and B. Szymanski, *Cost-Effective Multi-Period Spraying for Routing in Delay Tolerant Networks*, to appear in IEEE/ACM Transactions on Networking, 2010.

[6] T. Spyropoulos, K. Psounis, C. S. Raghavendra, *Efficient routing in intermittently connected mobile networks: The multi-copy case*, IEEE/ACM Transactions on Networking, 2008.

[7] Y. Wang, S. Jain, M. Martonosi, and K. Fall, *Erasure coding based routing for opportunistic networks*, in Proceedings of ACM SIGCOMM workshop on Delay Tolerant Networking (WDTN), 2005.

[8] A. Lindgren, A. Doria, and O. Schelen, *Probabilistic routing in intermittently connected networks*, SIGMOBILE Mobile Computing and Communication Review, vol. 7, no. 3, 2003.

Books:

[9] *Delay tolerant networking research group*, http://www.dtnrg.org.

[10] A European Union funded project in Situated and Autonomic Communications, *www.haggleproject.org*.

[11] D. Bertsekas, and R. Gallager, *Data networks (2nd ed.)*, 1992.

[12] C. Liu and J. Wu, *An Optimal Probabilistically Forwarding Protocol in Delay Tolerant Networks*, in Proceedings of MobiHoc, 2009.

[13] J. Leguay, A. Lindgren, J. Scott, T. Friedman, J. Crowcroft and P. Hui, *CRAWDAD data set upmc/content (v. 2006-11-17)*, downloaded from http://crawdad.cs.dartmouth.edu/upmc/content, 2006.

[14] T. Spyropoulos, K. Psounis,C. S. Raghavendra, *Spray and Wait: An Efficient Routing Scheme for Intermittently Connected Mobile Networks*, ACM SIGCOMM Workshop, 2005.

Theses:

[15] A. Vahdat and D. Becker, *Epidemic routing for partially connected adhoc networks*, Duke University, Tech. Rep. CS-200006, 2000.

[16] S. Jain, K. Fall, and R. Patra, *Routing in a delay tolerant network*, in Proceedings of ACM SIGCOMM, Aug. 2004.

[17] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, *Impact of Human Mobility on the Design of Opportunistic Forwarding Algorithms*, in Proceedings of INFOCOM, 2006.

[18] X. Zhang, J. F. Kurose, B. Levine, D. Towsley, and H. Zhang, *Study of a Bus-Based Disruption Tolerant Network: Mobility Modeling and Impact on Routing*, In Proceedings of ACM MobiCom, 2007.

[19] P. U. Tournoux, J. Leguay, F. Benbadis, V. Conan, M. Amorim, J. Whitbeck, *The Accordion Phenomenon: Analysis, Characterization, and Impact on DTN Routing*, in Proceedings of Infocom, 2009.

[20] C. Liu and J. Wu, *Routing in a Cyclic Mobispace*, In Proceedings of ACM Mobihoc, 2008.

[21] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli, *Age Matters: Efficient Route Discovery in Mobile Ad Hoc Networks Using Encounter Ages*, In Proceedings of ACM MobiHoc, 2003.

[22] E. Daly and M. Haahr, *Social network analysis for routing in disconnected delay-tolerant manets*, In Proceedings of ACM MobiHoc, 2007.

[23] P. Hui, J. Crowcroft, and E. Yoneki, *BUBBLE Rap: Social Based Forwarding in Delay Tolerant Networks*, In Proc. of ACM MobiHoc,2008.

[24] E. P. C. Jones, L. Li, and P. A. S. Ward, *Practical routing in delay tolerant networks*, in Proceedings of ACM SIGCOMM workshop on Delay Tolerant Networking (WDTN), 2005.

[25]Y.Wang,P.Zhang,T.Liu,C.SadlerandM.Martonosi,*http://crawdad.cs.dartmouth.edu/princeton/zebranet*, CRAWDAD data set princeton/zebranet (v. 2007-02-14), 2007.