小 型 微 型 计 算 机 系 统
Journal of Chinese Computer Systems
DOI:10.20009/j.cnki.21-1106/TP.2021-0803
2023 年 6 月 第 6 期
Vol.44 No.6 2023

DTN 中基于连接分离时间的概率路由算法

崔建群,邬 尧,常亚楠,孙佳悦,余东海

(华中师范大学 计算机学院,武汉 430079)
E-mail:jqcui@126.com

摘 要:延迟容忍网络(delay tolerant network,DTN)中,由于节点的移动性、休眠调度、资源受限以及网络误码率较高等因素,消息成功投递的可能性较低.为了解决上述 DTN 中存在的问题,有大量的路由算法被提出,其中 Prophet 路由算法作为 DTN 中重要的路由之一,主要思想是根据 DTN 中节点的相遇频率进行节点之间消息的传递,但该算法没有考虑节点的交互意愿和中继节点的缓存大小.针对这一问题,本文提出一种基于连接分离时间的概率路由算法 P-AVF(Prophet routing based on Average fluctuation).该算法主要根据节点在时间窗口 T 内的连接分离时间以及节点间运动轨迹的差异性来定义节点连接的平均波动,进而引出节点连接紧密性与可靠性的概念,使得连接能力更优异的节点与它相遇过的节点保持更大的投递预测值,从而能综合挑选出合适转发消息的中继节点.同时利用消息接收节点的缓存占用比和该节点与其他节点连接分离的总时间作为影响该节点投递预测值衰减的一部分,使得投递预测值的衰减更准确.仿真结果表明,基于连接分离时间的概率路由算法 P-AVF 在消息的投递率、网络负载率和平均跳数等方面均优于其他对比路由算法.

关 键 词:DTN;节点连接分离时间;节点连接的平均波动;ACK
中图分类号:TP393 **文献标识码**:A **文 章 编 号**:1000-1220(2023)06-1257-09

Probabilistic Routing Algorithm Based on Connection Separation Time in Delay Tolerant Network

CUI Jian-qun,WU Yao,CHANG Ya-nan,SUN Jia-yue,YU Dong-hai
(School of Computer,Central China Normal University,Wuhan 430079,China)

Abstract:In delay tolerant network(DTN),the probability of successful message delivery is low due to the mobility of nodes,dormant scheduling,resource constraints and high network BER. To solve the above problems in DTN,several routing algorithms have been proposed. Prophet routing algorithm is the most promising routes in DTN. The main idea is to deliver messages between nodes according to the encounter frequency of nodes in DTN. However,the algorithm does not consider the interaction willingness of nodes and the cache size of relay nodes. To address this problem,this paper proposes a probabilistic routing algorithm P-AVF(Prophet routing based on Average fluctuation) based on the connection separation time. It defines the average fluctuation of node connectivity mainly based on the connection separation time of nodes within a time window T and the variability of the movement trajectories between nodes,which leads to the concepts of node connectivity tightness and reliability,so that a node with better connectivity maintains a greater deliver probability with the nodes it has met,and thus can comprehensively select relay nodes that are suitable for forwarding messages. At the same time,the cache occupancy ratio of the receiving node and the total time of separation between the node and other nodes are used as part of the attenuation of the predicted value of the node to make the predicted value of the attenuation more accurate. The simulation results show that the probabilistic routing algorithm P-AVF based on connection separation time outperforms other comparative routing algorithms in terms of message delivery ratio,overhead ratio and average hop count.

Key words:DTN;node connection separation time;average fluctuation of node connections;ACK

1 引 言

延迟容忍网络(Delay Tolerant Network,DTN)[1],又称为容迟网络,是一种新型的端到端存储转发体系结构.DTN 中的节点移动会导致网络拓扑时刻发生变化,再加上节点不稳定、资源受限以及环境干扰等因素,发送端和接收端之间难以实时维护一条可靠的端对端通路.为了克服 DTN 的间歇连通问题,完成消息的成功投递,DTN 采用“存储-携带-转发”[2]的路由模式使得 DTN 适用于具有挑战性的环境,如野生动物追踪,移动车载网[3]与星际互联网络[4]等现实场景中,这种“无处不在的网络”被认为是发展潜力巨大的通信方式.DTN 中消息的投递成功率受网络拓扑、链路状态、网络资源以及节点能量受限等诸多因素的影响[5],而高效合理的路由算法能够使得消息以较小的代价与较短的时间从源节点成功投递到

收稿日期:2021-10-12 收修改稿日期:2021-11-22 基金项目:国家自然科学基金项目(61672257,61702210,61772377)资助. 作者简介:**崔建群**,女,1974 年生,博士,教授,CCF 会员,研究方向为机会网络、物联网、移动网络、应用层组播等;**邬 尧**,男,1995 年生,硕士研究生,研究方向为延迟容忍网络、社交网络和无线通信;**常亚楠**(通讯作者),女,1984 年生,博士,副教授,CCF 会员,研究方向为无线 mesh 网络、延迟容忍网络、社交网络等;**孙佳悦**,女,1996 年生,硕士研究生,研究方向为延迟容忍网络、社交网络和无线通信;**余东海**,男,1997 年生,硕士研究生,研究方向为延迟容忍网络、社交网络和无线通信.

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

目的节点,因此路由算法设计是 DTN 研究方向最为关键的问题.

基于上述分析,本文在 Prophet 算法基础上提出了一种基于节点连接分离时间的概率路由算法(Probabilistic routing algorithm based on connection separation time in DTN). 本文主要贡献如下:

1)节点的连接分离时间,衡量节点的交互度. 如果节点与其他节点分离时间越长就说明该节点与其他节点的交互度越低,表明该节点不愿意进行大范围移动或则在其运动范围内邻居节点较少. 因此在选择中继节点时就不应该把消息投递给这类节点,而选择交互度高的节点,这样才能使得消息在短时间内扩散至较多的节点,从而提高投递率,降低传输时延.

2)在连接分离时间的基础上衍生出平均波动的概念,其是在节点运动轨迹和连接分离时间的基础上来综合反映节点之间连接波动大小. 节点 v_i 与其他节点在相遇时,会计算上一个时间窗口 T 中的平均波动值. 当两个节点之间的连接波动较大,消息在传输的过程中随时可能会发生中断,将消息传给这样的节点会降低消息的投递率以及浪费网络资源. 从前面的描述可以得出,平均波动应该作为投递预测值影响因子的一部分,使得平均波动较大的节点保持较低的投递预测值,反之,平均波动较小的节点保持较高投递预测值.

3)以平均波动为基础衍生出了节点连接紧密性、节点连接可靠性. 衡量节点连接能力不能仅凭某方面的优势而下定论,因此本文考虑到节点连接紧密性与节点连接可靠性这两个影响因素,来综合定义节点连接能力. 若节点连接能力较大,当该节点与其他节点相遇更新投递预测值时,应使得这对节点间的投递预测值较大,这样才能使得优质的节点能够获得更多的消息副本,使消息在 DTN 网络中扩散的范围更大,进而提高消息到达目的节点的可能性.

4)ACK 机制,删除成功投递到目的节点的消息会释放节点缓存和节约网络中的资源,提高消息投递率,以及能明显降低网络开销.

2 相关工作

根据网络中转发消息副本的数量,DTN 中的路由算法通常可以分为基于转发的路由算法和基于洪泛的路由算法两大类. 其中基于转发策略的路由算法是利用网络拓扑选择一个到达目的节点的最优路径,然而在网络拓扑频繁发生变化的 DTN 中,单个节点想获得这样一条最优路径是几乎不可能的,因为其无法及时获取全部网络节点的状态信息[6]. 而洪泛式路由于无需获取全局的网络信息,只需将消息复制到足够多的节点,通过多节点携带消息副本的方式确保目的节点能成功地接受到这些消息. 为了能够更好的利用节点之间关系的信息,并且能够从源头上控制消息副本数目,研究学者提出 Prophet 路由算法[7]. 该算法利用历史相遇信息与传递性来定义投递预测值(Deliver Probability). 当节点 v_i 遇到 v_j 节点,并不会盲目地将消息复制转发给 v_j,只有当 v_j 的投递预测值比 v_i 高时,v_i 才将消息转发给 v_j. 该算法通过挑选合适中继节点与有选择的复制消息副本,提高消息传输率,降低了网络开销. 假设用 $P_{(i,j)}$ 来表示节点 v_i 和 v_j 之间的投递预测值,$P_{(i,j)}$ 的计算分为 3 个步骤:更新、衰减以及传递性.

更新:当节点 v_i 与节点 v_j 相遇,根据式(1)来更新投递预测值.

$$P_{(i,j)} = P_{(i,j)old} + (1-P_{(i,j)old}) \times P_{init} \tag{1}$$

其中,$P_{init} \in [0,1]$ 是初始化常数.

衰减:当节点 v_i 与节点 v_j 一段时间内没有相遇,投递预测值根据式(2)进行衰减.

$$P_{(i,j)} = P_{(i,j)} \times \gamma^k \tag{2}$$

其中,γ 是一个时间衰减常数,在[0,1)间取值;指数 k 为从上一次计算相遇概率开始所经的时间单元个数.

传递性:投递预测值具有传递性,若节点 v_i 与 v_j 经常相遇,v_j 与 v_k 也经常相遇,则可以认为 v_j 是 v_i 和 v_k 较为合适的中继节点,节点 v_i 和 v_k 间的投递预测值传递如式(3)所示.

$$P_{(i,k)} = P_{(i,k)old} + (1-P_{(i,k)old}) \times P_{(i,j)} \times P_{(j,k)} \times \beta \tag{3}$$

其中 $\beta \in [0,1]$ 是个常数,它决定了节点传递性对节点的投递预测值将产生影响的比重.

传统的 Prophet 没有考虑节点特殊性以及不同 DTN 场景间巨大的差异性,Prophet 只从节点间相遇频率来定义不够准确. 另外,由于消息大小是随机产生的而节点缓存大小一般是固定的[8],当节点剩余缓存较小时会为接收消息腾出缓存空间丢弃其他消息,为了尽可能避免丢弃有效消息,在挑选中继节点时应选择剩余缓存空间较大的节点. 并且 Prophet 算法在衰减阶段仅从时间这一维度出发也没有考虑节点自身属性或 DTN 环境等其他因素,具有一定的局限性和缺失[9]. 针对 Prophet 算法的主要弊端,目前国内外研究学习者已经在 Prophet 的改进上取得了一些成果.

文献[10]考虑到节点的剩余能量与空闲缓冲区,当两节点相遇时先交换与其他节点相遇次数,再比较相遇次数与剩余能量,携带消息的节点会将消息投递给较多的相遇次数和能量的节点. 文献[11]提出基于节点持续连接时间与节点缓存空闲率的理由算法 BAProphet. 该算法在挑选中继节点时,考虑到节点的连接持续时间以及节点缓存,将消息复制给当前性能更优的节点. 文献[12]基于节点相似性提出 S-Prophet 算法,其在计算消息转发概率时考虑了节点的历史相遇情况,并采用 ACK 删除机制来保证及时清除已成功投递消息的副本. 文献[13]基于距离的概率路由协议提出了 DiProphet 算法,该算法在投递预测值中添加近距离度量,提高了消息的投递率,降低了投递延迟. 文献[14]将 Prophet 算法与 Spray-and-Wait 算法的消息控制策略相结合,并运用到车载网络中有效地控制了消息副本复制的次数. 文献[15],结合节点接触时间与相遇频率提出 HPR 算法,同时为提高算法的消息传递率,该算法还通过消息复制的方式实现了多路径并行消息传输. 与 BSW 算法[16]类似,指定源节点的所携带的最大副本数 M($M>1$),然后基于二叉树生成 M 个消息副本,目的是为了减少有效的网络开销. 因此,消息总能朝着投递预测值递增的方向转发,这将有利于增大消息的投递成功率以及降低网络资源开销.

BAProphet 算法在消息传递的过程中考虑到节点移动模型的特性和节点缓存空间对传输概率的影响,选用节点间持续连接时间以及消息传递过程中的缓存空闲率改进,但在消息的传输过程中只是简单的选取了两节点间总的连接持续时间和当前仿真运行总时间的关系,仿真实验表明消息投递率

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

没有明显的提升,故不能充分说明时间这一参数对算法性能的影响力. S-Prophet 算法在消息传递的过程中,虽然考虑节点历史相遇节点集合的相似性,但在计算投递率时只是简单的把 P_{init} 替换成节点相似率,而且并未考虑节点移动这一特性对投递预测值的影响,因此在消息投递率表现上并不够理想.

基于上述分析,本文综合考虑不同 DTN 场景中节点特殊性,以节点连接分离时间作为基础,引入了节点平均波动、节点连接可靠性、节点连接紧密性以及节点连接能力的定义,并重新设计了传统 Prophet 算法中的更新与衰减公式,提出了一种基于连接分离时间的概率路由算法 P-AVF(Prophet routing based on Average fluctuation). 该算法中,当两节点相遇时,若当前时间 t 大于时间窗口 T,则选择 P-AVF 算法;否则,利用 Prophet 算法计算投递预测值. 在 P-AVF 算法中,两个相遇节点将利用的上一个时间窗口 T 内的历史信息分别各自计算的投递预测值,挑选出合适的中继节点使优质的节点能够携带更多消息并且扩散的更远. 将 P-AVF 的仿真与 Prophet, BAProphet, S-Prophet 这 3 种路由算法的仿真数据进行分析和比较,P-AVF 路由算法能够有效提高消息投递率,并且会有效地降低网络开销.

3　问题模型

3.1　网络模型

假设网络中包含 n 个节点,$G=(V,E)$ 为网络拓扑图,其中 $V=\{v_i \mid 1\leqslant i\leqslant n\}$ 表示网络中的节点集合,E 为定义在 G 上边的集合. 每个节点 v_i 维护了一个历史相遇节点集合 $S_i=\{\{v_j,\{t_s,t_d\}\}\mid i\neq j\}$,其中该集合记录与节点 v_j 起始相遇时刻 t_s 和分离时刻 t_d. 由于集合 S_i 是由 $\{v_j,\{t_s,t_d\}\}$ 三元组构成,当 v_i 在不同相遇时刻遇到同一节点 v_j 也会当成一个新的三元组并加入到集合中. 另外,每个节点 v_i 还维护了一个与其他相遇节点的投递预测值的集合 $S_i=\{v_j \mid P_{(i,j)}\}$,其中 $P_{(i,j)}$ 表示节点 v_i 和 v_j 的投递预测值. 若 v_i 与节点 v_j 在仿真过程中相遇多次,会根据上次的投递预测值更新此次投递预测值.

3.2　节点连接分离时间

在 DTN 中为了尽快将源节点的消息投递到目的节点,在选取中继节点时要挑选交互度更高的节点,交互度更高反映该节点愿以进行大范围的移动可以更频繁的遇到其他节点或该节点的邻居节点就较多,这样就可以将消息在短时间扩散至更多的节点. 由于 DTN 网络拓扑结构频繁变化,为保证从 DTN 场景中抽象出来的数据有效性,本文取一个时间窗口 T 来衡量该节点在最近时间窗口 T 内的交互度.

假设 2 个节点发生相遇时,若当前时间 $t<$ 时间窗口 T 则采用 Prophet 算法. 若当前时间 $t\geqslant$ 时间窗口 T,两个节点分别同时计算上一个时间窗口 T 内的节点分离时间,用来衡量节点最新交互度.

定义 1. 节点连接分离时间. 在每个时间 T 窗口内节点 v_i 都会记录 1 组矢量 $\boldsymbol{S}_i=[\{\mathrm{start}_{(i,1)},\mathrm{end}_{(i,1)}\},\{\mathrm{start}_{(i,2)},\mathrm{end}_{(i,2)}\},\cdots,\{\mathrm{start}_{(i,n)},\mathrm{end}_{(i,n)}\}]$. 其中,$\mathrm{start}_{(i,1)}$ 表示在当前时间窗口 T 内节点 v_i 第 1 次连接分离的时刻,$\mathrm{end}_{(i,1)}$ 表示节点在时间窗口 T 内节点 v_i 第 2 次建立连接的时刻. 在一个时间窗口 T 内,连接分离时间总时间为:$T_s=(\mathrm{end}_{(i,1)}-\mathrm{start}_{(i,1)})+(\mathrm{end}_{(i,2)}-\mathrm{start}_{(i,2)})+\cdots+(\mathrm{end}_{(i,n)}-\mathrm{start}_{(i,n)})$. 若 T_s 的值越小,则表明该节点与其他节点的交互性越强;反之,越弱.

3.3　节点平均波动

DTN 中有的节点移动性很强,当两个节点相遇频率较高,但节点间连接时间较短,消息在连接时间内没有被传输成功,这种情形将导致消息投递失败. 因此,节点连接波动情况将与消息的成功投递密切相关,本文定义节点平均波动来衡量节点连接情况.

定义 2. 节点平均波动. 节点平均波动是衡量节点 v_i 与其他节点 v_j 连接波动程度,记为 $AVF(D_i)$,公式如下.

$$AVF(D_i)=\frac{\int_{t\text{-}T}^{t}f(t)\,dt}{N_i} \tag{4}$$

其中,D_i 表示节点 v_i 与其他节点 v_j 的分离时间,t 表示当前时间,T 表示时间窗口,N_i 表示节点与其他节点的分离次数. 当前时间 t 大于时间窗口 T 时,节点 v_i 遇到其他节点时,其会以当前时间 t 为起点计算上一个时间窗口 T 中的 $AVF(D_i)$,以此作为依据来影响节点 v_i 与其他节点之间的投递预测值. 由式(4)可以看出当连接分离时间越长,并与其他节点分离次数越少时,$AVF(D_i)$ 越大,表明该节点的连接时波动性更大.

由于 DTN 中每个节点的缓存容量有限,在利用多副本策略下选择中继节点时应合理利用节点缓存,选择将消息复制给轨迹差异性更大的节点,减少轨迹相似节点携带过多相同的消息,使得消息能够短时间在 DTN 中扩散更远[17]. 故式(4)中 $f(t)$ 取值公式如下:

$$f(t)=\begin{cases}0 & \text{与其他节点通信}\\ e^{-\left(1-\frac{E(t)_i\cap E(t)_j}{E(t)_i\cup E(t)_j}\right)} & \text{与其他节点断开}\end{cases} \tag{5}$$

其中,$E(t)_i$ 表示节点 v_i 从仿真开始到时刻 t 历史相遇节点的个数,$E(t)_j$ 表示节点 v_j 从仿真开始到时刻 t 历史相遇节点的个数 $E(t)_i\cap E(t)_j$ 表示从仿真时间开始到时刻 t 节点 v_i 与最近分离节点 v_j 的历史相遇节点集合中相同节点的个数,如图 1 所示,$E(t)_i\cap E(t)_j=\{V_1,V_{13}\}$. $E(t)_i\cup E(t)_j$ 表示节点 v_i 与最近分离节点 v_j 从仿真开始到时刻 t 的历史相遇节点集合中总节点数,如图 1 所示,$E(t)_i\cup E(t)_j=\{V_1,V_2,V_3,V_5,V_7,V_8,V_{10},V_{11},V_{13},V_{14},V_{17},V_{19},V_{20}\}$.

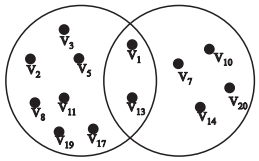

图 1　节点 v_i 和节点 v_j 的历史相遇节点集合间的关系

Fig. 1　Relationship between the historical encounters of nodes v_i and v_j

本文将 $1-\frac{E(t)_i\cap E(t)_j}{E(t)_i\cup E(t)_j}$ 简单的抽象为节点 v_i 和节点 v_j 运动轨迹的差异性. 故由 $-\left(1-\frac{E(t)_i\cap E(t)_j}{E(t)_i\cup E(t)_j}\right)$ 可知,当两个节点运动轨迹的差异性越大,节点间连接断开时 $f(t)$ 的取值越小,故 $AVF(D_i)$ 也越小. 由于 $AVF(D_i)$ 综合考虑了节点的分离时间和节点轨迹的差异性,在设计路由算法时应给 $AVF(D_i)$ 越小的节点分配更多的资源. 源节点选择中继节点时也会挑选 $AVF(D_i)$ 较小的节点作为中继节点进行消息转发时,这种策略可以使得消息在网络开销较小的前提下能快速从源节点到达目的节点.

3.4　节点连接紧密性

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

DTN 的网络拓扑处于动态变化中,单个节点要想实时获取全局或部分的拓扑信息几乎是不可能的,因此通过挑选合适的中继节点限制单个消息的副本数量是一种优秀的路由策略. 本文中节点的平均波动反映了节点在投递消息时成功概率,为了更好的说明分离时间与节点本身的关系,本文提出节点连接紧密性的概念,该概念与平均波动负相关. 当平均波动越大时,节点连接的紧密性越小;反之,越大.

定义 3. 节点连接紧密性. 为了更好的说明节点 v_i 自身的连接能力,采用指数函数来标准化 $AVF(D_i)$ 来定义节点连接紧密性. 记为 $CT(D_i)$,公式如下:

$$CT(D_i) = e^{-AVF(D_i)} = e^{\frac{\int_{t-T}^{t} f(t)\,dt}{N_i}} \tag{6}$$

由式(6)可以看出当节点的平均波动越小,节点的连接紧密性越强,它直接反映了节点 v_i 自身的连接能力.

3.5 节点连接可靠性

DTN 特殊的网络环境使得节点的移动性、自身缓存、能量和带宽等限制都可能使得消息不能够成功投递给目的节点,传统的 Prophet 中只是通过计算历史投递预测值,将消息投递给历史预测值高的节点,忽略了节点连接质量问题,当节点开启消息传输时,由于通信节点建立通信时间过短、节点间接性休眠或外界信号干扰等因素使得不能将消息完整的投递给对端节点,这造成了不必要的网络和能量开销. 本文提出节点连接可靠性来解决上述问题,节点可靠性是节点连接平均波动性的衍生概念,因为节点平均波动性只是从单个节点获取的信息进行简单积分运算,并不能充分表明节点间的连接是否可靠.

定义 4. 节点连接可靠性. 利用平均连接分离时间的标准差来定义该节点连接的可靠性. 记为 $CR(D_i)$,公式如下:

$$CR(D_i) = e^{-\sqrt{\frac{\sum_{i=1}^{N_i}(L_i - AVF(D_i))^2}{N_i}}} \tag{7}$$

其中,L_i 表示节点 v_i 与其他节点的连接分离时长.

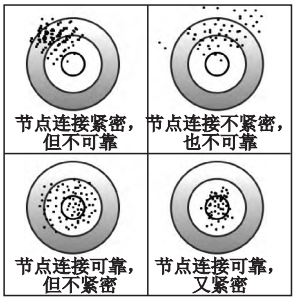

图 2 节点连接紧密性与可靠性的区别

Fig. 2 Difference between node connection tightness and reliability

为了更好的在含义上区别节点连接紧密性与可靠性,图 2 给出紧密性与可靠性的简单区别.

3.6 节点连接能力

在 DTN 网络中,衡量节点连接能力不能仅凭某一方面的优势而下定论,每种算法都有其优劣,因此本文考虑到节点连接紧密性与节点连接可靠性这两个影响因素,综合来衡量节点的连接能力.

定义 5. 节点连接能力. 节点连接能力是节点连接紧密性与节点连接可靠性加权平均值,记为 $NC(D_i)$,公式如下:

$$NC(D_i) = \alpha \times CT(D_i) + (1\text{-}\alpha) \times CR(D_i) \tag{8}$$

其中 $CT(D_i)$ 为节点连接紧密性见式(6),$CR(D_i)$ 为节点连接的可靠性见式(7). 在式(8)中 α 为节点连接能力的平滑因子,其取值范围在[0,1]. 在本文设置实验参数的仿真环境下,经大量仿真实验证明,当参数 $\alpha = 0.35$ 时,节点连接能力 $NC(D_i)$ 对网络性能提升最大. 连接能力综合考虑了节点连接可靠性和连接紧密性,在挑选中继节点时会更严苛,使得消息在投递的过程中能沿着更可靠的中继节点直至目的节点.

4 DTN 中基于连接分离时间的概率路由算法

现实的 DTN 场景中,由于环境因素及场景的需要,节点的属性各不相同,节点的属性从一方面来说也往往影响消息投递率. 设想下面情景:

当节点 v_i 在较长一段时间没有遇到中继节点,说明节点 v_i 遇到目的节点的几率较低,故它与其他相遇过节点的投递预测值应该随之动态的变化,这样才能更好的反映 DTN 网络中节点的特殊性.

本文就是在上述假设的基础上,把节点的连接分离时间作为改进 Prophet 路由算法的基础,在节点连接分离时间的基础上还衍生出节点连接紧密性以及节点连接可靠性,并将节点自身缓存占用率引入到概率路由中,得到了基于 Prophet 改进算法,即 P-AVF(Prophet routing based on Average fluctuation).

4.1 衰减公式的常数设计

传统的 Prophet 算法中没有考虑节点自身的属性与其运动的特点,只是简单选取 v_i 和 v_j 从上一次相遇到本次相遇所经历的时间单位个数 k,很难反映不同 DTN 场景以及不同节点间的差异性. 由于缓存占用率小的节点,它有更大的缓存来接收更多消息,随着节点接收更多消息又使得该节点投递率变小,这种惩罚机制也使得 DTN 中每个节点能动态确定节点本身的状态,故可以更公平的挑选出合适的中继节点进行转发. 因此本文为了使得缓存占用比小的节点承担更多中继节点的任务,结合接收消息节点缓存大小和参数 k,来表示消息接收节点缓存占用比.

$$C_r = k \times \frac{C_{total} - \sum_{i=1}^{n} M_i}{C_{total}} \tag{9}$$

其中,C_{total} 是节点 v_j 缓存大小,M_i 表示单个消息大小. k 是放大参数,它的大小决定了缓存占用比在较小的情况下并且在仿真的初期时衰减公式能够起到作用. 本文进行了大量的仿真实验,实验结果表明,当 $k = 120$ 时,C_r 对网络性能的提升效果最好.

节点 v_i 分离时间总和越大表明该节点与其他节点交互度越低,遇到目的节点的概率也就越低,因此消息投递率也会受到影响. 故本文利用消息接收节点缓存占用率和分离时间与运行时间的关系来修改式(2)中的参数 k,使得投递预测值的衰减不仅仅只依赖于分离时间. 在本文中,节点投递预测值衰减公式如式(10)所示:

$$P_{(i,j)} = P_{(i,j)old} \times \gamma^{\beta \times C_r + (1-\beta) \times \left(\frac{T_{total}}{T_s}\right)} \tag{10}$$

其中,T_{total} 表示当前时刻仿真运行总时间,T_s 表示当前时刻的上一个时间窗口 T 内节点 v_i 与其他节点连接分离的总时间.

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

在仿真实验运行初期$\frac{T_{total}}{T_s}$值较小,随着仿真时间的增加$\frac{T_{total}}{T_s}$值会越来越大,为了使节点缓存占用比这一属性能影响投递预测值,故在 C_r 中的采用放大参数 k.

式(10)主要从两个方面考虑:1)接收节点的缓存占用比,让空闲缓存越大的节点可以承担更多的转发任务;2)该节点从仿真运行总时间与其他节点连接分离的总时间比值,$\frac{T_{total}}{T_s}$运行时间是分离时间的倍数,反映该节点与其邻居节点通信总时长越小的节点,说明与目的节点相遇的概率越低. β 为参数,取值范围在 0 ~ 1 之间,β 的大小会直接影响到衰减公式变化率. 大量仿真实验证明,当 $\beta = 0.70$,对网络性能的提升最大.

4.2　更新相遇概率常数的设计

传统的 Prophet 算法用 $P_{(i,j)}$ 表示节点 v_i 与节点 v_j 之间的预测投递值,其中 $P_{init} \in [0,1]$ 是一个初始化的常量,经常相遇的两个节点之间能有更高的投递预测值. 当两个节点相遇频率足够高,但是建立连接的质量较差,每次都不能完整的投递整个消息,将会降低消息的投递率,并且不完整的消息也会占用节点的缓存,以及不完整消息传输过程中会损耗两个节点能量. 为了避免上述情况的发生,本文在挑选中继节点时采取了更为严苛的条件.

本文利用节点连接能力对传统 Prophet 算法中投递预测值的更新,对式(1)中的 P_{init} 进行重新设计,在本文中节点投递的更新公式如式(11)所示.

$$P_{(i,j)} = P_{(i,j)old} + (1-P_{(i,j)old}) \times e^{(NC(C_i)-1)} \tag{11}$$

其中,$NC(D_i)$ 为节点连接能力见式(8). 连接能力越大的节点在更新投递预测值的时候也会越大. 可以看出 P-AVF 算法在挑选中继节点时,也将节点连接能力考虑其中,从而能更好的挑选出合适的中继节点,从而控制消息副本数,降低网络开销.

从式(11)可以看出 *AVF* 的定义是本文整个算法的核心,为了更准确的描述 P-AVF 算法,本文下面给出平均波动 *AVF* 实现的伪代码.

算法 1. P-AVF 算法中平均波动 AVF

输入:时间窗口 T;

　　时间窗口 T 内节点 v_i 与其他节点相遇的节点集合为 *list*;s 为节点 v_i 与其他节点分离起始时间;

输出:*AVF*

```
1.  AVF ← 0
2.  IF 当前时间 t≥时间窗口 T THEN
3.      将 list 中的节点按 s 大小降序排列
4.      FOR node ∈ list
5.          IF node 正在与 v_i 传送消息 THEN
6.              式(4) 中的 f(t) ← 0
7.          ELSE
8.              计算节点 v_i 与 node 之间的并集
9.              计算节点 v_i 与 node 之间的交集
10.             按式(5)计算 f(t)
11.         END IF
12.     END FOR
13.  END IF
14.  N ← list 中的节点数
15.  按式(4)计算 AVF
16.  RETURN AVF
```

假设节点历史相遇节点总数为 M,由于将 *list* 中的节点按起始分离时间 s 进行排序,所以算法 1 的平均时间复杂度为 O(MlogM). 因此,计算 *AVF* 的平均时间复杂度为 O(MlogM). 假设网络中节点总数为 N,则 P-AVF 算法的平均时间复杂度为 O(NMlogM).

4.3　ACK 确认机制

DTN 中节点资源受限,为了能够充分利用节点的缓存和减少无效转发产生的网络开销,P-AVF 算法采用了 ACK 确认机制[18]以减少消息冗余的副本数量. 具体地说,当 DTN 中某个消息被成功投递到目的节点后,一些中继节点缓存空间中还存在该消息的副本,这些冗余的消息副本占用节点有限的缓存空间,使节点缓存利用率较低. 此外,由于冗余副本存在于中继节点缓存中,若这些中继节点继续对冗余消息继续进行复制转发将会浪费有限的网络带宽,导致算法的网络开销暴增,造成网络拥塞,从而使得待投递的消息资源被占用,最终会降低算法整体的投递率,并加剧网络负载率.

P-AVF 算法在设计时,给每个节点设置了一个 ACK 消息列表,当消息被成功投递到目的节点时,该消息就会被加入到节点的 ACK 列表中. 当两个节点相遇时,需要相互交换 ACK 列表,用以及时补充完整的 ACK 信息,交换完成后节点会遍历自身的消息列表,查看消息列表中是否含有 ACK 列表中存在的消息,若有则直接从本地消息缓存中删除,以减少无效的重复转发复制的操作,从而为其他待投递的消息腾出更多的缓存空间.

ACK 消息列表实现使用的是一个哈希表,查找某个消息的平均时间复杂度为 O(1),本身消息队列中的消息共有 N 个,则本文的 ACK 确认机制时间复杂度为 O(N).

由于 ACK 确认机制引入了额外的数据结构,会占用节点的一部分缓存,而且在消息投递过程中会对每个消息都进行哈希表的查找,这样额外的操作也存在能量的开销. 但与继续复制已成功投递的消息占用的缓存和能量,这部分开销是可以容忍的.

4.4　消息转发策略

本文接下来将介绍 P-AVF 路由算法的消息转发步骤以及消息转发的伪代码实现.

基于连接分离时间的概率路由过程与传统 Prophet 路由过程大致相同,只不过在传统 Prophet 的投递预测值更新、投递预测值衰减做了参数上的重新设计,使得 P-AVF 更加能够适应现实的 DTN 环境,也能更好提升消息的投递率.

路由转发机制如下:

Step 1. 节点 v_i 携带消息 m_i 在延迟容忍网络中移动,遇到节点 v_j,先判断 v_j 是否是目的节点,若是直接投递给 v_j,并执行 ACK 策略,删除 DTN 中其他节点缓存中 m_i 消息,完成本次消息的投递. 若不是,再判断当前时间 t 是否大于设定的时间窗口 T,若小于时间窗口 T,则按传统 Prophet 的方式计算投递预测值. 若大于等于时间窗口 T,则执行 P-AVF 算法. 继续执行 Step 2.

Step 2. 判断节点 v_j 到目的节点 v_d 的投递预测值是否大于节点 v_i 到 v_d 的投递预测值,若大于则 v_i 将消息 m_i 复制转发给节点 v_j. v_j 和 v_i 执行相同的功能,在 DTN 中寻找目的节点与合适的中继节点. 否则执行 Step 3.

Step 3. 节点 v_i 将继续携带消息直到遇到比节点 v_i 到目

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

的节点投递预测值更高的节点,则将消息转发给该节点,继续执行 Step 1.

上述路由转发机制的算法如下:

算法 2. P-AVF 路由转发机制

输入:时间窗口 T

输出:*AVF*

```
WHILE(TRUE)
    节点 v_i 携带消息 m_i 寻找目的节点 v_d
    遇到节点 v_j
    IF v_j == v_d THEN
        将 m_i 投递给 v_j
        BREAK;
    ELSE
        IF 当前时间 t < 时间窗口 T THEN
            执行传统 Prophet 算法
        ELSE
            按式(11)计算 P(i,j)
            按式(10)进行衰减
        END IF
    END IF
    IF P(i,d) > P(j,d) THEN
        不转发消息 m_i
    ELSE
        将消息 m_i 转发给节点 v_j
        v_j 携带消息寻找中继节点
    END IF
END WHILE
执行 ACK 机制
RETURN
```

5 仿真实验与结果分析

5.1 仿真环境配置

本实验采用 ONE[19] 仿真平台分别对 P-AVF、Prophet、BA Prophet以及S-Prophet算法进行仿真,本次仿真采用ONE自带的赫尔辛基市地图作为移动场景,区域大小为 4500m × 3400m. 具体的仿真参数设置如表 1 所示,其他参数均采用 ONE 仿真平台自带的默认设置[20].

表 1 仿真参数

Table 1 Simulation parameters

Simulation parameters	Parameter value
Simulation time/h	12
Bluetooth bandwidth/kBps	250
Message generation rate/s	23 ~ 35
TTL/min	300
Node cache/MB	5
Message size	500KB ~ 1MB
Transmission range/m	10
Number of nodes	6
γ	0.40
P_{init}	0.75

5.2 仿真设置

由式(4)可知给定时间窗口 T 的大小会直接影响节点的平均波动,从而影响到连接紧密性与可靠性的大小,进而会影响到消息投递率. 为挑选出效果最佳的时间窗口 T 值,本文在如上实验的参数的基础上做了大量的对比仿真实验,本文在此次实验中选择节点缓存大小为 15MB,绘制出如下时间窗口 T 与投递率的关系图,其他参数若无特殊说明均为 ONE 自带初始化设置,实验结果如图 3 所示.

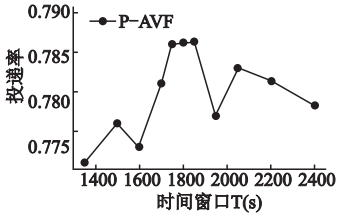

图 3 不同时间窗口 T 下 P-AVF 算法的投递率

Fig. 3 Delivery rate of P-AVF algorithm under different time window T

从图 3 中可以观察到,当 T 取值为 1800 时,P-AVF 算法的投递率最高,并且趋近 1800 时,如图 3 所示波动的平均值趋近一个固定值. 这是因为如果 T 值取的较小,节点没有足够多的历史分离次数以及足够长的分离时间来使得实验数据趋于稳定,可能会有部分的节点在此段时间内相遇其他节点的次数足够少,分离时间过长,导致该算法的投递率较低;如果 T 值较大,则 P-AVF 算法的投递预测值会被上一个时间窗口影响过大,不能及时的反映当前节点状态,致使该算法的投递率下降. 因此本文中的时间窗口 T 取值 1800.

5.3 仿真实验评价指标

本文采用以下指标来评估本次仿真实验所运行的四种路由算法的性能[21].

1)投递率(Delivery Ratio):仿真时间内由消息的源节点生成并成功投递到目的节点消息总数与网络中生成所有的消息数量之比,计算公式如式(12)所示:

$$R_{delivery\ prob}=\frac{delivered\ messages}{total\ created\ messages} \tag{12}$$

2)网络负载率(Overhead ratio):仿真时间内网络中冗余中继转发次数与成功投递消息数目之比,计算公式如式(13)所示:

$$R_{overhead}=\frac{relayed\ messages\text{-}delivered\ messages}{delivered\ messages} \tag{13}$$

3)平均时延(Average Latency):仿真环境中所有由源节点创建的消息到被成功投递目的节点的时长之和,与成功投递的消息数目之比,计算公式如式(14)所示:

$$R_{average\ latency}=\frac{\sum_{i=1}^{i=delivered\ messages} latency(i)}{delivered\ messages} \tag{14}$$

5.4 仿真结果及分析

为了验证 P-AVF 算法的高效性,本文通过改变消息的缓存大小、消息的生存周期、仿真时间以及消息产生间隔,进行了 4 组仿真实验,将 P-AVF 和 BAProphet、Prophet 以及 S-Prophet 4 种路由算法在投递率、网络负载率和平均传输时延的 3 个方面性能上进行对比,实验结果表明,P-AVF 路由算法对网络性能的提升最为明显.

5.4.1 不同的节点缓存大小

在如上的环境配置下,通过改变节点缓存容量的大小,对比四种算法在不同缓存大小下的投递率、网络负载率和平均时延的变化情况结果如图 4 所示.

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

图 4 展示了 4 种算法的路由性能随着单个节点缓存空间的变化情况,从图 4 可以看出,在 P-AVF、BAProphet、Prophet 以及 S-Prophet 算法中,P-AVF 算法的投递率最高,网络负载率最低,平均时延较大.

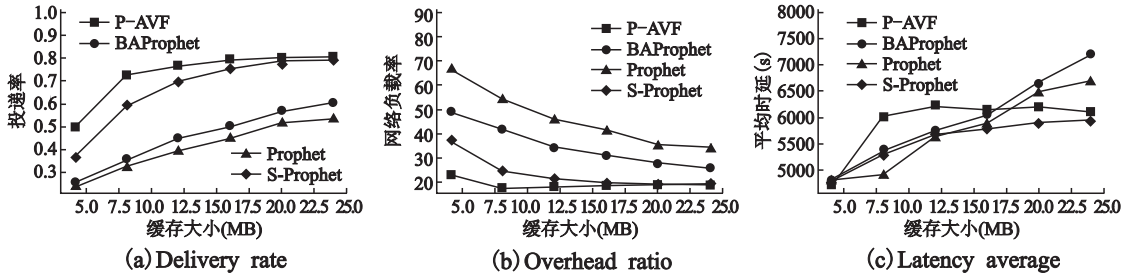

图 4　不同的节点缓存大小对算法路由性能的影响

Fig. 4　Impact of different node cache sizes on algorithm routing performance

如图 4(a)所示,随着缓存空间的增大,4 种算法的投递率都明显增大. 这是因为节点缓存空间越大,单个节点就可携带更多的消息并会降低因缓存大小不够造成的消息丢弃的概率,携带待投递消息节点遇到目的节点的概率也就随之增大,从而提高算法整体的投递率. P-AVF 算法的平均投递率相比 Prophet 算法提高 78. 73%,相比于 BAProphet 算法的平均投递率提高 61. 27%,相比于 S-Prophet 算法的平均投递率提高 10. 18%,这是因为 P-AVF 对中继节点的筛选比其他算法更加严格,并使得性能较优的节点上保持较高的投递预测值,从而提高了消息的投递率. 并且由于 ACK 确认机制,使得 P-AVF 路由算法与 S-Prophet 路由算法中的节点都有更多的缓存空间接收待投递的消息,因此这两种算法的节点可以携带更多有效消息,提升整体算法的投递率.

如图 4(b)所示 P-AVF 算法的平均网络负载率相比 Prophet 算法降低 58. 27%,相比于 BAProphet 算法的平均网络负载率降低 44. 66%,相比于 S-Prophet 算法的平均网络负载率降低 16. 98%,这是因为 P-AVF 相比较于其他 3 种来说,对中继节点的筛选更严格,在一定程度上减少了不必要的转发次数,而且 P-AVF 与 S-Prophet 都增加了 ACK 确认机制,能够及时删除已经成功投递的消息,为需要继续投递的消息腾出缓存空间,使得有效消息被丢弃的概率下降.

如图 4(c)所示随着缓存容量增大,BAProphet、Prophet 以及 S-Prophet 算法的时延都呈现上升,这是因为随着节点的缓存增大,消息能成功投递的概率变大,使原本不可能投递到目的节点的消息变为可能,整体平均时延呈现上升的趋势. P-AVF 在缓存 12. 5M-20. 0M 的范围内,网络平均延迟表现出下降趋势因为 12. 5M 后的 P-AVF 算法的投递率趋于稳定,而随着节点缓存的增大,节点同一时刻能携带更多的消息,消息被丢弃的次数也会降低,则消息能够被成功投递到目的节点的时延也会降低. P-AVF 算法的平均网络时延相比 Prophet 算法提高 2. 67%,相比于 BAProphet 算法的平均网络时延降低 1. 33%,相比于 S-Prophet 算法平均网络时延提高 6. 18%. 这是因为 P-AVF 算法选择中继节点时考虑的因素较多,从而能够有效避免将消息副本拷贝传输给综合性能较低的节点这样减少网络开销,但也降低了消息副本在当前网络中的扩散范围,所以在网络时延表现上不够优秀,另一方面也验证了 P-AVF 算法对中继节点选择的严苛,从而证实 P-AVF 算法的可行性.

5.4.2　不同的消息生存周期

图 5 展示了 4 种路由算法在不同的消息生存周期下的性能表现. 从图 5(a)可以看出,随着消息生存周期的增加,Prophet 算法与 BAProphet 算法的投递率在降低,这是因为在节点缓存空间较小的条件下,如果消息生存周期过大,消息以较长时间存在于 DTN 中,这对缺少 ACK 确认机制的 Prophet 与 BAProphet 算法在剩余缓存不足的条件下,被迫丢弃部分其他消息用于接收新消息. 另外从生存周期 200min 后 S-Prophet 算法的投递率没有明显增加,这是因为 S-Prophet 算法对中继节点的筛选过于简单,只是考虑投递预测值和节点相似性,并没有有效地挑选出优秀的中继节点. 从整体上来说 P-AVF 算法的平均投递率相比于 Prophet 算法提高 69. 20%,相比于 BAProphet 算法的平均投递率提高 51. 51%,相比于 S-Prophet 算法的平均投递率提高 12. 88%.

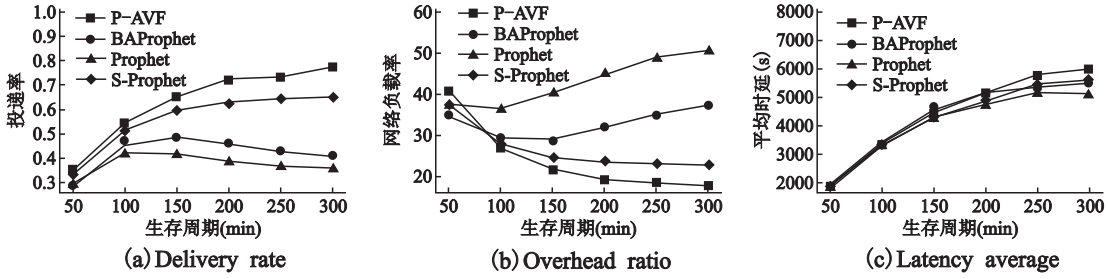

图 5　消息生存周期对算法路由性能的影响

Fig. 5　Impact of message life cycle on algorithm routing performance

从图 5(c)可以看出 S-Prophet 算法能够快速的将副本数分散到整个延迟容忍网络中,相比于 P-AVF 算法复杂的筛选中继节点的机制,S-Prophet 算法在选择中继节点机制上更为简单所需要的计算资源也更少,因此 S-Prophet 算法的平均时

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

延是最低的.

5.4.3 不同的消息产生间隔

在不同的消息产生间隔下,比较和分析各个算法在消息投递率(如图6(a))、网络负载率(如图6(b))、平均时延(如图6(c)),这3个方面变化情况.

图6展示了4种路由算法在不同的消息产生间隔下的性能表现.从图6(a)中可以得知,P-AVF算法的投递率稳定上升且上升的趋势较快,其他3种算法投递率都随着消息产生间隔增大而增大,但趋势都没P-AVF明显.从图6(b)可得知,P-AVF和S-Prophet的网络开销最小,且随着消息产生间隔增大网络开销增大不明显,而Prophet算法与BAProphet算法较为随着消息产生间隔增大网络开销增大较为明显.这是因为在消息产生间隔较小时,相同时间内整个网络会产生较多的消息,由于受限的节点缓存空间与缺乏消息冗余副本的去除机制,Prophet算法与BAProphet算法都会在消息产生间隔较小时,丢弃大量的消息,由于S-Prophet算法有ACK确认机制可以在一定程度上减少消息的丢弃,但P-AVF算法利用节点的缓存占用率以及节点连接能力等因素可以选择更合适的中继节点,在一定程度上比由于S-Prophet算法性能更优.从图6(c)可得知,P-AVF算法平均时延最大,这也再次验证了P-AVF算法的可行性,此处不再赘述原因.

5.4.4 不同的仿真时间

在不同的仿真时间下,比较和分析各个算法在消息投递率(如图7(a))、网络负载率(如图7(b))、平均时延(如图

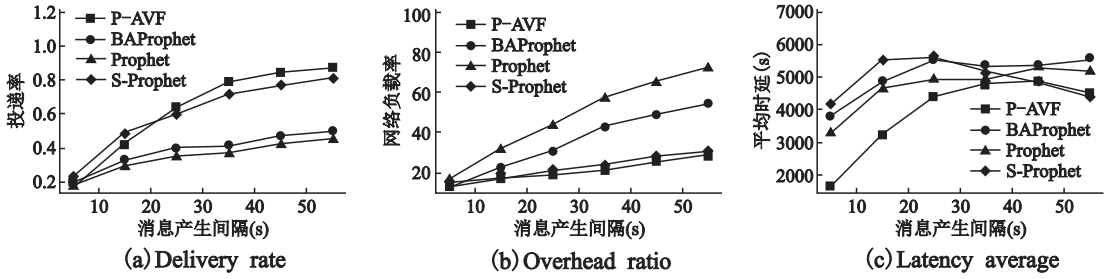

图6 消息产生间隔对算法路由性能的影响

Fig. 6 Effect of message generation interval on algorithm routing performance

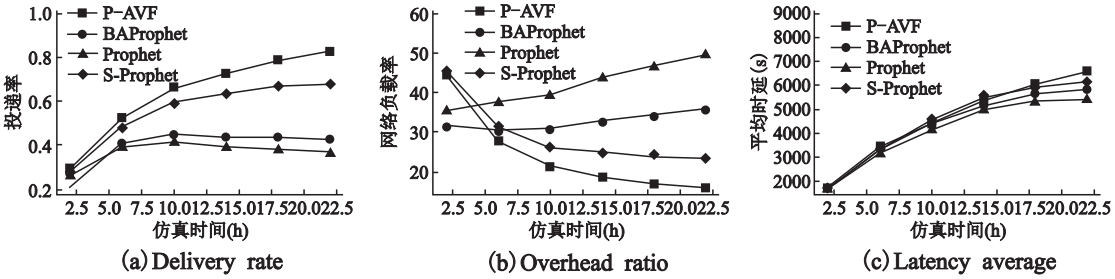

图7 仿真时间对算法路由性能的影响

Fig. 7 Influence of simulation time on algorithm routing performance

现.如图7(a)所示,在仿真进行到13小时所有算法的投递率基本维持一个定值,这是因为算法运行已达到了一个稳定状态.另外,P-AVF算法的平均投递率相比Prophet算法的平均投递率提高71.56%,相比BAProphet算法的平均投递率提高60.82%,相比S-Prophet算法的平均投递率提高14.63%.这主要是因为P-AVF算法利用时间窗口T内的分离时间等信7(c)),这3个方面变化情况.

图7展示了4种路由算法在不同的仿真时间下的性能表息与消息缓存占用率来评估节点的是否可靠,使得性能高效的节点总能保持较高的投递预测值,而Prophet算法与BA-Prophet算法的投递率较低,这是因为首先对中继节点的挑选过于简单,没有考虑节点自身能力与投递预测值的关系,并且没能及时的去除网络中成功投递的消息副本,导致网络资源被成功投递的消息消耗,引起网络拥塞并造成节点丢包.如图7(b)所示,P-AVF算法的网络负载率最低,相比于Prophet算法的平均网络负载率降低42.41%,相比于BAProphet算法的平均网络负载率降低25.61%,相比于S-Prophet算法的平均网络负载率降低16.81%,此现象原因不再赘述.如图7(c)所示,P-AVF算法的网络时延和其他参数环境下结果类似,原因不再赘述.

6 结束语

本文基于Prophet路由算法提出一种基于节点连接分离时间的概率路由算法P-AVF.该算法根据节点连接分离时间衍生出了平均波动、节点连接紧密性与可靠性以及节点能力等概念.该算法在挑选中继节点时,依据时间窗口T内的节点属性来选择性能优越的节点作为中继节点,由于对Prophet衰减公式参数进行重新设计,使得性能优异的节点能够长时间保持较高的投递预测值,同时还结合ACK确认机制降低网络负载,进一步提升了网络的整体性能.仿真实验结果表明,P-AVF算法在投递率、网络负载等方面均优于其他3种算法,但是对中继节点选择过于严苛,P-AVF算法在平均时延上也大于其他3种算法,接下来的研究应该从降低平均时延方向出发,在保持较高投递率与较低的网络负载率同时,使得算法的网络时延不要过高,从而整体上提升P-AVF的性能.

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

References:

[1] Caini C, Cruickshank H, Farrell S, et al. Delay-and disruption-tolerant networking (DTN): an alternative solution for future satellite networking applications[J]. Proceedings of the IEEE, 2011, 99(11):1980-1997.

[2] Wu Da-peng, Zhou Jian-er, Wang Ru-yan, et al. Message-redundancy estimating adaptive buffer management mechanism for opportunistic network[J]. Journal of Electronics and Information Technology, 2012, 34(1):101-107.

[3] Tornell S M, Calafate C T, Cano J C, et al. DTN protocols for vehicular networks: an application oriented overview[J]. IEEE Communications Surveys & Tutorials, 2014, 17(2):868-887.

[4] Zhang Long, Zhou Xian-wei, Guo Ji-wen. Noncooperative dynamic routing with bandwidth constraint in intermittently connected deep space information networks under scheduled contacts[J]. Wireless Personal Communications, 2013, 68(4):1255-1285.

[5] Ma Hua-dong, Yuan Pei-yan, Zhao Dong. Research progress on routing problem in mobile opportunistic networks[J]. Journal of Software, 2015, 26(3):600-616.

[6] Dai Chen-qu, Li Jian-bo, You Lei, et al. Routing algorithm based on node similarity in delay/disruption tolerant networks[J]. Journal of Computer Applications, 2014, 34(9):2476-2481.

[7] Lindgren A, Doria A, Schelén O. Probabilistic routing in intermittently connected networks[J]. ACM Sigmobile Mobile Computing and Communications Rreview, 2003, 7(3):19-20.

[8] Tao K, Abel F, Gao Q, et al. Tums: twitter-based user modeling service[C]//Extended Semantic Web Conference, Springer, Berlin, Heidelberg, 2011:269-283.

[9] Jian Tang, Zhaoshi Meng, Xuanlong Nguyen, et al. Understanding the limiting factors of topic modeling via posterior contraction analysis[C]//International Conference on Machine Learning (PMLR), 2014:190-198.

[10] Bista B B, Rawat D B. Enhancement of PRoPHET routing in delay tolerant networks from an energy prospective[C]//IEEE Region 10 Conference(TENCON), 2016:1579-1582.

[11] Xie Meng-hao, Qu Yu-gui, Zhao Bao-hua. Improvement in PRoPHET routing protocol in delay tolerant network[J]. Journal of Chinese Computer Systems, 2010, 31(1):40-44.

[12] Cui Jian-qun, Wu Shu-qing, Chang Ya-nan, et al. Probabilistic routing algorithm based on node similarity rate in opportunity network [J]. Journal of Chinese Computer Systems, 2021, 42(3):609-614.

[13] Sok P, Kim K. Distance-based PRoPHET routing protocol in disruption tolerant network[C]//International Conference on ICT Convergence(ICTC), IEEE, 2013:159-164.

[14] Du Z, Wu C, Yoshinaga Y, et al. A prophet-based DTN protocol for VANETs[C]//IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation, IEEE, 2018:1876-1879.

[15] Yu C, Tu Z, Yao D, et al. Probabilistic routing algorithm based on contact duration and message redundancy in delay tolerant network [J]. International Journal of Communication Systems, 2016, 29(16):2416-2426.

[16] Diana R, Lochin E. Modelling the delay distribution of binary spray and wait routing protocol[C]//International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), IEEE, 2012:1-6.

[17] Song You-mei, Li Jian-bo, He Tian-yue, et al. Probabilistic routing algorithm in delay tolerant network based on node similarity[J]. Computer Engineering, 2016, 42(9):63-70.

[18] Kawabata N, Yamasaki Y, Ohsaki H. On message delivery delay of epidemic DTN routing with broadcasting ACKs[C]//IEEE 41st Annual Computer Software and Applications Conference(COMPSAC), IEEE, 2017, 1:701-704.

[19] Keränen A, Kärkkäinen T, Ott J. Simulating mobility and DTNs with the ONE[J]. Journal of Communications, 2010, 5(2):92-105.

[20] Wang Zhen, Wang Xin-hua, Sui Jing-qi. Extending research for ONE simulator of opportunistic network[J]. Application Research of Computers, 2012, 29(1):272-276.

[21] Wang Zhi-hai, Gao Xue-yao. A social DTN network routing algorithm based on data link[J]. Journal of Harbin University of Science and Technology, 2020, 25(1):90-96.

附中文参考文献:

[2] 吴大鹏,周建二,王汝言,等. 机会网络中消息冗余度动态估计的缓存管理策略[J]. 电子与信息学报,2012,34(1):101-107.

[5] 马华东,袁培燕,赵 东. 移动机会网络路由问题研究进展[J]. 软件学报,2015,26(3):600-616.

[6] 戴晨曲,李建波,由 磊,等. 基于节点相似性的容延/容断网络路由算法[J]. 计算机应用,2014,34(9):2476-2481.

[11] 谢孟昊,屈玉贵,赵保华. DTN 网络中 PRoPHET 路由协议的改进[J]. 小型微型计算机系统,2010,31(1):40-44.

[12] 崔建群,吴淑庆,常亚楠,等. 机会网络中基于节点相似率的概率路由算法[J]. 小型微型计算机系统,2021,42(3):609-614.

[17] 宋有美,李建波,和天玥,等. 基于节点相似性的容迟网络概率路由算法[J]. 计算机工程,2016,42(9):63-70.

[20] 王　朕,王新华,隋敬麒. 机会网络模拟器 ONE 及其扩展研究[J]. 计算机应用研究,2012,29(1):272-276.

[21] 王志海,高雪瑶. 基于链路效能社会 DTN 网络的路由算法[J]. 哈尔滨理工大学学报,2020,25(1):90-96.

(C)1994-2024 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net