UWSN 中基于随机游走模型的可靠路由算法研究

朱 剑[1)] 刘 君[1)] 赵 海[1)] 徐 野[2)]

[1)](东北大学信息科学与工程学院 沈阳 110819)

[2)](沈阳理工大学信息科学与工程学院 沈阳 110159)

摘 要 UWSN(Underwater Wireless Sensor Networks)相较于传统的无线传感器网络采用了声信号进行数据传输,由于高传输延迟的引入,冲突类数据丢失现象凸显,网络可靠通信面临全新的挑战.为了在这样的环境中实现低消耗、高可靠网络通信这一目的,文中设计了一种最小化冲突概率路由算法 MCR(Minimum Conflict probability Routing).该算法融合了网络节点的度值和节点工作负载,形成了一种全新的路由策略 DBM(Degree and Buff based Metric).在该路由策略基础上,采用图论中的随机游走模型对源节点与 sink 节点之间的路径进行选择.MCR 算法的核心思想是选择两点之间冲突概率最低的路径完成数据传输,虽然该算法不能从 Mac 层解决冲突类丢包问题,但是从基于 NS-2 的仿真实验结果来看,在 UWSN 环境下,MCR 算法相较于传统路由算法有效地减少了路径中的冲突类丢包概率,提升了端到端链路可靠性、具有较高较稳定的网络吞吐量.

关键词 水下无线传感器网络;数据冲突;路由策略;路由算法;物联网

中图法分类号 TP393 **DOI 号** 10.11897/SP.J.1016.2016.01007

Research on Reliable Routing Protocol Based on Random-Walk Model in UWSN

ZHU Jian[1)] LIU Jun[1)] ZHAO Hai[1)] XU Ye[2)]

[1)](*Institute of Information Science and Engineering, Northeastern University, Shenyang* 110819)

[2)](*Institute of Information Science and Engineering, Shenyang Ligong University, Shenyang* 110159)

Abstract Compared with WSN, UWSN uses acoustical signal as the transmission medium, this change increases the probability of data loss in UWSN caused by signal conflict for the reason of high transmitting delay. In order to achieve a low consumption and reliable network communication in such environment, an algorithm called MCR (Minimum Clash probability Routing) is proposed in this paper. MCR gives out a routing metric called DBM (Degree and Buff based Metric) by combining nodes' degree with nodes' workload, and finds out the best communication path by random-walk theory in basis of DBM. The main idea of MCR is to select a path with lowest conflict probability between two nodes to finish the data transmission. The result of simulation experiment based NS-2 shows that, MCR can reduce the probability of data loss caused by signal conflict, increase the reliability between two nodes, and has a higher and more stable throughput than traditional routing algorithms, although it can't resolve the conflict in Mac layer.

Keywords UWSN; data conflict; routing metric; routing algorithm; Internet of Things

收稿日期:2014-07-29;在线出版日期:2015-07-23.本课题得到辽宁省科技项目教育厅一般项目(L2012088)、中国自然科学基金面上项目基金(61373159)资助.**朱 剑**,男,1981 年生,博士,讲师,主要研究方向为无线传感器网络、水下无线传感器网络.E-mail:zhujian@ise.neu.edu.cn.**刘 君**(通信作者),女,1982 年生,博士,讲师,中国计算机学会(CCF)会员,主要研究方向为网格计算.E-mail:liujun1@ise.neu.edu.cn.**赵 海**,男,1959 年生,博士,教授,博士生导师,主要研究领域为无线传感器网络、复杂网络.**徐 野**,男,1976 年生,博士,教授,主要研究领域为无线传感器网络、复杂网络.

1 引 言

地球 71%的表面被海洋所覆盖,这些未被探索的区域内蕴含着丰富的资源和未知的可能性,与陆地的探索进度相比,对于海洋环境的了解,人类仍停留在一个初级阶段.随着沉船探索等一系列水下探索行为的开展,人们对海洋等深水环境的关注程度也与日俱增.但是深水环境中充满着不确定性和危险,很多区域是人类无法涉足的,例如马里亚纳海沟最深的地方有 10 000 多米,一些特制的探测仪都会被高水压破坏,并且单个探测仪器探测区域受限、探测成本很高.在这样的一个形势下,人们开始考虑将无线传感器网络应用在该深水环境下,形成一种水下无线传感器网络 UWSN(Underwater Wireless Sensor Networks)[1-5].

水下环境与陆地环境之间存在很大的差别,例如电磁波无法传输、节点易受水流影响而发生位移等,这些物理特性使得传统无线传感器网络无法直接应用于水下环境.在传输媒介上 UWSN 采用声频信号取代原来的电磁波,使得节点之间能够通过无线的方式进行远距离通信,解决了电磁波的局限性,但是传输媒介的改变导致了水下无线传感器网络很多关键技术需要重新设计.声音传播的速度与电磁波相比少了 5 个数量级,两节点之间的信号传输延迟上升到了秒级,这不但使得传统的一些路由算法变得低效,也使得传统无线网络中类似隐藏终端的冲突丢包问题变得凸显[6].水下无线传感器面临的这些问题对研究者而言是一个全新的挑战,本文的主要工作是针对 UWSN 中冲突类丢包现象提出一种高效、高可靠的路由算法,提高 UWSN 的网络传输性能.

传统 WSN(Wireless Sensor Networks)中,路由算法往往会先建立好源节点到目的节点的路径,当源节点有数据需要发送时,它仅仅按照路径的设定将数据包发送给下一跳节点,下一跳节点也会按照路径继续将数据向后传输,通过这种方式,数据包最终会被传送至目的节点.这种路由协议的主要任务是建立路由,一旦路由建立成功,则每个节点按照路由的标定进行传输即可,AODV 就是一个经典算法,通过 REQ 探测包建立源节点与目的节点间跳数最少的路径,这种路由协议中的路径建立需要正向路径和反向路径的闭环通信成功才能完成,在 UWSN 中,丢包现象严重,闭环通信成功的几率很低,成功建立一次路由需要大量探测包和时延.此外,即使成功建立了一条路由,该路由的性能和寿命都不确定.

在 UWSN 中直接应用传统的路由算法将会导致低效的性能,这在后文中的实验中将会进一步描述.UWSN 环境恶劣,需要快捷有效的方法才能获得较高的路由性能.常用的算法往往是基于位置信息的定向扩散,当源节点有数据要发送时,中继节点即时地通过路由计算方法计算出最优的下一跳,并将数据发送给这个下一跳.对于这样的协议,不需要花费大量的能量、时间去建立或者维护路由信息.这类协议中的最关键问题是如何设计算法实时地找到最优的下一跳.一些路由协议采用广播技术和地理位置信息来计算下一跳,这样的路由协议比较适用于移动性较强的网络中,但是在路径的 QoS 服务性能方面欠缺考虑,并且对于大规模网络往往是通过局部算法进行计算,最终可能导致数据无法到达目的节点,降低了其实用性[7].针对上述问题,本文提出了一种 MCR 路由算法.

UWSN 中丢包原因主要有硬件类、冲突类和信号衰减类.硬件类丢包是指由于设备故障造成数据包丢失,冲突类丢包是指信号由于互相干扰造成数据丢失,信号衰减类丢包是指节点由于移动或障碍物阻隔等原因造成信号强度较弱产生丢包[8].本文主要针对 UWSN 中冲突类丢包进行了研究,基于网络拓扑结构和网络节点工作状态设计了一种路由策略 DBM(Degree and Buff based Metric),在此基础上,采用随机游走思想完成 UWSN 中源节点与目的节点之间的路由建立,降低数据冲突概率,提升网络性能.

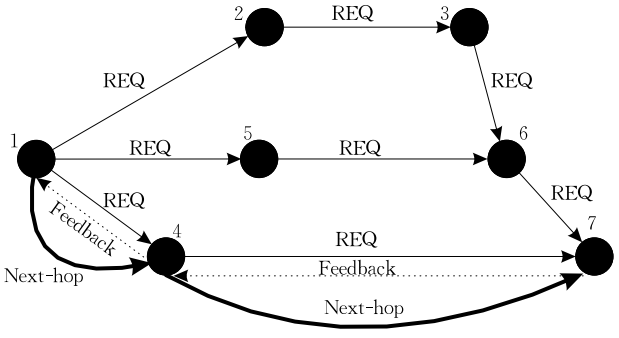
2 相关工作

在近几年内,一些针对 UWSN 的路由算法产生了,大部分协议主要是基于位置信息的[9-12].文献[10]提出了一种 H2-DAB,该协议将 UWSN 从水下至水面分成若干个层,每一层都会有一个 sink 节点,然后通过两个阶段的工作来完成数据的路由.第 1 阶段,sink 节点通过发送 Hello 包为每个悬浮节点分配一个动态的 *ID*,每个节点收到 Hello 包后会更新信息并且继续进行广播.因此,最终每个节点都会获得一个 *ID*,*ID* 的数值和层是相关的,越是深处的节点对应的 *ID* 的数值越大.第 2 阶段,数据包通过这些 *ID* 进行传输,数据包总是被发送到较小的 *ID* 节点,最终数据包就像物体浮出水面一样会被传送到水面.通过这种协议能够简单地进行数据采集,

但是该协议对水下网络拓扑布局有着非常严格的分层要求，并且分配 *ID* 也会花费较多的时间和带宽消耗. 此外，节点的频繁漂移使得 *ID* 的更新要非常及时才能确保可靠通信. 为了提升该协议的可靠性，文献[11]基于 H2-DAB 提出了一种 2H-ACK 模型.

VBF[12]是一种经典的基于泛洪和位置信息的路由算法，每个节点都知道自身的位置坐标，一次数据发送过程内，在源节点的邻居节点中，距离目的节点距离向量最短的节点将会被选为源节点的下一跳. 在 VBF 算法中，数据包可能无法传输到水面上的 sink 节点，这是因为当网络较为稀疏时，可能会出现一种情况：计算出的源节点下一跳节点，无法继续与后续节点通信. 为了提升 VBF 的可靠性，文献[9]提出了一种 HH-VBF 协议，该协议为每一个节点都使用了距离向量，和之前的静态源节点至目的节点距离向量相比，具有更高的动态性和可靠性，但是 HH-VBF 仍属于局部最优求解方式，并不能完全解决 VBF 存在的问题. 针对上述问题，本文提出了 MCR(Minimum Clash probability Routing)算法.

图 1 显示了一种经典路由算法 AODV 的流程示意图[13]. 图 1(a)中 1 号节点是源节点，7 号节点是目的节点，1 号节点没有 1→7 的有效路由. 当 1 号节点要给 7 号节点发送数据时，1 号节点会首先通过泛洪方式广播 REQ 探测包，数据包格式如图 1(b)，REQ 数据包每次被转发时将会更新 Path 与 Hops 两个字段.



(a)

类型	标志位	Path	Hops
REQ标识(可选)			
目的节点的IP地址			
目的节点序列号			
源节点IP地址			
源节点序列号			

(b)

图 1　AODV 协议中的路由建立示意图

例如当 1 号节点发出的 REQ_1 到达 4 号节点的时候，Path 字段被更新为 1→4，Hops 字段被更新为 1，这两个字段更新完毕后就会产生一个新的 REQ_4 数据包，4 号节点判断自己是否为该 REQ 的目的节点，若不是，则 4 号节点会继续广播该新的 REQ_4. 当 REQ_4 到达 7 号节点时，Path 字段被更新为 1→4→7，Hops 字段被更新为 2，与此同时 7 号节点判断出自己就是目的节点，则 7 号节点不再广播 REQ，而是等待一段时间 T，然后开始整理收集到的 REQ 数据包，并选择 Hops 最小的路径作为最终路径，若存在多条，则随机选择一条. 例如对于图 1(a)，7 号节点最终会获取代表 3 条路径的 REQ：1→4→7，1→5→6→7 与 1→2→3→6→7，对应的 Hops 分别为 2、3、4，7 号节点最终会选择 1→4→7 作为最终通信路径. 为了告知 1 号节点这条路径的成功建立，7 号节点将会按照 REQ 的逆向路径发送 Feedback 数据包，与 REQ 不同，Feedback 数据包不是广播而是根据 REQ 中记录的路径逆向定点传输，与 REQ 类似，Feedback 数据包每被转发一次将会更新 Path 字段，这样 1 号节点收到 7 号节点的 Feedback 数据包后，就可以从 Path 字段中提取出建立的路径 1→4→7. 类似 AODV 传输协议在传统无线传感器网络中应用较多，其优点在于建立的路径可靠有效，并且可以通过 Hops 字段选择最短路径，提高通信效率，节约能量. 即使网络节点发生变化，也可以通过定期发送 REQ 更新路由信息的方式解决该问题.

与传统的无线传感器网络应用环境不同，UWSN 的通信无法通过电磁波进行传递，因为电磁波在水介质中损耗很大. 目前在 UWSN 通信中应用较多的通信方式是声通信，声通信具有如下两个特点[14]：

(1) 声传播的速度与电磁波传播速度差距较大.

(2) 声传播的带宽较窄，通信速率较低.

UWSN 的应用环境较为复杂，各类丢包现象凸显. 若在这样的环境中应用类似 AODV 的传输协议，则会出现不同的结果. 在采用声通信的 UWSN 中，1 号节点通过泛洪方式发送 REQ 继而等待目的节点反馈回来的 Feedback 对应的过程将会非常“漫长”，大幅度降低了网络端到端吞吐量[15]；由于 UWSN

信号传播速率的降低,网络同步、CSMA/CA 等机制无法体现出以电磁波为介质时的性能,网络冲突可能性急剧增加,例如隐藏终端冲突现象如图 2 所示.

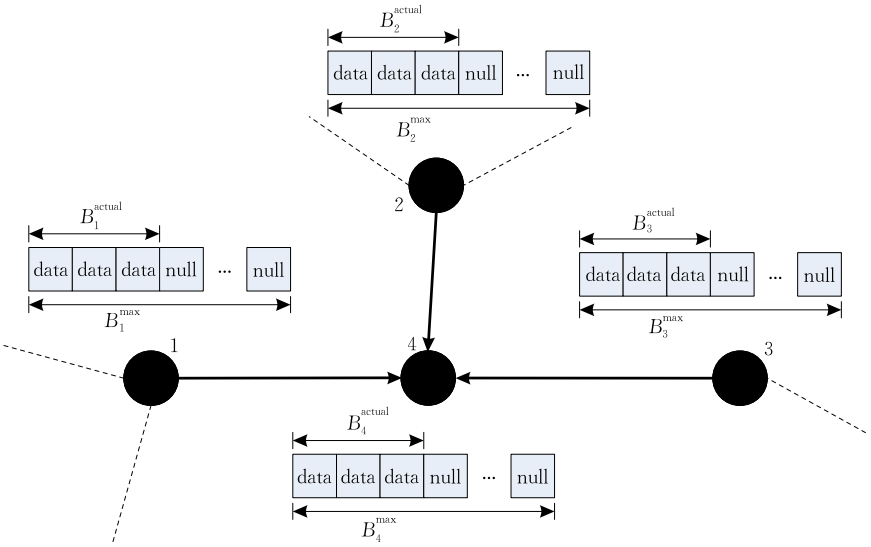


图 2 隐藏终端问题

图 2 中,1、2、3 号节点为 4 号节点的邻居节点,且 1、2、3 号节点彼此均不连通,此时当 1 号节点给 4 号节点发送数据时,2、3 号节点探测到信道仍为空闲,也同时向 4 号节点发送数据,从而信号在 4 号节点处冲突,造成数据丢失.

目前解决隐藏终端的方法有两种:一种是接收节点在接收的同时发送忙音来通知邻居节点,即 BTMA(Busy Tone Multiple Access)系列;另一种方法是发送节点在数据发送前与接收节点进行一次短控制消息握手交换,以短消息的方式通知邻居节点它即将进行接收,即 RTS/CTS 方式. 这种方式是目前解决这个问题的主要趋势,如已经提出来的 CSMA/CA、MACA、MACAW 等. 还有将两种方法结合起来使用的多址协议,如 DBTMA. 但是上述思路在 UWSN 中却并不适用,因为 UWSN 中信号传输时间不可忽略[16],即使 4 号节点及时发出了忙音,但是 2、3 号节点可能在忙音到达之前发出了数据[17-18]. 此外,在高传输延迟的 UWSN 中很难实现时钟同步,基于时钟同步的隐藏终端解决方法也难以取得较好的收益. 为此本文从路由协议设计角度,设计了 MCR 路由算法,优化端到端的路径选择,降低端到端路径中冲突类丢包发生的几率,提高网络性能.

3 MCR 协议的提出

本节主要分为两部分,3.1 节主要介绍了 DBM 路由策略及其相关结论;3.2 节在 DBM 基础之上采用随机游走算法形成了 MCR 路由协议.

3.1 DBM 路由策略

本文主要关注的是数据在传输中冲突类数据包丢失问题. UWSN 和其他类型的无线通信节点一样,每个节点均具有一个接收缓存 RxBuff 和发送缓存 TxBuff. 当节点接收到一个数据包后,该数据包首先会存入 RxBuff 中,然后按照协议格式解析该数据包,获得信息;当节点想发送一个数据包时,该数据包会被放置在 TxBuff 中,等待硬件控制信号来进行数据发送. 为了简化描述本文将这两个缓存统一标记为 $B_{node\_id}^{length}$,其中 node_id 代表节点的编号,length 代表缓存的大小,在后文中 length 可以取两个值:max 和 actual,其中 $B_{node\_id}^{max}$ 表示该节点缓存的最大值,该值通常是预先设定的,$B_{node\_id}^{actual}$ 表示该节点缓存的实际利用值,表示节点在工作中被占用的缓存空间长度.

对于两点之间路径上的信号冲突概率计算有一种方法如下:节点 i 与 j 之间有 m 个中间节点,当节点 i 向节点 j 发送数据时,统计这 m 个中间节点的所有邻居节点数量之和 Mum,认为 Mum 越大的路径对应的冲突概率也越大. 但是若两条路径的 Mum 值相同(例如均为 10),其中一条路径的中间节点个数为 3,中间节点对应的邻居节点数依次为 3、3、4;另一条路径的中间节点个数为 2,中间节点对应的邻居节点数依次为 5、5,最终的两条路径的冲突概率存在着很大差别,因此上述方法存在局限性.

本文将端到端之间链路的冲突概率转变为路径

上两点之间冲突概率的乘积. 假设路径中有 n 个中间节点，每个中间节点 i 的度值为 d_i，节点 i 成功转发数据的概率为 $1/d_i$，端到端的成功交付概率的计算方法如下：

$$S_n=\prod_{i=1}^{n}\frac{1}{d_i},\ d_i\in[1,+\infty) \tag{1}$$

式中 S_n 表示由 n 个节点形成的路径内，在冲突干扰情况下成功进行数据交互的概率. 从式(1)可以看出，当转发节点的度值趋于无穷大时，成功转发的概率为 0；当转发节点的度值为 1 时，成功交互概率为 1. 式(1)属于拓扑层面的概率估计，在实际应用中式(1)存在一个问题，某个节点的度值较大，但是该节点的邻居节点并没有数据要发送，即每个邻居节点均不抢占信道，也就不会与该节点发生冲突，此时若以式(1)进行计算，则计算结果和实际情况将会出现较大差别. 针对上述问题，本文将邻居节点的工作状态引入到了式(1)中. UWSN 中难以实现时钟同步，基于时隙分配的节点工作机制难以执行，本文研究认为 UWSN 中的节点均采用自由竞争的方式抢占信道，即当某个节点发现发送缓存中有数据需要发送，则通过自由竞争的方式进行数据发送，若没有数据需要发送，则进入节能休眠状态. 缓存占用率能够标示节点竞争信道的频度，频度影响着冲突概率，因此本文引入了缓存占用比率 $\frac{B_k^{\text{actual}}}{B_k^{\max}}$ 对式(1)中的概率进行修正.

图 3 描述了一条通信路径 1→2→3，这条路径存在两个转发过程：1→2 与 2→3. 为了简化分析，本文做出如下两点假设：

(1) 冲突发生在转发过程中的接收端(本文主要针对隐藏终端类冲突问题)，例如本文称转发过程 1→2 中的转发节点为 2 号节点，该转发过程的成功率称为 2 号节点转发成功率.

(2) 1 号节点向 2 号节点转发数据后将不会成为 2 号节点的干扰源，2 号节点的干扰源是指除了转发过程中源节点外的其他邻居节点(因为通常 1 号节点转发数据后需要收到 2 号节点的应答才会继续发送数据，之间有良好的协调).

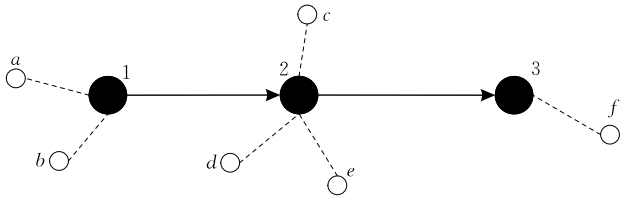


图 3　隐藏终端类冲突问题

依据上述两点假设，可认为干扰源 a、b 并不会对这条路径进行干扰，在转发过程 1→2 中，成功概率等于 1 减去 2 号节点干扰源干扰的概率，并且 2 号节点的干扰源不包括转发源节点 1 号节点，以此类推可以得出节点 i 的转发成功率 S_i 的计算方法：

$$S_i=\begin{cases}1-\dfrac{1}{d_i}\displaystyle\sum_{l=1}^{d_i-1}\frac{B_l^{\text{actual}}}{B_l^{\max}}, & d_i>1\\ 1, & d_i=1\end{cases} \tag{2}$$

式(2)中 d_i 是节点 i 的度值，取值范围为 1 到无穷大之间的正整数，l 表示节点 i 的邻居节点逻辑编号. 依据假设 1 可知，节点 i 是指一条路径中除源节点外的其他节点，例如图 3 中的 2、3 号节点. 当 d_i 为 1 时，即两点通信，依据假设 2 可知转发成功率为 1. 即使转发节点 i 的度值为无穷大，若干扰节点中缓存占用比率均为 0，则 i 节点转发数据成功率仍为 1，缓存占用率与度值共同决定了 i 节点转发成功率.

DBM 路由策略定义：在一条路径 l 中，有 n 个节点，则该路径的 DBM 值计算方法如下：

$$DBM_l=\prod_{i=1}^{n-1}S_i \tag{3}$$

式(3)中，除源节点外还有 $n-1$ 个转发节点. 通过式(2)、(3)可以看出，当两个相邻的节点 i、j 进行 $i\to j$ 数据转发的时候，可以总结得出如下几点结论：

(1) 当节点 j 的邻居节点数目趋于无穷大时，若每个邻居节点的缓存占用率均为 1，则转发过程 $i\to j$ 成功的概率也将趋于 0.

(2) 当节点 j 的邻居节点数量趋于无穷大时，但是每个邻居节点的待发数据量均为 0 时，此时转发过程 $i\to j$ 的成功概率为 1.

(3) 无论节点 j 的邻居节点数量多少，邻居节点中的待发数据量是多少，转发过程 $i\to j$ 均是有成功概率的，可以趋于 0 但是不会等于 0.

(4) 当节点 j 有 m 个邻居节点，且每个邻居节点的缓存均为满负荷状态时(即 $\frac{B_k^{\text{actual}}}{B_k^{\max}}=1$)，转发过程 $i\to j$ 成功的概率为 $1/m$.

由式(2)、(3)可知，DBM 受缓存占用率影响，而缓存占用率会随着时间的推移而改变，因此通过 DBM 策略选择的路由可能并不是当前网络最优的路由. 降低该可能性的方法主要有如下两种：

(1) 和 AODV 协议中周期发送 REQ 探测包维护路由的时效性一样，本文可以周期地通过单跳 hello 包更新各个节点对应邻居节点的缓存占用

率,使得 DBM 值具有较高的时效性,从而降低上述现象发生的可能性.相应地,网络带宽消耗也会增加,因此需要结合具体网络设定恰当的更新周期.

(2)规范网络布局和节点的数据采集流程.UWSN 主要功能是周期采集数据,若能够对节点布局和数据采集进行规范化,则在任何时刻网络中各个节点的缓存占用率均可以通过数学推导形成占用率与时间的函数,从而降低上述现象发生的可能性.但相应地计算复杂度和布设成本将会随着网络规模的增加而增加.

3.2 MCR 协议

上文介绍了 DBM 路由策略的概念及其计算方法,本节主要介绍如何将该策略应用于路由算法中.图 4 描述了 UWSN 的一种网络布局,UWSN 网络中存在一个 sink 节点和若干普通节点,sink 节点通常具有较高的性能,通常被放置在水面上,主要负责收集 UWSN 中的普通节点采集到的数据并通过 GPRS 等服务将这些数据转发至 Internet.普通节点打开电源后开始进行数据采集,采集到的数据会存入自身缓存中,等待竞争信道向 sink 节点发送数据.对于网络中的一个普通节点,当其工作时并不知道通过哪条路由才可以将数据传送到 sink 节点,此时需要进行路由建立.UWSN 环境中冲突干扰频繁,

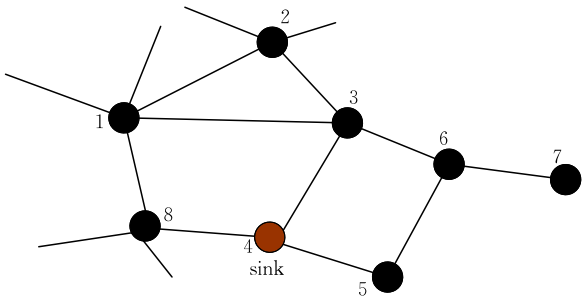


图 4 网络拓扑示意图

在这样的不稳定环境下,若想获得一条性能较高的路由,通常有如下两种思路:

(1)通过周期广播足够数量的探测包对路由质量进行评估.

(2)基于网络相关参数的计算进行路由质量评估.

UWSN 中的传感器节点对数据的采集通常是规律性的行为,即缓存中数据的增长速率是相对稳定的;尽管网络会出现变化,但总体来看拓扑结构仍是相对稳定的.因此,上文的 DBM 计算结果也会相对稳定.DBM 策略带宽消耗主要集中在基于一跳 hello 包的邻居信息采集上,在后续的路径建立过程中源节点只需要发起一次路由建立请求 REQ 广播就可以完成路由建立,避免了类似 AODV 协议中的 Feedback 数据包导致的带宽消耗(在 UWSN 中,REQ 与 Feedback 反馈包均成功被交付的几率很低,因此 AODV 协议在 UWSN 中将会导致大量带宽消耗).综上,采用 DBM 对路由质量进行评估相比于传统路由协议能够在减小一部分带宽消耗的基础上获得较为稳定的评估结果,具有较高的实用性,为此本文选择了方法 2.

图 4 中,当 7 号节点有数据需要发送给 sink 节点时,7 号节点发起一次路由建立请求 REQ 广播,REQ 数据报文格式和图 1(b)基本一致,但是在 hops 字段后增加了邻居信息表 ToN_i 字段,邻居信息表如图 5 所示,该表属于链表结构,记录了接收到 REQ 节点的邻居节点 ID 和邻居节点对应的缓存占用率,该表的建立主要通过周期发送 hello 包,hello 包只有一跳的寿命,通过 hello 包除了能获得 ToN_i 同时也能及时发现邻居的变化,发送周期并不需要太频繁.当网络中某个节点接收到 REQ 后,会将自己的邻居信息表添加到 REQ 中的 ToN_i 字段中,然后将更新后的 REQ 继续向前传递,ToN_i 字段对应的链表内容也会逐渐增加.

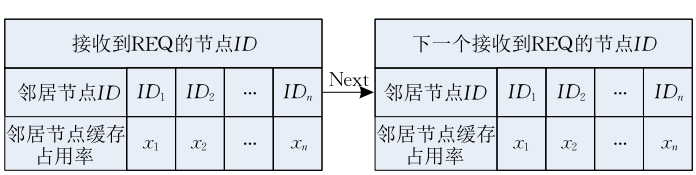


图 5 邻居信息表 ToN_i 的结构示意图

为了防止广播风暴的发生,可以设置 REQ 的生命周期为 L hops,REQ 每被转发一次 L 就会减 1,当 L 值为 0 时,REQ 不再被发送,且广播中通过 ID 标志解决循环广播问题,即当 REQ 被节点 i 发到节点 j,则节点 j 广播后 i 将不会接收.例如在图 4 中,由 7 号节点发起一次 REQ,将 L 设置为 3,则这次 REQ 广播涉及到的节点 ID 分别为 1、2、3、4、5、6、7.依照这种 REQ 机制,最终 sink 节点会得到一个全面的 ToN_i 字段,并进行路由选择计算.为了简化描述计算过程,本文假设图 4 中所有节点的缓存

占用率均为 50%，$L=3$，7 号节点向 sink 节点发送的 REQ 数据包将会涉及到 1～7 号节点，各节点度值和缓存占用率见表 1.

表 1　节点度值及缓存占用率

	占用率/%	度值
1	50	5
2	50	4
3	50	4
4	50	3
5	50	2
6	50	3
7	50	1

由 3.1 节介绍可知，当节点 i 向节点 j 发送数据时，节点 j 是该过程的转发节点，式(2)中，只考虑了节点 j 的转发成功率，并没有考虑节点 i 与节点 j 的连通关系，当节点 i 与 j 不连通时，无论节点 j 转发成功率多大，则 $i\rightarrow j$ 过程的转发成功率均为 0. 为此，对式(2)进行了修正，设 $i\rightarrow j$ 转发过程的成功率为 Sp_j，则该成功率的计算方法如下：

$$Sp_j=S_i\times l_{i,j} \tag{4}$$

其中

$$l_{i,j}=\begin{cases}1, & \text{节点 } i \text{ 和 } j \text{ 相连}\\ 0, & \text{节点 } i \text{ 和 } j \text{ 不相连}\end{cases} \tag{5}$$

S_i的定义见式(2). 式(4)的核心思想是节点 j 的度值和邻居节点的缓存占用率一旦确定，则节点 j 转发数据的成功率也将确定，无论是哪个源节点与 j 通信，转发成功率均相同，但是若某个节点与 j 不相连，则 j 转发该节点数据的成功率为 0，通过邻居信息表内容可以容易地判断出两节点是否相连. 通过式(4)可以计算出网络中任意两点间的转发成功率，并形成一个 $M\times M$ 的矩阵$\widetilde{\boldsymbol{L}}$($M$ 为节点数量)，结合随机游走理论模型，可以计算出从源节点出发到达网络中任意节点(包含 sink 节点)的成功概率，计算方法如下：

$$\widetilde{\boldsymbol{S}}_s^{f+1}=\widetilde{\boldsymbol{L}}\times\widetilde{\boldsymbol{S}}_s^{f} \tag{6}$$

其中，$\widetilde{\boldsymbol{S}}_s^{f}$是一个 $M\times 1$ 矩阵，该矩阵代表数据由节点 s 出发经历 $f(f\geq 0)$ 步随机游走后，到达网络中其他节点的成功概率. 依据上文给出的转发成功概率定义可知当 $f=0$ 时，$\widetilde{\boldsymbol{S}}_s^{0}$即为起始矩阵，该矩阵中，除了源节点对应的位置为 1，其他均为 0.

为了更直观理解 MCR 协议原理，本文给出了随机游走算法在路径搜索中的具体操作细节：

(1)结合网络拓扑中各节点度值与各节点邻居节点的缓存占用率，形成一个 $M\times M$ 的矩阵$\widetilde{\boldsymbol{L}}$.

(2)依据数据源节点的 ID 号，形成一个初始矩阵 $\widetilde{\boldsymbol{S}}_s^{0}$.

(3)依据网络规模或者经验值，设计好随机游走的步数 f.

(4)基于式(6)，可以得到 f 步随机游走之后，源节点到达网络中各个节点的成功概率，概率值将会被存储在矩阵 $\widetilde{\boldsymbol{S}}_s^{f}$中.

以图 4 为例，设 7 号节点为数据源节点，该节点将会产生数据，并要将数据传至 sink 节点. 设 REQ 的生命周期为 3 hops，节点相关参数见表 1，节点 4 为 sink 节点，也就是 REQ 的目的节点，则 7 号节点与 sink 节点间各条路径转发成功率的计算流程如下：

$$\widetilde{\boldsymbol{S}}_7^{1}=\widetilde{\boldsymbol{L}}\times\widetilde{\boldsymbol{S}}_7^{0}=$$

$$\begin{Bmatrix}
ID & 1 & 2 & 3 & 4 & 5 & 6 & 7\\
node_1 & \backslash & 1\times\frac{7}{12} & 1\times\frac{7}{12} & 0 & 0 & 0 & 0\\
node_2 & 1\times\frac{5}{8} & \backslash & 1\times\frac{5}{8} & 0 & 0 & 0 & 0\\
node_3 & 1\times\frac{5}{8} & 1\times\frac{5}{8} & \backslash & 1\times\frac{5}{8} & 0 & 1\times\frac{5}{8} & 0\\
node_4 & 0 & 0 & 1\times\frac{2}{3} & \backslash & 1\times\frac{2}{3} & 0 & 0\\
node_5 & 0 & 0 & 0 & 1\times\frac{3}{4} & \backslash & 1\times\frac{3}{4} & 0\\
node_6 & 0 & 0 & 1\times\frac{2}{3} & 0 & 1\times\frac{2}{3} & \backslash & 1\times\frac{2}{3}\\
node_7 & 0 & 0 & 0 & 0 & 0 & 1 & \backslash
\end{Bmatrix}\times$$

$$\begin{Bmatrix}
S_{7\rightarrow 1}^{0} & 0\\
S_{7\rightarrow 2}^{0} & 0\\
S_{7\rightarrow 3}^{0} & 0\\
S_{7\rightarrow 4}^{0} & 0\\
S_{7\rightarrow 5}^{0} & 0\\
S_{7\rightarrow 6}^{0} & 0\\
S_{7\rightarrow 7}^{0} & 1
\end{Bmatrix}=\begin{Bmatrix}
S_{7\rightarrow 1}^{1} & 0\\
S_{7\rightarrow 2}^{1} & 0\\
S_{7\rightarrow 3}^{1} & 0\\
S_{7\rightarrow 4}^{1} & 0\\
S_{7\rightarrow 5}^{1} & 0\\
S_{7\rightarrow 6}^{1} & \frac{2}{3}\\
S_{7\rightarrow 7}^{1} & \backslash
\end{Bmatrix} \tag{7}$$

式(7)中，$node_i$行代表节点 i 的成功转发概率，1～7 代表节点序号，例如节点 1 转发节点 2 或 3 的成功概率均为 7/12. $S_{7\rightarrow i}^{0}$代表节点 7 经过 0 步随机游走时到达各个节点 i 的成功率. 因此 7 号节点通过 1 跳(即 1 步随机游走)到达各个节点的成功概率计算结果见式(7)，不难看出 7 号节点在 1 跳到达 4 号节点的成功率 $S_{7\rightarrow 4}^{1}$为 0. 从图 4 中可以看出，当 L 为 3 时，7 号节点将会存在成功传输给 sink 节点的概率，因此为了简化描述，本文直接计算 3 跳后 7 号节点成功交付数据给网络中其他节点的概率矩阵，计算结果如下：

$$\tilde{\boldsymbol{S}}_7^2=\tilde{\boldsymbol{L}}\times\begin{Bmatrix}S_{7\to1}^1 & 0\\ S_{7\to2}^1 & 0\\ S_{7\to3}^1 & 0\\ S_{7\to4}^1 & 0\\ S_{7\to5}^1 & 0\\ S_{7\to6}^1 & \frac{2}{3}\\ S_{7\to7}^1 & \backslash\end{Bmatrix}=\begin{Bmatrix}S_{7\to6\to1}^2 & 0\\ S_{7\to6\to2}^2 & 0\\ S_{7\to6\to3}^2 & \frac{5}{12}\\ S_{7\to6\to4}^2 & 0\\ S_{7\to6\to5}^2 & \frac{1}{2}\\ S_{7\to6\to6}^2 & \backslash\\ S_{7\to7\to7}^2 & \backslash\end{Bmatrix},$$

$$\tilde{\boldsymbol{S}}_7^3=\tilde{\boldsymbol{L}}\times$$

$$\begin{Bmatrix}S_{7\to6\to1}^2 & 0\\ S_{7\to6\to2}^2 & 0\\ S_{7\to6\to3}^2 & \frac{5}{12}\\ S_{7\to6\to4}^2 & 0\\ S_{7\to6\to5}^2 & \frac{1}{2}\\ S_{7\to6\to6}^2 & \backslash\\ S_{7\to7\to7}^2 & \backslash\end{Bmatrix}=\begin{Bmatrix}S_{(7\to6\to3\to1),(7\to6\to5\to1)}^3 & \frac{35}{144}+0=\frac{35}{144}\\ S_{(7\to6\to3\to2)(7\to6\to5\to2)}^3 & \frac{25}{96}+0=\frac{25}{96}\\ S_{(7\to6\to3\to3),(7\to6\to5\to3)}^3 & \backslash+0=\backslash\\ S_{(7\to6\to3\to4),(7\to6\to5\to4)}^3 & \frac{5}{18}+\frac{6}{18}=\frac{11}{18}\\ S_{(7\to6\to3\to5),(7\to6\to5\to5)}^3 & 0+\backslash=\backslash\\ S_{(7\to6\to3\to6),(7\to6\to5\to6)}^3 & \backslash+\backslash=\backslash\\ S_{(7\to6\to3\to7),(7\to6\to5\to7)}^3 & \backslash+\backslash=\backslash\end{Bmatrix}\tag{8}$$

式(8)中类似 $S_{(7\to6\to3\to1)(7\to6\to5\to1)}^3$ 的标记本文称之为路径标记 Pm(Path mark),代表 3 跳后,7 号节点到达 1 号节点的路径有两条,分别为 7→6→3→1(成功概率为 35/144)和 7→6→5→1(成功概率为 0,因为 5 和 1 不相连). Pm 对应的路径若满足如下条件之一的时候,则该路径对应的成功转发率标记为无意义——"\":

(1)路径存在闭环,则该链路记为"\"(例如式(7)中的 $S_{7\to7}^1$ 和式(8)中的 $S_{7\to6\to6}^2$).

(2)若干个"\"的和等于"\".

(3)"\"+X(非 0 且非"\")=X.

(4)"\"+0="\".

"\"标记也能有效预防算法复杂度随着 REQ 生命周期 L 值的增加而无限增加的情况. 通常 L 值不应该被设置为较大的值,一方面会增加计算复杂度,另一方面也会造成多跳路径的不可靠性增加. 在上例中,可以看出当 f=0、1、2 时,节点 7 与 sink 节点之间的路径转发成功率均为 0,这是因为在 2 跳之内,节点 7 与 sink 节点无法建立连接. 当 f=3 时,Pm 为 $S_{(7\to6\to3\to4)(7\to6\to5\to4)}^3$,其中路径 7→6→3→4 的转发成功率为 5/18,7→6→5→4 的转发成功率为 6/18. 简单比较可以看出路径 7→6→5→4 的转发成功率较优,则 sink 节点将会沿着 4→5→6→7 发送 Feedback 数据包通知 7 号节点选择 7→6→5→4 路径进行数据发送,至此,整个路由选择计算流程就结束了.

综上所述,可以形成一种路由建立算法 MCR.

算法 MCR.

Procedure MCR(*Source*,*Destination*)

{

1. *get_ToN*$_i$(); //初始化网络,获取网络中各个节点 i 的邻居信息表
2. *Init_REQ*(L,f,$f_{\max}$); //初始化 REQ 数据包,设定其最大跳数 L,当 L 为 0 时,REQ 将不会继续被转发. L 值设置的足够大的时候,将会得到全局最优路径,但是会消耗较大的带宽和计算量. 初始化随机游走步数 f 与 f 最大可能值 $f_{\max}$
3. *Form_matrix*_$\tilde{\boldsymbol{L}}$(*Destination*); //在目的节点处基于 REQ 形成式(6)中的矩阵 $\tilde{\boldsymbol{L}}$
4. FOR f:=0 to f {$\tilde{\boldsymbol{S}}_s^{f+1}=\tilde{\boldsymbol{L}}\times\tilde{\boldsymbol{S}}_s^f$} //执行上文中的矩阵叠加计算,形成最终的 $\tilde{\boldsymbol{S}}_s^{f+1}$
5. IF ($\tilde{\boldsymbol{S}}_s^{f+1}$[*Destination*]>0)
   {*select_path_with_max_Pm*();} //如果最终的 $\tilde{\boldsymbol{S}}_s^{f+1}$ 矩阵中对应于 *Destination ID* 号的项大于 0,则选择具有最大路径标记 Pm 的那一条路径作路由
   ELSE IF ($f<f_{\max}$) //如果 f 小于最大设定值
   {f++;
   *back_to_step*_4 } //进入 step4,重新计算
   ELSE
   { END; //源节点和目的节点无法连接上}
6. *Send _Feedback* (*Destination*, *Source*);//目的节点通过Feedback 数据包告知源节点最优的通信路径
7. END

}

3.3 MCR 算法的科学意义与理论分析

水下无线传感器网络中信号传输延迟较高、节点移动性较强,类似 AODV 的全局最短路由算法并不具有较高的性能. AODV 需要通过大量端到端的探测包来发现路由、维护路由,但是实际上,在高信号延迟、高移动性的环境中这类算法会一直疲于路由发现与维护,减少有效数据的传输量,在后文的实验环节中,该现象非常凸显(每一次路由发现或者维护,都需要进行全路径探测,有效数据传输所占总时间比例很少). MCR 算法的本质可以认为是一种概率路由,避免了 UWSN 环境导致有效数据一直无法传输的现象,从而增加了网络吞吐量.

MCR 能在路径断开后会重新建立路由,但是与类似 AODV 算法不同之处在于,MCR 发起一次 REQ 就可以完成路径的建立,而 AODV 需要多次 REQ、Feedback 闭环成功才能完成路径的建立(为了确保路径的可靠). 从后文中的实验数据来看,大

量的探测包存在于网络中反而会增加冲突概率，降低可靠性，MCR 在可靠性上具有较高的提升.

在步骤 4 与 5 中，有两种方法可以建立路由，第 1 种是 sink 节点收到一个 REQ 立即建立一条路径；第 2 种是 sink 节点收集到所有 REQ 后通过随机游走算法建立路径. 这两种方法有一个很大的区别，例如在图 4 中，5 号节点给 4 号节点发送 REQ 失败了，但是 5 号节点将自身的邻居信息表通过 3 号节点传送给了 4 号节点，则最终的$\tilde{\boldsymbol{L}}$矩阵中，仍然保留了 5 号节点的信息，在路由计算结束后，若 7→6→5→4 对应的转发成功率较高，则该路径不会因为一次偶然的 REQ 丢失而被忽略，这就是 MCR 路由算法的一个优势，增加了路由的鲁棒性.

4　实验分析

为了对 MCR 协议性能进行评估，本节进行了一系列仿真实验，实验平台为 NS-2 仿真平台. 为了更加直观地描述 MCR 协议性能，本文引入了典型的源驱动式 AODV 协议、表驱动式 DSDV 协议[19]以及 UWSN 中较为成熟的位置信息驱动 VBF 算法与其进行性能对比.

4.1　实验环境

在基于 NS-2 的仿真实验中[20-21]，20 个节点被均匀放置在 600 m×450 m 的矩形区域内，拓扑形状如图 6 所示，任意相邻两节点间距离均为 150 m，仿真相关参数详细设置如表 2 所示. 为了使得仿真环境与 UWSN 的环境接近，进行了一些如下设置：(1) 带宽被设置为低速率，带宽值为 256 kbps；(2) 将数据包缓存数量最大值设置成较小值，增加冲突类丢包概率，取值为 50；(3) Mac 层两节点间的丢包率与距离关系如表 2 所示，为了模拟 UWSN 链路的不可靠特性，将节点距离设置为 150 m；(4) 关闭 RTS/CTS 功能，增加隐藏终端发生可能性.

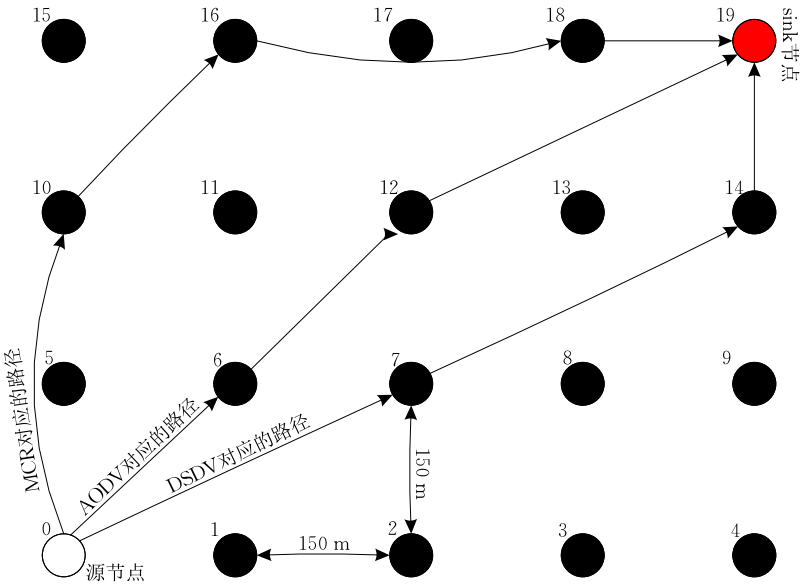


图 6　节点布局图

表 2　相关仿真参数

参数项	取值
信道类型	WirelessChannel
衰减模型	Shadowing
网络接口类型	WirelessPhy
Mac 层协议	802_11
Interface queue type (ifq)	PriQueue
Link layer type	LL
天线类型	OmniAntenna
Max packet in ifq	10
节点数量(nn)	20
路由协议	AODV/DSDV/MCR

表 3　距离与丢包率对照表

距离/m	丢包率
560	0.43
510	0.37
460	0.29
360	0.13
310	0.11
210	0.03
160	0.006

图 6 中，节点 0 代表数据源节点，采用的 Agent 为 TCP，节点 19 代表 sink 节点，采用的 Agent 为

TCP/sink. 仿真时间为 1500 s，节点 0 向节点 19 发送恒定速率 cbr 数据流，cbr 数据包长度为 64 byte，无发送间隔，发送速率为 16 kb. 图 6 中的虚线箭头，描述了 MCR、DSDV、AODV 在仿真过程中的某一时刻建立的路由.

4.2 性能评价参数

本文主要从端到端数据丢失率 *drop rate*、端到端数据延迟、端到端数据延迟抖动、网络吞吐量等 4 个参数对路由协议性能进行评估. 具体参数定义如下：

(1) 端到端数据丢失率. 在仿真过程中的各个时间点上记录节点 0 发出的数据包总数 T_0，记录节点 19 接收到的数据包总数 R_0，则有 $drop\ rate = T_0/R_0$.

(2) 端到端数据延迟. 设数据包 P_i（i 为数据包序列号）为节点 0 成功发送至节点 19 的数据包，记录每一个 P_i 在节点 0 产生的时刻 t_start，记录节点 19 收到来自节点 0 的 P_i 数据包对应时刻 t_end，则有序列号为 i 的数据延迟为 $t\_end\text{-}t\_start$.

(3) 端到端数据延迟抖动[22]. 设 $recvTime(j)$ 代表节点 19 收到序列号为 j 的数据包时刻，$recvTime(i)$ 为节点 19 收到序列号为 i 的数据包时刻，i 和 j 是节点 19 收到的相邻数据包序号，则有数据延迟计算公式如下：

$$Jitter = \frac{recvTime(j) - recvTime(i)}{j - i} \tag{9}$$

(4) 网络吞吐量. 记录节点 19 在经历 t 秒时，收到的有效数据总比特数目 $sum(t)$，则有吞吐量为 $sum(t)/t$.

4.3 实验结果分析

(1) 端到端数据丢失率分析

在仿真实验中，REQ 数据包的寿命均被设置为 5 跳，图 7 给出了 3 种路由算法对应的丢包率. 图 7 中横坐标表示仿真记录的时间点，横坐标空缺表示在仿真过程中该时刻点没有对应任何行为.

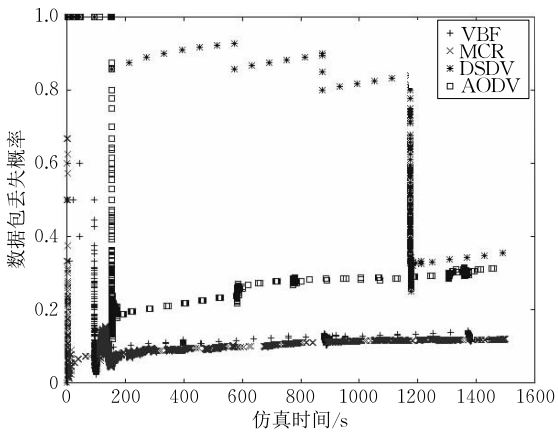


图 7 丢包率仿真结果

从图 7 中可以看出 AODV 和 DSDV 相似的丢包率没有明显的规律，这是因为 AODV 和 DSDV 均考虑最短路径，因此在丢包率上没有必然的趋势，这和路径选择的偶然性有关. MCR 的丢包率一直较低，这是因为 MCR 算法基于节点缓存和度值考虑了路径转发成功率，会选择干扰源较少的路径，在对 AODV 和 DSDV 的 trace 文件分析中发现，数据丢失主要有两个原因：①数据冲突，占 39.3%；②路径损坏，占 51.7%. 而在 MCR 的 trace 中，数据冲突占 18.4%，路径损坏占 45.2%. 可以看出 MCR 丢包率的降低主要受益于冲突类丢包的减少. VBF 算法的核心思想是在源节点通信范围内寻找与目的节点位置最为接近的节点作为下一跳，并以此类推直到数据被交付至目的节点. 在图 6 规则布局网络中 VBF 算法与 MCR 算法相似，具有较低的丢包率，并且算法简单实用，但是 VBF 算法属于局部最优算法，在不规则的网络拓扑中很容易出现数据在中继过程进入“死胡同”的现象，无法找到下一跳，此时数据将会丢失.

在图 7 中还可以看出，在 0～150 s 之间，AODV 和 DSDV 的丢包率均为 1，数据在 150 s 后才开始正式传输. 通过 trace 文件分析发现，在前 150 s 内 AODV 和 DSDV 一直处于频繁路径建立阶段，经历了很久的 REQ 广播，建立了一条链路，结果很快就发现该链路被损坏，又需要重新建立. VBF 算法在一开始需要通过广播收集各个节点的位置信息，每个节点需要知道自己通信范围内各个邻居节点的位置信息，这个过程需要消耗一部分时间，从图 7 中可以看出，大概消耗了 100 s 左右. 而 MCR 算法并不完全依赖 REQ 和 Feedback 构成的环路，通过随机游走的方法，即使有一些 REQ 或 Feedback 在传输过程中丢失了，也不会造成路径丢失(见本文 3.3 节)，因此，更能快速全面地完成有效路由的建立.

(2) 端到端数据延迟分析

图 8 中横坐标是某个数据包产生的时刻，纵坐标代表该时刻产生的数据端到端延迟大小. 横坐标的空缺代表着在仿真实验中该时刻点上节点 0 没有能够成功将数据传输给节点 19. 通过图 8 可以看出在成功发送的数据样本中，MCR 的端到端延迟较高，而 AODV、DSDV、VBF 的端到端延迟较为相似，这主要是因为 AODV、DSDV、VBF 均偏好选择

跳数最少的路径，在数据传输过程中跳数是影响端到端延迟的一个重要因素，因此 AODV、DSDV、VBF 对应的延迟较低；而 MCR 没有考虑跳数问题，在路径选择过程中可能会选择一条跳数较大的路径，从而增加了延迟.

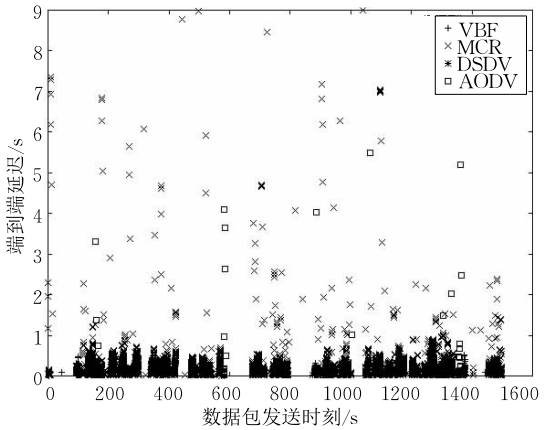


图 8　端到端延迟仿真结果

图 8 还显示了 AODV、DSDV、VBF 对应的横坐标相比于 MCR 空缺很大，这表明在仿真过程中 AODV、DSDV、VBF 的实际成功数据交付只是出现在几个短暂的时间区间上，空缺的地方代表网络一直在建立路径或者是传输失败. 产生这个现象的主要原因是 AODV、DSDV、VBF 偏好追求最短路径而没有考虑路径传输的可靠性和干扰源，从而导致不论在路径建立阶段还是数据传输阶段频繁出现数据包丢失. 此外，图 8 与图 7 是可以对应上的，例如在图 7 和图 8 的 1200 s 附近，图 7 的 DSDV 算法丢包率之所以能够急剧下降，是因为图 8 中 DSDV 算法在 1200 s 附近成功交付了大量数据.

(3) 端到端数据延迟抖动分析

节点 19 接收到数据包的序号排序可能与节点 0 发送的数据包序号排序不同，例如节点 0 发送序号为{1,2,3}，而节点 19 接收到的序号可能为{3,2,1}. 根据端到端延迟抖动的定义可知，jitter 值可能出现正值也可能出现负值. jitter 值可以反映网络的稳定性，若 jitter 值较大，则可以认为链路在某个地方出现拥堵，网络不稳定. 图 9 中的 jitter 数据有很多看起来很接近于零，但是实际上并不为 0，因为 NS-2 仿真实验中，数据包的时间间隔非常短，基本是在毫秒级，因此计算出的 jitter 值看起来才会很小.

图 9 的横坐标是成功交付数据包的序号，在 NS-2 仿真中，AODV 与 MCR 协议的数据包序号与有效数据绑定，即路由层的 ARQ、relpy 等数据不

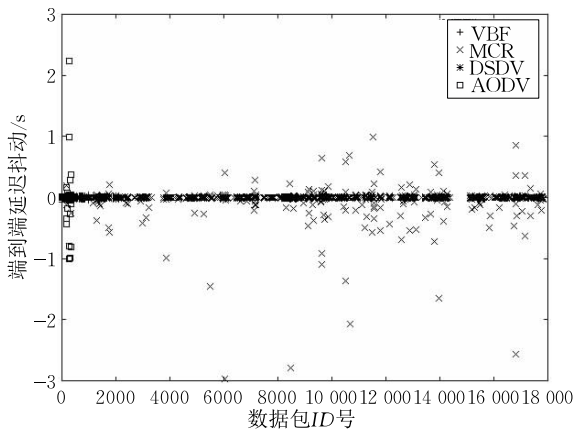


图 9　端到端延迟抖动仿真结果

会导致序号增加，而 DSDV、VBF 与 AODV、MCR 不同，路由层的 message 会使序号增加，因此图 9 中的 DSDV 数据包序号要比 AODV 序号大很多，实际上两者传输的有效数据量很接近. 纵轴是成功接收到的某序号数据包对应的延迟抖动值. 由于 MCR 选择的路径可靠性较高，路由建立和维护时间开销低，因此大部分仿真时间均被用来进行有效数据传输，所以有效传输的数据包数量较大；而 AODV 与 DSDV 需要消耗大量时间去建立、维护路由，而且频繁发生数据丢失，导致有效传输的数据量很小；VBF 属于源驱动算法，每个节点没有维护路由表，每当一个节点接收到数据包后，总是会计算其邻居节点与目的节点的距离，并选择最为接近的作为下一跳，这导致在数据通信过程中消耗一些时间，导致有效数据传输的时间变少.

从图 9 中还可以看出在成功传输的数据样本中，AODV 和 MCR 对应的延迟抖动较为相似，而 DSDV、VBF 对应的延迟抖动较小. 这是因为 DSDV 属于表驱动路由协议，在一次路径确定和下一次路径更新之间，不再会有路由搜索行为，数据将会严格按照确定的路由进行传输，即使多次发送失败也不会随意改变路径；VBF 算法在整个通信过程中几乎没有多余的探测包消耗，在规则拓扑网络中出现数据交付失败的概率也很低，避免了数据重传的现象，因此延迟抖动较小，但在不规则网络中，很可能会因为数据回传造成大量数据拥塞，导致较高延迟抖动；而 AODV 和 MCR 属于源驱动路由协议，一旦发现路径不可用则立刻会发起 REQ 广播进行路径寻找、建立，从而为网络注入了大量探测包，使得网络的正常数据传输变得不稳定.

(4) 网络吞吐量分析

吞吐量是衡量一个网络性能的重要参数，吞吐

量的大小、吞吐量的稳定性，对网络应用具有重要的指导价值.

图 10 的横坐标是仿真过程中记录的时间点，纵坐标是各个时间点上统计的节点 19 与节点 0 之间的端到端网络吞吐量，单位是 kbps. 从图 10 中可以看出 AODV 协议在 200 s 附近时有一个吞吐量峰值，这和图 7 相对应，AODV 协议在 200 s 附近的一段时间内，成功交付率很高，此时的吞吐量也随之上涨，随着仿真时间的推移，AODV 的交付率逐渐降低，并且网络上大量时间消耗在路径探索和维护上，有效数据传输量越来越少，从而导致 AODV 网络吞吐量急剧下降. DSDV 吞吐量一直处于极低的状态，虽然在 1200 s 时的交付率较高，但是由于仿真时间基数已经很大，因此最终吞吐量结果仍然很低. VBF 吞吐量一直较低，这主要是因为 VBF 每次传输数据时总是会消耗一部分时间来计算各个节点与目的节点的距离.

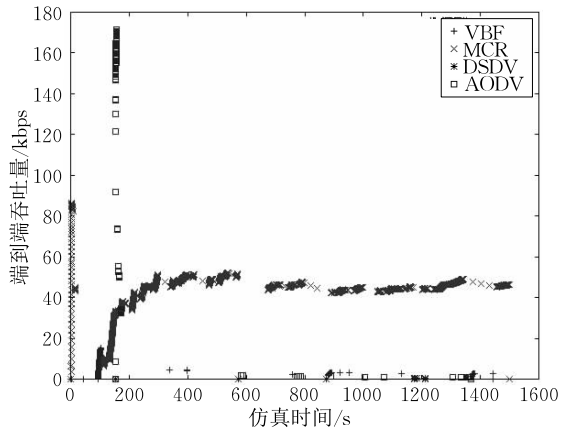


图 10 端到端吞吐量仿真结果

相比于 AODV、DSDV、VBF，MCR 具有较稳定的吞吐量，这主要取决于两方面：(1) MCR 协议降低了冲突类数据丢包概率，提高了端到端数据包交付率；(2) MCR 在整个仿真时间内，在路径建立、维护方面只消耗了较少的一部分时间(不需要等待 REQ 和 Feedback 构成环路就可以计算出有效路径)，大部分的时间均被用来传输有效数据. 因此 MCR 在吞吐量性能方面优越于 AODV、DSDV、VBF.

5 结 论

本文针对水下无线传感器网络的物理特性，设计了一种最小化冲突概率的路由算法 MCR，该算法综合考虑网络中节点的度值和通信负载，融合随机游走思想，为数据的传输选择了一条冲突概率较低的路径. 基于 NS-2 仿真平台分别对 AODV、DSDV、VBF 和 MCR 路由算法进行了实验评估，并得出以下几点结论：(1) 在 UWSN 中，冲突类丢包在丢包总数内占据很大比例，MCR 算法能够有效避免冲突类丢包，相比于其他两种具有更低的丢包率；(2) 由于 MCR 算法没有选择跳数最少的路径，因此在端到端延迟性能上不如 DSDV、AODV 与 VBF；(3) AODV、MCR 相比于 VBF、DSDV 协议会不定期的进行路径探索和维护，因此在延迟抖动性能上 AODV、MCR 不如 DSDV、规则拓扑下的 VBF；(4) 相比于 AODV、VBF、DSDV，MCR 具有更高和更稳定的网络吞吐量，这主要是因为 MCR 的丢包率较低，并且大部分仿真时间 MCR 均处于有效数据传输阶段，而 AODV 与 DSDV 大部分仿真时间花费在路径探索和维护上，并且丢包现象严重，VBF 没有维护路由表，导致每次数据转发时均需要消耗一些时间来计算最优下一跳；(5) MCR 采用随机游走思想进行路由计算，相比于 AODV 和 DSDV，MCR 协议无需等待 REQ 和 Feedback 构成回路(在链路较差的环境中，很难构成回路)就可以完成路径建立，路由建立速度更快；(6) MCR 算法并不会因一次 REQ 或者 Feedback 的丢失而导致一条路径被忽略，具有更鲁棒的路径发现性能.

UWSN 与传统 WSN 不同，由于通信环境的改变，冲突类丢包现象剧增，MCR 针对该现象进行了路由策略和路由搜索算法的设计，在丢包率、吞吐量等方面获得了收益，但是在延迟和延迟抖动方面仍存在不足. 实验仿真环境主要是面向 UWSN 构建的，在其他环境下性能对比结果如何，是本文下一步的工作. 此外，UWSN 中，除了冲突类丢包，还有链路质量不稳定导致的丢包，如何全面地考虑各种丢包因素，进一步提高 MCR 协议的性能也是本文的下一步主要工作.

致 谢 感谢辽宁省面向先进装备制造业的嵌入式技术重点实验室对本论文的实验平台支持. 感谢辽宁省教育厅对本论文的项目经费支持!

参 考 文 献

[1] Partan J, Kurose J, Levine B N. A survey of practical issues in underwater networks//Proceedings of the 1st ACM International Workshop on Underwater Networks (WUWNet'06). Los Angeles, USA, 2006: 17-24

[2] Tran K T M, Oh S H. UWSNs: A round-based clustering scheme for data redundancy resolve. International Journal of Distributed Sensor Networks, 2014, 2014(383912): 19-26

[3] Cui J-H, Kong J, Gerla M, Zhou S. The challenges of building scalable mobile underwater wireless sensor networks for aquatic applications. IEEE Network, 2006, 20(3): 12-18

[4] Guo Zhong-Wen, Luo Han-Jiang. Current progress and research issues in underwater sensor networks. Journal of Computer Research and Development, 2010, 47(3): 377-389 (in Chinese)
(郭忠文，罗汉江. 水下无线传感器网络的研究进展. 计算机研究与进展，2010，47(3)：377-389)

[5] Zhou Z, Cui J H, Bagtzoglou C A. Scalable localization with mobility prediction for underwater sensor networks// Proceedings of the IEEE INFOCOM 2008 Mini-Conference. Phoenix, USA, 2008: 2198-2206

[6] Partan J, Kurose J, Levine B N. A survey of practical issues in underwater networks//Proceedings of the Special Section on ACM WUWNet 2006. Los Angeles, USA, 2007: 23-33

[7] Aarti, Tomar S K. A reliable and cost effective next hop selection for aggregative path generation in UWSN// Proceedings of the International Conference on Wireless and Optical Communications Networks (WOCN' 2013). Bhopal, India, 2013: 231-236

[8] Su R Y, Venkatesan R, Li C. A new node coordination scheme for data gathering in underwater acoustic sensor networks using autonomous underwater vehicle//Proceedings of the 2013 IEEE Wireless Communications and Networking Conference (WCNC). Shanghai, China, 2013: 4370-4374

[9] Ayaz M, Baig I, Abdullah A, Faye I. A survey on routing techniques in underwater wireless sensor networks. Journal of Network and Computer Applications, 2011, 34(6): 1908-1927

[10] Ayaz M, Abdullah A. Hop-by-hop dynamic addressing based (H2-DAB) routing protocol for underwater wireless sensor networks//Proceedings of the International Conference on Information and Multimedia Technology (ICIMT'09). Jeju Island, Korea, 2009: 436-441

[11] Ayaz M, Abdullah A, Faye I. Hop-by-hop reliable data deliveries for underwater wireless sensor networks// Proceedings of the 5th International Conference on Broadband Wireless Computing, Communication and Applications (BWCCA'10). Fukuoka, Japan, 2010: 363-368

[12] Xie P, Cui J-H, Lao L. VBF: Vector-based forwarding protocol for underwater sensor networks//Proceedings of the IFIP (Networking'06). Coimbra, Portugal, 2006: 171-189

[13] Bai Rendong, Singhal M. DOA: DSR over AODV routing for mobile ad hoc networks. IEEE Transactions on Mobile Computing, 2006, 5(10): 1403-1416

[14] Zorzi M, Casari P, Baldo N, Harris III A F. Energy-efficient routing schemes for underwater acoustic networks. IEEE Journal on Selected Areas in Communications, 2008, 26(9): 1754-1766

[15] Syed A A, Heidemann J. Time synchronization for high latency acoustic networks//Proceedings of the INFOCOM 2006, 25th IEEE International Conference on Computer Communications. Barcelona, Spain, 2006: 892-903

[16] Domingo M C. A distributed energy-aware routing protocol for underwater wireless sensor networks. Wireless Personal Communications, 2011, 57(4): 607-627

[17] Wahid A, Lee S, Kim D. An energy-efficient routing protocol for UWSNs using physical distance and residual energy// Proceedings of the Oceanic Engineering Society and the Marine Technology Society (OCEANS'11). Santander, Spain, 2011: 6-11

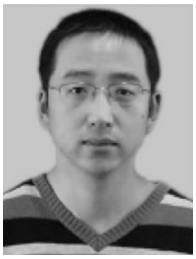
[18] Gopi S, Govindan K, Chander D, et al. E-PULRP: Energy optimized path unaware layered routing protocol for underwater sensor networks. IEEE Transactions on Wireless Communications, 2010, 9(11): 3391-3401

[19] Liu Ting, Liu Kai. Improvements on DSDV in mobile ad hoc networks//Proceedings of the 3rd International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM 2007). Shanghai, China, 2007: 1637-1640

[20] Xie P, Zhou Z, Peng Z et al. Aqua-sim: An NS-2 based simulator for under water sensor networks//Proceedings of the MTS/IEEE Marine Technology for Our Future: Global and Local Challenges (OCEANS'09). Biloxi Miss, USA, 2009: 119-127

[21] Wang Y, Jin Z G, Su Y S. Simulator-to-emulator: Analysis and design of experiment platform for underwater sensor networks. Applied Mechanics and Materials, 2014, 473(1): 219-225

[22] Tao Liqiang, Yu Fengqi. A TDMA jitter minimization algorithm for real-time applications in wireless sensor networks// Proceedings of the 6th IEEE Symposium on Computers and Communications (ISCC). Kerkyra, Greece, 2011: 117-121



ZHU Jian, born in 1981, Ph. D., lecturer. His research interests include under-water wireless sensor networks and wireless sensor networks.

LIU Jun, born in 1982, Ph. D., lecturer. Her research interest is grid computing.

ZHAO Hai, born in 1959, Ph. D., professor, Ph. D. supervisor. His research interests include wireless sensor networks and complex network.

XU Ye, born in 1976, Ph. D., professor, Ph. D. supervisor. His research interests include wireless sensor networks and complex network.

Background

This paper mainly focuses on the communication problems in UWSN(Underwater Sensor Networks). There are many differences between UWSN and WSN, the most defining one is that, the propagation speed of acoustics is lower than electromagnetic wave. This difference makes the propagation latency of acoustics can't be ignored, and, many routing algorithms which aim at WSN can't be used in UWSN directly.

Focusing on the problem above, many researches are carried out, most of them use node's position as the routing metric, such as VBF. The routing algorithms based on node's position have the same problems that, they can't work in irregular networks, and they will cost a lot on the localization of nodes. This paper avoids the problems above, the MCR routing algorithm can be used in any networks, and uses the DBM metric for path selection.

This paper is supported by the general project technology projects in department of education of Liaoning province. The research in this paper can speed up the development of UWSN, and has contributions to the explorations of underwater environment for human.