ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.21, No.3, March 2010, pp.473−489
doi: 10.3724/SP.J.1001.2010.03413

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

传感器网络中基于蚁群算法的实时查询处理*

余建平[1,2], 林亚平[2,3+]

[1](湖南师范大学 数学与计算机科学学院,湖南 长沙 410081)

[2](湖南大学 计算机与通信学院,湖南 长沙 410082)

[3](湖南大学 软件学院,湖南 长沙 410082)

Real-Time Query Processing for Sensor Networks Based on Ant Algorithm

YU Jian-Ping[1,2], LIN Ya-Ping[2,3+]

[1](College of Mathematics and Computer Science, Hu'nan Normal University, Changsha 410081, China)

[2](College of Computer and Communication, Hu'nan University, Changsha 410082, China)

[3](College of Software, Hu'nan University, Changsha 410082, China)

+ Corresponding author: E-mail: yplin@hnu.cn

Yu JP, Lin YP. Real-Time query processing for sensor networks based on ant algorithm. *Journal of Software*, 2010,21(3):473−489. http://www.jos.org.cn/1000-9825/3413.htm

Abstract: Wireless sensor networks are often deployed in diverse application specific contexts, which can be treated essentially as distributed databases. The event-involved responses can be obtained by issuing queries to this kind of database. The applications with real-time requirement have tight constraints on query delay. However, the existing query algorithms cannot meet the demands of the real-time query applications. With regard to the special applications, a real-time query processing algorithm based on ant colony optimization is proposed. In this algorithm, priority-based multiple-rings storage scheme and ant-based distributed search mechanism are adopted to improve the integrated performance of energy-efficiency, delay and query reception rate. It takes advantage of the self-organization and positive feedback characteristics of ant colony optimization algorithm. The proposed algorithm provide a new idea for distributed dynamic parallel real-time query applications, demanding merely local information to obtain named events efficiently and determine the number and allocation of event replicas adaptively. Theoretical analysis and experiments prove that compared with other existing query algorithms, this algorithm cannot only improve the performance of energy-efficiency and reception, it can also shorten the query delay considerably.

Key words: wireless sensor network; real-time query processing; data replication; ant colony algorithm

摘 要: 无线传感器网络因不同应用而被广泛部署于各种场合,通常被视为分布式数据库.可以通过向该类数据库发布查询请求来获取事件相关的响应信息.一些具有实时需求的应用对查询时延要求较高,而目前存在的查询算法通常不能很好地满足实时查询应用的需求.针对此类特定应用,提出了基于蚁群优化的实时查询处理

* Supported by the National Natural Science Foundation of China under Grant No.60903168 (国家自然科学基金); the Science and Technology Plan of Hu'nan Province of China under Grant No.2009FJ3083 (湖南省科技计划项目)

Received 2007-09-22; Revised 2008-03-11; Accepted 2008-07-02

算法,该算法采用基于事件重要性的分环存储策略和基于蚁群算法的分布式搜索机制,充分利用蚁群优化算法的自组织和正反馈等特征,综合提高查询处理算法的节能性、实时性及查询请求接受率,为分布式动态并行实时查询应用提供新的思路.执行过程仅需局部环境信息即可有效获取指定事件信息并动态确定事件副本的数目及其位置.理论分析和实验表明,此算法既能在一定程度上提高节能性和查询成功率,又能显著缩短查询时延,与现有的算法相比,具有明显的优越性.

关键词: 无线传感器网络;实时查询处理;数据复制;蚁群算法

中图法分类号: TP393 **文献标识码**: A

近几年,无线传感器网络技术得到了飞速发展.该类网络通常用来完成环境监控、军事探测等特定任务.由于节点缺乏保护且能量通常不可追加,怎样充分利用有限的网络能量等资源是算法设计的关键[1].

大多数情况下,传感器网络是以数据中心存储及路由方式进行工作,与该类网络相关的任何应用都必须考虑其数据管理及处理技术,尤其是数据查询处理机制[2].对于节点规模巨大的传感器网络,传统的集中式查询处理方式通常不可行.从数据查询角度的观点,可以将传感器网络视为一个分布式数据库[3,4].从该类数据库获取信息的一种简单机制是通过在网内洪泛指定数据的查询请求,来获取相关节点的响应.由于该类网络节点的能量、计算能力和存储容量都极为有限,为了在获取有效数据的同时最大限度地延长网络生命期,提出一种节能的分布式内网查询处理算法至关重要[5].文献[6]利用传感节点固有的空间、语义特征,提出了一种支持传感器网络查询处理的基础结构.文献[7]提出了 ACQUIRE 查询处理机制,该机制把查询视为一个活跃实体,通过实体在网络内搜索相关的待查信息.文献还根据不同的标准,对传感器网络查询进行了简单分类.ACQUIRE 机制为针对复制数据的合成、单一查询提供了一定的优化,但未考虑事件数据的聚合.文献[8,9]提出了几种聚合查询处理算法.然而,几乎没有哪一种查询处理技术能够有效解决所有类型的查询[7].对于某些实时应用,查询响应时间往往比能耗更为重要,直接影响到查询结果的价值,尤其是针对多优先级的事件查询.如何缩短查询时延以满足不同优先级事件的时延需求是一个具有挑战性的难题.本文着重考虑针对该类实时应用的查询处理机制.

对于特定的传感器网络应用,所关注的事件类型有限.当某区域有相关事件产生时,产生地四周的传感节点收集事件数据并传至本地聚合节点.由于查询节点未知,聚合节点在收到查询请求前如何存储事件数据是设计查询处理算法必须考虑的关键因素.通常情况下,聚合节点完成事件聚合及编码后,将事件编码存储于网内某存储节点以供查询.存储节点的选取可预先确定,亦可随机选择.前者以基于哈希表的数据中心存储技术 GHT[10] 和 DIM[11]为代表.由于查询目的节点已知,故查询效率较高.然而,由于网络环境具有动态性,在全网范围内维护共享的预先确定的位置信息异常困难,故某些情形下利用随机存储更具有优势.但查询节点必须借助于盲目搜索方能找到感兴趣的事件信息.为了进一步提高查询处理的实时性,可将事件编码复制多份存储于多个分散的存储点.文献[12]提出了基于扩展环的查询技术,该技术通过逐步扩大 TTL 值的受限洪泛方式来搜寻被查事件的最近副本.文献[13]对如何确定基于扩展环查询的最优复制标准进行了分析.但在事件查询率等全局信息未知的情况下性能并不理想.上述查询处理算法均着眼于节能性,未对算法的实时性作重点关注.文献[14]提出了一种基于环的负载平衡数据存储方法及其对应的查询处理算法,该算法在平衡能耗方面具有显著的优越性.但由于它采用的是一种全局环存储机制,需要环内存储节点定时对事件数据进行环间迁移,需消耗较多的能量,且 Sink 节点的固定布置不利于算法的扩展,故在节能性和实时性方面均不理想.针对实时应用需要提出新的查询机制.文献[15]提出了一种动态分布式多蚁群任播路由算法 MACR.该算法不仅针对因特网服务进行快速有效地搜寻,还为针对传感器网络的实时查询提供了一定的启示.

蚂蚁算法[16]采用一组称作“蚂蚁”的智能代理来执行本地搜索,利用历史搜索信息来指导当前搜索行为.该算法在全局信息未知的情况下可快速找到较优路径,具有群集智能的特征.本文基于群集智能的思想,提出了一种带正反馈的分布式多蚁群实时查询处理算法 ARTQP(ant-based real-time query processing).该算法首先由聚合节点根据事件优先级对事件编码进行复制并分环存储,然后通过多个查询节点连续发送带有特定事件编码等标志的多只蚂蚁来开拓到被查事件最近副本的较优路径以削减查询过程中的复制和搜索总能耗,缩短查询

时延,提高查询处理效率.在查询过程中,可根据需要动态地确定事件副本数量,具有一定的智能性和灵活度.

本文第 1 节简单介绍群集智能的特征和原始蚂蚁算法.第 2 节提出基于多蚁群算法的实时查询处理算法 ARTQP.第 3 节对提出的实时查询处理算法 ARTQP 进行相应的理论分析.第 4 节通过模拟实验对 ARTQP 算法进行验证.第 5 节对全文的工作进行总结.

1 群集智能及原始蚂蚁算法

1.1 群集智能

群集智能[17]主要通过对社会性昆虫行为的模拟来产生解决问题的新方案.该类昆虫可视为具有微弱智能的一组代理,但它们可通过相互协作呈现整体的智能性.群集智能的主要特征如下:

A) 群体中相互协作的个体是完全分布式的,这能够适应规模较大的网络,尤其是节点数目巨大的无线传感器网络;B) 在没有集中控制的情况下,系统仍可保持鲁棒性,故少数个体出错或死亡不会对整体的性能造成很大影响;C) 系统具有很强的扩展性,因为个体代理仅通过环境信息与其他代理交互;D) 由于个体行为简单,系统执行相对容易,耗时相对较少,从而使系统呈现简单性.

1.2 原始蚂蚁算法

原始蚂蚁算法是基于群集智能的思想,模拟自然界中真实蚁群的觅食行为而形成的一种模拟进化算法.研究表明,蚂蚁具有找到蚁穴至食物之间较优路径的能力.作为一种社会性昆虫,当遇到个体不能完成的任务时,群体可以通过相互协作来完成,协作过程表现出一定的智能性.

Dorigo 等学者较早提出用蚂蚁算法解决 TSP 问题,通过模拟蚂蚁的搜索行为及信息素通信方式来实现.多个蚂蚁在相互连通的城市之间根据信息素浓度进行遍历.遍历过程中,越短的路径集聚的信息素浓度越高.蚂蚁正是根据这种正反馈机制寻找到较优的遍历路径.蚂蚁通过信息素浓度寻找优化路径的过程如图 1 所示.

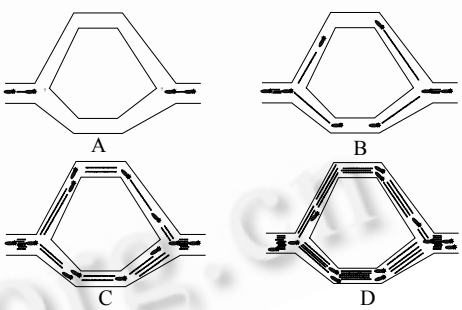

Fig.1 Illustration of how real ants find optimum path

图 1 真实蚂蚁寻找优化路径图解

图 1 中,A 表示蚂蚁到达某一交叉路口,需选择前进路线;B 表示蚂蚁随机选择上下两条路径;C 表示选择下方较短路径的蚂蚁返回较快,返回过程释放更多信息素;D 表示下方较短路径上聚集的信息素远多于上方路径,导致更多蚂蚁选择下方较短路径(图中虚线表示信息素浓度).图 1 表明蚂蚁具有在选定路径释放信息素指导后继蚂蚁选择下一跳的能力.

蚁群算法不仅可用于解决 TSP 问题,还可用于解决组播、任播等诸多问题[15,18,19].一些新的蚂蚁算法[20,21]已经陆续提出.然而,据我们所知,用蚁群算法优化求解无线传感器网络实时查询处理问题仍鲜有文献涉及.

2 传感器网络中基于蚁群算法的实时查询处理

2.1 传感器网络模型及问题描述

无线传感器网络可用无向图 $G=(V,E)$来表示.其中,$V=(V_1,V_2,\ldots,V_N)$表示顶点集;E 表示边集.假设 V 中有 N 个节点,节点通信半径为 R.每个节点可直接与在其通信半径之内的所有节点通信,且在感知半径内可以多模方式采集数据.由于传感器网络具有基于特定任务的特征,任意节点在部署前均刻入特定应用所关注的所有事件的类型,或相应的事件编码及对应的属性值特定范围.

由于任意节点均可作为查询节点,本文关注的问题可以简单描述为:一旦某事件发生且有关于它的查询发布,1) 如何优化从事件聚合节点复制数据到多个存储节点所导致的复制能耗;2) 如何优化从查询节点搜索最

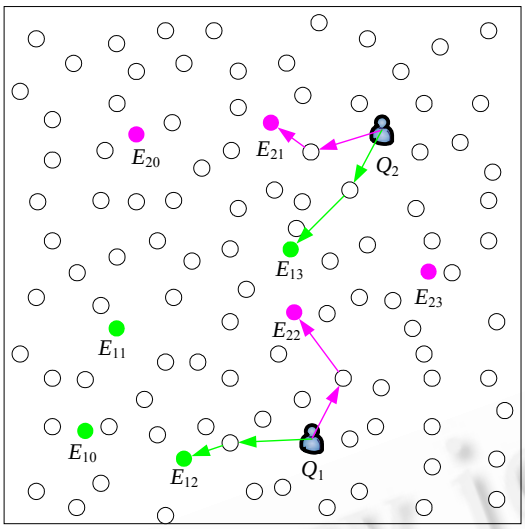

Fig.2 Illustration of queries for replicated data in sensor network
图 2 传感器网络实时查询图解

近的事件副本所导致的搜索时延及能耗;3) 如何按需进行事件数据的动态二次复制.如果在搜索过程中存在事件数据复制,则该部分复制能耗也应考虑在总能耗之内.为不失一般性,我们考虑了多事件多查询节点并存的情况.即,事件可以是发生在不同区域的不同类型的多个事件,查询可以是来自多个不同的查询节点、关注不同的事件类型且可以在同一时刻并行发布.以事件复制存储为前提的传感器网络实时查询如图 2 所示.

图 2 中,Q_1 和 Q_2 代表查询节点,E_{10} 和 E_{20} 代表原始事件信息,E_{11},E_{12},E_{13} 为 E_{10} 的副本,E_{21},E_{22},E_{23} 为 E_{20} 的副本.Q_1 和 Q_2 发布一系列查询请求以获取 2 个事件最近的副本所包含的信息,返回响应.一般来说,增加事件副本的数量将会降低查询时延,削减搜索能耗,但复制能耗会相应增大.

无线传感器网络节点规模庞大,对网络进行分簇可提高其伸缩性[1].文献[22]提出了一种高能效的分布式成簇协议 EADEEG 以保证成簇后的传感器网络每个簇范围内仅有一个簇头且簇头分布均匀.文献[23]指出,对于所有的数据相关性标准,保持簇内节点数为 20 左右可获得较优性能.据此,我们结合网络部署范围的大小确定需要的传感节点数.然后,确定最优簇半径取值和簇头数目.根据文献[22]可知,理论最优簇半径 $R_C^*=2\sqrt{\frac{2\pi\|S\|E_{DA}}{27N\varepsilon_{fs}}}$,理论簇头数目 $N_C^*=\left\lceil\frac{4\|S\|}{3\sqrt{3}R_C^{*2}}\right\rceil$.其中,$||S||$为网络部署区域面积,$E_{DA}$=5nJ/bit/signal, ε_{fs}=10pJ,N 为网络总节点数.这种分簇结构本质上提供了很好的节点管理方案,并同时保证成簇后的网络具有一定的连通度.

2.2 事件聚合及事件编码

2.2.1 事件数据聚合

对于传感器网络某一特定应用,假设其关注的最大事件类型数目为 M_0,每个事件的最大属性数为 s,则事件属性可表示为$(A_1,\ldots,A_i,\ldots,A_s)(1\le i\le s)$.每个事件所对应的每个属性都有其特定的取值范围.用$|A_i|$表示属性 A_i 的取值,对某一具体事件,若$|A_i|\in[a_i,b_i]$,则该事件可表示为$([a_1,b_1],\ldots,[a_i,b_i],\ldots,[a_s,b_s])$.若考虑事件的时空属性,假设位于坐标$(x_i,y_i)$的节点 i 在时段 t_1~t_2 内采集到属性值为$|A_1|$,$|A_2|$和$|A_3|$的事件 E_j,该事件可表示为$(E_j,x_i,y_i,t_1-t_2,|A_1|,|A_2|,|A_3|)$.一旦某一事件发生,节点获取了该事件各个$|A_i|$值后,即可确定事件的类型.从 QoS 角度考虑,可根据事件的重要性给每类事件赋优先级值.为了节省节点能量及存储空间,需根据事件数据的时空属性和感知属性来进行时空聚合操作.聚合后的事件信息可包含不同时段及不同感知属性值,但必须共享同一事件类型值.节点一般对数据先进行时间聚合,再把聚合后的数据送给其簇头处理.簇头在一段时间内收集到一定量的来自簇内普通节点的事件数据后,需进一步进行空间聚合.利用时空聚合方法抽取事件关键信息对于特定查询应用来说,既节省了存储开销,又在一定程度上削减了通信开销.

2.2.2 事件数据编码

为设计查询处理算法的方便,对应用相关的所有事件类型进行编码十分必要.假设事件编码长度 l 是事件属性个数的整数倍,即 $l=ms$(m 为整数).为方便叙述,假定所有属性值都被规格化为 0~1 之间的一个小数.本文采用二分法对事件进行编码.如果 m=1(即 $l=s$),属性值域被分为 2 段:[0,0.5]和[0.5,1.0].若$|A_i|\in[0,0.5]$,则事件编码的第 i 位置 0,否则置 1($1\le i\le s$).但 m=1 时编码不足以准确描绘相应事件.如果 m=2,每个属性值域被分为 4 段:[0,0.25],[0.25,0.5],[0.5,0.75]和[0.75,1].不失一般性,如果 m>2,每个属性值域被分为 2^m 段,每个属性子值域可用

m 位二进制数表示.重复上述过程直到 ms 位编码全部赋值为止.故通过二分法用 ms 位对具有 s 个属性的事件编码,属性值的精度可以达到 $1/2^m$.例如,考虑具有 3 个属性的事件 E=(0.2,0.4,0.8),如果 m=3,则对应的 9 位事件编码为 code(E)=001011110.本文仅考虑与特定应用相关的 M_0 类事件.把定义好的这 M_0 类事件及其编码预先载入节点中.在算法执行过程中可对与具体查询处理无关的事件数据进行屏蔽,节省节点能量.

2.3 传感器网络中基于事件优先级的分环存储策略

充分考虑到传感器网络实时应用的需求,本节给出一种基于事件优先级的分环存储策略,使得重要性级别高的事件产生更多的副本存储于离事件源所属簇头更远的环上,更大程度地降低搜索能耗,缩短查询时延.

假设特定应用关注的事件有 M_0 类,依据对应用的重要性指标,分为 k_0 个优先级,重要性 k 取值为 $1\le k\le k_0$ 且 k 为正整数.k 值越大,优先级越高.簇头负责把收集的事件信息编码后按优先级高低存储到附近的存储点(存储节点可不属于本地簇).当簇头收集到重要性为 k 的事件后,则把经过编码的数据传到距离自己 kR 的环内存储,以保证重要性大的事件有更多的机会最先被查询到.假设网内活跃节点足够多且分布较均匀,则环上必有充分多的节点充当存储点.存储环如图 3 所示.图 3 中,位于 B_7 的簇头收集到优先级为 k 的事件数据,则以半径 kR 确定存储环轨迹.然后把该事件编码复制到环上节点.在存储事件的过程中,环上节点的选取较为关键.要保证来自环外的查询请求不能跳过存储环上的节点直接进入环内,必须使环上的存储节点足够密集,但为了节约存储空间,不能过于密集.我们提出一种构环方法,可保证用最少的环上节点构成密不可透的存储链.

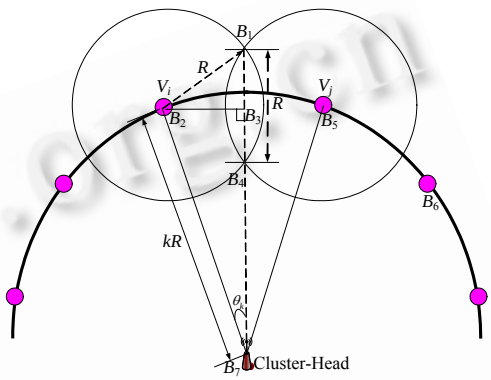

Fig.3 Illustration of storage ring in WSN

图 3 传感器网络存储环图解

假设位于 B_2 的 V_i 和位于 B_5 的 V_j 是环上相邻的存储节点.要使环外查询请求不能直接越过存储环,V_i 和 V_j 相交区域的最短割线 B_1B_4 必须满足$|B_1B_4|\ge R$.设$|B_1B_4|=R$,则 $|B_2B_3|=\frac{\sqrt{3}}{2}R$.令 $\angle B_2B_7B_3=\theta_k$,簇头到该环的距离 $|B_2B_7|\ge kR$,则 $\sin\theta_k=B_2B_3\big/B_2B_7=\frac{\sqrt{3}}{2k}$.设环上最少需均匀布置 N_k 个点才能满足设计条件,则有:

$$N_k=\lceil 2\pi/2\theta_k\rceil=\left\lceil \pi/\arcsin\frac{\sqrt{3}}{2k}\right\rceil \quad (\lceil\ \rceil\text{表示取上整}) \tag{1}$$

簇头对每类事件数据均执行环存储,从而形成以簇头为中心的多层存储环链.同一优先级的多类事件可存储于同环的相同存储节点.若查询请求来自存储环外层,则可优先搜索到较重要的事件信息.当最外层环的事件优先级低于待查事件的优先级时,表明内层环亦无待查事件,查询请求没有必要继续深入内层环;当最外层环的事件优先级高于待查事件的优先级时,可继续深入内层环进行查询.若查询请求来自存储环内层,则可根据实际情况确定继续进行外层环搜索或内层环搜索.多环存储链上的事件信息均附带含有存储有效期字段.这可以通过在事件信息中加入时间戳来实现,重要性级别高的可保留较长的存储期限.

对于单个簇头来说,基于事件优先级的分环存储策略具有一定的结构性.但对于整个网络或查询节点而言,仍属于随机存储,因为不存在全局存储节点,且查询节点对待查事件数据存储位置无法预知.针对此类随机存储,如何迅速找到感兴趣的事件信息,尤其是重要性强的事件信息,是设计实时查询处理算法的关键.

2.4 传感器网络中基于蚁群优化的实时查询处理算法

本节基于蚁群算法思想,提出传感器网络中带正反馈机制的分布式多蚁群实时查询处理算法.

2.4.1 传感器网络查询处理环境

为了阐明ARTQP算法,首先介绍查询处理应用的网络环境.本文采用第2.1节介绍的簇状网络.在某一时刻,网络仅部分节点处于活跃状态.所有活跃节点均可收集事件信息、与邻节点通信以及存储簇头传来的经过聚合的事件信息,即,所有活跃节点可同时充当路由节点和存储节点.簇内活跃节点可收集应用相关的事件数据并进行简单处理后发给簇头;簇头负责对送来的数据进行时空聚合及编码处理并复制一定份数的副本发送给相应活跃节点存储.由于分簇算法保证各簇与相邻簇适当相交,故整个拓扑具有较好的连通性.ARTQP 算法的网络环境如图 4 所示.

图 4 中忽略了休眠节点和部分存储环间活跃节点.假设簇 CL_1,CL_2 和 CL_3 有应用相关事件产生.相应簇头完成了事件编码后,即开始复制事件信息 N_k 份存储于距离簇头 kR 的环 SR_k 上的 N_k 个存储节点(k 为事件优先级值).存储节点应尽量均匀分布在环上,N_k 的取值见公式(1).事件信息除了附带含有优先级值、存储有效期、时间戳外,还应含有其对应的簇头信息.由于事件产生的随机性,簇头进行复制存储也具有随机性.对于整个网络而言,这种事件复制存储方式属于随机存储.查询者若发出针对某类优先级事件的查询请求,由于无固定存储点,必须借助于一定的盲目搜索.

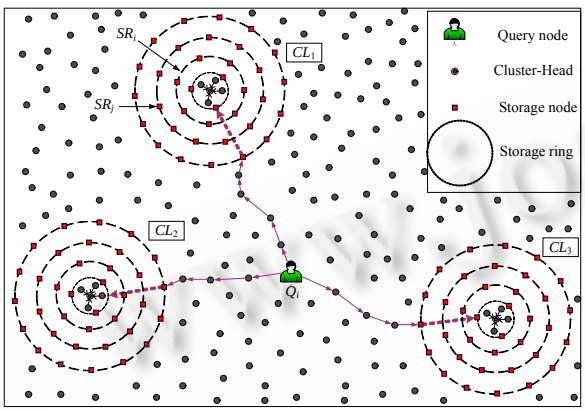

Fig.4 Network environment of ARTQP

图 4 ARTQP 算法网络环境

为了获取查询响应,只需发布查询请求探测到待查事件的一份副本即可.传感器网络针对复制事件探测的实时查询处理问题与因特网中针对复制服务获取的任播路由问题非常相似.鉴于蚁群算法在解决因特网任播路由问题时表现出的优良性能,下文将重点阐述如何利用蚁群算法优化求解针对传感器网络的实时查询处理问题.

2.4.2 查询请求的管理

每个查询请求根据具体的应用需求而发布,返回的是关于事件信息的简单响应而非事件数据本身.但发布的查询请求一般包含所关注事件的优先级、事件类型或编码.

查询请求通常包含一些公共属性:事件编码 *EC*、查询节点标识符 *ID_S*、查询点产生的蚂蚁标识 *ID_ANT*、蚂蚁当前所在节点的标识 *ID_CUR*、蚂蚁出发时间 *T_ANT*、请求开始时间 *BT*、已走跳数 *HN_CUR*、能量约束 *ENE_R* 和时延约束 *DUR_R*.每个查询源节点的初始请求以泊松分布产生,能量约束 *ENE_R* 在给定范围内服从均匀分布,时延约束 *DUR_R* 取值根据具体应用而设定.额外的查询请求属性(时空属性等)也可根据需要而加入.

在 ARTQP 算法中,每个查询源节点对应一个蚁穴,每个初始请求对应一只蚂蚁,每只蚂蚁携带有事件编码等查询请求属性.当待查事件从未被探测到时,对该事件的第 1 次查询对应多个初始查询请求(即多只蚂蚁),直到蚂蚁首次探测到该事件数据为止.对同一事件的后继查询每次发一个初始请求,仅对应一只蚂蚁,除非网络状态发生急剧变化.每只蚂蚁每前进一步,对应一个前向请求的产生.每个前向请求的 *ID_S*,*ID_ANT*,*T_ANT*,*ENE_R* 和 *DUR_R* 取值都不改变,每个当前请求将其下一跳标识符赋给新前向请求的 *ID_CUR*,将其 *HN_CUR* 值加 1 赋给新前向请求的 *HN_CUR*.设当前节点 i 到下一跳 j 的延时为 $d_1(i,j)$,每个当前请求将其 *BT* 值加 $d_1(i,j)$赋给新的前向请求的 *BT*.所有请求将依时间先后顺序置于节点请求管理队列.

2.4.3 ARTQP 算法的选路策略

在处理初始请求和前向请求的过程中,由于网络规模巨大且全局信息未知,仅依靠当前节点邻节点的局部信息来确定下一跳的走向.在探测到存储节点之前,邻节点一定为无事件信息的活跃节点,可一定程度地降低计算复杂性,提高处理效率.在选路时,首先依据链路上的信息素、链路时延和邻节点的剩余能量值计算当前节点到所有可能的邻节点之间的权重.再依据式(2)计算当前节点到每个邻节点之间的选路概率.概率值最大的下一

跳将被选取.设从某蚁穴出发的第 k 只蚂蚁当前节点为 n_1,任意邻节点为 n_2,则选路概率计算公式见式(2):

$$P_k(n_1,n_2)=\frac{\pi(n_1,n_2)\cdot[\lambda(n_1,n_2)]^{\beta}\cdot[\mu(n_1,n_2)]^{\eta}}{\sum_{d\in L(n_1)}\pi(n_1,n_2)\cdot[\lambda(n_1,n_2)]^{\beta}\cdot[\mu(n_1,n_2)]^{\eta}} \tag{2}$$

式(2)中,$\pi(n_1,n_2)$为链路(n_1,n_2)上的信息素强度,$L(n_1)$表示节点 n_1 的待选邻节点集合.$\lambda(n_1,n_2)=1/\overline{d}_1(n_1,n_2)$, $\overline{d}_1(n_1,n_2)$是链路(n_1,n_2)上的平均时延,$\mu(n_1,n_2)$为邻节点 n_2 上的剩余能量;β和η分别为确定链路时延和邻节点剩余能量重要性的参数.

从式(2)可以看出,蚂蚁趋向于选择链路上信息素更强、时延更小、对应节点剩余能量更多的邻节点作为下一跳.如果蚂蚁发现有邻节点已经在它的路径表中,则把该邻节点从候选节点集中删除以避免环路.为了判断是否满足能量约束 *ENE_R* 和时延约束 *DUR_R*,需借助于邻节点的能量信息和当前请求的 *BT* 信息.若$\mu(n_1,n_2)<ENE_R$或$BT>DUR_R$,则该请求不可接受.如果所有的邻节点均不满足约束条件,则对应的蚂蚁死亡.请求接受率即为通过蚂蚁探测到待查事件信息的查询次数与同一蚁穴发出的针对该事件的总查询次数之比.

2.4.4 ARTQP 算法的信息素更新规则

为保持网络信息的新鲜性,算法引入两条信息素更新规则:全局更新和局部更新.蚂蚁每前进一步都将加强链路信息素,故从同一蚁穴出发的蚂蚁趋向于走相同的路径.但每只蚂蚁的约束条件因查询应用的不同而不尽相同,它们有选择不同路径的可能性.局部信息素更新公式为

$$\pi(n_1,n_2)\leftarrow(1-\rho_1)\cdot\pi(n_1,n_2)+\Phi\cdot\Delta\pi^k(n_1,n_2) \tag{3}$$

式(3)中,$\pi(n_1,n_2)$为链路(n_1,n_2)上的信息素强度,ρ_1 为局部信息素挥发系数且 $0<\rho_1<1$.Φ为可调参数,取决的不同的查询任务.$\Delta\pi^k(n_1,n_2)=1/\overline{d}_1(n_1,n_2)$, $\overline{d}_1(n_1,n_2)$为链路(n_1,n_2)上的平均时延.局部更新公式可基于链路时延动态调整链路信息素强度.链路平均时延大,表明该链路待传数据量大或距离较长,使得信息素增强量变小,最终导致后继蚂蚁不再选择该时延较大的链路.

当蚂蚁找到某理想路径后,它将增强整条路径的信息素,同时,不在该路径上的其他链路信息素随时间而挥发.故链路上的信息素不会永远增强,这在一定程度上起到了平衡负载的作用.全局信息素更新公式如下:

$$\pi(n_1,n_2)\leftarrow\begin{cases}(1-\rho_2)\cdot\pi(n_1,n_2)+\Delta\hat{\pi}(n_1,n_2), & \text{if}(n_1,n_2)\in p\\ (1-\rho_2)\cdot\pi(n_1,n_2), & \text{else}\end{cases} \tag{4}$$

式(4)中,p 代表从查询源节点到事件存储点的路径,$\Delta\hat{\pi}(n_1,n_2)=(\mu_{\min})^{\sigma}/\theta$,$\mu_{\min}$ 为路径 p 上的瓶颈能量值.θ为可调参数,取决于具体任务.σ为信息素对节点能量的响应度.ρ_2 为全局信息素挥发系数$(0<\rho_2<1)$.

2.4.5 ARTQP 算法的动态二次复制

ARTQP 算法事件复制能耗是本文要考虑的一个重要方面.第 2.3 节对 ARTQP 算法第 1 阶段的事件复制存储策略做了介绍.当完成了第 1 阶段的复制存储之后,针对事件的查询即可发布.对于某一事件信息,一旦它的一个副本被某次查询的一只蚂蚁搜索到,蚂蚁将执行关于该事件数据的复制,产生一个新的副本.然后,蚂蚁释放后向请求并启动全局信息素更新机制从存储环原路返回,携带相应的响应信息和新的事件副本.当到达路径中央时,新的副本被蚂蚁存储于中央节点,查询的响应信息仍随蚂蚁继续返回查询源节点.当有关于该事件的后继查询发布时,后继蚂蚁将依照路径上的信息素痕迹在首次成功查询的半路上找到前驱蚂蚁复制并存储的副本,所用搜索时间和能耗均减半.该蚂蚁返回时亦执行相同复制操作.此查询复制过程重复执行,直到没有关于该事件的新查询请求发布或新事件副本离查询节点仅一跳距离.通过蚂蚁动态按需复制事件副本且增强路径信息素,不但对同蚁穴的后继蚂蚁搜索行为具有指导作用,还可以在一定程度上引导来自不同蚁穴但针对同一事件的其他蚂蚁.故通过动态二次复制技术,能在很大程度上缩短搜索时间,减少搜索能耗,且实现了事件副本的按需复制,尽量节省复制能耗.ARTQP 的动态二次复制技术如图 5 所示.

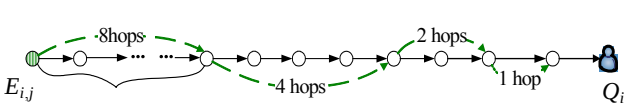

Fig.5 Illustration of the second replication operation of ARTQP algorithm

图 5 ARTQP 算法二次复制策略图解

2.4.6 ARTQP 算法的具体步骤

无线传感器网络基于多蚁群的 ARTQP 复制数据查询处理算法具体步骤如下:

Step 1. 每个查询源节点(蚁穴)按泊松分布产生对 M 个感兴趣事件的带不同 DUR_R 和 ENE_R 约束的初始请求,每个初始请求的产生对应一只蚂蚁的出发.这些初始请求按产生先后顺序插入请求管理器队列;

Step 2. 请求管理器按序处理各查询初始请求.每处理一个初始请求,就马上产生相应的前向请求.新产生的前向请求也将按产生时间先后顺序插入请求管理器队列;

Step 3. 请求管理器继续按序处理各初始请求、前向请求或后向请求.处理的顺序取决于各请求的产生时间;

Step 4. 在处理某一节点对应的请求时,必须先判断该节点是否存在满足查询请求约束条件的相邻链路(或节点).若没有满足条件的邻节点,则对应的蚂蚁死亡.否则,判断请求类型,按类型区分处理;

A) 对于初始请求或前向请求,蚂蚁首先判断是否有待查事件数据在当前节点或其邻节点,通过匹配当前请求的 EC 属性值和节点内事件数据的编码来初步确定:

1) 若当前节点或其邻节点有事件信息且编码属于查询请求的 EC 编码范围,则继续判断该事件的其他属性值是否与请求的对应属性值匹配.如果当前节点或邻节点所存储的数据确实是待查数据,则触发后向请求,启动二次复制及全局信息素更新;

2) 若当前节点或其邻节点有事件信息但编码与待查事件不符,则节点首先比较待查事件与存储事件的优先级.设待查事件优先级为 PRI_U,存储事件优先级为 PRI_S.若 $PRI_U \geq PRI_S$,则向存储事件对应的源簇头反向作外环搜索.若搜索成功,转 1),否则,转 3);若 $PRI_U < PRI_S$,则向该存储事件对应的源簇头方向作内环搜索以找到优先级等于 PRI_U 的事件存储点.搜索过程中,若某存储点具有来自不同簇头的不同优先级的事件,且待查事件优先级更接近另一簇头维持的环优先级,则可尝试转移到异簇存储环继续搜索,直到请求过期导致蚂蚁死亡;

3) 若当前节点或其邻节点没有任何事件信息,则蚂蚁根据选路概率在可能的邻节点中选取下一跳且避免环路,对选择的链路进行信息素局部更新.该请求完成下一跳选择后,触发新的前向请求.判断新的前向请求是否满足约束,若不满足,则对应蚂蚁死亡.否则,把该新产生的前向请求按时间插入队列.

B) 对于后向请求,主要任务是把查询响应结果返回查询节点,记录蚂蚁从蚁穴到事件副本所走路径;启动二次动态复制和信息素全局更新.

Step 5.如果各节点没有待处理的请求,统计各蚁穴能够成功找到待查事件的查询次数,计算查询成功率;计算整个查询处理过程的时延、搜索能耗及复制能耗.

3 ARTQP 算法理论分析

定义 1. 设在一定的时间间隔 ΔT 内,产生事件的簇头占整个簇头的百分比为 ψ($0.6\% < \psi < 5\%$),则称 ψ 为 ΔT 内的事件密度.

在相同的网络规模下,事件密度 ψ 越大,同一时段网络内产生的事件越多.显然,当 ψ 较小时,最大存储环的半径越大越有利于提高查询效率.但环半径一般应小于整个网络的最大半径.

为深入分析 ARTQP 算法的性能,我们做了如下假设:

1) 传感器网络部署于宽度为 W 米的正方形空间,网络监测面积为 $\|S\|=W^2$,节点数目为 N,以均匀簇结构组织,每簇节点个数约为 20,簇半径为 R_C,故密度约为 $\frac{20}{\pi R_C^2}$ 且 $N=\frac{20}{\pi R_C^2}\|S\|$.在整个网络部署期 T 的任意时间间隔 ΔT 内,事件密度为 ψ 且产生原始事件的簇头分布均匀.部署期的某时段网络将产生 M 个应用相关的事件数据.

2) 传感器网络内每个原始事件根据其优先级 k 的取值产生 $r_i(k)$ 个副本,故网络中每个事件均有 $r_i(k)+1$ 份相同的数据存在.每个事件对应的查询次数为 $40k(i)$ 次.查询节点为 N_S 个($1 \leq N_S \leq 10$),均匀分布于网内.每个查询均为单一查询,只要找到待查事件信息,返回简单的响应即可完成.

3) 查询处理过程的复制、搜索能耗及时延均与节点转发次数成正比,且每转发一次消耗 1 个单位的能量,

耗时 100ξ个时间单位(ξ是衡量网络中链路负载的参数且 $0.1\le\xi<1$).

根据以上假设,结合第 2.1 节的分簇方法,可知:

$$R_C=2\sqrt[4]{\frac{2\pi\|S\|E_{DA}}{27N\varepsilon_{fs}}}=2\sqrt[4]{\frac{1000\pi\|S\|}{27N}}=2\sqrt[4]{\frac{1000\pi\|S\|}{27N}\cdot\frac{\pi R_C^2}{20\|S\|}}.$$

解方程可得:R_C=17.1 米.此外,在保证网络无盲点的前提下,为减少簇开销,本文簇头数目的计算如下:

$$N_C=\left\lceil\left(\frac{4\|S\|}{3\sqrt{3}R_C^2}+\frac{2\|S\|}{3\sqrt{3}R_C^2}\right)\Big/2\right\rceil=\left\lceil\frac{1}{\sqrt{3}R_C^2}\cdot\frac{\pi R_C^2N}{20}\right\rceil=\left\lceil\frac{\pi}{20\sqrt{3}}N\right\rceil \tag{5}$$

由第 2.3 节的环存储策略可知,事件优先级的最大值 k_0 决定了环的最大半径.为保证时间间隔ΔT 内产生事件的各簇头维持的存储环尽量不相交,可令 $2k_0R\sqrt{\psi\cdot N_C}\le W$,结合式(5),则有

$$2k_0R\sqrt{\left\lceil\frac{\pi}{20\sqrt{3}}N\right\rceil\psi}\le\sqrt{\frac{\pi R_C^2N}{20}} \tag{6}$$

设节点通信半径 R=30,而最优簇半径 R_C=17.1,由式(6)可得: $k_0\le0.113\sqrt{\frac{N}{\left\lceil\frac{\pi}{20\sqrt{3}}N\right\rceil\psi}}$.故在假设 1)的基础上,可得 k_0 的理论值如下:

$$k_0=\left\lceil0.113\sqrt{\frac{N}{\left\lceil\frac{\pi}{20\sqrt{3}}N\right\rceil\psi}}\right\rceil=\left\lceil\frac{0.38}{\sqrt{\psi}}\right\rceil \tag{7}$$

本文令 ψ=3%,则 k_0=3.若ΔT 内事件密度小于 3%,k_0 的理论值可相应增大,以牺牲复制能耗来减少搜索能耗,缩短搜索时延.当然,若事件优先级低于 k_0,则选择半径小于 k_0R 的存储环进行存储.此外,根据文献[13]可知,正方形区域任意两节点间的平均欧氏距离为

$$\bar{d}=W\underbrace{\int\cdots\int_0^1}_{4}\sqrt{(x_1-y_1)^2+(x_2-y_2)^2}\,\mathrm{d}x_1\mathrm{d}y_1\mathrm{d}x_2\mathrm{d}y_2=W\frac{2+\sqrt{2}+5\ln(1+\sqrt{2})}{15}\approx0.52W,$$

故网络中任意两节点间的平均跳数为

$$\bar{h}=0.52W/R=0.52\sqrt{\frac{N\pi R_C^2}{20}\Big/R^2}=0.12\sqrt{N} \tag{8}$$

3.1 ARTQP算法复制能耗分析

在 ARTQP 算法中,数据复制存在两个阶段.第 1 阶段,簇头把编码后的优先级为 $k(i)$的原始事件 i 复制 $N_k(i)$个副本并把它们发送到距离为 $R\cdot k(i)$左右的活跃节点进行环存储(若环超越网络边界,则超越部分由边界上的活跃点充当存储节点),其能耗为 $C_{R1}^A(i)$;第 2 阶段,找到待查事件的蚂蚁执行数据复制并把新复制的副本存储于路径上的中央节点(第 2.4.5 节对此阶段进行了详细描述),其能耗为 $C_{R2}^A(i)$.由于蚁穴的数目为 N_S 且每个蚁穴到待查事件 i 都有其特定的路径,故 ARTQP 算法的总复制能耗可表达如下:

$$C_R^A=\sum_{i=1}^{M}C_{R1}^A(i)+N_S\cdot\sum_{i=1}^{M}C_{R2}^A(i) \tag{9}$$

为了计算第 1 复制阶段的能耗 $C_{R1}^A(i)$,令 $\bar{k}(i)=\left\lceil\frac{k_0}{2}\right\rceil=2$,则第 1 复制阶段平均复制能耗表达式如下:

$$C_{R1}^A=\sum_{i=1}^{M}C_{R1}^A(i)=\sum_{i=1}^{M}N_k(i)\cdot k(i)=M\cdot N_2\cdot2=2M\left\lceil\pi/\arcsin\frac{\sqrt{3}}{4}\right\rceil=14M \tag{10}$$

下面进一步考虑第 2 复制阶段的能耗.

引理 1. 设第 1 次查询蚂蚁成功找到待查事件所用的平均跳数为 $\bar{h}_A$,则第 2 复制阶段的复制能耗满足

$C_{R2}^{A}(i) \le \overline{h}_A - 1$.

证明:令 h_j 表示蚂蚁执行第 j 次复制所需的跳数,f_0 表示复制执行的次数,则 $1 \le j \le f_0$.

1) $$h_1 = \begin{cases} \overline{h}_A/2, & \text{若}\overline{h}_A\text{为偶数} \\ (\overline{h}_A - 1)/2, & \text{若}\overline{h}_A\text{为奇数} \end{cases}.$$

2) $$h_{j+1} = \begin{cases} h_j/2, & \text{若}h_j\text{为偶数} \\ (h_j - 1)/2, & \text{若}h_j\text{为奇数} \end{cases}.$$

而第 2 复制阶段的复制能耗可表示为

$$C_{R2}^{A}(i) = \sum_{j=1}^{f_0} h_j \tag{11}$$

显然,当 $\overline{h}_A$ 和所有 h_j 均为偶数且针对事件 i 的查询次数足够大时,$C_{R2}^{A}(i)$ 取最大值.此时,复制操作一直持续到最新的副本距查询点仅 1 跳为止.易知: $f_0 = \log_2 \overline{h}_A$.第 2 复制阶段复制能耗最大值表达式如下:

$$\max\{C_{R2}^{A}(i)\} = \sum_{j=1}^{\log_2 \overline{h}_A} h_j = \frac{\overline{h}_A}{2} + \frac{\overline{h}_A}{4} + \ldots + 1 = \frac{1 - 2^{\log_2 \overline{h}_A}}{1 - 2} = \overline{h}_A - 1 \tag{12}$$

□

定理 1. 设某时段与应用相关的事件总数为 M,首次查询蚂蚁找到相应事件 i 的平均跳数为 $\overline{h}_A(i) = \overline{h} - \overline{k}(i) + 1$,查询次数 $q_i \ge f_0$,蚁穴数为 N_S,则 ARTQP 算法的平均总复制能耗约为

$$C_R^A = M(0.12N_S \cdot \sqrt{N} - 2N_S + 14) \tag{13}$$

证明:因为 $\overline{h}_A(i) = \overline{h} - \overline{k}(i) + 1 = \overline{h} - 1$ 且 $q_i \ge f_0$,故可把式(10)、式(12)、式(8)代入式(9),有:

$$C_R^A = \sum_{i=1}^{M} N_k(i) \cdot k(i) + N_S \cdot \sum_{i=1}^{M} [\overline{h}_A(i) - 1] = M(N_S \cdot \overline{h} - 2N_S + 14) = M(0.12N_S \cdot \sqrt{N} - 2N_S + 14).$$ □

由定理 1 可知,当网络规模 N 和蚁穴数 N_S 一定时,ARTQP 算法的平均总复制能耗 C_R^A 与事件总数 M 成正比.当事件数 M 一定时,C_R^A 随网络规模增大而增加.在相同规模的网络环境下,C_R^A 随蚁穴数递增而增大.这是因为随着 N_S 增多,产生的搜索路径数增加,使得第 2 复制阶段所需能耗增大,从而导致总复制能耗相应增大.

3.2 ARTQP算法搜索能耗分析

ARTQP 算法的搜索行为实质上也分两个阶段.设网络中某时段事件总数为 M,当针对某事件 i 的第 1 复制阶段完成后,每个蚁穴发布一定数量的蚂蚁以搜索该事件较近的副本.此为第 1 搜索阶段,其能耗为 $C_{S1}^{A}(i)$.一旦事件 i 的一个副本被某蚁穴的一只蚂蚁发现,从该蚁穴出发的后继蚂蚁将根据路径上的信息素痕迹选择较优路径,一般可在原路径的中央节点获取待查事件信息.此为第 2 搜索阶段,其能耗为 $C_{S2}^{A}(i)$.设网络中蚁穴总数为 N_S,每个事件查询次数为 $q_i = 40k(i)$(平均值为 $\overline{q}_i = 40\overline{k}(i) = 80$),则每个蚁穴对某事件 i 的平均查询次数为 $40k(i)/N_S$.ARTQP 算法第 1 搜索阶段的总搜索能耗可表示如下:

$$C_{S1}^{A} = N_S \cdot \sum_{i=1}^{M} C_{S1}^{A}(i) \tag{14}$$

ARTQP 算法第 2 搜索阶段的搜索能耗与第 2 复制阶段的二次动态复制策略紧密相关.一旦某蚁穴的某只蚂蚁首次发现待查事件 i 的一个副本,其后继蚂蚁通过路径信息素强度等信息可较快找到事件新的副本.

引理 2. 设第 1 次查询蚂蚁成功找到待查事件 i 所用跳数为 $h_A(i)$,第 2 搜索阶段蚁穴对某事件平均查询次数为 $q' = 40k(i)/N_S - 1$.则当 q' 足够大时,该蚁穴针对事件 i 第 2 搜索阶段的搜索能耗 $C_{S2}^{A}(i) \le q' + h_A(i) - \log_2 h_A(i) - 1$.

证明:可用类似引理 1 的证明方法,易得如下表达式:

$$C_{S2}^{A}(i)\le\begin{cases}\sum_{j=1}^{q'}h_A(i)/2^j, & 0<q'\le\log_2 h_A(i)-1\\ \sum_{j=1}^{\log_2 h_A-1}\frac{h_A(i)}{2^j}+(q'-\log_2 h_A(i)+1), & q'>\log_2 h_A(i)-1\end{cases}\tag{15}$$

由于 q'足够大,故有下式成立:

$$\begin{aligned}\max\{C_{S2}^{A}(i)\}&=\sum_{j=1}^{\log_2 h_A(i)-1}\frac{h_A(i)}{2^j}+(q'-\log_2 h_A(i)+1)\\&=\frac{1-2^{\log_2 h_A(i)}}{1-2}+(q'-\log_2 h_A(i))=q'+h_A(i)-\log_2 h_A(i)-1\end{aligned}\tag{16}$$

□

定理 2. 设某时段与应用相关的事件总数为 M,首次查询蚂蚁找到相应事件 i 的平均跳数为 $\bar{h}_A(i)=\bar{h}-\bar{k}(i)+1$ ($\bar{h}_A(i)=\bar{h}_A$),首次查询所需蚂蚁平均数为 $\bar{N}_{Ant}(i)$ ($\bar{N}_{Ant}(i)=\bar{N}_{Ant}$),事件平均查询次数为 $\bar{q}_i$ ($\bar{q}_i=\bar{q}=(40\times\bar{k}(i))/N_S-1=80/N_S-1$ 且足够大),蚁穴数为 N_S,则 ARTQP 算法的平均总搜索能耗为

$$C_S^A=M[N_S(\bar{N}_{Ant}+1)(0.12\sqrt{N}-1)-N_S\log_2(0.12\sqrt{N}-1)-2N_S+80]\tag{17}$$

证明:因为 $C_S^A=C_{S1}^A+C_{S2}^A=N_S\sum_{i=1}^{M}C_{S1}^A(i)+N_S\sum_{i=1}^{M}C_{S2}^A(i)$,结合式(16)可得:

$$C_S^A=N_S\sum_{i=1}^{M}[N_{Ant}(i)\bar{h}_A(i)+\frac{80}{N_S}+\bar{h}_A(i)-\log_2\bar{h}_A(i)-2)]\tag{18}$$

而 $\bar{h}_A(i)=\bar{h}-\bar{k}(i)+1=\bar{h}-1=0.12\sqrt{N}-1$ 且 $\bar{N}_{Ant}(i)=\bar{N}_{Ant}$,代入式(18),可得:

$$\begin{aligned}C_S^A&=N_S\cdot M\left[(\bar{N}_{Ant}+1)\bar{h}_A-\log_2\bar{h}_A+\frac{80}{N_S}-2\right]\\&=M[N_S(\bar{N}_{Ant}+1)(\bar{h}-1)-N_S\log_2(\bar{h}-1)-2N_S+80]\\&=M[N_S(\bar{N}_{Ant}+1)(0.12\sqrt{N}-1)-N_S\log_2(0.12\sqrt{N}-1)-2N_S+80].\end{aligned}$$

□

由定理 2 可知,当网络规模和蚁穴数一定时,ARTQP 的平均总搜索能耗 C_S^A 亦与事件数 M 成正比.当事件数一定时, C_S^A 随网络规模增大而增加.在相同规模网络环境下, C_S^A 随蚁穴数 N_S 递增而增大.这是因为随着 N_S 增多,第 1 搜索阶段搜索源增加导致第 2 阶段搜索路径数增加,两搜索阶段所需能耗均存在一定程度的增加.由于蚁群算法的正反馈及采用的二次复制策略,第 2 搜索阶段路径长度变小,从而使得总搜索能耗的递增趋于平缓.

3.3 ARTQP算法查询时延分析

定义 2. 设查询者 $Q(u)(1\le u\le N_S)$从发出对优先级为 $k(i)$的事件 i 的查询请求到获取关于该事件响应的这段时间为 $D(u,i)(1\le i\le M)$,则称 $\bar{D}(i)=\frac{1}{\bar{q}_i/N_S}\cdot\frac{1}{N_S}\sum_{u=1}^{N_S}D(u,i)$ 为事件 i 的平均查询时延, $\bar{D}=\frac{1}{M}\sum_{i=1}^{M}\bar{D}(i)$ 为 M 个事件的平均查询时延.设优先级等于 k 的事件有 $M_k(M_k<M)$个,则称 $\bar{D}_k=\frac{1}{M_k}\sum_{k(j)=k}\bar{D}(j)$ 为优先级等于 k 的事件的平均查询时延.

ARTQP 算法的查询时延实质上由两部分组成:首次查询时延 D_{Fir} 和后继查询时延 D_{Fol}.首次搜索时延是指查询者针对某一事件发送第 1 个查询请求至返回该事件响应所经历的时间;后继查询时延是指查询者针对某一已查事件发送后继查询请求至返回该事件对应的后继响应所经历的时间.首次查询时延与采用蚁群算法的搜索策略密切相关.由于蚁群算法天然的并行性,某查询节点发出的多只蚂蚁并行搜索目标事件,第 1 只搜索到目标事件的蚂蚁所历时延即为首次查询时延.后继查询时延与算法的动态二次复制策略紧密联系,可结合第 2 搜索阶段进行分析.根据定义 2 可知:

$$D(u,i)=D_{Fir}(u,i)+D_{Fol}(u,i)\tag{19}$$

若事件查询次数足够多,则首次查询时延 $D_{Fir}(u,i)=2[100\xi\cdot h_A(i)]=200\xi\cdot h_A(i)$ $(0.1\le\xi<1)$,后继查询时延 $D_{Fol}(u,i)=2\times100\xi(q'+h_A(i)-\log_2 h_A(i)-1)$.令 $\bar{\xi}=0.5$,$h_A(i)$ 的平均值 $\bar{h}_A(i)=\bar{h}-\bar{k}(i)+1=\bar{h}-1$,q' 的平均值 $\bar{q}'=\dfrac{40\bar{k}(i)}{N_S}-1=80/N_S-1$,则结合定义 2 及式(19)可知,针对某时段内产生的 M 个事件的平均查询时延为

$$\begin{aligned}\bar{D}&=\frac{1}{M}\sum_{i=1}^{M}\bar{D}(i)=\frac{1}{\bar{q}_i/N_S}\cdot\frac{1}{M}\sum_{i=1}^{M}\frac{1}{N_S}\sum_{u=1}^{N_S}D(u,i)=\frac{1}{\bar{q}_i/N_S}\cdot\bar{D}(u,i)=\frac{1}{\bar{q}_i/N_S}[\bar{D}_{Fir}(u,i)+\bar{D}_{Fol}(u,i)]\\&=\frac{1}{80/N_S}\cdot200\bar{\xi}\cdot[\bar{h}_A(i)+80/N_S+\bar{h}_A(i)-\log_2\bar{h}_A(i)-2]=\frac{5N_S}{2}\cdot\bar{\xi}[2(\bar{h}-1)+80/N_S-1-\log_2(\bar{h}-1)-1]\\&=\frac{5N_S}{4}[2\bar{h}-\log_2(\bar{h}-1)+80/N_S-4]=\frac{5N_S}{4}[0.24\sqrt{N}-\log_2(0.12\sqrt{N}-1)-4]+100\end{aligned}\tag{20}$$

结合定义 2 可知,针对某时段内产生的 M 个事件中优先级等于 k 的 M_k 个事件平均查询时延为

$$\begin{aligned}\bar{D}_k&=\frac{1}{M_k}\sum_{k(j)=k}\bar{D}(j)=\frac{1}{q_i/N_S}[\bar{D}_{k\_Fir}(u,i)+\bar{D}_{k\_Fol}(u,i)]\\&=\frac{1}{40k(i)/N_S}\cdot200\bar{\xi}[(\bar{h}-k(i)+1)+\frac{40k(i)}{N_S}+\bar{h}-k(i)-\log_2(\bar{h}-k(i)+1)-1]\\&=\frac{5N_S}{2k(i)}[\frac{40k(i)}{N_S}+2(\bar{h}-k(i))-\log_2(\bar{h}-k(i)+1)]\\&=\frac{5N_S}{2k(i)}[2(0.12\sqrt{N}-k(i))-\log_2(0.12\sqrt{N}-k(i)+1)]+100\end{aligned}\tag{21}$$

由式(20)可知,ARTQP 算法的平均查询时延 $\bar{D}$ 仅与网络规模 N 和蚁穴数目 N_S 有关,随网络规模的增大而增大.当网络规模较小时,N_S 取值越大,查询时延越小;当网络规模增大($N\ge800$)时,N_S 取值越小,查询时延越小.这可以解释为,虽然 N_S 增大导致路径数增加,但对于小规模网络,路径长度较短且存在重复链路的路径较多,故时延变小;对于大规模网络,初始路径长度相应较长且存在重复链路的路径相对较少,故查询时延相应增大.

由式(21)可知,对于多个具有相同优先级 $k(i)$的事件,ARTQP 算法对应的查询时延 $\bar{D}_k$ 随网络规模增大而增大.在相同网络规模下,对于优先级高的事件,ARTQP 算法查询时延较小,这主要是由于优先级高的事件对应的存储环半径大,使得搜索路径变短.此外,高优先级事件的副本数量相对较多,使得搜索命中率提高.

4 仿真实验

为了验证 ARTQP 算法的实时性及节能性,我们在 Windows XP 系统的 Visual C++环境下进行了模拟实验,并将实验结果与其他 4 种数据查询处理策略进行了比较.比较算法分别为:1) 基于蚁群算法的初始随机复制查询处理算法 ARIRQ(ant-based querying with random initial replication),该算法实质上是 ARTQP 算法的初始版,但数据的初始存储采用随机复制策略,并未按环结构存储;2) 基于扩展环的查询处理算法 ERQ(expanding ring-based querying[13]),该算法数据初始存储采用随机复制方法,搜索过程采用逐步扩大搜索环半径的受限洪泛机制,且搜索到目标事件后,不执行事件数据二次复制;3) 基于复制策略的随机游走查询处理算法 RWQ_R (random-walk based querying with replication),该算法数据初始存储亦采用随机复制方法,但搜索过程采用 Random Walk 策略[24],且不执行事件二次复制;4) 非复制随机游走查询处理算法 RWQ_NR(random-walk based querying without replication),该算法与 RWQ_R 算法的主要区别在于它不执行事件数据的任何复制.

实验采用随机拓扑生成器产生传感器节点,节点初始能量设为 800 单位,通信半径设为 30 米,节点数在 200~8000 范围内递增变化,保持每簇节点数约为 20,网络规模随节点数的增加而增大.模拟时间控制在 0~5000 单位,在模拟期间某时段内随机产生 12 个不同类型的原始事件数据,大致均匀分布于网络内,这 12 个事件数据分为 3 个优先级,每 4 个事件共享一个优先级.整个网络随机产生 2~8 个查询节点(蚁穴),且每个蚁穴对每个事件数据均有查询请求且平均查询次数约为 10,事件优先级越高,对应的查询次数越多.每只蚂蚁 *ENE_R* 均值为 80 单位,*DUR_R* 均值为 2 000 单位,信息素挥发系数设为 0.5.为了获取算法的复制能耗、搜索能耗和相应的查询时

延,分别在节点数为 200~8000 范围内的 11 种不同规模网络下进行了模拟.实验结果及其分析如下.

4.1 ARTQP算法与其他算法查询能耗比较

ARTQP 算法和其他几种算法查询能耗比较如图 6~图 9 所示.图 6 给出了 ARTQP 和其他算法平均复制能耗比较情况.其中,AvgRepCost of Algorithm-*X* 表示算法 Algorithm-*X* 对应的平均复制能耗值.由于 RWQ-NR 不执行任何复制,故复制能耗为 0,图 6 中不予绘出.从图中可以看出,ERQ 复制能耗最大,这是因为它初始阶段产生的副本比其他算法均高,且复制所用跳数亦高于其他算法.RWQ-R 初始复制阶段,产生的副本参照 ARTQP 标准,但副本复制的跳数参照 ERQ 标准,且不执行二次复制,故其复制能耗介于 ERQ 和 ARTQP 之间.ARTQP 和 ARIRQ 均通过蚂蚁执行事件动态二次复制,但总的来说,其副本数相对较少,平均复制能耗低于 ERQ 和 RWQ-R 算法,且 ARTQP 的复制能耗与定理 1 结论吻合,进一步证明了定理 1 的正确性.ARIRQ 的复制能耗比 ARTQP 略低,这得益于其初始复制不分事件优先级,且复制最大跳数不超过 ARTQP 最大优先级事件环对应跳数.虽然 ARTQP 在复制能耗方面的性能略逊于 ARIRQ,但采用的按优先级环存储策略将很大程度降低搜索阶段能耗.

图 7 给出了 ARTQP 算法和其他算法平均搜索能耗比较情况.其中,AvgSchCost of Algorithm-*X* 表示算法 Algorithm-*X* 对应的平均搜索能耗值.由图 7 可知,采用蚁群二次复制策略的算法(ARTQP 和 ARIRQ)对应的平均搜索能耗远低于其他 3 种未采用二次复制的算法对应的搜索能耗.这表明,二次复制策略对于降低后继查询的搜索能耗极为有效.此外,ARTQP 的平均搜索能耗与定理 2 的结论相互吻合,进一步验证了定理 2 的正确性.在采用蚁群二次复制策略的两种算法中,ARTQP 算法的搜索能耗小于 ARIRQ 算法,这主要由于 ARTQP 算法初始复制阶段采用按优先级环存储方法,为蚂蚁初始搜索提供了一定的指导.在未采用二次复制策略的 3 种算法中,执行初始复制的两种算法(ERQ 和 RWQ-R)比不执行任何复制的 RWQ-NR 算法搜索能耗要低,进一步表明复制策略在查询处理中的重要作用.由于 RWQ-R 算法搜索过程具有很强的随机性,其所需搜索能耗略高于 ERQ 算法.

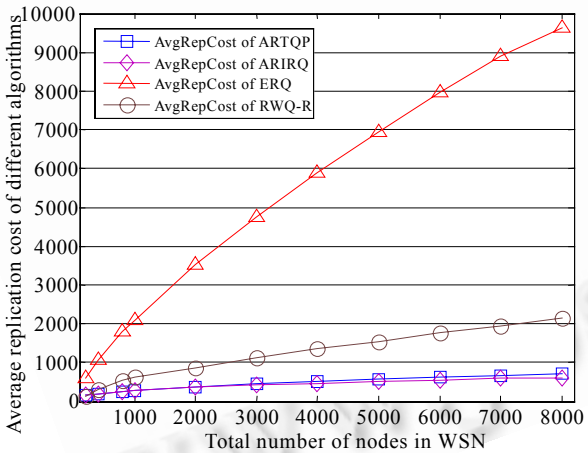

Fig.6 Comparison of average replication cost between ARTQP and other algorithms

图 6 ARTQP 与其他算法平均复制能耗比较

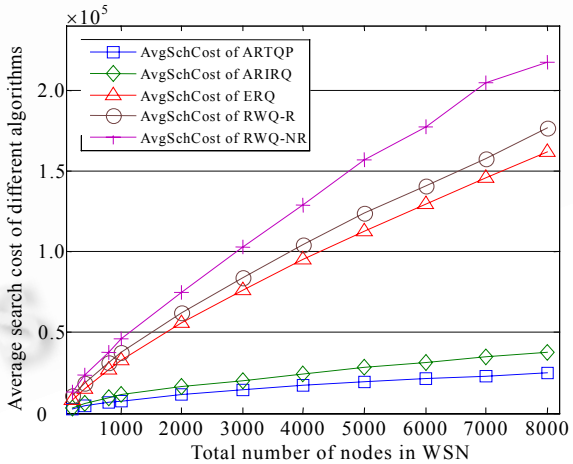

Fig.7 Comparison of average search cost between ARTQP and other algorithms

图 7 ARTQP 与其他算法平均搜索能耗比较

图 8 给出了 ARTQP 算法和其他算法的总能耗比较情况.图中 AvgTotalCost of Algorithm-*X* 表示算法 Algorithm-*X* 对应的平均总能耗值.从图 8 可以看出,5 种算法对应的总能耗趋势与图 7 中搜索能耗大致相似,这是由于搜索能耗占总能耗比重较大,很大程度上决定了总能耗趋势.大多数情况下,基于蚁群优化的 ARTQP 和 ARIRQ 算法对应的总能耗明显低于其他几种算法,这得益于蚁群算法的正反馈效应以及搜索过程中采用的动态复制策略.而 ARTQP 采用了按优先级环存储策略,一定程度上降低了搜索能耗,故总能耗比 ARIRQ 略低.

图 9 描绘了 ARTQP 算法在查询源节点数量变化情况下,平均总查询能耗的变化情况.图中 ARTQP with *X* Query Nodes 表示 ARTQP 算法在查询节点数为 *X* 时对应的总能耗值.由图 9 可知,随着查询节点的增多,总能耗

逐渐增大.这是因为查询节点的增加使得搜索路径增多,一定程度上削弱了蚁群二次复制策略的节能效应.但随着查询节点的增多,总能耗增大的速度明显变慢.其主要原因在于查询节点的增多,使得各条路径的信息素产生交互影响的概率增大,导致正反馈效应增强,在一定程度上降低了搜索能耗,使 ARTQP 总能耗增速变小.

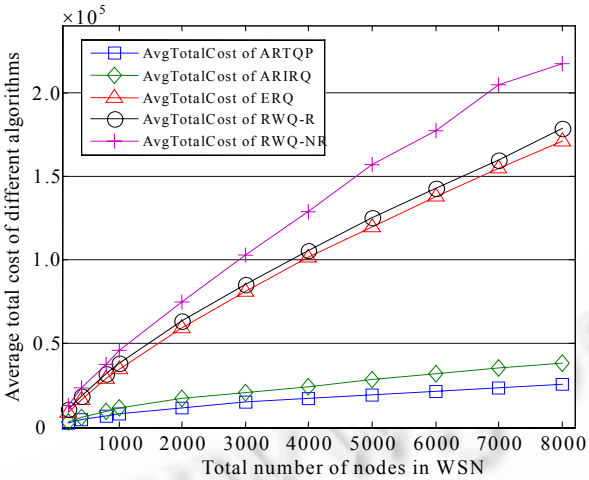

Fig.8 Comparison of average total cost between ARTQP and other algorithms

图 8 ARTQP 与其他算法平均总能耗比较

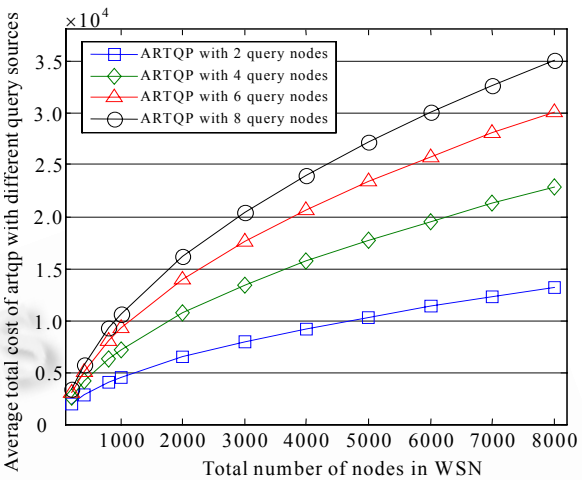

Fig.9 Average total cost of ARTQP with different number of query sources

图 9 ARTQP 平均总能耗随查询节点数变化图

4.2 ARTQP算法与其他算法查询时延比较

ARTQP 算法与其他几种算法对应的查询时延比较如图 10 所示.图中 AvgDelay of Algorithm-*X* 表示算法 Algorithm-*X* 对应的平均查询时延值.由图 10 可知,采用蚁群二次复制策略的算法(ARTQP 和 ARIRQ)对应的平均查询时延远低于其他 3 种未采用二次复制的算法对应的查询时延.这表明,二次复制策略对于降低后继查询的时延极为有效.在采用二次复制策略的两种算法中,ARTQP 算法的查询时延小于 ARIRQ 算法,这主要由于 ARTQP 算法初始复制阶段采用按优先级环存储方法,为蚂蚁初始搜索提供了一定的指导,降低了初始查询时延.其中,ARTQP 算法的平均查询时延与第 3.3 节式(20)给出的结论相符合.在未采用二次复制策略的 3 种算法中,执行初始复制的两种算法(ERQ 和 RWQ-R)比不执行任何复制的 RWQ-NR 算法时延要低,进一步表明复制策略能够较大程度地降低查询时延,提高查询处理效率.RWQ-R 算法所需查询时延略高于 ERQ 算法,主要原因在于其搜索过程具有较强的随机性.

ARTQP 算法对于不同优先级的事件数据,其查询时延区别明显,如图 11 所示.图中 AvgDelay with k=X 表示 ARTQP 算法在执行优先级等于 X 的事件查询时对应的平均时延值.由图 11 可知,事件的优先级越大,所需平均查询时延越小,与第 3.3 节的式(21)给出的结论相吻合.这是因为优先级大的事件对应的副本较多,初始存储环半径较大,有更大的可能性接近查询节点,从而使得查询时延相对较小.ARTQP 算法能够较好地满足以尽量短的时间查询到优先级高的事件这一 QoS 需求,而优先级高的事件对应的查询次数往往相对较多,故针对所有事件的平均查询时延得以显著降低.

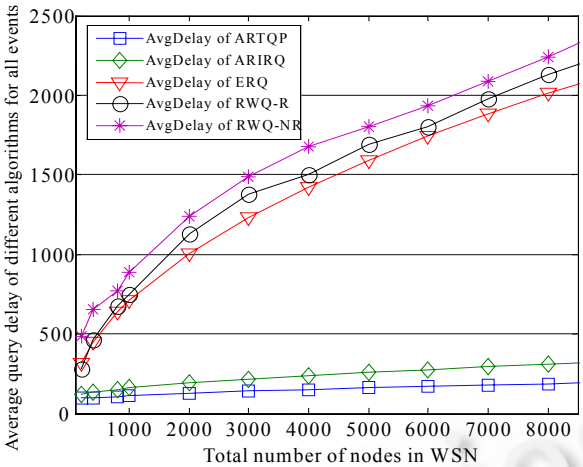

Fig.10 Comparison of average query delay between ARTQP and other algorithms

图 10 ARTQP 与其他算法平均查询时延比较

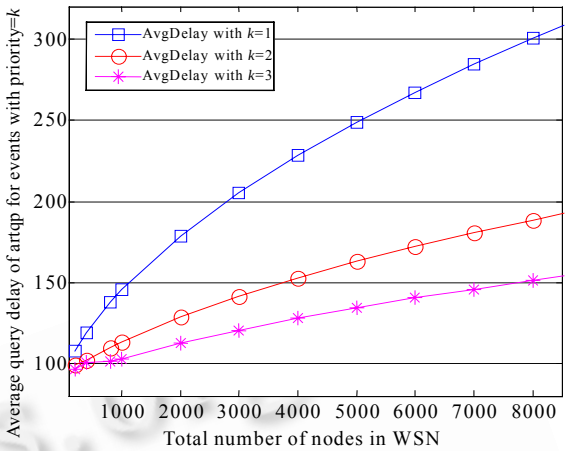

Fig.11 Average query delay of ARTQP for events with different priorities

图 11 ARTQP 查询时延随事件优先级变化图

4.3 ARTQP算法与其他算法查询成功率比较

上述针对查询能耗及时延的比较实验是在假设待查的事件数据均已产生且存储于网络某处,每只蚂蚁对应的能量约束 *ENE_R* 均值为 80 单位,时延约束 *DUR_R* 均值为 2 000 单位的前提下进行的,QoS 约束条件比较宽松.本节模拟实验设定传感器网络规模为 N=4000,查询节点 N_S=5 且位置随机产生.针对具有严格 QoS 约束的应用需求,考虑两种情况下算法的查询成功率(查询请求接受率):(1) 能量约束相对宽松(*ENE_R*=80),考察查询成功率对时延约束的敏感程度;(2) 时延约束属中等严格(*DUR_R*=800),考察查询成功率对能量约束的敏感程度.

图 12 给出了在查询时延约束变化情况下,各算法平均查询成功率变化曲线.图中 AvgDRRate of Algorithm-*X* 表示算法 Algorithm-*X* 在时延约束变化时对应的平均查询成功率.由图 12 可知,随着时延约束取值的增大,各算法查询成功率逐渐增大,这是因为约束值增大意味着查询可用更多时间搜索待查事件,搜索成功的概率理应增大.结合图 10 还可以看出,基于蚁群优化的 ARTQP 和 ARIRQ 算法查询成功率高于其他 3 种算法,而 ARTQP 算法对应的查询成功率最高.这可以理解为:在时延约束一定的情况下,查询时延较小的算法具有更高的查询成功率.

图 13 给出了在查询局部能量约束变化情况下,各算法平均查询成功率对应的变化曲线.图中 AvgERRate of Algorithm-*X* 表示算法 Algorithm-*X* 在局部能量约束变化时对应的平均查询成功率.由图 13 可知,能量约束值越大,各算法对应的查询成功率越高,这是因为约束值增大意味着查询可选择的较优邻节点数增多,其搜索成功的概率相对增大.结合图 8 还可以看出,基于蚁群算法的 ARTQP 和 ARIRQ 算法查询成功率高于其他 3 种算法,而 ARTQP 算法对应的查询成功率最高.这可以解释为:在局部能量约束一定的情况下,查询能耗较小的算法具有更高的查询成功率.总的来看,各算法的查询成功率小于图 12 给出的相应取值.这是因为图 13 的取值是在时延约束为中等严格的情况下取得的,其较严格的全局时延和局部能耗双约束特征导致各算法查询成功率相对下降.然而,ARTQP 算法因采用动态复制策略及其天然的分布式、正反馈等特征,其表现的性能较其他算法更为稳健.

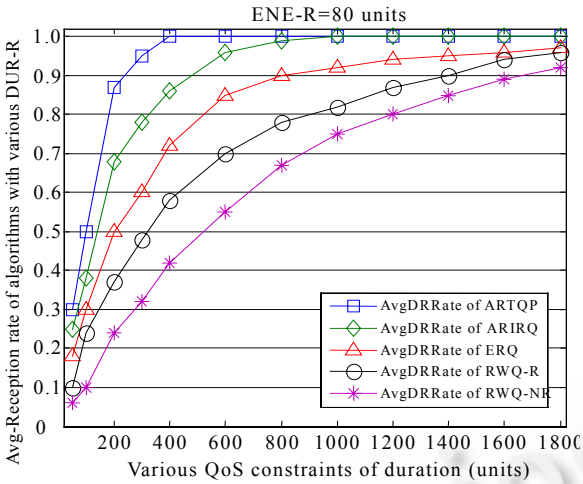

Fig.12 Average query success rate of different algorithms with various delay constraints

图 12 各算法平均查询成功率随时延约束变化图

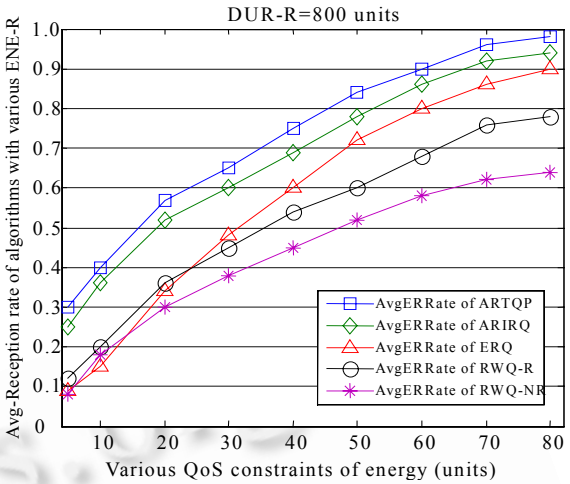

Fig.13 Average query success rate of different algorithms with various energy constraints

图 13 各算法平均查询成功率随能量约束变化图

5 总 结

无线传感器网络实时查询处理是针对该类网络实时应用的关键技术.本文考虑到传感器网络能量资源极为有限,提出了带正反馈特征的分布式多蚁群实时查询处理算法 ARTQP.在该算法中,各蚂蚁通过相互协作来调整个体行为,使群体行为具有智能性.算法具有天然的分布式及自组织特征,执行过程仅需局部环境信息即可有效找到指定数据并通过二次复制动态确定事件副本的数目及其位置,无需事件查询次数等全局信息.理论分析和实验表明,新算法具有一定智能性,与已存算法相比,其在实时性和节能性等方面性能增益明显,且网络规模越大,其性能优势越显著,在 QoS 约束较严格的情况下亦能取得良好的性能.如何改进蚁群算法,使其能够有效解决传感器网络中其他类型查询处理问题是我们下一步研究的重点.

References:

[1] Akyildiz IF, Su W, Sankarasubramaniam Y, Cayirci E. Wireless sensor networks: A survey. Computer Networks Journal, 2002,38(4):393−422.

[2] Li JZ, Li JB, Shi SF. Concepts, issues and advance of sensor networks and data management of sensor networks. Journal of Software, 2003,14(10):1717−1727(in Chinese with English abstract). http:// www.jos.org.cn/1000-9825/14/1717.htm

[3] Bonnet P, Gehrke JE, Seshadri P. Towards sensor database systems. In: Tan KL, Franklin MJ, Lui JCS, eds. Proc. of the 2nd Int'l Conf. on Mobile Data Management. Hong Kong: Springer-Verlag, 2001. 3−14.

[4] Madden SR, Franklin MJ, Hellerstein JM, Hong W. The design of an acquisitional query processor for sensor networks. In: Franklin MJ, Moon B, Ailamaki A, eds. Proc. of the SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2003. 491−502.

[5] Yao Y, Gehrke J. Query processing for sensor networks. In: Proc. (online) of the 1st Biennial Conf. in Innovative Data Systems Research. 2003. 21−32. http://www-db.cs.wisc.edu/cidr/cidr2003/program/p21.pdf

[6] Ross R, Lee WC. Decentralizing query processing in sensor networks. In: Kalis A, ed. Proc. of the 2nd Annual Int'l Conf. on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous 2005). IEEE CS Press, 2005. 270−280.

[7] Sadagopan N, Krishnamachari B, Helmy A. Active query forwarding in sensor networks (ACQUIRE). Ad Hoc Networks Journal, Elsevier Science, 2005,3(1):91−113.

[8] Madden SR, Szewczyk R, Franklin MJ, Culler D. Supporting aggregate queries over ad-hoc wireless sensor networks. In: Kindberg T, ed. Proc. of the Workshop on Mobile Computing and Systems Applications. Los Alamitos: IEEE Computer Press, 2002. 49−58.

[9] Li JZ, Guo LJ, Zhang DD, Wang WP. Processing algorithms for predictive aggregate queries over data streams. Journal of Software, 2005,16(7):1252−1261 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/16/1252.htm

[10] Ratnasamy S, Karp B, Shenker S, Estrin D, Govindan R, Yin L, Yu F. Data-centric storage in sensornets with GHT, a geographic hash table. Mobile Networks and Applications (MONET), 2003,8(4):427−442.

[11] Xin L, Young JK, Ramesh G, Wei H. Multi-Dimensional range queries in sensor networks. In: Ian A, Deborah E, eds. Proc. of the 1st Int'l Conf. on Embedded Networked Sensor Systems. New York: ACM Press, 2003. 509−517.

[12] Chang N, Liu M. Revisiting the TTL-based controlled flooding search: Optimality and randomization. In: Haas ZJ, ed. Proc. of the 10th Annual Int'l Conf. on Mobile Computing and Networks (MobiCom 2004). New York: ACM Press, 2004. 85−99.

[13] Krishnamachari B, Ahn J. Optimizing data replication for expanding ring-based queries in wireless sensor networks. In: Noubir G, ed. Proc. of the Int'l Symp. on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt 2006). Washington: IEEE CS Press, 2006. 361−370.

[14] Li GL, Gao H. A load balance data storage method based on ring for sensor networks. Journal of Software, 2007,18(5):1173−1185 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/18/1173.htm

[15] Yu JP, Lin YP, Lin M, Yi YQ. Multi-Constrained anycast routing based on ant algorithm. Chinese Journal of Electronics (CJE), 2006,15(1):133−137.

[16] Dorigo M, Gambardella LM. Ant colonies for the traveling salesman problem. BioSystems, 1997,43(2):73−81.

[17] David H, Kagan T. An introduction to collective intelligence. Technical Report, NASA tech rep NASA-ARC-IC-99-63, AAAI Press/MIT Press, 2000.

[18] Wang Y, Xie JY. Ant colony optimization for multicast routing. In: Proc. of the IEEE Asia-Pacific Conf. on Circuits and Systems on Electronic Communication Systems (APCCAS 2000). Tianjin: IEEE Press, 2000. 54−57. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=913404

[19] Dorigo M, Birattari M, Stützle T. Ant colony optimization: Artificial ants as a computational intelligence technique. IEEE Computational Intelligence Magazine, 2006,1(4):28−39.

[20] Dorigo M, Bonabeau E, Theralulaz G. Ant algorithms and stigmergy. Future Generation Computer System, 2000,16(8):851−871.

[21] Dorigo M, Blum C. Ant colony optimization theory: A survey. Theoretical Computer Science, 2005,344(2-3):243−278.

[22] Liu M, Cao JN, Chen GH, Chen LJ, Wang XM, Gong HG. EADEEG: An energy-aware data gathering protocol for wireless sensor networks. Journal of Software, 2007,18(5):1092−1109 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/18/1092.htm

[23] Krishnamachari B. Modeling data gathering in wireless sensor networks. In: Li Y, Thai MT, Wu W, eds. Book Chapter in Wireless Sensor Networks and Applications. Heidelberg: Springer-Verlag, 2005. 572−591.

[24] Avin C, Brito C. Efficient and robust query processing in dynamic environments using random walk techniques. In: Proc. of the 3rd Int'l Symp. on Information Processing in Sensor Networks (IPSN 2004). Berkeley: IEEE Signal Processing Society, 2004. 277−286.

附中文参考文献:

[2] 李建中,李金宝,石胜飞.传感器网络及其数据管理的概念、问题与进展.软件学报,2003,14(10):1717−1727. http://www.jos.org.cn/1000-9825/14/1717.htm

[9] 李建中,郭龙江,张冬冬,王伟平.数据流上的预测聚集查询处理算法.软件学报,2005,16(7):1252−1261. http://www.jos.org.cn/1000-9825/16/1252.htm

[14] 李贵林,高宏.传感器网络中基于环的负载平衡数据存储方法.软件学报,2007,18(5):1173−1185. http://www.jos.org.cn/1000-9825/18/1173.htm

[22] 刘明,曹建农,陈贵海,陈力军,王晓敏,龚海刚.EADEEG:能量感知的无线传感器网络数据收集协议.软件学报,2007,18(5):1092−1109.http://www.jos.org.cn/1000-9825/18/1092.htm

余建平(1979－),男,湖南怀化人,博士生,主要研究领域为传感器网络数据查询处理及路由算法,群集智能模型.

林亚平(1955－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为计算机网络,机器学习.