第36卷 第14期 计 算 机 工 程 2010年7月
Vol.36 No.14 Computer Engineering July 2010
·网络与通信· 文章编号：1000—3428(2010)14—0105—03 文献标识码：A 中图分类号：TP391

基于动态联盟和蚁群算法的任务协同框架

张荣雨，李士宁，李志刚，杨丽平
(西北工业大学计算机学院，西安 710129)

摘 要：单个节点能力受限，无线传感器节点需要协同完成任务。针对该问题，将协同任务分为感知子任务和计算子任务，提出基于动态联盟和蚁群算法的任务协同框架。根据应用需求选择感知节点形成初始联盟分配感知任务，当感知节点与节点总数的比值小于32%时，网络监测性能最优，引入自适应蚁群算法构建数据汇集路由树，利用同一任务数据的强相关性优化数据传输路径，从而降低通信能耗。

关键词：无线传感器网络；任务协同；动态联盟；蚁群算法

Task Collaboration Framework Based on Dynamic Coalition and Ant Colony Algorithm

ZHANG Rong-yu, LI Shi-ning, LI Zhi-gang, YANG Li-ping
(College of Computer Science, Northwestern Polytechnical University, Xi'an 710129)

【Abstract】 Single node capability is limited, node in Wireless Sensor Network(WSN) need collaborate with neighbors to complete a task. Aiming at this problem, this paper divides the collaborate task into sensor subtask and computing subtask, and proposes a task collaboration framework based on dynamic coalition and Ant Colony Algorithm(ACO). It chooses the perception node according to the application need, those nodes form an initial dynamic coalition, and assign perception task on the coalition member. When the ratio of perception nodes to the whole nodes is below 32%, the network monitoring performance is best. It adopts self-adaptive ACO to construct data aggregation routing tree, optimizes data transfer path based on strong data relation of a same task, and reduces communication energy consumption.

【Key words】 Wireless Sensor Network(WSN); task collaboration; dynamic coalition; Ant Colony Algorithm(ACO)

1 概述

由于无线传感器网络(Wireless Sensor Network, WSN)和多代理系统(Multi-Agent System, MAS)相似，因此MAS理论常被用以解决WSN中的任务协同问题[1-2]。动态联盟是多代理中常用的协同方法之一，其基本思想是当某个事件发生时形成联盟来解决由此产生的问题，当事件结束或问题解决后联盟解散[2]。动态联盟的形成分为3个阶段：联盟初始化，联盟形成和联盟确认阶段。能量受限、无人值守是WSN协同的关键技术难题。通信能耗是节点能耗的主要部分，为降低系统能耗，原始数据需经过网内汇聚处理后向上传输。蚁群算法是一种新兴的仿生优化算法，常用于WSN路由以及数据汇聚优化问题。本文提出的任务协同框架利用同一任务数据的相关性，在联盟内部基于蚁群算法形成数据上报汇聚路由。

2 WSN任务协同

WSN本质上是一个分布式系统，单个节点只能获取监测环境局部信息，WSN需在约束时间内对发生的事件做出反应，完成约定任务。WSN中任务协同的目标是：在时间约束范围内构成特定任务的各个不可或少的子任务至少被一个节点执行，各子任务在同一节点或不同节点上有秩序地被执行。节点需协同执行任务的主要原因有：

(1)单个节点的能力和资源有限[3]

1)人工部署的传感器节点能力各异，不同的传感器节点具有不同的感知能力，传感器节点上的感知模块可能会部分或全部失效，如节点A可以感知光照强度、湿度和温度，节点B因为温度传感器失效只能感知光照强度和湿度。

2)节点可能为节约能量关掉部分传感器。

3)由于内存有限，因此部分感知数据无法存储。

(2)节点之间存在依赖关系

单个节点的决策可能会影响团体中其他节点的决策，节点之间需要协商达成共识。

(3)单个节点与整体利益间的冲突

由于WSN资源受限，任务有时效要求，任务执行会有一定的能耗和时间约束，若各节点只关注自身行为的优化则可能无法满足整体需求，如各节点以最短路径传输源数据而不考虑整体能耗，会导致任务无法满足能耗约束，缩短网络的生存周期，因此各节点之间应通过协商构建一棵最小代价数据汇聚路由树，减少通信能耗，延长网络生存周期。另外，节点可能会因为处于忙碌状态无法接收新任务，需要与管理节点协商任务的迁移，优化任务分配方案。

WSN协同的基本内容[2]有：协同资源的使用，协同任务的分配与执行，协同信号及协同信息处理。协同任务的分配和执行与无线传感器网络的功能相关，指如何进行任务的描

基金项目：国家科技支撑计划基金资助项目(2007BAD79B03, 2007BAD79B02)；陕西省自然科学基金资助项目(2007F29)；陕西省科技攻关计划基金资助项目(2007K04-01)

作者简介：张荣雨(1983－)，女，硕士研究生，主研方向：无线传感器网络；李士宁，教授、博士生导师；李志刚，博士；杨丽平，硕士研究生

收稿日期：2010-01-24 **E-mail：**zhangrongyu@hotmail.com

述、分解、分配、调度和执行，包括冲突的检测与消除。任务协同的主要内容是协同任务分配。

本文借鉴并结合动态联盟和蚁群算法思想解决 WSN 中的任务协同问题。基于无回路有向图(Directed Acyclic Graph, DAG)的 WSN 任务描述如图 1 所示，任务可分解为感知和计算子任务，汇聚任务是空任务，只标识信息的汇聚。感知任务感知物理环境信息，计算任务对源数据进行计算处理形成相应结果，如平均温度等。为降低通信能耗，计算节点应尽量靠近感知节点。

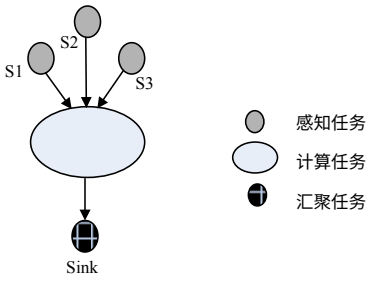

图 1　基于 DAG 图的任务协同

基于动态联盟和蚁群算法的任务协同框架的主要思想是：在联盟初始化阶段，盟主采用特定的感知节点选择算法进行静态感知任务分配；在联盟形成阶段，被选节点和盟主协商，基于自身状态和任务要求进行任务微调整，如将任务迁移到邻居节点或要求邻居节点完成部分任务等；在联盟形成阶段，联盟成员之间形成一个数据汇聚路由树，将计算任务分配在有 2 个或 2 个以上分支的节点上。各感知节点执行感知任务并依照协商好的数据汇聚路由向上传输数据，执行计算任务的节点汇聚各分支的数据并向上传输直至盟主节点，然后由盟主提交给簇头节点。

基于案例推理的动态联盟能适应动态变化的环境，具有较好的自适应性，信息库和案例库的建立和维护要求节点消耗一定能量；基于历史信息的成员选择不能保证节点当前是可用的。本文框架简化了联盟初始化和协商过程，针对动态联盟不适合大规模应用的问题，引入组织结构设计的特点，在动态分簇的基础上添加助理管理员的角色。

3　基于动态联盟和蚁群算法的任务协同框架

基于动态联盟和蚁群算法的任务协同框架如图 2 所示，其主要角色有：

(1)CH：簇头节点，担任区域管理员的角色，负责管理本区的活动，连接区域内各个节点，包括：任务分配分解，通过地址簿存储提供有关传感器节点信息、动态指定助理管理员。

(2)AM：助理管理员，辅助簇头节点进行任务分配。

(3)MN：盟主，主要功能是分配任务，由 CH 或 AM 担任。当 CH 接到监测任务时，如果 CH 当前空闲，则 CH 成为 MN，否则选择其邻居节点中能量最高者作为 AM 并与其协商请求其担任 MN 分配任务。如果协商成功，则向 AM 发送确认消息以及簇内传感器节点信息快照(节点当前状态、位置等)。

(4)SN：感知源节点。

(5)CN：执行计算任务的节点，可以是 SN。

联盟形成和任务分配的具体过程包括联盟初始化、联盟的形成以及联盟的确认。

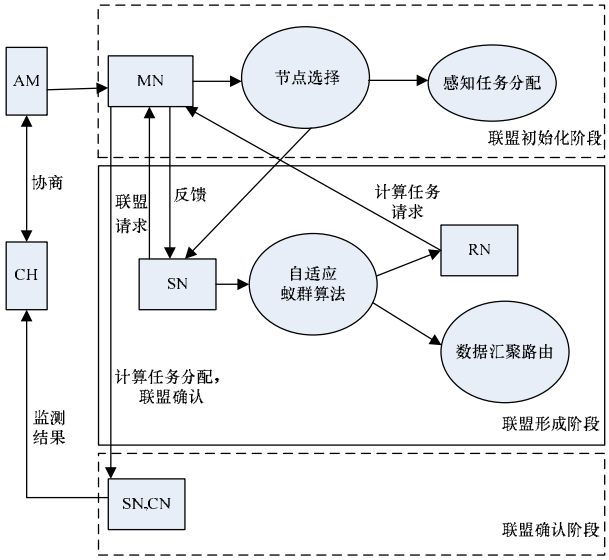

图 2　基于动态联盟和蚁群算法的任务协同框架

3.1　联盟初始化

MN 运行节点选择算法依据任务描述、当前的网络拓扑以及资源状况选择合适的感知节点形成初始联盟，可根据具体应用采用不同的节点选择算法(基于覆盖控制、感知效益等)。MN 依据联盟成员的潜在效用进行感知任务分配，该过程只在盟主节点上进行，节点之间没有任何协商过程，这样可以减少节点之间的通信量，提高联盟形成的速度。

3.2　联盟的形成

联盟的形成阶段主要是盟主和候选者协商感知任务分配以及联盟内部计算任务的协商分配过程。盟主向候选者发送感知任务请求，请求信息包括：任务类型(光照、温度等)，数据类型以及等待协商时间等。候选者接到请求后，依据自身状态决定是否接受，如节点同意协商，向盟主节点发送任务执行时间和成本预算，并附带等待盟主仲裁的时间。如果节点无法执行任务，可依据邻居节点状态向盟主推荐合适的邻居节点，盟主进一步和邻居节点协商。

感知任务确定后，基于感知任务部署结构图，在各个源节点上运行蚁群算法，构建一棵以源节点为叶节点，盟主节点为根节点的数据汇聚树，使其通信代价最小。数据汇聚树中含有 2 个以及 2 个以上分支的节点在数据传输阶段进行数据汇聚处理，执行计算任务的非源节点向盟主发送加入联盟请求，盟主回复信息为其分配计算任务。

3.3　联盟的确认

当协商过程结束后，发起者进行联盟确认，若失败则向联盟成员发送丢弃信息，则解散联盟，否则向所有联盟成员(包括新成员)发送联盟确认消息。

4　数据汇聚树的形成过程

在感知节点确定后，簇结构可以用图 $G=(v,e,\omega)$ 表示，其中，v 代表节点集；E 表示边集；ω 表示 v_i 到 v_j 的单位数据通信费用 $\omega=E_{tra}+E_{re}$，E_{tra} 和 E_{re} 分别表示单位数据接收和发送的能耗。因为无线通信半径有限，所以 G 是不完全图。设 $S\in V$ 为源节点集，CH 为簇头节点。假设汇聚后的数据包大小和输入的数据包大小相同。最优处理任务分配问题可以转换为在图 G 中构建一棵以簇头节点为根的 Steiner 树 T，使式(1)的值最小：

$$Cost=\sum_{i=1}^{n}\omega_i \tag{1}$$

采用自适应蚁群算法 Ant-Q[4]构建数据汇聚路由树。假设簇内有 N 个节点，联盟中有 S 个源节点，在每个源节点上随机部署 R 只蚂蚁，设置最大迭代次数 M ，每步循环中节点释放一只蚂蚁直至放完，可构建 R 棵树，记录本次迭代最小代价树并更新信息素，进入下次迭代。

在第 m 次迭代过程中，设 $Ant_k(r)$ 代表从第 k 个源节点出发的第 r 只蚂蚁，现位于节点 s ，其产生的部分树枝 B_k 为空，由前 $k-1$ 个节点构建的部分树为 $T_{k-1}(r)$ ，则当 $s\notin T_{k-1}(r)$ 时 $Ant_k(r)$ 选择下一个节点 u 的概率为

$$P_{s,u}(m,B_k)=\begin{cases}\arg\max\limits_{(s,v)\in且v\notin B_k}\{[t_{sv}(m)^{\alpha}[\eta_{sv}]^{\beta}\} & 若q\le q_0\\ J & 否则\end{cases} \tag{2}$$

$$J=\begin{cases}\dfrac{[\tau_{su}(m)]^{\alpha}[\eta_{su}]^{\beta}}{\sum\limits_{v\notin B_k且(s,v)\in E}[\tau_{sv}(m)]^{\alpha}[\eta_{sv}]^{\beta}} & u\notin B_k且(s,v)\in E\\ 0 & 否则\end{cases} \tag{3}$$

当 $s\in T_{k-1}(r)$ 时

$$P_{su}(m,B_s)=\begin{cases}1 & 当u是T_{k-1}(r)中s的下一跳节点\\ 0 & 其他\end{cases} \tag{4}$$

其中，q_0 是区间[0,1]上的一个随机数；$\tau_{su}(m)$ 代表第 m 次迭代中 $e(s,u)$ 上的信息素量，η_{su} 是启发因子，则 $\eta_{su}=c_{su}/\omega_{su}$ ，c_{su} 代表计算收益 $c_{su}=P_{re}/P_{tra}$ ， P_{re} 为节点 u 从节点 s 接收到的数据及其自身数据之和，并设非源节点自身数据包大小为零， P_{tra} 为节点 u 执行汇聚操作后输出数据包大小。

信息量 τ_{su} 更新如下：

$$\tau_{su}(m)\leftarrow(1-\delta)\cdot\tau_{su}(m-1)+\delta\cdot\left(\Delta\tau_{su}+\gamma\cdot\max_{v\notin B_k且(s,v)\in E}\tau_{uv}(m-1)\right) \tag{5}$$

设前 m 次迭代产生的最优化树为 T_m，其分支总数为 L_{T_m} 信息素增量为

$$\Delta\tau_{su}=\begin{cases}W/L_{T_m} & 若(s,u)\in T_m\\ 0 & 否则\end{cases} \tag{6}$$

CNSA(Computing Node Selection Algorithm)算法如下：

Step 1 初始化所有参数，令 m=0，对所有(s,u)∈E，令 $t_{su}(0)=t_0$，在每一个源节点上放置 R 只蚂蚁

Step 2 对于迭代次数 m=1,2,…,M

{对于蚂蚁 r=1,2,…,R

{对于源节点 s=1,2, …,S

{设 $Ant_k(r)$当前位置 s 为第 k 个源节点，同时设 $Ant_k(r)$当前的部分树枝 B_k 为空

当(s,u)∈E 且 s≠CH

{依照式(2)~式(4)选择下一跳节点，将弧(s,u)添加到当前枝 B_k 中，移动 $Ant_k(r)$到节点 u，并令 s=u}

更新 $T_k(r)=T_{k-1}(r)\cup B_k$

}

记录树 T(r)的代价 Cost(Tr)

}

记录到目前为止找到的最优树 T_m，并按照式(5)、式(6)进行信息素更新

}

5 仿真实验分析

5.1 仿真环境和参数设置

使用 Matlab 仿真工具模拟上述算法的实现过程。仿真参数设置如表 1 所示。

表 1 仿真参数

参数名称	参数值
仿真区域/m	100×100
节点数	50
簇头节点编号	1
簇头节点坐标	[2,100]
通信半径/m	30
传感器覆盖半径/m	20
数据包大小/bit	4 000
接收能耗/mJ	0.2
发送能耗/mJ	0.56
仿真次数	10
蚁群算法参数	n=50,R=10,M=20,q_0=0.9,α=1,β=2,γ=0.3,W=10,δ=0.1, $\tau_{uv}(0)$=0.1, J_0=1 J

能耗和网络检测性能是衡量协同任务分配机制的 2 项重要指标，获取两者间的平衡是协同任务分配的主要目的[5]。本文框架利用节点选择算法保证网络监测效率和一定的能耗，并利用同一任务的数据相关性进行网内处理降低通信能耗，一般任务执行是源数据依 SPT(Shortest Path Tree)算法上传，在数据相遇处压缩汇聚并由簇头节点执行计算任务，通信能耗比较大。

5.2 通信能耗分析

图 3 显示了选择不同数目感知节点时 CNSA 算法和 SPT 算法的通信能耗对比。从图 3 可以看出，由于 CNSA 算法利用同一任务数据的强相关性，优化数据传输路径，从而较大降低了通信能耗，同时由于 SPT 中节点沿着固定路径传输数据会造成节点能量消耗不均，降低网络生存周期。

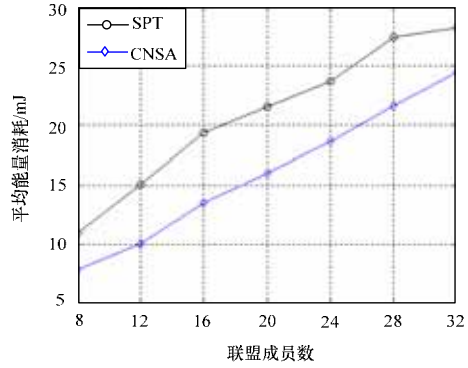

图 3 通信能耗对比

5.3 网络监测性能分析

面向监测类应用中的网络检测性能主要体现在源数据精度，普通任务执行流程为：感知节点选择→感知→路由(相遇处汇聚)→簇头执行计算任务。簇头节点与感知节点间没有协商过程，感知节点被动执行感知命令。在 WSN 中，由于网络状态随时间动态变化，因此簇头节点不能准确获悉感知节点状态，导致簇头选择了执行其他感知任务的节点，降低了监测精度。本文框架基于动态联盟，若感知节点忙，则它将拒绝盟主的请求并依据其邻居节点状态向盟主推荐替换节点，只有当其覆盖范围内全是忙节点或执行同一任务的节点时，感知任务不能被执行。本节主要从被执行的感知任务与总感知任务的比值来分析网络监测性能。

实验中随机设定 10 个忙节点，处于该状态下的节点无法执行新任务，实验运行 10 次并取其均值比较分析。图 4 显示了普通框架和任务协同框架在选择不同感知节点数目时数据精度对比。 (下转第 110 页)