考虑时延约束的无线传感器网络中继节点部署算法

马超凡^{①②③}, 郑萌^{①②}, 梁炜^{①②*}

① 中国科学院网络化控制系统重点实验室, 沈阳 110016

② 中国科学院沈阳自动化研究所, 沈阳 110016

③ 中国科学院大学, 北京 100049

* 通信作者. E-mail: weiliang@sia.cn

收稿日期: 2014-10-31; 接受日期: 2015-04-06; 网络出版日期: 2015-05-14

中国科学院战略性先导科技专项 (批准号: XDA06021000) 资助

摘要 无线传感器网络中考虑时延约束的中继节点部署 (delay constrained relay node placement, DCRNP) 问题通过部署最少数量的中继节点使得所有传感器节点与汇聚节点之间都有满足时延约束条件的通路. DCRNP 问题已经被证明是 NP-hard 问题. 本文提出基于汇合点及剪枝技术的中继节点部署 (convergence-pruning-based relay node placement, CPRNP) 算法近似求解 DCRNP 问题. CPRNP 算法分为两个阶段, 第 1 阶段找出所有可能使满足时延约束的通路相交的汇合点, 并生成一棵以汇聚节点为根节点连接所有传感器节点的最短路径树; 在第 2 阶段中 CPRNP 算法通过不断删除或替代最短路径树中的节点, 逐步缩减中继节点的部署规模. 仿真结果表明, 本文提出的 CPRNP 算法与已有算法相比可以明显地减少中继节点部署数量.

关键词 无线传感器网络 时延约束 中继节点部署 最短路径树 NP-hard 问题

1 引言

鉴于在战场侦测、环境监测、灾害应急管理、工业自动化等方面的巨大应用潜力, 无线传感器网络 (wireless sensor networks, WSNs) 于近些年受到极大的关注^[1,2]. WSNs 通过在监测区域布设大量的传感器节点来收集所需信息. 通常传感器节点比较廉价且能量受限, 这导致传感器节点通信半径较小, 需要在 WSNs 中部署中继节点以保证网络连通性及延长网络寿命^[3,4]. 这些中继节点一般比较昂贵, 且具有充足的能量及较大的通信半径, 它们的主要功能在于转发由传感器节点采集的信息. 一个有效的中继节点部署算法能够以较低的网络布设成本保证网络连通性、降低传感器节点能耗及延长网络寿命.

目前关于中继节点部署 (relay node placement, RNP) 问题的研究主要可以分为如下两类: (1) 不考虑时延的 RNP 问题^[5~13]; (2) 考虑时延的 RNP (delay constrained relay node placement, DCRNP) 问题^[14~17]. 不考虑时延的 RNP 问题在近些年被广泛而深入地研究, 而研究 DCRNP 问题的文献十分有限. 针对 DCRNP 问题, 文献^[14~16]提出基于最短路径树的算法. 该算法基于剪枝技术通过不

引用格式: 马超凡, 郑萌, 梁炜. 考虑时延约束的无线传感器网络中继节点部署算法. 中国科学: 信息科学, 2015, 45: 766–782, doi: 10.1360/N112014-00340

断修剪上一轮返回的生成树来逐步缩减中继节点的部署数量. 但这些算法只能通过删除原始生成树上的节点来节省开销, 这个缺陷使得该算法会错过不在原始生成树上且能大量节省开销的中继节点, 从而导致局部最优. 文献 [17] 提出基于分支 — 切割法的算法, 但该算法要求给定的 DCRNP 问题必须满足“任何节点的删除不能使 DCRNP 问题无解”的条件, 且其时间复杂度呈指数级增长.

针对已有算法的局部最优、严格约束等问题^[14~17], 本文提出基于汇合点及剪枝技术的中继节点部署 (convergence-pruning-based relay node placement, CPRNP) 算法. 观察到连接汇聚节点与传感器节点且满足时延约束的通路上的节点都位于一个椭圆形区域内 (该椭圆形以汇聚节点和该传感器节点为焦点, 其长轴长度与节点通信半径及延迟约束条件相关), CPRNP 算法首先搜索所有位于不同椭圆相交区域的节点, 即汇合点. 然后 CPRNP 算法执行一个迭代步骤, 在每次迭代中 CPRNP 算法通过删除或利用汇合点替代生成树上的节点来缩减生成树的规模, 直到生成树上所有节点不能被删除或替代为止, 最终的生成树给出可部署中继节点的候选部署位置. CPRNP 算法利用汇合点解决了文献 [14~16] 的局部最优缺陷, 同时 CPRNP 算法还适用于节点具有通信半径不同的异构网络, 这是文献 [14~17] 所未考虑到的. 本文最后通过大量仿真验证了 CPRNP 算法的性能.

本文主要贡献叙述如下:

- 本文首先证明对于任意传感器节点, 所有满足其时延约束的节点都位于一个以汇聚节点和该传感器节点为焦点的椭圆形区域内.
- 基于上述证明, 本文提出 CPRNP 算法, 该算法不仅能够通过删除而且能够通过替代已部署中继节点来进一步节省部署成本, 不再受已有算法^[14~16]所遭受的约束.
- 文中对算法时间复杂度进行了严格的分析, 并进行了大量仿真实验. 与之前算法相比, CPRNP 算法在百点规模时 (其中传感器节点为 100 个, 候选部署位置为 400 个) 最多可节省 9.628 (9.628/45.6047 $\approx$ 21.1%) 个中继节点.

本文结构如下: 第 2 节回顾了相关工作; 第 3 节给出问题描述; 第 4 节提出 CPRNP 算法, 并给出详细的分析描述; 第 5 节给出仿真结果及分析; 第 6 节最后总结全文.

2 相关工作

2.1 不考虑时延的 RNP 问题

不考虑时延的 RNP 问题可进一步划分为不受部署位置限制的 RNP 问题及受部署位置限制的 RNP 问题.

不受部署位置限制的 RNP 问题 文献 [5~10] 对不受部署位置约束的 RNP 问题进行深入研究. Lin 和 Xue^[5] 把单层网络模型中的 RNP 问题抽象为使用最少 Steiner 点的边长度受限的 Steiner 树问题 (Steiner tree problem with minimum number of Steiner points and bounded edge-length, STP-MSP), 并证明该问题是 NP-hard 问题. 接着作者提出一个 5 逼近的算法来解决 STP-MSP 问题. Chen 等^[6] 首先证明文献 [5] 中提出的算法其实是 4 逼近的算法, 然后基于所谓的 4 星结构提出一个 3 逼近的算法. Cheng 等^[7] 研究同一问题, 并基于 3 星结构提出 3 逼近算法. Tang 等^[8] 研究双层网络模型中的 RNP 问题, 并针对单覆盖 1 连通中继部署问题提出两个分别具有 8 和 4.5 逼近的算法, 接着作者针对双覆盖 2 连通问题提出两个分别具有 6 和 4.5 逼近的算法. Lloyd 和 Xue^[9] 首先研究单层网络模型中的 RNP 问题并提出一个 7 逼近的算法, 进一步针对双层网络模型中的 RNP 问题提出一个 $(5 + \epsilon)$ 逼近的算法, 其中 ϵ 为任意小的正实数. Wang 等^[10] 把双层网络模型中的 RNP 问题划分为两个阶段

——覆盖及连通阶段, 其研究主要集中在连通阶段并提出 3 个启发式算法来解决该问题。

受部署位置限制的 RNP 问题 文献 [11 ~ 13] 深入地研究受部署位置约束的 RNP 问题. Misra 等 [11, 12] 研究单层网络模型中的受部署位置限制的 RNP 问题并提出一个 $O(1)$ 逼近的算法. Yang 等 [13] 研究双层网络模型中受部署位置限制的 RNP 问题, 首先针对单覆盖 1 连通问题提出一个 $O(1)$ 逼近的算法, 然后针对双覆盖 2 连通问题的特殊情况提出一个 $O(1)$ 逼近的算法, 最后作者针对任意情况的双覆盖 2 连通问题提出一个 $O(\ln n)$ 逼近的算法。

2.2 DCRNP 问题

文献 [14~17] 对受时延约束的 RNP 问题进行了研究. Bhattacharya 和 Kumar [14~16] 首先证明 DCR-NP 问题是 NP-hard 问题, 并提出一 $O(n)$ 逼近的算法. 该算法基于剪枝技术通过不断修剪上一步生成的最短路径树来减小中继节点的部署数量. 如果算法第一步错过所有最优解, 将导致最差部署方案. Nigam 和 Agarwal [17] 把 DCRNP 问题归结为一类线性规划问题, 并基于分支—切割法提出一最优算法, 但是该算法只适用于“删除其中任意给定的节点或候选部署位置不能使 DC RNP 问题无解”的特殊情况, 且该算法的时间复杂度呈指数级增长, 不适用于大规模实际应用。

DCRNP 问题与受时延约束的 Steiner 树 (delay constrained Steiner tree, DCST) 问题相似. 对于给定的带权重有向图, DCST 问题为在同一分组中的节点寻求一棵以源节点为根连通所有目的节点的最小权重生成树, 并且每条由目的节点到源节点通路的端到端时延必须满足给定的时延约束. Kompella 等 [18] 最早研究 DCST 问题并提出一启发式算法. 该算法首先构建一封闭图, 然后在封闭图中寻找一棵最小生成树. 该算法要求每条边的延迟及延迟约束必须为整数, 这使得该算法无法应用于大多数实际应用场景. 在实际问题中时延可以由跳数或传输距离表示. 文献 [19] 中, Voss 将时延用跳数表示, 研究了受跳数约束的 Steiner 树问题, 并证明该问题是 NP-hard 问题, 最后作者基于禁忌搜索及最小生成树提出一启发式算法, 但是基于禁忌搜索的启发式算法无法保证时间复杂度为多项式时间. Costa 等 [20] 研究了受收益、预算及跳数约束的 Steiner 树问题, 针对该问题提出一贪婪算法来生成初始解, 并采用禁忌搜索进一步改善算法性能, 同样该算法无法保证其时间复杂度为多项式时间。

3 问题描述

数据在网络中传输的总时延通常由 4 部分组成, 分别是: 处理时延、排队时延、传输时延以及传播时延. 在 WSNs 中, 传感器节点或中继节点之间仅相距几十或数百米, 因此相较于其他 3 种时延, 传播时延对网络时延的影响较小, 可以忽略不计. 而其他 3 种时延又与路由跳数成正比, 因此在本文中使用跳数代表时延. 这样时延约束的中继部署问题就转化为跳数约束的中继部署问题 [14~16]。

对于给定的一个传感器节点集合 $S = \{s_1, s_2, \dots, s_n\}$, 一个候选部署位置 (candidate deployment location, CDL) 集合 $C = \{c_1, c_2, \dots, c_m\}$ 以及一个汇聚节点 K , DCRNP 问题从 CDL 集合中选择位置部署中继节点. 从给定信息中可构造出一幅无向图 $G = \{V, E\}$, 其中 $V = S \cup C \cup \{K\}$ 为节点集合, E 为所有边的集合. 令 $u, v \in V$ ($u \neq v$), 如 u 与 v 之间存在一条属于 E 的边, 则 u 与 v 满足下列条件:

- 如 $u \in S$ 或者 $v \in S$, 则 u 与 v 必须满足 $\|u - v\| \leq r$,
- 如 $u \notin S$ 且 $v \notin S$, 则 u 与 v 必须满足 $\|u - v\| \leq R$,

其中 $\|u - v\|$ 表示 u 与 v 之间的欧氏距离 (Euclidean distance), r 与 R ($r \leq R$) 分别表示传感器节点与中继节点的通信半径。

图 1 (网络版彩图) 椭圆形区域与 DCRNP 问题的关系示例

Figure 1 (Color online) An illustration of the relationship between the ellipse area and the DCRNP problem

定义1 (DCRNP 问题) 对于无向图 $G = \{V, E\}$, DCRNP 问题寻求 G 的一个子图 $G' = \{V', E'\}$, 其中 $V' = S \cup C' \cup \{K\}$, $C' \subseteq C$ 且 $|C'|$ 为满足下述条件的最小值: G' 中每个传感器节点至少有一条到汇聚节点 K 的通路, 且该通路要满足给定的跳数约束.

令 $P_i = \{p_{i,1}, p_{i,2}, \dots, p_{i,k_i}\}$ ($1 \leq i \leq n$) 为从传感器节点 s_i 到汇聚节点 K 的所有 k_i 条通路的集合, 其中 $p_{i,j}$ 为第 j ($1 \leq j \leq k_i$) 条连接传感器节点 s_i 与汇聚节点 K 的通路. 然后令 $P = \{P_1, P_2, \dots, P_n\}$ 为 G' 中所有连接传感器节点与汇聚节点的通路的集合. 令 $\mathcal{N}(p)$ 与 $\mathcal{P}(p)$ 分别表示通路 p 上所有节点 (不包含汇聚节点) 的集合及通路 p 上所有 CDL 的集合, 则 $|\mathcal{N}(p)|$ 表示通路 p 的跳数. 至此, DCRNP 问题可表述为一个最优化问题:

$$\text{Minimize } |C'|, \quad (1a)$$

$$\text{s.t. } \forall P_i \in P \ (1 \leq i \leq n), \quad \exists p_{i,j} \in P_i \ (1 \leq j \leq |P_i|), |\mathcal{N}(p_{i,j})| \leq \Delta_i, \quad (1b)$$

其中, Δ_i 为第 i 个传感器节点的跳数约束条件, 它表示不同区域的监视关键程度.

文献 [14] 证明 DCRNP 问题为一个 NP-hard 问题, 因此本文提出一个启发式算法近似求解 DCRNP 问题. 本文中称满足跳数约束条件的通路 $p_{i,j}$ 为传感器节点 s_i 的可行通路, 称满足式 (1b) 的图为 DCRNP 问题的可行解.

4 DCRNP

4.1 相关引理

根据 DCRNP 问题的定义可知, 如 G' 为一个可行解, 则对于任一传感器节点 s_i 则 G' 中必有一条可行通路 p 连接 s_i 与汇聚节点 K . 如图 1 所示, p 为图中边的集合, 其中虚线表示 p 中未标示出节点. 令 q (图 1 中黑色圆点) 为 p 上某个节点, 则从 q 到 s_i 的通路和从 q 到 K 的通路的跳数之和不超过 Δ_i . 由于 p 上任意相邻的两个节点之间的欧氏距离不超过 R (R 为中继节点的通信半径), 因此从 q 到 s_i 与从 q 到 K 的欧氏距离之和不超过 $\Delta_i R$, 如下式所示

$$\|q - s_i\| + \|q - K\| \leq \Delta_i R. \quad (2)$$

式 (2) 说明, 对于所有连接汇聚节点 K 与传感器节点 s_i 的通路, 其中可行通路上的节点在这样的椭圆形区域内:

- 该椭圆以汇聚节点 K 及传感器节点 s_i 为焦点;

图 2 (网络版彩图) 对于引理 1 的图例说明

Figure 2 (Color online) An illustration of Lemma 1

- 该椭圆的长轴长度等于 $\Delta_i R/2$, 短轴长度为 $\sqrt{(\Delta_i R/2)^2 - (\|K - s_i\|)^2}$.

引理1 所有连接汇聚节点 K 与传感器节点 s_i 的可行通路上的点都位于式 (2) 所定义的椭圆形区域内.

证明 采用反证法证明引理 1. 令 q 表示任意一个位于式 (2) 所示椭圆形区域外的节点, p 为一条经过 q 且连接传感器节点 s_i 的与汇聚节点 K 可行通路. 根据式 (2), 可知 q 满足下式

$$\|q - s_i\| + \|q - K\| > \Delta_i R. \quad (3)$$

令 ϵ 为一个满足下式的正实数

$$\|q - s_i\| + \|q - K\| = \Delta_i R + \epsilon. \quad (4)$$

因为每跳的最大欧氏距离为 R , 据此可以得出

$$|\mathcal{N}(p)| \geq \left\lceil \frac{\|q - s_i\| + \|q - K\|}{R} \right\rceil = \left\lceil \frac{\Delta_i R + \epsilon}{R} \right\rceil = \lceil \Delta_i + \epsilon/R \rceil \geq \Delta_i + 1, \quad (5)$$

不等式 (5) 说明 p 不能满足跳数约束条件, 这与假设 p 为可行通路矛盾. 因此可行通路经过的节点只能在式 (2) 所定义的椭圆形区域内, 至此证明完毕.

图 2 中的圆与三角分别代表了 CDL 与传感器节点. 由引理 1 可知, 只须考虑采用式 (2) 所示的椭圆区域内的点去搭建一条连接汇聚节点 K 与传感器节点 s_i 的通路. 相应的, 在图 2 中, 只须考虑采用图中椭圆内的 CDL 与传感器节点连接 K 与 s_i .

定义2 (覆盖) 令 $\mathcal{E}_i$ ($1 \leq i \leq n$) 为一个以汇聚节点 K 及传感器节点 s_i 为焦点, 且其长轴长度为 $\Delta_i R/2$ 的椭圆区域. 当某 CDL 或传感器节点位于 $\mathcal{E}_i$ 区域内时, 则称该 CDL 或传感器节点被传感器节点 s_i 的椭圆形区域 $\mathcal{E}_i$ 覆盖.

定义3 (邻居) 令所有被 $\mathcal{E}_i$ 所覆盖的 CDL 及传感器节点的集合为 $\mathcal{K}_i$. 当 $\mathcal{K}_i \cap \mathcal{K}_j \neq \emptyset$ ($i \neq j, 1 \leq i, j \leq n$) 时, 则称 s_i 与 s_j 互为邻居.

定义4 (汇合点) 如 $\bigcap_{i=1}^k \mathcal{K}_i \neq \emptyset$, 其中 $k = |A|$ ($A \subseteq S$), 则称每个在 $\bigcap_{i=1}^k \mathcal{K}_i$ 中的元素为 A 中传感器节点的汇合点.

令 G' 为一个给定 DCRNP 问题的可行解, 则 G' 中包含以汇聚节点 K 为根的生成树. 因此, 在对 DCRNP 问题求解过程中只需构建一棵以汇聚节点 K 为根连接所有传感器节点的生成树即可. 与文献 [14~16] 相同, 本文也采用最短路径树算法构造生成树.

文中称满足式 (1b) 的可行解中的生成树为可行树. 令 $T = \{p_1, p_2, \dots, p_n\}$ 为一棵可行树, 则 $\tilde{T} = \{\tilde{p}_1, \tilde{p}_2, \dots, \tilde{p}_n\}$ 为 T 删除某个 CDL 后由 T 的剩余节点生成的一棵可行树, 其中 $\tilde{p}_i$ 与 p_i 分别表示 $\tilde{T}$ 和 T 中连接传感器节点 s_i 与汇聚节点 K 的通路. 令 $\tilde{T}$ 与 T 中节点的集合分别由 $\bigcup_{i=1}^n \mathcal{N}(\tilde{p}_i) = X = \{x_1, x_2, \dots, x_N\}$ 及 $\bigcup_{i=1}^n \mathcal{N}(p_i) = Y = \{y_1, y_2, \dots, y_M\}$ 表示, 其中 N 与 M 为 $\tilde{T}$ 和 T 的节点数.

定义5 (共享率) 令 α_i 与 β_i 分别表示通过节点 x_i 与 y_i 的通路数量, 并称其为该节点的共享率. 称 $\bar{\alpha}(X)$ 及 $\bar{\beta}(Y)$ 分别为节点 $\{x_1, x_2, \dots, x_N\}$ 与 $\{y_1, y_2, \dots, y_M\}$ 的平均共享率, 其中 $\bar{\alpha}(X)$ 与 $\bar{\beta}(Y)$ 表示为

$$\bar{\alpha}(X) = \frac{\sum_{i=1}^N \alpha_i}{N}, \quad (6a)$$

$$\bar{\beta}(Y) = \frac{\sum_{i=1}^M \beta_i}{M}. \quad (6b)$$

由共享率定义可知, $\tilde{T}$ 中某节点的共享率 α 表示该节点至少被 α 个椭圆形区域覆盖, 而 $\tilde{T}$ 中所有节点的平均共享率 $\bar{\alpha}(X)$ 表示每个节点平均至少被 $\bar{\alpha}(X)$ 个椭圆形区域覆盖.

引理2 $\bar{\alpha}(X) > \bar{\beta}(Y)$.

证明 因每条通路都是由最短路径算法生成的, 且 $\tilde{T}$ 为可行树 T 删除某一 CDL 后生成的一棵可行树, 因此可知 $\forall i \in \{0, 1, \dots, n\}$, $|\mathcal{N}(\tilde{p}_i)| \geq |\mathcal{N}(p_i)|$, 即

$$\sum_{i=1}^n |\mathcal{N}(\tilde{p}_i)| \geq \sum_{i=1}^n |\mathcal{N}(p_i)|. \quad (7)$$

由共享率及 $\mathcal{N}(\bullet)$ 的定义可知

$$\sum_{i=1}^n |\mathcal{N}(\tilde{p}_i)| = \sum_{i=1}^N \alpha_i = N\bar{\alpha}(X). \quad (8)$$

同理可以得到

$$\sum_{i=1}^n |\mathcal{N}(p_i)| = \sum_{i=1}^M \beta_i = M\bar{\beta}(Y). \quad (9)$$

把式 (8) 和 (9) 代入式 (7) 中可以得到

$$N\bar{\alpha}(X) \geq M\bar{\beta}(Y). \quad (10)$$

因 $\tilde{T}$ 为 T 删除一个 CDL 后生成的可行树, 因此有

$$N < M. \quad (11)$$

要使式 (10) 和 (11) 成立, 则必有 $\bar{\alpha}(X) > \bar{\beta}(Y)$. 至此证明完毕.

图 3 通过汇合点节省中继节点部署数量示例

Figure 3 An example of how to save deployed relay nodes by using covergences

由引理 2 可知, 如一可行树 T 中的 CDL 能被删除, 则删除后生成的可行树 $\tilde{T}$ 中节点的平均共享率大于 T 中节点的平均共享率. 在剪枝过程中, 初始最短路径树上的 CDL 被不断删除, 这导致可行树中节点的平均共享率不断增大, 即平均通过每个节点的通路增多, 但是每个 CDL 所能通过的可行通路数量 (即覆盖该 CDL 的椭圆形区域个数) 有限, 这说明可行树中 CDL 平均所能通过的通路越多, 能够进行的删除操作可能就越多, 也就是说能够节省的中继节点可能越多. 因此, 为了保证可行树中的 CDL 能够通过较多的通路, 在删除过程中, 应首先考虑删除可行树中被较少椭圆形区域覆盖的 CDL.

引理 3 如用汇合点 q 替代可行树中某个 CDL 后, 使可行通路 $p_1, p_2, \dots, p_i$ 相交于 q , 为构造一棵可行树则 $p_1, p_2, \dots, p_i$ 应在 q 合并, 即只有一条通路连接 q 与汇聚节点.

证明 若存在多条通路连接 q 与汇聚节点, 则说明 q 与汇聚节点间存在一条环路, 即这几条通路不属于一棵树, 因为要构造一棵可行树, 因此 $p_1, p_2, \dots, p_i$ 必在 q 合并. 至此证明完毕.

由引理 3 可知, 当采用汇合点替代可行树中 CDL 造成几条通路相交于该汇合点时, 必须使这几条通路在该汇合点合并.

如图 3 所示, 原始可行树由线段表示, 经过节点删除或替代后生成的可行树由点线段表示. 由于 c_1 只被一个椭圆形区域覆盖且删除 c_1 后, 仍可用原始可行树中的剩余节点生成一棵可行树, 根据引理 2 应删除 c_1 ; 当算法采用 c_4 替代 c_2 和 c_3 后, p_1 与 p_2 相交于 c_4 , 根据引理 3 可知, 应使这两条通路在 c_4 合并, 其中点划线表示合并前的通路. 由图可知, 通过删除及替代一共可以节省 5 个中继节点.

4.2 算法描述

CPRNP 算法把给定的传感器节点分为无邻居和有邻居两类, 并针对各自情况设计了相应的算法. 因此 CPRNP 算法由这两部分算法组成.

4.2.1 传感器节点无邻居情况

令 G^* 为给定 DCRNP 问题的一个最优解, T^* 为 G^* 中以汇聚节点为根的生成树, p_i^* 为 T^* 中连接汇聚节点 K 与传感器节点 s_i 的一条通路. 同时令 $\bar{p}_i$ 为一条由最短路径算法生成的连接汇聚节点 K 与传感器节点 s_i 的通路. 假设 s_i 没有邻居, 则根据引理 1 可知, 所有连接汇聚节点 K 与传感器节点 s_i 的可行通路不能经过被其他传感器节点的椭圆形区域覆盖的节点. 加之 $p_i^* \in T^*$, 可以得到如下

结论:

$$|\mathcal{P}(\bar{p}_i)| \geq |\mathcal{P}(p_i^*)|. \quad (12)$$

若有 $|\mathcal{P}(\bar{p}_i)| < |\mathcal{P}(p_i^*)|$, 则可用 $\bar{p}_i$ 在 T^* 中替代 p_i^* 生成一棵比 T^* 使用更少 CDL 的可行树, 这与 T^* 为最优解矛盾, 因此式 (12) 成立.

另一方面, 由于 $\bar{p}_i$ 是由最短路径算法生成的, 同时可以得到

$$|\mathcal{C}(\bar{p}_i)| \leq |\mathcal{C}(p_i^*)|. \quad (13)$$

不等式 (12) 和 (13) 说明对于没有邻居的传感器节点来说, 尽管使用最短路径算法可以生成跳数最少的通路, 但是不能保证所生成的通路部署了最少的中继节点. 因此对于没有邻居的传感器节点, 可以通过不断修剪由最短路径算法生成的通路来达到节省中继节点部署的目的. 通路修剪算法具体描述见算法 1. 算法中 $\text{ShortestPath}(u, v, C)$ 表示一条在传感器、汇聚节点及 CDL 集合 C 上由最短路径算法生成的连接节点 u 与 v 的通路.

算法 1 Path pruning algorithm

Require: sensor node s_i without neighbors, the CDL set C , the sink K ;

Ensure: a feasible path connecting K and s_i ;

```

1:  $p = \text{ShortestPath}(s_i, K, C)$ ;
2:  $\text{tmpND} = \mathcal{N}(p) - s_i$ ;
3:  $\text{tmpN} = \text{the first node in set tmpND}$ ;
4: while ( $\text{tmpN} \neq \text{Null}$ ) do
5:    $\text{tmpC} = C - \text{tmpN}$ ;
6:    $\text{tmpP} = \text{ShortestPath}(s_i, K, \text{tmpC})$ ;
7:   if ( $(\text{tmpP} \neq \text{Null}) \ \&\&(|p| > |\text{tmpP}|)$ ) then
8:      $p = \text{tmpP}$ ;
9:      $\text{tmpND} = \mathcal{N}(p) - s_i$ ;
10:     $\text{tmpN} = \text{the first node in set tmpND}$ ;
11:   else
12:      $\text{tmpN} = \text{the next node in set tmpND}$ ;
13:   end if
14: end while
15: return  $p$ ;
```

4.2.2 传感器节点有邻居情况

对于有邻居的传感器节点, 为使生成的最短路径树中含有较少的 CDL, 即生成树中可被删除的节点较少, 由引理 2 可知, 应使生成的通路尽可能经过汇合点, 即采用被较多传感器节点的椭圆形区域覆盖的 CDL. 基于这种思路, 本文提出 CPRNP 算法来近似求解 DCRNP 问题.

本文提出的 CPRNP 算法包括两个阶段: 检测阶段与替代阶段.

4.2.3 检测阶段

由最短路径树算法生成的树中, 每条连接根节点与其他节点的通路具有最少的跳数, 因此可以通过构建一棵最短路径树来检查所选择的 CDL 与给定的传感器节点能否生成一棵以汇聚节点为根的可

行树. 因此, 针对有邻居的传感器节点, CPRNP 算法采用最短路径树算法来生成并检测算法当前所做的选择是否合适.

在检测阶段, 首先检测在给定的传感器节点、CDL、汇聚节点以及跳数约束条件下, 能否采用最短路径树算法生成一棵以汇聚节点为根节点的可行树 T . 如不能生成 T , 则宣告 DCRNP 问题无解. 如能够生成 T , 则 CPRNP 算法接着检测是否存在一棵不包含 CDL 的可行树. 为此, 尝试只使用传感器节点与汇聚节点构建一棵最短路径树 T' , 如能生成 T' 且满足跳数约束条件, 则输出 T' 并结束运算, 否则进入检测阶段的下一步.

在这一步中, 首先从给定的传感器节点中找出没有邻居的传感器节点, 然后采用算法 1 连接这些传感器节点与汇聚节点. 此步中采用算法 1 生成的通路被标记为锁定状态且不代入 CPRNP 算法的下一阶段. 同时, 找出并记录所有汇合点.

4.2.4 替代阶段

替代阶段大致划分为两个子阶段. 在第 1 子阶段中, 由检测阶段返回的可行树被逐步修剪. 然后在第 2 子阶段中, 不断地采用汇合点替代可行树中的 CDL, 以此来进一步减少可行树中的 CDL.

第 1 子阶段与文献 [14~16] 中提出的算法 SPTiRP 相似, 但是两者的关键差异在于对 CDL 及通路权重的定义. 由引理 2 可知, 可使构建的通路尽可能多地共享其所经过的汇合点, 以此来减少结果中包含的 CDL. 为此在 CPRNP 算法中 CDL 的权重定义为覆盖该 CDL 的传感器节点数量, 而通路的权重相应地定义为通路上所有 CDL 的权重之和. CPRNP 算法将通路及每条通路中的 CDL 按升序排列. 每次修剪过程都从权重最小的通路开始, 并从该通路中权重最小的 CDL 开始修剪. 由于修剪过程与 SPTiRP 相似, 具体步骤可以参照文献 [14~16].

对于一条连接汇聚节点 K 与传感器节点 s_i 的通路, 把从 K 到 s_i 的方向成为下方. 令 q 为这条通路上的一个节点, 则与 q 相邻并在其下方的节点 $\bar{q}$ 称为 q 的子节点, 同时 q 称为 $\bar{q}$ 的父节点. 如图 1 所示, q 的子节点为 c_h , 同时 q 为 c_h 的父节点.

首先引入满足条件 1 与条件 2 的节点, 以此引出替代阶段的第 2 子阶段算法.

条件 1 该节点能够与其他通路中的 CDL 或传感器节点直接通信;

条件 2 该节点存在合格 1 跳邻居汇合点, 其中一个合格 1 跳邻居汇合点, 首先必须为非本条通路上的汇合点, 其次该汇合点必须能够与其他通路上的 CDL 或传感器节点直接通信.

对于满足条件 1 的节点, 说明多条通路相交于该节点, 根据引理 3, 应在该节点使这几条通路合并. 可通过删除其父节点, 并在剩余节点上生成一棵最短路径树来检测其父节点的删除能否产生一棵可行树, 具体步骤如算法 2 中 7~15 行所示.

对于满足条件 2 的节点, 其合格 1 跳邻居汇合点可能比其父节点被更多传感器节点的椭圆形区域覆盖. 根据引理 2, CPRNP 算法试图用其合格 1 跳邻居汇合点替代其父节点. 具体步骤如算法 2 中 16~34 行所示.

第 2 子阶段中, CPRNP 算法不断删除满足条件 1 的节点的父节点, 且用合格 1 跳邻居汇合点替代满足条件 2 的节点的父节点, 以此逐步缩减中继节点的部署数量. 第 2 子阶段具体描述如算法 2 所示. 算法 2 中 $SPT(C, S, K)$ 表示一棵在 CDL 集合 C 、传感器节点集合 S 及汇聚节点 K 上生成的最短路径树, 该树以汇聚节点为根节点且连接所有传感器节点.

如图 4 所示, 其中所有黑色线段表示前一步所生成的可行树, 而不同样式的虚线表示当一些 CDL 被删除或替代后新生成的可行树. 在检测 c_2 时可知, s_1 是 c_2 的子节点而 s_1 的 1 跳邻居为 s_2 , 因此

算法 2 The second substage

Require: the shortest path tree T returned in the first substage, the sensor nodes set $\mathbb{S}$, the set of convergences $\mathbb{C}$, the sink K ;

Ensure: a feasible tree $\mathbb{T}$;

```

1:  $\mathbb{T} = T$ ,  $p$  = the least weighted path in  $\mathbb{T}$ ;
2: while ( $p \neq \text{Null}$ ) do
3:    $\text{tmpN} = \mathcal{P}(p)$ , and sort the nodes in  $\text{tmpN}$  in a descending order according to their hop counts to the sink;
4:    $\text{tmp} =$  the CDL with the largest hop count in  $\text{tmpN}$ ;
5:   while ( $\text{tmp} \neq \text{Null}$ ) do
6:      $\text{child} =$  the child node of  $\text{tmp}$ ;
7:     if ( $\text{child}$  fulfils condition 1) then
8:        $TN =$  all the CDLs in  $\mathbb{T}$ ;
9:        $\text{tmpTN} = TN - \text{tmp}$ ;
10:       $\text{tmpT} = \text{SPT}(\text{tmpTN}, \mathbb{S}, K)$ ;
11:      if ( $\text{tmpT} \neq \text{Null}$ ) then
12:         $\mathbb{T} = \text{tmpT}$ ;
13:         $p =$  the least weighted path in  $\mathbb{T}$ , break;
14:      end if
15:    end if
16:    if ( $\text{child}$  fulfils condition 2) then
17:       $\text{neighbor} =$  all the qualified 1-hop neighbor convergences of  $\text{child}$  in  $\mathbb{C}$ ;
18:      if ( $\text{neighbor} \neq \text{Null}$ ) then
19:         $\text{tmpNei} =$  the node with the largest weight in  $\text{neighbor}$ ;
20:        while ( $\text{tmpNei} \neq \text{Null}$ ) do
21:           $TN =$  all the CDLs in  $\mathbb{T}$ ;
22:           $\text{tmpTN} = (TN - \text{tmp}) \cup \text{tmpNei}$ ;
23:           $\text{tmpT} = \text{SPT}(\text{tmpTN}, \mathbb{S}, K)$ ;
24:          if ( $\text{tmpT} \neq \text{Null}$ ) then
25:            break;
26:          end if
27:        end while
28:      end if
29:      if ( $\text{tmpT} = \text{Null}$ ) then
30:         $\text{tmp} =$  the next CDL with the largest hop count in  $p$ , continue;
31:      else
32:         $\mathbb{T} = \text{tmpT}$ ;
33:         $p =$  the least weighted path in  $\mathbb{T}$ , break;
34:      end if
35:    else
36:       $\text{tmp} =$  the next CDL with the largest hop count in  $p$ ;
37:    end if
38:  end while
39:   $p =$  the least weighted path in  $\mathbb{T}$ ;
40: end while
41: return  $\mathbb{T}$ ;

```

图 4 (网络版彩图) 对于第二子阶段中条件 1 和 2 的说明

Figure 4 (Color online) An illustration for the conditions 1 and 2 in the second substage

当 c_2 被删除后, 可以生成一棵新的可行树, 在图中由短划线表示, 其中 s_1 与 s_2 在同一条通路中且可以节省两个中继节点.

如图 4 所示 c_4 的子节点为 s_3 , 当 c_4 被 c_3 替代后, 同样可以生成一棵可行树, 图中由点线表示, 并且通过替代可以节省两个中继节点.

尽管可以通过替代上一步选择的 CDL 来减少中继节点部署量, 但是对于距离通路上节点大于 1 跳的汇合点来说, 替代过程需要多个 CDL 参与, 且很难确定哪些 CDL 应该和这个汇合点一起参与替代. 如图 4 所示, 令 p_3 (p_4) 表示连接 s_3 (s_4) 与汇聚节点的通路, c_4 是 c_5 的子节点, 但是 c_4 没有合格的 1 跳邻居汇合点. 如图中点 — 划线所示, 尽管通过使 p_3 与 p_4 在 c_9 汇合可以节省一个中继节点, 但是如何令 p_3 与 p_4 在 c_9 汇合十分困难, 因为它本质上仍是一个 DCRNP 问题, 因此本文提出的 CPRNP 算法仅搜索 1 跳邻居汇合点.

4.3 算法复杂度分析

令 $N = |C| + |S|$, 则最短路径树算法的时间复杂度为 $N \lg N$ [21], 其中函数 $\lg$ 表示以 10 为底的对数 $\log_{10}$. 首先给出通路修剪算法的时间复杂度, 通路修剪算法每试图删除一个通路上的节点就需执行一次最短路径树算法, 而通路上至多有 N 个节点, 因此算法 1 的一次迭代复杂度为 $O(N^2 \lg N)$, 但是一旦某节点成功删除需从头执行算法 1, 而算法 1 至多被执行 N 次, 最终算法 1 的时间复杂度为 $O(N^3 \lg N)$.

在检测阶段需生成两次最短路径树, 因此这个阶段的时间复杂度为 $O(N \lg N)$. 寻找汇合点需要至多 $|S||C| \leq N^2/4$ 次运算.

根据文献 [14~16] 可知, 替代阶段的第 1 个子阶段需要 $O(N^3 \lg N)$ 次运算. 在第 2 子阶段中, 至多需要进行 N 次替代, 即至多需要 $N^2 \lg N$ 次运算. 当一次替代成功需重新开始替代, 因此第 2 子阶段的时间复杂度为 $O(N^3 \lg N)$. 最终, CPRNP 算法的时间复杂度 T_{CPRNP} 为

$$T_{\text{CPRNP}} = O(N^3 \lg N) + O(N \lg N) + O(N^2) + O(N^3 \lg N) = O(N^3 \lg N), \quad (14)$$

式 (14) 说明 CPRNP 算法的时间复杂度与文献 [14~16] 所提出算法相同.

图 5 同构情况下, 不同算法的输出结果 ($r = R = 10 \text{ m}$, $\Delta = 15$)

Figure 5 The results returned by different algorithms in the homogeneous network, where $r = R = 10 \text{ m}$, $\Delta = 15$. (a) Shortest path tree; (b) SPTiRP; (c) CPRNP

图 6 异构情况下, 不同算法的输出结果 ($r = 10 \text{ m}$, $R = 15 \text{ m}$, $\Delta = 15$)

Figure 6 The results returned by different algorithms in the heterogeneous network, where $r = 10 \text{ m}$, $R = 15 \text{ m}$, $\Delta = 15$. (a) Shortest path tree; (b) SPTiRP; (c) CPRNP

5 仿真及其结果

考虑一个 $100 \text{ m} \times 100 \text{ m}$ 的正方形部署区域. 为确保生成的 DCRNP 问题有解, 部署区域被划分为连续的边长为 5 的方格, 并在每个方格中随机部署 1 个 CDL. 因此 CDL 一共为 400 个. 在仿真中, 传感器节点随机放置, 数量从 5 变化到 100. 仿真同样考虑到通信半径对算法的影响, 首先对同构 (即网络中传感器节点与中继节点通信半径相同) 情况进行仿真, 然后测试在异构 (即网络中传感器节点与中继节点通信半径不同) 情况下的算法效果. 本文将 CPRNP 算法与文献 [14~16] 提出的 SPTiRP 算法进行对比. 为了保证对比的公平性, 本文在每个传感器数量下分别进行了 100 组仿真比较, 并对结果取平均值. 鉴于文献 [14~16] 提出算法只针对同构网络, 对比仿真中对 SPTiRP 算法进行修改使其能够应用于异构网络. 在仿真实验中, 为了方便对比且不失一般性, 传感器节点具有相同的时延约束, 即 $\forall i \in \{1, 2, \dots, n\}$, $\Delta_i = \Delta$.

5.1 输出 CDL 拓扑对比

针对仿真中生成的网络拓扑任意选取一组进行对比, 其中图 5 和 6 展示了同构与异构情况下两种

图 7 (网络版彩图) SPTiRP 算法与 CPRNP 算法中继节点部署数量对比 ($r = R = 20$ m)

Figure 7 (Color online) A comparison between the SPTiRP and the CPRNP on the deployed relay nodes ($r = R = 20$ m). (a) The number of deployed relay nodes by the SPTiRP and the CPRNP under different Δ s; (b) the difference of the deployed relay nodes between the SPTiRP and the CPRNP under different Δ s

图 8 (网络版彩图) SPTiRP 算法与 CPRNP 算法中继节点部署数量对比 ($r = R = 10$ m)

Figure 8 (Color online) A comparison between the SPTiRP and the CPRNP on the deployed relay nodes ($r = R = 10$ m). (a) The number of deployed relay nodes by the SPTiRP and the CPRNP under different Δ s; (b) the difference of the deployed relay nodes between the SPTiRP and the CPRNP under different Δ s

算法的对比。两幅图显示不同算法对同一给定的传感器节点集合、CDL 集合及汇聚节点的返回结果, 图中方块为汇聚节点, 圆点为 CDL, 星型为传感器节点。图 5 和 6 分别为最短路径树算法、SPTiRP 算法及 CPRNP 算法返回的结果。图 5(b) 和 6(b) 中可以看出 SPTiRP 输出结果中的 CDL 只能是那些被原始最短路径树算法选中的 CDL, 其中被最短路径树选中的 CDL 如图 5(a) 和 6(a) 所示。相比之下, CPRNP 算法的输出结果中包含合适的汇合点, 且这些汇合点不必被原始最短路径树算法选中, 如图 5(c) 和 6(c) 所示。这导致 SPTiRP 算法在图 5(b) 和图 6(b) 中部署的中继节点数量分别为 30 与 24, 而 CPRNP 算法在图 5(c) 和 6(c) 中部署的中继节点数量分别为 26 与 20。

5.2 算法性能

图 7 和 8 显示了同构情况下通信半径及跳数约束条件对算法的影响。图 7 中通信半径设置为 $r = R = 20$ m。显然 CPRNP 算法在各个条件下的性能都要优于 SPTiRP 算法, 其中 CPRNP 算法最多比 SPTiRP 算法节省 1.8511 ($1.8511/9.8511 \approx 18.8\%$) 个中继节点。图 8 中通信半径设置为 $r = R = 10$ m, 两种算法 SPTiRP 及 CPRNP 的仿真结果如图 8 所示, CPRNP 算法最多比 SPTiRP

图 9 (网络版彩图) SPTiRP 算法与 CPRNP 算法中继节点部署数量对比 ($r = 10$, $R = 15$ m)

Figure 9 (Color online) A comparison between the SPTiRP and the CPRNP on the deployed relay nodes ($r = 10$ m, $R = 15$ m). (a) The number of deployed relay nodes by the SPTiRP and the CPRNP under different Δ s; (b) the difference of the deployed relay nodes between the SPTiRP and the CPRNP under different Δ s

算法节省 9.628 ($\approx 21.1\%$) 个中继节点.

通过对比图 7 与 8 可知, 随着节点通信半径的减小, CPRNP 算法比 SPTiRP 算法节省的中继节点数增加. 这种现象可以通过以下描述来解释: SPTiRP 算法只能通过删除可以与其他通路节点直接通信的 CDL 来缩小中继节点部署规模, 而随着通信半径的减小, 不同通路上各节点之间的跳数在增加, 这样就导致 SPTiRP 算法所能删除的 CDL 数量大幅减少. 相比 SPTiRP, CPRNP 则可以通过选择 1- 跳邻居汇合点来抵消通信半径减小带来的影响.

如图 7(a) 及 8(a) 所示, 这两个子图显示随着跳数约束条件的松弛, 这两种算法部署的中继节点数量会随之减少. 这主要由于随着约束条件的松弛, 算法会寻找那些包含较少 CDL 的通路来代替现有通路, 即使这些通路的跳数会大于现有通路.

异构情况下两种算法的对比如图 9 所示. 可以看出, 在异构情况下, 节点通信半径及跳数约束条件对两种算法的影响与同构情况一致, 即节点通信半径越小, CPRNP 算法越优于 SPTiRP 算法, 且两种算法执行时间越长, 跳跃约束条件越松弛, 两种算法部署的中继节点越少.

5.3 算法执行时间

SPTiRP 算法与 CPRNP 算法的运行时间如图 10 所示. 由图 10(a) 及 10(c) 可知, 在同构或异构情况下, CPRNP 算法的运行时间为 SPTiRP 算法运行时间的 2 至 3 倍, 其中 CPRNP 算法最长执行时间为 2228.5 ms, 这只是 SPTiRP 算法执行时间的 1.733 倍, 且当节点通信半径增加, 即 $r = R = 20$ m 时, 如图 10(b) 所示, CPRNP 算法的运行时间几乎与 SPTiRP 算法运行时间相同, 这也验证了 CPRNP 算法与 SPTiRP 算法有着相同的时间复杂度. 另外可以看出, 当跳数约束条件松弛时, 两个算法的运行时间都会随之增加, 这说明随着约束地松弛, 两种算法都会尝试更多的通路来节省中继节点的部署数量.

6 总结

本文提出 CPRNP 算法来解决 DCRNP 问题. 与之前的算法不同, CPRNP 算法首先找出所有汇合点; 然后执行迭代剪枝, 每次通过删除或利用汇合点替代最短路径树中的节点来缩减部署中继节

图 10 SPTiRP 算法与 CPRNP 算法运算时间对比

Figure 10 A comparison between the SPTiRP and the CPRNP on the running time. (a) $r = R = 10$ m; (b) $r = R = 20$ m; (c) $r = 10$ m, $R = 15$ m

点的数量, 直到最短路径树上的节点不能再被删除或替代为止, 将所生成最短路径树作为最终结果输出. 仿真结果显示, 相较于已有算法, CPRNP 算法能够在可容忍运行时间内显著节省中继节点的部署规模.

虽然本文提出的 CPRNP 算法可以有效地减少中继节点部署数量, 但同样也会增加剩余中继节点的能耗开销, 因此在后继工作中将考虑同时面向时延和网络生命周期的中继节点部署算法.

参考文献

- 1 Akyildiz I F, Su W, Sankarasubramaniam Y, et al. Wireless sensor networks: a survey. *Comput Netw J*, 2002, 38: 393–422
- 2 Estrin D, Govindan R, Heidemann J, et al. Next century challenges: scalable coordination in sensor networks. In: *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, New York, 1999. 263–270
- 3 Yick J, Mukherjee B, Ghosal D. Wireless sensor network survey. *Comput Netw*, 2008, 52: 2292–2330
- 4 Bao X R, Zhang S, Xue D Y, et al. An improved centralized voronoi tessellation algorithm for wireless sensor network coverage problem. *Inf Control*, 2009, 38: 620–623
- 5 Lin G, Xue G. Steiner tree problem with minimum number of steiner points and bounded edge-length. *Inf Proc Lett*, 1999, 69: 53–57
- 6 Chen D, Du D Z, Hu X D, et al. Approximations for steiner trees with minimum number of steiner points. *J Global Optim*, 2000, 18: 17–33
- 7 Cheng X Z, Du D Z, Wang L S, et al. Relay sensor placement in wireless sensor networks. *Wirel Netw*, 2007, 14: 347–355
- 8 Tang J, Hao B, Sen A. Relay node placement in large scale wireless sensor networks. *Comput Commun*, 2006, 29: 490–501
- 9 Lloyd E, Xue G. Relay node placement in wireless sensor networks. *IEEE Trans Comput*, 2007, 56: 134–138
- 10 Wang Q, Xu K, Takahara G, et al. Device placement for heterogeneous wireless sensor networks: minimum cost with lifetime constraints. *IEEE Trans Wirel Commun*, 2007, 6: 2444–2453
- 11 Misra S, Hong S, Xue G, et al. Constrained relay node placement in wireless sensor networks to meet connectivity and survivability requirement. In: *Proceedings of IEEE INFOCOM*, Phoenix, 2008. 879–887
- 12 Misra S, Hong S, Xue G, et al. Constrained relay node placement in wireless sensor networks: formulation and

- approximations. *IEEE/ACM Trans Netw*, 2010, 18: 434–447
- 13 Yang D J, Misra S, Fang X, et al. Two-tiered constrained relay node placement in wireless sensor networks: computational complexity and efficient approximations. *IEEE Trans Mobile Comput*, 2012, 11: 1399–1411
  - 14 Bhattacharya A, Kumar A. Delay constrained optimal relay placement for planned wireless sensor networks. In: *Proceedings of IEEE IWQoS*, Beijing, 2010. 1–9
  - 15 Bhattacharya A, Kumar A. A shortest path tree based algorithm for relay placement in a wireless sensor network and its performance analysis. *Comput Netw*, 2014, 71: 48–62
  - 16 Bhattacharya A, Kumar A. QoS aware and survivable network design for planned wireless sensor networks. Technical report, arxiv.org/pdf/1110.4746. 2013
  - 17 Nigam A, Agarwal Y K. Optimal relay node placement in delay constrained wireless sensor network design. *Eur J Oper Res*, 2014, 233: 220–233
  - 18 Kompella V P, Pasquale J C, Polyzos G C. Multicast routing for multimedia communication. *IEEE/ACM Trans Netw*, 1993, 1: 286–292
  - 19 Voss S. The steiner tree problem with hop constraints. *Ann Oper Res*, 1999, 86: 321–345
  - 20 Costa A M, Cordeau J F, Laporte G. Fast heuristics for the steiner tree problem with revenues, budget and hop constraints. *Eur J Oper Res*, 2008, 190: 68–78
  - 21 Cormen T H, Leiserson C E, Rivest R L, et al. *Introduction to Algorithm*. London: The MIT Press and McGraw-Hill, 2001. 40–50

Delay constrained relay node placement in wireless sensor networks

MA ChaoFan^{1,2,3}, ZHENG Meng^{1,2} & LIANG Wei^{1,2*}

¹ Key Laboratory of Networked Control Systems, Chinese Academy of Sciences, Shenyang 110016, China;

² Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China;

³ University of Chinese Academy of Sciences, Beijing 100049, China

*E-mail: weiliang@sia.cn

Abstract The delay constrained relay node placement (DCRNP) problem minimizes the quantity of deployed relay nodes that are employed to build at least one path between a sink and each sensor node and guarantees that the delay constraints for the built paths are fulfilled. It has been proven that the DCRNP problem is NP-hard. This paper proposes a convergence-pruning-based relay node placement (CPRNP) algorithm to approximately solve the DCRNP problem. The CPRNP algorithm consists of two stages. In the first stage, CPRNP identifies the convergences of all the paths meeting the delay constraint and forms a shortest-path tree that is rooted at the sink and connects all the sensors. In the second stage, the CPRNP gradually reduces the number of deployed relay nodes by deleting or substituting the nodes on the shortest-path tree. The simulation results confirm that CPRNP can significantly save deployed relay nodes compared to existing algorithms.

Keywords wireless sensor networks, delay constraint, relay node placement, shortest-path tree, NP-hard problem

MA ChaoFan was born in 1985. He received a B.S. degree in Electrical Engineering and Automation from Henan Polytechnic University, China in 2008. He received an M.S. degree in Circuits and Systems from Dalian University of Technology, China in 2013. He is currently working toward a Ph.D. degree in Detection Technology and Automatic Equipment at Shenyang Institute of Automation, Chinese Academy

of Sciences. His research interests are in wireless sensor networking, and embedded systems.

ZHENG Meng was born in 1983. He received a B.S. degree in information and computing science and an M.S. degree in operational research and cybernetics from Northeastern University, Shenyang, P.R. China and a Ph.D. degree in mechatronic engineering from Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, P.R. China in 2005, 2008, and 2012, respectively. He was a visiting student

from 2010 to 2012 at the Fraunhofer Institute for Telecommunication, Heinrich-Hertz-Institute, Berlin, Germany. Currently, he is serving as an associate professor at the Key Laboratory of Networked Control Systems, Shenyang Institute of Automation, Chinese Academy of Sciences. His research interests are in the areas of wireless ad hoc and sensor networks and cognitive radio networks and security in the smart grid.

LIANG Wei was born in 1974. She received a Ph.D. degree in Mechatronic Engineering from Shenyang Institute of Automation, Chinese Academy of Sciences in 2002. She is currently serving as a professor at the Shenyang Institute of Automation, Chinese Academy of Sciences. Her research interests are in the areas of wireless sensor networks, dynamic scheduling theory, and system simulation.