Available online at www.sciencedirect.com
ScienceDirect
Computer Communications 30 (2007) 2931–2953
computer communications
www.elsevier.com/locate/comcom

Demand-scalable geographic multicasting in wireless sensor networks

Shibo Wu *, K. Selçuk Candan

Department of Computer Science, Arizona State University, Tempe, AZ 85287, USA

Available online 21 June 2007

Abstract

In this paper, we focus on the challenge of demand-scalable multicast routing in wireless sensor networks. Due to the ad-hoc nature of the placement of the sensor nodes as well as the variations in the available power of the nodes, centralized or stateful routing schemes are not applicable. Thus, in this paper, we first introduce a *Geographic Multicast routing Protocol* (GMP) for wireless sensor networks.[1] The protocol is fully distributed and stateless. Given a set of the destinations, the transmitting node first constructs a *virtual* Euclidean Steiner tree rooted at itself and including the destinations, using a novel and highly efficient *reduction ratio* heuristic (called rrSTR). The simulation results on NS2 show that GMP requires 25% less hops and energy than the existing *Position Based Multicasting*, PBM, *Location-Guided Steiner trees*, LGS, approaches. The GMP algorithm as well as LGS and PBM all assume that each recipient receives the same copy of the multicast message. In reality, however, especially when the transmission includes streamed media, different recipients have different demands (in terms of the frequency of packets or the quality of media). Thus, in this paper, we investigate the suitability of the geographic multicasting schemes for situations where scalable transmission paths can save power. In particular, we propose intuitive mechanisms to extend the three schemes to cases where the data transmission can scale based on the demand. This leads to three new weighted multicast routing algorithms: wGMP, wLGS, and wPBM. The results show that the wGMP algorithm provides the best opportunities for scalability due to its flexible self-correcting decision making process, while other schemes, such as wLGS and wPBM are not directly suitable for scalable multicasting, due to their naively greedy structures.

Keywords: Wireless sensor networks; Geographic multicast; Group communication; Localized routing; Scalable streaming media

1. Motivation and related work

In this paper, we address the challenge of demand-scalable multicast routing in wireless sensor networks. Applications of wireless sensor networks vary from personal area networks to wide-area networks, where sensors are placed on very large terrains for *in situ* observations. Since *architected configurations* are not feasible in most cases, the network has to function in a fully distributed and scalable manner. Furthermore, to save power of the sensors and routing nodes, network protocols have to be energy efficient. Due to the ad-hoc nature of the placement of sensor nodes as well as the variations in the available power of the nodes, centralized or stateful routing schemes are not applicable in wireless sensor networks.

Geographic routing in sensor networks. Unlike most wired networks, such as the Internet, where network neighborhood does not necessarily correspond to the physical neighborhood of the nodes, wireless sensor networks can benefit from the available geographic information to eliminate expensive network operations, like flooding. For example, [6] and [23] present two similar greedy unicast routing algorithms called GFG and GPSR, respectively, which use the knowledge of the geographic positions of the nodes to minimize the number of hops. Essentially, each transmitting node chooses the neighbor closest to the destination as the next hop. In [62,64], we proposed single- and multi-path geographic routing algorithms (GPER and MGPER, respetively) which use local neighborhood information to identify *power-efficient* unicast routes.

* Corresponding author.
E-mail addresses: shibo.wu@asu.edu (S. Wu), candan@asu.edu (K.S. Candan).

[1] This work is supported by NSF Grant # 0308268, "ARIA – Quality-Adaptive Media-Flow Architectures for Sensor Data Management".

0140-3664/$ - see front matter © 2007 Published by Elsevier B.V.
doi:10.1016/j.comcom.2007.05.029

Geographic unicast protocols reported in the literature include [31,65,57,30,6,55,1,26].

Geocasting and geographic multicast routing. In contrast to single-source single-destination unicast schemes, group communication schemes like geocasting and multicasting aim at identifying one-to-many transmission paths. During the past decade, there has been a flurry of activities for multicasting in wired networks. However, there are fundamental differences between multicasting in wired and wireless networks. The simplest group communication scheme in a sensor network is *geocasting*, where a message is delivered to all nodes within a specified geographic region. In most geocasting schemes, the packet is first forwarded to any node in the destination region and then flooded to others within this region. Various geocasting protocols are presented in the literature [25,28,56,3,13,27].

Multicasting is another technology for supporting group communication among a group of nodes in wireless sensor networks. However, unlike geocasting, in multicasting, the set of destinations cannot be simply represented by any particular region: (a) individual destinations may be distributed in a wide area and (b) in a given area, not all nodes may be included in the list of destinations. Examples of multicast applications in wireless sensor networks include when a prime node is interested in the attributes of a collection of nodes and thus needs to communicate commands to this distributed set of wireless sensors in an observation network [53,22,19,36]. Another use example is data centric sensor networks [45], where data are named based on attributes and are hashed to different geographic locations. Since each data object may have multiple attributes, it may be hashed to different geographic locations. In such cases, updates to the copies of a data object requires dissemination to multiple sensor nodes in the network. For military use, multicast is also important for group communication, for example in a battlefield. Multicasting can also be used in multi-source single-destination data collection applications for identifying a suitable *reverse* data delivery tree. Multicasting (as opposed to multiple unicasting) preserves network resources by reducing redundant messaging. Therefore, the *quality* of multicast trees created by the system has a big impact on the survivability of a sensor network with large numbers of group communications.

Most existing multicast routing protocols maintain a distributed structure for the delivery of multicast packets. In tree-based structures [11,5,61,42,48,54,21], there is only one path for each destination; multiple destinations may share parts of their paths. In mesh-based structures [32,16,33,12], there may be multiple paths from a given source to each destination. Scoped flooding based multicast protocol can be found in [9]. As in our GPER single- and multipath geographic routing algorithms [62,64] aims to leverage variable radio ranges of the available nodes to take advantage of the reduced energy consumption (but paying a price in terms of increased hop counts). Unfortunately, topology changes, node failures, and group membership changes can render the communication and reconfiguration overheads of maintaining a distributed tree or mesh structure unacceptably high. In stateless multicast routing protocols, there is no need to maintain multicast session information. Consequently, in most of these schemes a fixed distributed structure is not maintained. Instead, the membership information is controlled by the data sources and the destinations and subdestinations are inserted into the packet at the time of forwarding. Several works [22,10,8,36] addressed stateless geographic multicast routing problem.

Although it is stateless, *Differential Destination Multicast* (DDM) [22] algorithm relies on an underlying unicast routing protocol for forwarding packets towards their destinations. Therefore, the multicast tree is implicit and cannot be controlled by DDM itself. In contrast, in source-routing based schemes (such as *Dynamic Source Multicast*, DSM [10]), the entire multicast tree is created by the source node in advance. In DSM, each node knows its own geographic location and periodically floods this information to the network so that every node in the network knows location of the others. Every time a source node needs to send a packet to a multicast group, it computes a multicast graph of nodes from the cache of locations it knows. In DSM, a *minimum spanning tree* based heuristic is used to create this routing graph. The resulting multicast tree is encoded using a *Prüfer sequence* [43] and included in the outgoing packet. Each receiving node on this path decodes the multicast tree information and routes the packet to the next nodes as decided by the source. In *Location-Guided Steiner Trees* (LGS) [8], each node only needs to know its own location and the locations of its neighbors. The source node locally constructs a multicast tree (using a minimum spanning tree based heuristic) consisting of itself and the destination nodes. A copy of the packet is then forwarded greedily towards the root of each subtree. The corresponding destinations in the subtree are also encapsulated in each copy of the packet. *Position Based Multicasting* (PBM) [36] is another protocol which makes forwarding decision based on local knowledge. Unlike LGS, however, it jointly optimizes (a) the progress of the packets towards the destinations and (b) the bandwidth usage.[2] GMR[49] proposes a cost over progress criteria to select next hops among neighbors, where the cost function is proportional to the number of next hops selected and progress is the expected distance advance by forwarding through the next hops. uCast [7] uses a scoreboard algorithm to calculate the next hops. In each iteration, the neighbor which *covers* most destinations gets the highest score and is chosen as a next hop. Once a neighbor is selected as next hop, the next iteration continues after this neighbor and those destinations covered by this neighbor are removed. There are other works which takes both spatial and temporal information into consideration to achieve just-in-time multicast, [20].

[2] In PBM, from the current node's point of view, the bandwidth usage is proportional to the number of next hops forwarding the packet.

Layered multicasting in wired and wireless networks. An underlying assumption in these geographic multicasting schemes is that all recipients receive the same copy of the multicast message. In reality, however, especially when the transmission includes streamed media, different recipients have different demands (in terms of the frequency of packets or the quality of media). A solution to varying bandwidth constraints is transcoding [4], where intermediate network nodes re-encode the video down to a lower rate before forwarding when a lower bandwidth is available. This approach however is computationally expensive and is not suitable for wireless scenarios where high degrees of computation should be avoided to save power. A more scalable solution to multicasting of streaming media is the use of hierarchical or layered encoding schemes [52]. In multi-layered media (image or video) encoding, the data is encoded into one or more easily separable layers, each with a different priority. Usually, a base layer contains the most important information, while enhancement layers contain data with lower priorities that can be dropped based on bandwidth availability. In spatial layering, a multiresolution representation is used to split each frame into layers. In SNR layering, the various layers are obtained through varying quantization degrees. DCT and DWT are also used to obtain subbands of a video or image such that more subbands provide higher quality. Layered video encoding schemes commonly used include MPEG-2 scalable profile, MPEG-4, and H.263+. In receiver-driven multicast schemes, such as [37], the source generates a number of layers and the destinations subscribe to as many layers as they need. In sender-driven (or source-adaptive) schemes, such as [41], the source uses congestion feedback to adjust the number and the bit rates of the layers. For example, in [60,59] uses multilayered video encoding schemes along with explicit range based congestion feedback to adjust trasnmission rates.

1.1. Contributions of this paper: Demand-scalable geographic multicasting through an efficient and effective euclidean steiner tree heuristic

In this paper, we first introduce *Geographic Multicast routing Protocol* (GMP) for wireless sensor networks. We then show that GMP can be expanded to multicasting scenarios with differing client-demands better than alternative geographic multicasting schemes.

1.1.1. Geographing multicasting with virtual euclidean steiner trees

As discussed above, while creating the minimum spanning tree for partitioning the destinations, LGS [8] does not consider any geographic points other than the actual destinations themselves. Thus, it significantly constrains the trees that it can generate. Consequently, this may result in both greater number of hops in the resulting multicast trees and larger per-destination hop counts. PBM [36] tries to balance the per-destination hop count with the total number of hops needed to reach all destinations, with the help of a trade-off parameter. The optimal value of this parameter, however, changes from task to task and depends on the number of neighbors and the distribution of destinations. Thus, choosing a single suitable parameter value is not easy.

In this paper, we first introduce a *virtual* Euclidean Steiner tree based multicast routing protocol where (a) transmitting nodes do not require any global knowledge to create a tree which will be used for partitioning the destination nodes into groups and (b) they use only local information during actual route selection. Note that the general Euclidean Steiner tree problem is NP-hard [24]. However, a special case where there are only three nodes, the Steiner point can be calculated efficiently as in [40]. Our algorithm, *Geographic Multicast routing Protocol* (GMP, initially sketched at [63]), exploits this to create heuristic Euclidean Steiner trees efficiently (in polynomial time) and effectively, without any global knowledge about where the sensor nodes are. The progressive nature of the routing scheme enables continuous refinement of the resulting trees, providing better multicast trees than similar schemes discussed above.

The GMP protocol is a Euclidean Steiner tree based multicast protocol. The underlying idea of GMP is that

> each transmitting node constructs a heuristic Euclidean Steiner tree, including the source and all destinations. The tree is *virtual* in the sense that it may include interior vertices that do not correspond to any actual wireless sensor nodes.

The destinations are divided into groups based on this tree. As in LGS, a copy of the packet is then forwarded to a suitable next hop and the process is repeated by the receiving nodes until all destinations are reached. Essentially, there are two main differences between GMP and LGS: (1) GMP uses an efficient and effective heuristic to construct Euclidean Steiner tree (which allows all possible Euclidean points) while splitting destinations into partitions and (2) the subdestination (root of the subtree) towards which the packet is forwarded is constrained to be an actual destination in LGS, while in GMP it can be any Euclidean Steiner point. As shown in Section 7.1, these flexibilities result in significantly better multicast trees.

1.1.2. wGMP – Demand-scalable geographing multicasting

In this paper, we also investigate the suitability of the geographic multicasting schemes, including LGS, PBM, and GMP, for situations where scalable, layered transmissions can save valuable power. In particular we propose mechanisms to adapt the existing schemes to the cases when the data can scale through layering mechanisms based on the demand.

Section 7.1 shows that GMP algorithm provides significant savings over both LGS and PBM algorithms in cases where all the destinations have identical data demands. Naturally, this may not be true for the weighted versions

of these algorithms adopted for the cases where consumers have varying quality and data demands. In Section 7.2, we provide simulation results (through NS2 network simulator) which compare wLGS, wPBM, and wGMP in terms of the number of packets transmissions as well as the total power consumption. The results show that the wGMP algorithm provides the best opportunities for scalability while wLGS and wPBM are not directly suitable for demand-scalable multicasting.

1.2. Organization of the paper

The remainder of this paper is organized as follows. Section 2 describes the wireless network model we assumed in this paper. Section 3 describes two existing geographic multicast routing protocols, LGS and PBM. Section 4 describes the rrSTR heuristic, based on a novel *reduction ratio* concept, for the construction of the Euclidean Steiner trees. Section 5 describes the GMP multicast routing protocols based on the Euclidean Steiner tree. Section 6 describes the weighted version of the three multicast protocols. Section 7 presents experiment results which evaluate the performance of GMP and compares it against alternative protocols. In addition, the performance of the weighted version of the multicast protocols are also presented in Section 7. Finally we present our conclusions in Section 8.

2. Wireless network model

In this paper, we adopt a commonly used sensor network model [62,23,66,47]: A set, S, of nodes is located in a two dimensional geographic area, G. Each node $v_i \in S$ has coordinates, $coord(v_i) = \langle x_i, y_i \rangle$. Each node knows its own coordinates. This can be achieved either through an internal GPS device or through a separate calibration process [23]. The location of a node acts as its ID and its network address. Therefore, there is no need for a separate ID establishment protocol. Each packet is marked with the location of the next hop and the corresponding node picks up the packet. The source node (generally a prime node) knows the destinations and their requested data consumption rates prior to the dissemination of the data packet. In this paper, we do not focus on the problem of how to establish and maintain multicast groups. In the literature, there are works ranging from static group membership [22,10] to highly dynamic scenarios supported by the source node [45,8] or a separate group management service [35]. In this paper, we focus on the problem of achieving efficient multicast routes between a source and destinations.

We assume that each node in the network has the same radio range, r. The wireless channels are assumed to be lossless. Note that the wireless channels in real sensor networks can be unreliable. Routing strategies for such lossy sensor networks have been proposed in [51]. In [51], the current node is assumed to know the packet reception rate of its neighbors and only those neighbors with reception rate larger than a threshold are considered for forwarding. In this paper, we do not consider channel loss explicitly. However, we note that the GMP protocol proposed in this paper can be extended for lossy wireless networks, for example by setting appropriate threshold value as in [51].

In this paper, we assume the commonly accepted channel power model, $\rho = a*\delta^{\gamma} + b$, where ρ denotes the transmission power, δ denotes the distance between the sender and the receiver, and γ is the power loss constant, typically between 2 and 4 [44]. In the power model, a and b are the distance-relative and constant terms of the power consumption. The energy consumption for transmitting a packet is therefore $E = \rho*t$, where t is the transmission time of the packet and t equals to the length of the packet divided by the channel data rate. The receiving power of each listening node in the neighborhood of the transmitting node is l, which is constant (or distance-insensitive).

To reduce the idle listening power consumption, various wireless network platforms use sleep and wake-up based solutions, where nodes may turn off to save power. Some algorithms, like STEM [50], use two radios, one for data and the other, low-duty-cycle mode radio, for waking-up sleeping nodes. Others, like GeRaF [69], on the other hand, do not assume that a sleeping node may be forced to wake up. Instead, in GeRaF, when a node wants to transmit a packet towards a destination, zero or more active nodes close to the target will receive the message. The actual node that will receive and forward the message is not known a priori by the sender, but rather is decided (probabilistically) after the transmission has taken place, according to nodes' own locations towards the destination. In this paper, we do not assume a specific mechanism that deals with sleeping or failed nodes. However, the incremental nature of the proposed GMP enables various GeRaF like extensions.

3. Background

In this section, we describe two *existing* geographic multicast protocols: LGS [8], PBM [36]. Both multicast protocols are stateless and use only local information; i.e., each node needs to know only its own location and the locations of its neighbors.

3.1. LGS: Location-Guided Steiner tree

LGS [8] aims at minimizing the overall bandwidth cost of the multicast paths. In LGS, the source node computes an approximate Steiner tree that consists of only the source node itself and the destination nodes. Although Steiner trees are commonly referred as multicast distributions tree for efficient delivery of packets in a fixed network, the Steiner tree problem in a graph is NP-hard, even if the global topology knowledge is given, i.e., even when the current node knows the locations of all sensor nodes in the network [24]. Thus, LGS approximates the Steiner tree using a minimum spanning tree based heuristic, which incrementally builds the multicast tree as follows: (1) Initially, there is only one node in the tree, which is the source node; (2)

At each iteration the node that is closest to the partially constructed tree is added to the tree. Note that LGS scheme overly constrains the multicast trees it can generate, because only the source node and the actual destination nodes can be included in the constructed tree.

After creating the multicast tree, the source node then creates a copy of the data packet for each subtree and encapsulates the corresponding destinations in the subtree in the copy of the packet. The copy is then forwarded greedily towards the root of the subtree, i.e., the packet is forwarded to a neighbor that is closest to the root of the subtree. After receiving a packet, the next hop node extracts the corresponding set of destinations and then repeats the same process to partition this set into subsets of destinations. After this, copies of the packets are forwarded to roots of each partition and the process is repeated by those roots.

LGS assumes the network density is large enough that an appropriate next hop can always be found for the group of destinations in each subtree. However, there are cases that no neighbor of current node is closer to the root of the subtree. LGS will fail in such cases.

3.2. PBM: Position Based Multicast

PBM on the other hand does not construct a complete multicast tree structure for routing. PBM only tries to find an appropriate set of neighbors as next hops and assigns each destination to a neighbor in the set that is closest to this destination. To find such a set, PBM jointly optimizes (a) the progress of the packets towards the destinations and (b) the bandwidth usage, using the following optimization criterion [36]:

$$Cost_{PBM} = \lambda * \frac{|M|}{|N|} + (1-\lambda) * \frac{\sum_{d\in D} \min_{m\in M} distance(m,d)}{\sum_{d\in D} distance(s,d)} \quad (1)$$

where s is the source node, N is the set of neighbors, $M \subseteq N$ is the set of next hops, D is the set of destinations. Therefore, the optimization criterion includes two components: the overall remaining distance to all destinations and the number of neighbors being used as next hops (the more the next hops are, the more the bandwidth usage is from the current node's view). By considering all possible subsets of its neighbors, PBM identifies one that minimizes the optimization criterion. Each destination is assigned to a neighbor in the set that is closest to itself. The destinations are then partitioned based on the identified subset; each neighbor in this subset becomes the forwarding node for this packet towards the corresponding partition.

There are cases when a destination cannot be assigned to any neighbor because no neighbor is closer to this destination than the current node. PBM groups all such *void* destinations into one group and performs perimeter routing for this group (which avoids the void by using a fixed routing strategy).

Note that since each possible subset of the neighborhood has to be considered, the PBM algorithm may be very costly, especially when there are large numbers of neighbors and destinations. Furthermore, the behavior of PBM is governed by a not-so-easy-to-set parameter, λ, which describes the preference between the two decision criteria: (a) tradeoff between remaining overall distance towards the destinations and (b) bandwidth usage (proportional to the number of next hops). The value of this parameter is fixed during actual routing; however, for a particular multicast scenario, the routing performance (e.g., number of transmissions, energy consumption) depends on the value λ. The optimal value of this λ parameter depends on the number of neighbors and the distribution of the destinations relative to the current node. Therefore, choosing the right parameter value is not trivial. As opposed to PBM, GMR [49] tries to eliminates the need for such a parameter by defining the optimization criterion as bandwidth usage divided by the progress towards the destinations.

4. rrSTR: A novel *Reduction ratio* measure for generating Euclidean Steiner trees

As described in Section 1, the GMP multicast routing algorithm, we initially sketched at [63] and we are detailing in this paper, is based on Euclidean Steiner trees [24]. The Euclidean Steiner tree problem is to connect a given set of points in the Euclidean space together, possibly by introducing extra points that are not in the given set, and minimize the total length of the line segments connecting all the points. Note that generating optimal Euclidean trees is a costly operation [24] which needs to be avoided. There are many heuristics [39,2,67] and approximation algorithms [68,18,46] to address this problem. Although they are much cheaper than optimal solutions, most approximation algorithms are still too costly to be deployed at sensor nodes. Some existing heuristics [38] apply to rectilinear Steiner problems and some are minimum spanning tree based algorithms [39,46,67].

4.1. Reduction ratio

These general purpose heuristics do not necessarily fit well for geographic multicasting. In fact, we note that (a) due to the lack of up-to-date global knowledge of the state of the wireless network, the Steiner tree points computed by any algorithm are not likely to be actual hops that will be used in the resulting route and (b) each receiving node in the network will have the opportunity to readjust the Steiner tree based on its own position. Therefore, instead of relying on expensive approximations or general purpose heuristics, in this paper, we focus on the following observations:

- *Observation 1:* when two destinations are far away from the source but are close to each other, they are likely to share subpaths. For example, in Fig. 1, destinations $\{u, v\}$ are likely to share subpaths on the multicast tree.

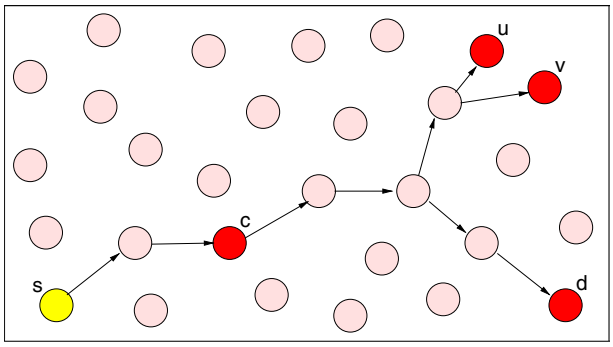

Fig. 1. Destinations $\{u, v\}$ are far away from the source and close to each other; therefore they are likely to share subpaths on a multicast tree. The cross angle of line segments $\overline{sc}$ and $\overline{sd}$ is small; thus destinations $\{c, d\}$ are also likely to share subpaths paths. Similarly, destinations $\{u, v\}$ are likely to share subpaths with c. How this multicast tree is created is described in Section 5.

- *Observation 2:* when the angle of the line segments connecting the source node and the destinations are small, the nodes are likely to share subpaths. For example, in Fig. 1, destinations $\{u, v, d\}$ are likely to share subpaths with destination c.

To uniformly capture these two observations, we introduce a novel measure called *reduction ratio*. Given a pair of destination points, (u, v), and a source node, s, reduction ratio, $RR(s, u, v)$, is defined as follows:

$$RR(s,u,v) = 1 - \frac{d(s,t) + d(t,u) + d(t,v)}{d(s,u) + d(s,v)} \quad (2)$$

Here t is the exact Euclidean Steiner point of these three nodes $\{s, u, v\}$, with the following property: the three edges from the given three nodes incident to the Steinter point form three 120° angles [40], that is, $\angle stu$, $\angle stv$ and $\angle utv$ are both 120°.

The reduction ratio measure has two major advantages: First of all, although the general Euclidean Steiner tree problem is NP-complete, the special case where there are only three nodes, u, v, and s in the Euclidean space, finding the Steiner point of these three points t such that $d(t, u) + d(t, v) + d(t, s)$ is minimum can be calculated very efficiently, in polynomial time [40]. Furthermore, the reduction ratio measure has the following properties, first two of which mirror our two motivational observations guiding the likelihood of two nodes on the network sharing a sub-path:

- *Property 1:* Given a source and two equidistant destination nodes, the reduction ratio would be larger if these two destinations are located further away from the source. For example, in Fig. 2(a), the reduction ratio of (p, q) is larger than the reduction ratio of (u, v).
- *Property 2:* Given a source and a pair of destinations, the reduction ratio would be larger if the angle between the two line segments connecting the source node and the two destinations is smaller. For example, in Fig. 2(b), the reduction ratio of (u, v) is larger than the reduction ratio of (u, v').

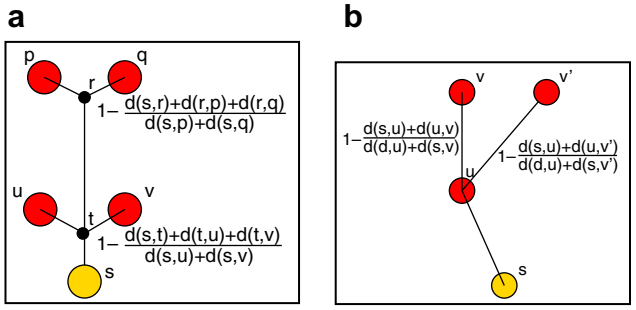

Fig. 2. Reduction ratio. (a) The reduction ratio of the pair (p, q) is larger than that of (u, v) because the effect of smaller $d(s, t)$ to the reduction ratio is less than the effect of smaller $d(s, u)$. (b) The reduction ratio of the pair (u, v) is larger than that of (u, v') because $d(u, v')$ is larger than $d(u, v)$.

- *Property 3:* The value of reduction ratio is always less than 1/2.

The proof of these properties are presented next.

Proof 4.1 (*Property 1*). Let us denote the source node as s and the two destinations as u and v. Let $d(u, v)$ be equal to L and let $\angle tsu$ and $\angle tsv$ be θ as shown in Fig. 3(a). We prove Property 1 by showing that the reduction ratio of nodes $\{s, u, v\}$ is either 0 or it decreases when θ increases.

Steiner points have the property that, when θ is greater or equal to $\pi/3$, the Steiner point t is exactly at the position of s. Therefore, in that case, the reduction ratio is 0.

Next we consider the case where θ is less than $\pi/3$. In this case, $\angle vut$ and $\angle uvt$ are both $\pi/6$, and $\angle utv$ is $2\pi/3$. Thus, we have, $d(u,t) = d(v,t) = L/\sqrt{3}$ and $d(s,t) = L/(2 * tan(\theta)) - L/(2 * \sqrt{3})$. Therefore, the reduction ratio can be computed as

$$RR(s,u,v) = 1 - \frac{2 * L/\sqrt{3} + L/(2 * tan(\theta)) - L/(2 * \sqrt{3})}{2 * L/(2 * sin(\theta)}$$
$$= 1 - \frac{\sqrt{3} * sin(\theta) + cos(\theta)}{2} \quad (3)$$

The derivative of Eq. (3) is

$$df_\theta = \frac{sin(\theta) - \sqrt{3} * cos(\theta)}{2} = \frac{1}{2 * cos(\theta)}(tan(\theta) - \sqrt{3}) \quad (4)$$

Note that df_θ is always less than 0 for $\theta < \pi/3$. Thus, the reduction ratio decreases when θ increases and $\theta < \pi/3$. This and the fact that reduction ratio is 0 for the case where $\theta \geqslant \pi/3$ completes the proof. □

Proof 4.2 (*Property 2*). Let us again denote the source node as s and the two destinations as u and v. We show that (a) reduction ratio is 0 when $\angle usv = \alpha$ is greater or equal to $2\pi/3$ and (b) reduction ratio decreases when α increases and $\alpha < 2\pi/3$.

Part (a) holds because the Steiner point t will be at the position of s when $\angle usv \geqslant 2\pi/3$.

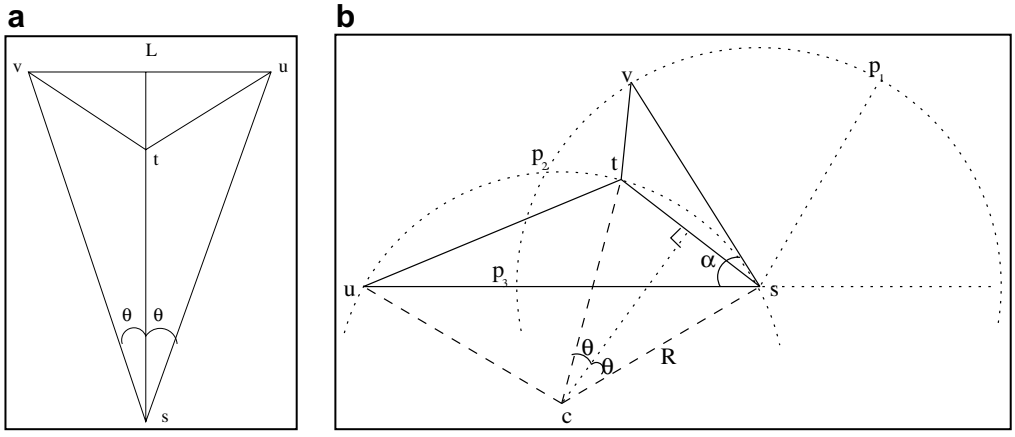

Fig. 3. Reduction ratio proof.

Next we prove part (b). Consider Fig. 3(b), in which c is the center of the circle circumventing the triangle of $\{s, u, t\}$. Note that, since $\angle stu = 2\pi/3$, $\angle scu = \widehat{stu} = 2\pi/3$.

Since for any point t', $\angle st'u = 2\pi/3$, the arc between the points s and u has the property that it hosts all Steiner points for point triples involving s and u. Thus, when node v moves from p_1 to p_2 on a circle centered around s (i.e., keeping the distance from s constant), the Steiner point moves along the arc $\widehat{stu}$. In particular, as v approaches p_2 (decreasing α), t also moves closer to p_2 along the arc $\widehat{stu}$, increasing the angle $\angle sct = 2{*}\theta$. Given this, we prove Property 2 by showing that when θ increases and $\theta < \pi/3$, the reduction ratio also increases.

Let R be the radius of circle centered at c. Then, we have $d(s,\ t) = 2{*}R{*}sin(\theta)$ and $d(u,t) = 2 * R * sin(\pi/3 - \theta) = \sqrt{3} * R * cos(\theta) - R * sin(\theta)$. Since $\angle stv$ is $2{*}\pi/3$, we also have $d(v,\ t)^2 + d(s,\ t)^2 + d(v,\ t){*}d(s,\ t) = d(s,\ v)^2$. Using these, we can solve for $d(v,\ t)$ as follows:

$$d(v,t) = \frac{-d(s,t) + \sqrt{d(s,t)^2 - 4 * (d(s,t)^2 - d(s,v)^2)}}{2}$$
$$= -R * sin(\theta) + \sqrt{4 * d(s,v)^2 - 3 * R^2 * sin(\theta)^2}$$

Given this, we can rewrite $d(s,\ t) + d(u,\ t) + d(v,\ t)$ as follows:

$$d(s,t) + d(u,t) + d(v,t) = 2 * R * sin(\theta) + (\sqrt{3} * R * cos(\theta) - R * sin(\theta) + \left(-R * sin(\theta) + \sqrt{4 * d(s,v)^2 - 3 * R^2 * sin(\theta)^2}\right)$$
$$= \sqrt{3} * R * cos(\theta) + \sqrt{4 * d(s,v)^2 - 3 * R^2 * sin(\theta)^2}$$

When θ increases from 0 to $\pi/3$, both terms, $\sqrt{3} * R * cos(\theta)$ and $\sqrt{4 * d(s,v)^2 - 3 * R^2 * sin(\theta)^2}$, decreases. Therefore $d(s,\ t) + d(u,\ t) + d(v,\ t)$ also decreases. Since $d(s,\ u)$ and $d(s,\ v)$, are both fixed, the reduction ratio measure $(1 - \frac{d(s,t)+d(t,u)+d(t,v)}{d(s,u)+d(s,v)})$ will increase with the increase of θ. In other words, the reduction ratio will increase when the $\angle usv = \alpha$ decreases.

Finally, if the node v falls within the circle centered at c (i.e., on the arc between points p_2 to p_3), the Steiner point will be at the same position as the node. As the node approaches from p_2 to p_3, the reduction ratio will also increase, as $d(u,\ t) = d(u,\ v)$ decreases while $d(s,\ t) = d(s,\ v)$ remains the same. □

Proof 4.3 (***Property 3***). According to Property 2, given two destinations u and v whose distances from the source node s are fixed, the maximum value of reduction ratio of $\{s, u, v\}$ occurs when $\angle usv$ is 0. □

Let $d(s,\ u)$ be x, let $d(s,\ v)$ be y, and without loss of generality, assume $0 < x \leqslant y$. Then, the reduction ratio of $\{s, u, v\}$ when $\angle usv = 0$ is equal to $1 - \frac{y}{x+y}$, which has the maximum value, 1/2, when $x = y$.

The efficient solution for the special case of three node Steiner tree and the properties described above enable us to develop a novel algorithm to create heuristic Euclidean Steiner trees efficiently (in polynomial time).

4.2. Basic rrSTR algorithm

In Fig. 4, we present an iterative algorithm, rrSTR, to construct a Euclidean Steiner tree, based on these properties of the reduction ratio measure. In this subsection, we provide an overview of this algorithm.

Given a set of destination points, initially, source node s marks all its multicast destinations as *active*; that is, none of the destinations are covered yet. In each iteration, the algorithm identifies a destination pair, $(u,\ v)$, with the largest reduction ratio. Given this pair, rrSTR creates a *virtual destination* w at the location of the Steiner point of nodes $\{s,\ u,\ v\}$. Note that when there are only three nodes, the Steiner point can be calculated efficiently [40]. The corresponding two edges, $\overline{wu}$ and $\overline{wv}$, are then added to the tree, such that node w becomes the parent of nodes u and v. Since they are already covered, both u and v are marked as *inactive* so that any

rrSTR(s, $destv$, rr)
This function calculates the Steiner tree for nodes s and destinations in $destv$

- s is the current node;
- $destv$, the list of all destinations;
- rr, the radio range of s;
- T is the tree to be returned;

{

1. set all destinations and all pairs of destinations to be active;
2. calculate the reduction ratios and Steiner points for all pairs of destinations with s;
3. repeat
   - find an active pair (u, v) with largest reduction ratio and let t be the Steiner point of $\{s, u, v\}$
   - if no such pair can be found, break;
   - if u is the same node with v
     - add edge $\overline{su}$ to T, deactivate node u;
   - else if t is collocated with s
     - add edges $\overline{su}$ and $\overline{sv}$ to T, deactivate both u and v;
   - else if t is collocated with u/v
     - add edge $\overline{uv}/\overline{vu}$ to T, deactivate v/u;
   - else if both $d(s, u)$ and $d(s, v)$ less than rr
     - deactivate the pair (u, v);
   - else if $d(s, u)/d(s, v)$ is less than rr
     - if $rr + d(t, u) + d(t, v)$ is larger than $d(s, u) + d(s, v)$, deactivate pair (u, v);
     - else add edge $\overline{uv}/\overline{vu}$ to T, and deactivate v/u;
   - else $d(s, t)$ is less than rr, and $rr + d(t, u) + d(t, v)$ is larger than $d(s, u) + d(s, v)$
     - add edges $\overline{su}$ and $\overline{sv}$ to T, deactivate both u and v;
   - else /**create a new virtual destination**/
     - create a new virtual destination w at t, activate w, add edges $\overline{wu}$ and $\overline{wv}$ to T, calculate the reduction ratios and Steiner points for all pairs (w, i), where i is an active destination
4. return T;

}

Fig. 4. Reduction-ratio-based heuristic for Euclidean Steiner tree generation.

destination pair that contains either u or v will not be considered in the remaining iterations. The new virtual destination w is added in the set of destinations and marked *active*. Thus, rrSTR will calculate reduction ratios for destination pairs consisting w and all remaining active destinations.

Note that, in the extreme case, the Steiner point can be collocated with u or v, i.e., the Steiner point is exactly at the location of u or v. If for example the Steiner point is collocated with u, then no new virtual destination needs to be created. Instead, the edge $\overline{uv}$ is inserted into the tree and v is marked *inactive*. u stays *active*. Also note that the Steiner point can be collocated with the source node s itself. In this case, edges $\overline{su}$ and $\overline{sv}$ are inserted to the tree and nodes, u and v, are marked *inactive*.

Fig. 5 illustrates the process with an example. In the first iteration, pair (u, v) is identified since they have the largest reduction ratio, and a virtual destination w_1 is created at the Steiner point of {s, u, v}. Edges $\overline{w_1u}$ and $\overline{w_1v}$ are added to the tree. Nodes u and v are then deactivated. In the sec-

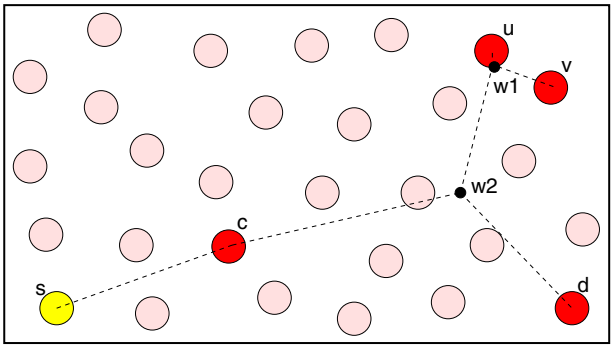

Fig. 5. An example Euclidean Steiner tree generated by rrSTR.

ond iteration, pair (w_1, d) is identified, and another virtual destination w_2 is created. Edges $\overline{w_2w_1}$ and $\overline{w_2d}$ are added to the tree. In the third iteration, pair (w_2, c) is identified. No virtual destination is created in this step since the Steiner point of $\{s, c, w_2\}$ is at node c itself. Instead, edge $\overline{cw_2}$ is added to the tree. At last, pair (c, c) is found and edge $\overline{sc}$ is added to the tree.

This algorithm is related to but different from conventional contraction based algorithms, such as [68,46]. In these algorithms a full connected component is identified and replaced with a new point, iteratively. In rrSTR, instead, a destination pair is identified and replaced with a virtual destination. One difference is that the source node is never contracted. More importantly, the construction of the Steiner tree is guided with the *reduction ratio*, which identifies those pairs that are more likely to share subpaths.

4.3. Radio range aware rrSTR

The basic rrSTR algorithm described above uses the reduction ratio measure to guide the construction of Euclidean Steiner trees. However, the fact that it is not always good to use extremely short steps, especially within the *radio range* of a transmitting node, is not properly captured by this algorithm. Intuitively, when the Steiner point t is within the radio range, creating a virtual destination at t may not always be beneficial (Fig. 6(b)). Therefore, in some cases the basic form of rrSTR may result in *redundant hops*. Thus, overly eager virtual destination assignments should be avoided. The following are the three cases in which, given destinations u and v, it is not appropriate to create a new virtual destination:

- When both u and v are in the radio range of the current node, a new virtual destination would increase the number of hops to reach u and v by 1. Therefore, it is not appropriate to create a virtual destination at their Steiner point. Instead, we mark the pair (u, v) as *inactive*, so that this pair will not be considered in the future. Note that marking a pair (u, v) as *inactive* is different from marking two nodes u and v as *inactive*, in that pairs containing u or v other than the pair (u, v) can still be active.
- If neither u nor v, but the corresponding Steiner point t of $\{s, u, v\}$ is in the radio range of s, then a virtual destination may be beneficial in some cases. Since routing through the virtual destination will cost one hop on the resulting multicast tree, this will be acceptable only if

$$1 + \frac{d(t,u) + d(t,v)}{rr} < \frac{d(s,u) + d(s,v)}{rr},$$

  where rr is the radio range of current node. If the left-hand side is larger, then there is no benefit of using the virtual destination; therefore, we add edges $\overline{su}$ and $\overline{sv}$ and mark pair (u, v) as *inactive*. If the lefthand side is smaller, then we can create a virtual destination w at t. For example in Fig. 6(a), it is beneficial to create the virtual destination w at t, but in Fig. 6(b), it is not appropriate to create the virtual destination.
- When only u is in the radio range, using a new virtual destination at the Steiner point may not be useful. The decision is once again based on the truth of the above inequality. If the righthand side is larger, then instead w, u will be used as the Steiner point. In this case, edge $\overline{uv}$ will be added into the tree and v will be marked as *inactive*. If the left hand side is larger, s will be used as the Steiner point instead of w; edges $\overline{su}$ and $\overline{sv}$ will be added to the tree and marked both u and v will be marked *inactive*. For example, in Fig. 7(a) the Steiner point becomes u, and in Fig. 7(b) s becomes the Steiner point instead.

The algorithm presented in Fig. 4 is radio range aware and implements these three special cases to prevent redundant hop generation and, thus, to save network resources.

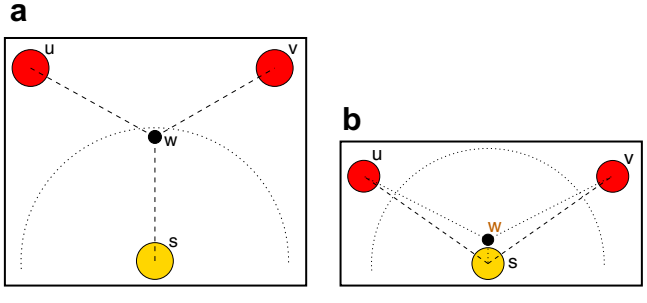

Fig. 6. It may or may not be beneficial to create a virtual destination when the destinations themselves are out, but their Steiner point is in the radio range: (a) a case where it is beneficial to create the virtual destination w, (b) a case where it is not beneficial. Dotted edges are constructed by the basic rrSTR algorithm without considering the radio range, and dashed edges are constructed by rrSTR by taking the radio range into consideration.

5. GMP routing based on rrSTR trees

The outline of the GMP routing algorithm based on the rrSTR trees introduced in the previous section is presented in Fig. 8. In this section, we describe how the GMP algorithm operates in detail. Let s be a source node.

Destination grouping: s first efficiently computes a *virtual* Euclidean Steiner tree as described in the previous section.

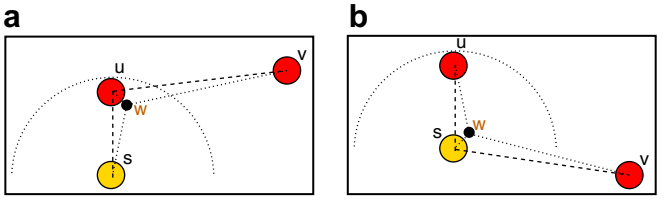

Fig. 7. It is not beneficial to create a virtual destination when only one of the destinations is in radio range: (a) u should be used as the Steiner point and (b) s should be used as the Steiner point. Dotted edges are constructed by the basic rrSTR algorithm without considering the radio range, and dashed edges are constructed by rrSTR by taking the radio range into consideration.

GMP(s, rr, $pack$)
This function splits the destinations into appropriate groups and forward a copy of the packet to a next hop for each group of destinations

- s is the current node;
- rr, the radio range of s;
- $pack$ is the packet

{

1. extract destinations from $pack$ to $destv$;
2. set T to be rrSTR(s, $destv$, rr);
3. set $pivots$ to be the children of s in T
4. for each p in $pivots$;
   - find a neighbor n, that minimizes $d(n,p)$ subject to $\sum_{v(p)} d(n,v(p)) < \sum_{v(p)} d(s,v(p))$, where $v(p)$ is a non-virtual destination in the subtree rooted at p;
   - if n is found, clear PERIMODE flag in $pack$, remove p from $pivots$, forward a copy of $pack$ to n with all $v(p)$s saved in the copy;
   - else remove edge $\overline{pl}$ and add edge $\overline{sl}$, where l is the last child of p, add l to $pivots$;
     - if p has only one child o left, and p is a virtual destination, then remove edge $\overline{po}$ and add edge $\overline{so}$, add o to $pivots$, remove p from $pivots$;
     - else continue with the same p;
5. if $pivots$ is empty, return;
6. /** all destinations in $pivots$ are void now **/
   if PERIMODE flag is not set in $pack$, set the flag on;
7. calculate a next hop n by perimeter routing based on the average location of the destinations in $pivots$;
8. forward a copy of $pack$ to n with destinations in $pivots$ saved in the copy;

}

Fig. 8. GMP routing algorithm.

s then uses this Steiner tree to split the destinations into groups. We refer to the direct (terminal or non-terminal) descendants (children) of s in the Steiner tree as *pivots*. Note that pivots may be actual nodes or they may be virtual, in the sense that a pivot may not always correspond to an actual sensor node. For each pivot p, s identifies all the non-virtual destinations in the subtree corresponding to this pivot. This set is referred to as the *group* of this pivot (*group*(p)).

Next hop selection: For each pivot p, s then identifies a next hop within its own neighborhood. In most cases, this node (*hop*(p)) is the neighbor closest to the pivot. However, to prevent routing loops, s also requires that the total distance from the next hop, *hop*(p), to all destinations in *group*(p) to be less than the total distance from s to all destinations in this group. For each pivot p, a copy of the packet as well as the destinations in *group*(p) are sent to *hop*(p). When the next hop receives the packet, it (1) extracts the corresponding destinations, (2) removes itself from this list if it is one of those destinations, and (3) repeats the above procedure, constructing a new Steiner tree to split the destinations into groups, and selecting next hops. This process is repeated by all hops until all the destinations are reached. Fig. 9 illustrates the execution of the GMP algorithm:

1. In Fig. 9(a), s constructs a Euclidean Steiner tree and chooses c as the pivot for destinations $\{c, u, v, d\}$. Then, s chooses n_1, the neighbor of s closest to c as next hop.
2. After a similar process, node n_1 forwards the message to node c, which in this case is both the pivot and the next hop (this step is omitted in the figure).
3. When node c receives the message, it extracts the destination list from the packet, and removes itself from the list. Based on the Steiner tree it constructs, as shown in Fig. 9(b), it decides not to split the destinations and forwards the message to node n_2 which is closest to the pivot w_2 it computed for destinations $\{u, v, d\}$.
4. After a similar process, n_2 forwards the message to n_3 without altering the destination list.
5. n_3 constructs a virtual Euclidean Steiner tree (shown by the dashed line in Fig. 9(c)). Note that, in this case, w_2, which is at the Steiner point of $\{n_3, w_1, d\}$, is not used as a virtual destination. This is because w_2 is in the radio range of n_3 and not close to the destinations $\{w_1, d\}$. Instead,
   - the pivot for destinations $\{u, v\}$ is w_1 and
   - the pivot for destination d is d itself.

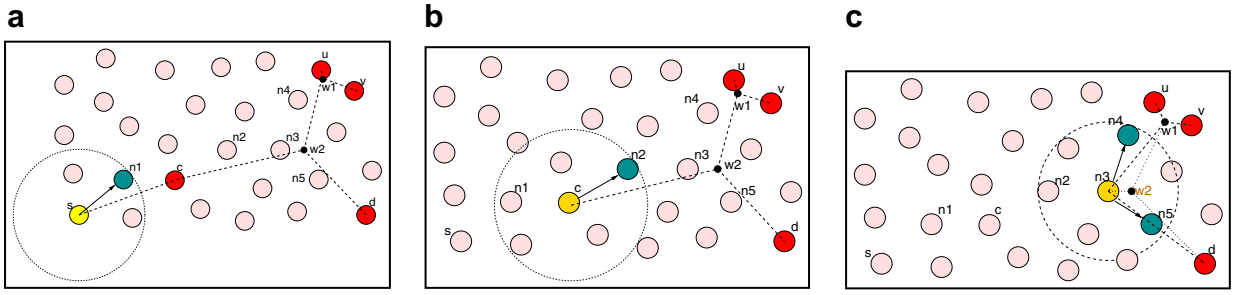

Fig. 9. Example of GMP routing. The Steiner tree calculated by a current node is indicated by dashed edges. (a) All the destinations are in one group with pivot as c and the next hop for this group is n_1. (b) The remaining destinations $\{u, v, d\}$ are in one group with pivot as w_2 and the next hop for these destinations are n_2. (c) Destinations are split into two groups: $\{u, v\}$ and $\{d\}$ The pivot and the next hop for group $\{u, v\}$ are w_1 and n_4, respectively, while the pivot and the next hop for group $\{d\}$ are d and n_5, respectively. Dotted edges in (c) are constructed by basic rrSTR and they are shown for comparison with the Steiner tree calculated by the radio range aware version of rrSTR.

6. A copy of the message is sent to the next hop n_4 for pivot w_1 with destinations specified as $\{u, v\}$. Another copy of the message is sent to n_5 with the only remaining destination d.
7. n_4 sends the message to destination u and v, respectively.
8. n_5 sends the message to destination d.

The resulting tree was already shown in Fig. 1 in Section 4.

5.1. Dealing with voids

Note that the process used to construct the Steiner tree does not consider the neighbor locations of the current node. Although this is not an issue in most cases (as an appropriate next hop can be found for each selected pivot in a dense network), it is possible that in some cases there will be no suitable neighbor with smaller total distance to the destinations in the pivot's group than the current node.

In GMP, the source considers if there exists a neighbor that has a smaller total distance to part of the destinations. If there exists such a neighbor, the source further splits the group into two smaller parts:

1. The source s removes the last child l of the pivot p from p's children list. (The children of p are the direct descendants of p in the Steiner tree. The last child of p can easily be found if the order in which edges are included to the Steiner tree is saved along with this edge.)
2. s makes l a new pivot by adding it to its own children list.
3. If there is only one child (say o) left in p's list of children, then
   - if p is a virtual destination, s makes o a new pivot and removes p from its children list
   - if p is an actual destination node, s does not remove p from its children list.

After splitting a group, next hops are calculated for each newly created or updated pivots. The splitting process continues if no valid next hop can be found for any pivot and its group. Fig. 10 shows an example: Node s constructs the Steiner tree with w_3 as the pivot for the group of destinations $\{u, v, c, d\}$. However, neither one of the two possible neighbors, n_1 or n_2, has a smaller total distance to the destinations than s itself. Therefore, there is no valid next hop for this group of destinations. In this case, s will split the destinations into two groups and w_1 and w_2 will be assigned as new pivots. Since n_1 and n_2 are now valid next hops for the groups of w_1 and w_2, respectively, copies of messages will be sent to n_1 and n_2.

Consider a case where one or more groups contain a single non-virtual destination, where no neighbor is closer to any of the destinations than the current node. In unicast schemes (where by definition there is only one destination), a similar situation is dealt with by placing the packet into a *perimeter mode* [6,23,62]. Once in perimeter mode, the packet traverses the boundaries of the void area following the right hand rule until a node that is closer to the destination than the point where the packet enters the perimeter mode is reached. To correctly perform the right hand rule, the graph of the wireless nodes has to be planarized first, based on Relative Neighborhood or Gabriel Graphs [58,14]. Such planarization can be done by the current node with only local information [6,23,62]. In multicasting, where there may be multiple such destinations, GMP takes the following actions:

1. The source creates a single group which contains all such destinations and sets the packet for this group to be in perimeter mode.

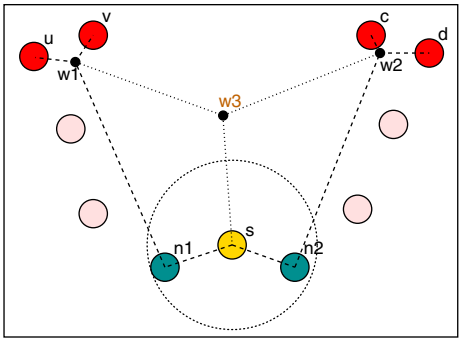

Fig. 10. Splitting the set of destinations when there is no valid next hop.

2. s calculates the average of the geographic locations of these destinations and identifies the next hop in perimeter mode based on this average location, as in [36].
3. The packet is forwarded to this next hop with all destinations in this group recorded in the packet.
4. When a node receives a packet in perimeter mode, it first runs GMP to try splitting the destinations into groups and finds a valid next hop for each group.
5. If valid next hops are found for all groups, then the packet for each group is out of perimeter mode.
6. If no valid next hop can be found for any of the groups, then the packet remains in the perimeter mode and traverses the network with the same previous average destination.
7. If valid next hops are found for some but not all of the groups, then a new perimeter group will replace uncovered groups and a new average destination location is calculated for them. The packet starts a fresh round of perimeter routing with this new average destination.

The perimeter mode operation described above is similar to the one used by PBM [36] in the sense that the calculation of next hop in perimeter mode is based on the planarized graph of wireless nodes. However, GMP does allow a *void* destination, for which no suitable next hop exists, to join other destinations in a group, so that a valid next hop can be found for all these destinations as a group. In contrast, in PBM [36], once a void destination is identified, the packet for this destination will enter perimeter mode. We note that this is not always necessary. For example, in Fig. 11, node s has no neighbor closer than itself to the destination v. In PBM, s will not group u and v together, and the packet for v will enter perimeter mode at s and is forwarded to n_1. In GMP, u and v are in one group and node n is a valid next hop for this group. Therefore the packet is forwarded to n for destinations u and v. Node n then forwards the packet to u, which will forward the packet to v.

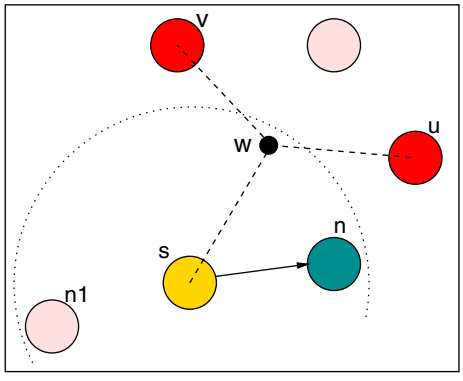

Fig. 11. An example where a group $\{u, v\}$ has a destination v, for which there is a no valid destination. Unlike other approaches, GMP allows such *void* destinations to enter in groups with other nodes. In the example, n will be the next hop for this group.

5.2. Computational complexity of the GMP and comparison with LGS and PBM

The computational complexity of GMP has two parts: (a) construction of the Steiner tree and (b) next hop selection:

- Let us assume there are n destinations and m neighbors for the current node, s. For constructing the Steiner tree, we use a priority queue to maintain the reduction ratios of all possible destination pairs (See step 3 in the algorithm in Fig. 4). In a given iteration, if no virtual destination is created, the complexity of this step is $O(\log n)$ for removing a destination pair from the priority queue. There can be at most $O(n^2)$ such iterations. On the other hand, if a virtual destination is created in an iteration, then the complexity of such a step is $O(n*\log n)$ for inserting $O(n)$ pairs into the priority queue. There are at most $O(n)$ virtual destinations. Therefore, the complexity for the construction of the approximate Steiner tree is: $O(n^2)*O(\log n) + O(n)*O(n\log n) = O(n^2* \log n)$.
- In the next hop selection step, the complexity for calculating a next hop for each pivot is $O(m)$, and there are at most $O(n)$ pivots.

Hence, the complexity of each step of the GMP algorithm is $O(n^2*\log n + n*m)$. In contrast, since PBM [36] considers all subsets of neighbors, it is exponential in m. Therefore, GMP is significantly more efficient than PBM, especially for dense networks. On the other hand, the complexity, $O(n^2 + n*m)$, of LGS is slightly less than that of GMP. This is expected as LGS limits the tree creation to the geographic locations of the network nodes. As we demonstrate in the next section, this over-constrains the possible trees and results in less effective multicasting.

6. Demand-scalable geographic multicasting

The geographic multicast protocols described in Section 3 and GMP protocol are designed based on the assumption that all destinations demand the same amount of data. Thus, they are not directly applicable to the cases where the data and quality demands are different among data consumers. Therefore, in this section, we extend the multicast protocols discussed in Section 3 in such a way that they can deal with different data demands among destinations. An important criteria in the design of these *weighted implementations* is that the localized nature of LGS, PBM, and GMP (essential for their adaptive operation) should be preserved. We will see that this constraint renders the weighted version of the LGS and PBM a disadvantage, while wGMP can provide the required scalability while ensuring localized routing decisions.

6.1. wLGS: Weighted LGS

We first start with the weighted implementation of the LGS protocol, wLGS, to deal with destinations of different packet ratio demands. As discussed above, LGS algorithm first constructs a tree consisting of only real nodes. The tree is not a Steiner tree, but actually an minimum spanning tree based approximation and is constructed incrementally: at each step of the protocol, the node that has the least distance to the partially constructed tree is added to the tree. In the original LGS, since the packet ratio demands is not considered in the tree construction, this may result in larger number of hops. To see this, consider the multicasting task in Fig. 12, in which the source node is s and destination nodes are u and v, with packet ratio demands as 1 and 0.1, respectively. If s wants to multicast 10 data packets, node u should receive all the 10 packets and node v only needs 1 of the 10 packets. Note that the node v is closer to node s than node u. As shown in Fig. 12a, by LGS, all the data packets for destination u will be forwarded through node v. When for example 10 packets are to be multicast to u and v, the total number of hops for the multicast routing is $3*10 = 30$. However, if forwarding the data packets to node v through u as in Fig. 12b, the total number of packets will be $2*10 + 1 = 21$ because node v only demands one of the 10 multicast packets.

From the above example, we see that destinations that demand more may be added to the tree earlier to avoid forwarding through destinations with less demands to destinations with more demands. Therefore, we propose to modify LGS by introducing a ranking condition, $C(u) = \frac{D(u,T)}{w_u}$, where u is a destination, T is the partially constructed tree, and w_u is the demands of destination u, as the criteria of selecting the next node to be added to the tree. Based on the ranking criteria C, the source node can construct an approximate MST tree as before and split the destinations into groups based on this approximate tree. The last thing to do for the source node then is to decide, based on the packet ratio demand, how much data should be forwarded to each next hop: (a) For each group of destinations, the source node will identify a next hop node within its neighborhood. Each next hop is assigned a packet ratio demand as the maximum demand of the destinations in the corresponding group. (b) The source node creates a copy of the data packet for each group and saves the destination information of this group in the copy, setting a new packet ratio demand for each destination. The new demand value is obtained by dividing the original demand with the maximum demand of this group. At last, (c) the source node forwards to each next hop as much data required by the its packet ratio demand. Naturally, some nodes may end up receiving more data packets than they request if they act as a relay node forwarding the data packets to other destinations.

Note that wLGS may result in larger numbers of routing hops than needed even when forwarding through larger demand destinations to lesser demand destinations. Consider the case shown in Fig. 13(a), where the source node s calculates the approximate Steiner tree in which node v is the child of node u. Obviously, it is not optimal to forward packets along this path (i.e., forwarding to v through u) because forwarding to v through z could reduce the total number of hops. In fact, the packet does not have to follow the tree structure specified by s during forwarding. In Fig. 13(b), when the next hop z receives the packet, it adds u first to the tree, however, this time v is no longer added as the child of the u but a child of z itself.

6.2. wPBM: Weighted PBM

In this subsection, we extend the PBM routing protocol for destinations with different packet ratio demands. Remember that the PBM is trying to balance between the bandwidth usage at current node (i.e., the number of next hops) and the overall distance from the next hops to the remaining destinations.

In PBM, the overall distance is calculated based purely on the distance from a neighbor to a destination. When considering the packet ratio demands, it is natural to account for the packet ratio demands of the destinations along with their distances. Therefore, wPBM uses a modified minimization objective function,

$$\lambda * \frac{|M|}{|N|} + (1-\lambda) * \frac{\sum_{d\in D} \min_{m\in M} distance(m,d) * w_d}{\sum_{d\in D} distance(s,d) * w_d} \quad (5)$$

when calculating the distance to the remaining destinations.

As before, s is the source node, N is the set of neighbors, $M \subseteq N$, D is the set of destinations. The first part of the equation describes the normalized bandwidth usage, i.e., dividing the number of selected next hop neighbors by the total number of neighbors. Note that weights are not

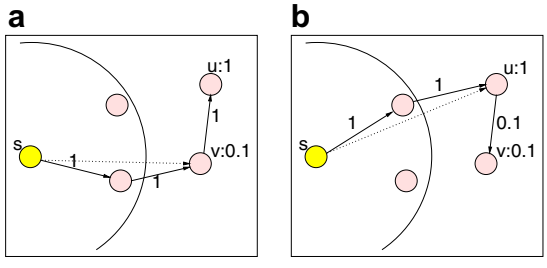

Fig. 12. LGS results in redundant transmissions.

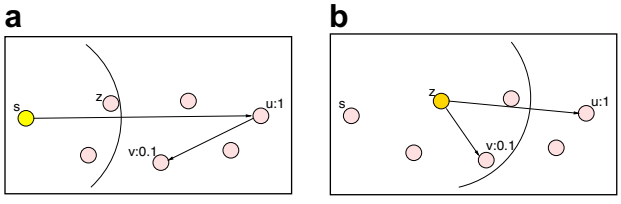

Fig. 13. wLGS: (a) the source node s calculates undesirable paths and (b) the next hop z makes adjustment and calculates better paths.

applicable here. The second part accounts for the weighted overall distance from the set of neighbors to the destinations, also normalized by dividing it by the weighted remaining overall distance from the current node.

Note that weights of the distances makes it even harder to select an appropriate value for the constant parameter, λ. This is evidenced in the experiment section.

6.3. wGMP: Weighted GMP

So far, we presented weighted versions of the LGS and PBM algorithms through intuitive which preserves the stateless, localized decision making property. In this subsection, we extend GMP for the situations where destinations have different packet ratio demands. The experiment results in Section 7.2 will show that the resulting protocol, wGMP, will be highly effective in reducing the amount of transmission and energy consumption.

As discussed in Section 4, GMP is based on the observation that reduction ratio reflects the likelihood of sharing subpaths for a pair of destination. The reduction ratio is calculated based on the position of the Steiner point of the source node and two destination nodes. Therefore, the first step for wGMP is the calculation of the reduction ratio for the destinations. However, the main question at this stage is whether the distance value used in the computation of reduction ratios should reflect the data demands of the destinations. Note that the reduction ratio is designed to capture the following two observations as described in Section 4: (1) when two destinations are far away from the source but are close to each other, they are likely to share subpaths. (2) when the angle of the line segments connecting the source node and the destinations are small, the nodes are likely to share subpaths. Therefore, as long as the two destinations are close to each other and far from the source, they are likely to share path, no matter what the packet ratio demands are for these two destinations. To see this, consider the multicast task in Fig. 14 where nodes u, v, x and y are the four multicast destinations with packet ratio 0.5, 0.5, 1, and 0.1, respectively. Nodes u and v are close to each other and are likely to share subpath, therefore the pair u, v should be considered early in the construction of the Steiner tree. The same observation is also true for nodes x and y, despite the fact that (unlike the case for nodes u and v) the packet ratio demands of x and y are quite different from each other. Therefore, the reduction ratio should be calculated in the same way as in GMP using Eq. (2).

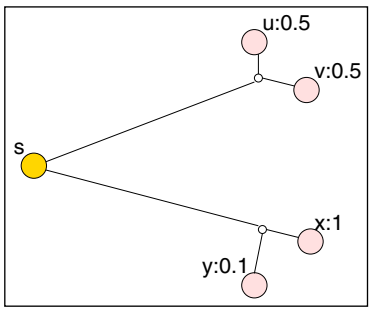

Fig. 14. An example showing that, in wGMP, the reduction ratio should be calculated in the same way as in GMP despite the differing demands of the destinations.

After the reduction ratios are computed and a pair of destinations are selected for targeting, the GMP algorithm computes the Steiner point between the source and these two destinations. At this step, GMP assumes that the destinations have the same packet ratio demands, in the sense that, in the calculation of Steiner point, the weights of the distances between the Steiner point to both destinations are the same. Therefore, when the packet ratio demands of destinations are different, routing based on the tree constructed by GMP may result in unnecessarily large amount of transmissions. Fig. 15 shows such an example. In this figure, node s is the source node and node u and v are the two destinations with packet ratio demands 1 and 0.1, respectively. Fig. 15(a) shows the paths routed using GMP. For 10 multicast data packets, the total number of hops to the two destinations is $4*10+1=41$. However, since most packets are forwarded to node u, the path from source node s to node u should be optimized as opposed in GMP. Fig. 15(b) shows another routing tree in which the path from s to u is shortest and the total number of hops from s to u and v are $3*10+2=32$, as compared to 41 by using GMP. On the other hand, as shown in Fig. 15(b), it is possible to reduce the required network transmissions, if we could consider the packet ratio demands of the destinations when calculating the Steiner point.

As already discussed in Section 4, for the case of three points, the exact Steiner point can be calculated efficiently. To take the packet ratio demand into consideration, we need to introduce the concept of weighted Steiner point: given three points A, B and C in the Euclidean space, and their corresponding weights w_A, w_B and w_C, the weighted Steiner point of these three points is the point p in the Euclidean space such that:

$$wd(p) = w_A * d(p,A) + w_B * d(p,B) + w_C * d(p,C) \qquad (6)$$

is minimum. To apply the weighted Steiner point for source s and destinations u and v with packet ratio demands w_u and w_v, respectively, we set the weight for s to be 1, and the weight of u to be $\frac{w_u}{max(w_u,w_v)}$ and the weight of v to be $\frac{w_v}{max(w_u,w_v)}$. From now on, we call the original Euclidean Steiner point as the un-weighted Steiner point. Although there is efficient solution for calculating the exact un-weighted Steiner point,[3] it turns out that the weighted Steiner point cannot be calculated as efficiently.[4] Therefore, in order to extend GMP for supporting weighted multicast trees, we

[3] The solution in [40] for calculating the exact Steiner point is by a geometric method.

[4] The geometric method in [40] is not applicable when the distances have weights.

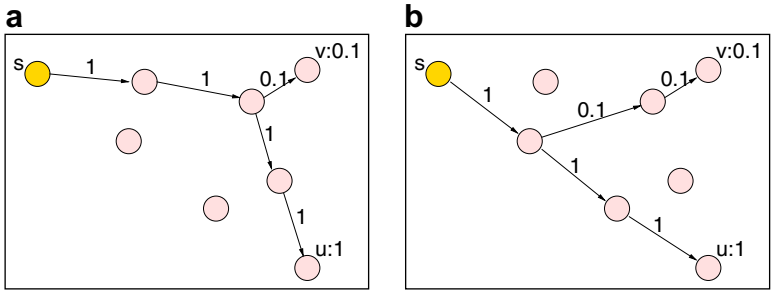

Fig. 15. (a) GMP algorithm results in an unnecessarily large number of transmissions, (b) an alternative route could save transmission and energy.

need an alternative approach to selecting the point p, which is efficiently computable and serves well as the target point for directing packets.

One approach would be to use a numerical approximation [17]. In this paper, we rely on the observation that, although Eq. (6) is hard to minimize, the following equation can be minimized for q exactly very efficiently

$$wsd(q) = w_A * d(q,A)^2 + w_B * d(q,B)^2 + w_C * d(q,C)^2 \quad (7)$$

Eq. (7) has two advantages: first of all, in general the difference between p (which minimizes Eq. (6)) and q tends to be small.[5] When we also consider that the calculated point may not actually correspond to a node in the network and that each forwarding node will anyhow adjust the tree based on it own information, this tradeoff between efficiency and accuracy pays off. Secondly, since the transmission power consumption in wireless networks is generally sensitive to the square of the distance i.e., $\rho = \alpha + \beta * d^{\gamma}$, where α and β are constants, d is the transmission distance, and the value of γ is normally between 2 and 4 [44], minimizing the squared value of the distances might in fact reflect the power consumption in wireless sensor networks better.

Thus, wGMP proceeds with the weighted *rrSTR* algorithm, $rrSTR_w$. In each round of $rrSTR_w$, the destination pair with the largest reduction ratio is selected (without considering weights). Then the weighted point, q, is calculated for this pair and a new virtual destination is created at this point. The packet ratio demand of the virtual destination is set to the maximum packet ratio demands of the two destinations. Consequently, the data rate demand of each node in the resulting tree is the maximum packet ratio within the subtree rooted at this node. As in GMP, the source node then splits the destinations into groups based on the resulting tree and identifies an appropriate next hop for each group. A copy of the data packet is created for each group and the destinations in this group are saved in the copy with new demand value being set for each destination. The source node then forwards the copy to its corresponding next hop based on the corresponding demand value. In case there is no appropriate next hop for any group of destinations, wGMP operates the same as GMP in Section 5.

[5] To test the difference between points p and q, we calculated the weighted total distance in Eq. (6) based on the approximate weighted Steiner point obtained by minimizing the squared version of Eq. (7) for a set of triangles whose vertices are randomly generated. We also calculated the minimum weighted total distance by using Maple software [15], which can achieve error value less than 0.001%. The average of the normalized differences between the weighted total distance obtained by the approximation and the Maple software was around 3% with a variance of 1% for a total of 100 triangles.

7. Evaluation of GMP and wGMP

7.1. Comparing GMP against existing geographic multicast protocols, LGS and PBM

In this section, we present the results of the experiments we carried to evaluate and compare the performance of the proposed GMP multicast protocol with the a set of location-aware multicast protocols shown in Table 1. To see the impact of radio range awareness, we also experimented with GMP^{nr}, the version of GMP in which radio range aware decisions have been turned off. For the sake of completeness, we also implemented a centralized algorithm [29], denoted by SRT, to calculate a static routing tree. This centralized algorithm assumes that the source node knows the positions of all sensor nodes in the network; thus the source node can calculate an approximate Steiner tree connecting itself and all destinations using the algorithm [29], which provides a 2 bound. The source node forwards a copy of the data packet with the routing information embedded in the packet. Naturally, acquiring the up-to-date global knowledge of the network topology is not practical for large sensor networks. Therefore, we include this centralized algorithm only for comparison purpose. We also experimented with an algorithm we referred as

Table 1
Algorithms compared in the experiments

Algorithm	Description
PBM	Position Based Multicast [36]
LGS	Location-Guided Steiner Tree [8]
SRT	Static routing tree constructed using global knowledge of the network [29]
GRD	Packets routed to each destination independently using a greedy algorithm
GMP	Geographic Multicast Protocol proposed in this paper
GMP^{nr}	A version of GMP with radio range aware feature turned off

GRD, which corresponds to the extreme case, where packets are independently routed for each destination in a greedy fashion. This algorithm explicitly minimizes the per-destination hop count and serves well as a lower-bound for the average number hops for each destination.

The simulation setup is described in Table 2. The NS2 simulator has been modified to support multicasting for wireless sensor networks. In each experiment, we generate 100 tasks. For each task, we randomly pick a node as the source node and randomly pick k nodes as the destination nodes. The value of k varied from 3 to 25. Each experiment is run on 10 different networks and results are averaged.

7.1.1. Total number of hops in the multicast tree

The total number of hops needed for a single multicasting task is the number of transmissions (forwarding) required to reach all destinations. Fig. 16 shows the results obtained using four different multicast routing protocols: PBM, LGS, GMP, and GMP^{nr}. In [36], it is noted that in PBM the minimum total number of hops is achieved for a λ value between 0 and 0.6. We have thus run the same routing task with the value of λ varying from 0 to 0.6. Among the results corresponding to these λ values, only the best (minimum number of hops) one is included for PBM in Fig. 16. In this figure, we see that, of the five protocols, GMP results in the least number of hops. The reduction of GMP compared to both PBM and LGS is up to 25%. The results indicate that by using rrSTR destinations are divided into groups more effectively and the way the next hop is calculated based on the Steiner point of the destinations in each group is more effective than the ways PBM and LGS calculate the next hops. As discussed in Section 4, when the radio range is not considered during the construction the Steiner tree, GMP^{nr} may generate redundant hops. The probability for redundancy becomes larger as the number of destinations becomes larger. Therefore, GMP^{nr} uses more hops than GMP. Note, however, that even without radio range awareness, GMP^{nr} works better than PBM and LGS.

As discussed earlier, in PBM the optimal value of the trade-off parameter λ depends on the number of neighbors and the distribution of the destinations. The value of λ, however, is fixed during routing, therefore, the destinations are split into groups by a receiving node sometimes earlier and sometimes later rather than being split by the node closest to the Steiner point of the destinations. Therefore, PBM may generate a larger total number of hops. LGS uses a MST heuristic in which geographic locations other than locations of nodes in the network are not taken into consideration. Therefore the groups identified by LGS may not be as good as those identified by GMP. Furthermore, the calculation of next hop in LGS is based on one of the destinations in the group. In GMP, the next hop is calculated based on the Steiner point of the destinations in this group relative to the current node, therefore, this results in better paths.

Note that, GMP performs even better than the centralized algorithm, SRT, despite SRT's use of global knowledge. We conjecture that this is because although both approaches have approximate/heuristic nature, the intermediate nodes in GMP are capable of adjusting the Steiner tree continuously with available local information.

Table 2
Simulation setup

SIM.Parameter	Value
Simulator	ns−2.27
Network size	1000 m × 1000 m
Number of nodes	1000
Channel data rate	1 Mbps
Mac protocol	Mac802.11
Transmission power	1.3 W
Receiving power	0.9 W
Message size	128 B
Antenna	OmniAntenna
Radio range	150 m

7.1.2. Per-destination hop count

Fig. 17 shows the average per-destination hop count. As discussed in Section 1, it is not always possible to minimize

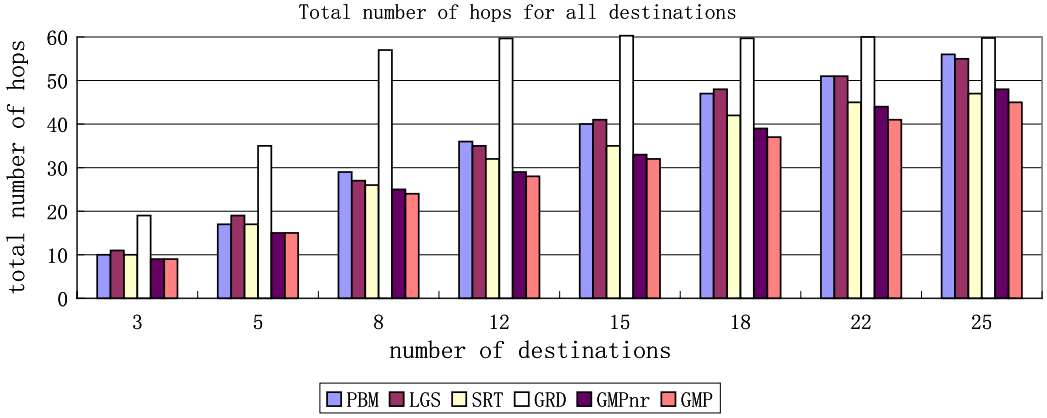

Fig. 16. Total hop count for the multicast tree (For large number of destinations, values of GRD are not shown completely since the required total number of hops using GRD are too high compared to others).

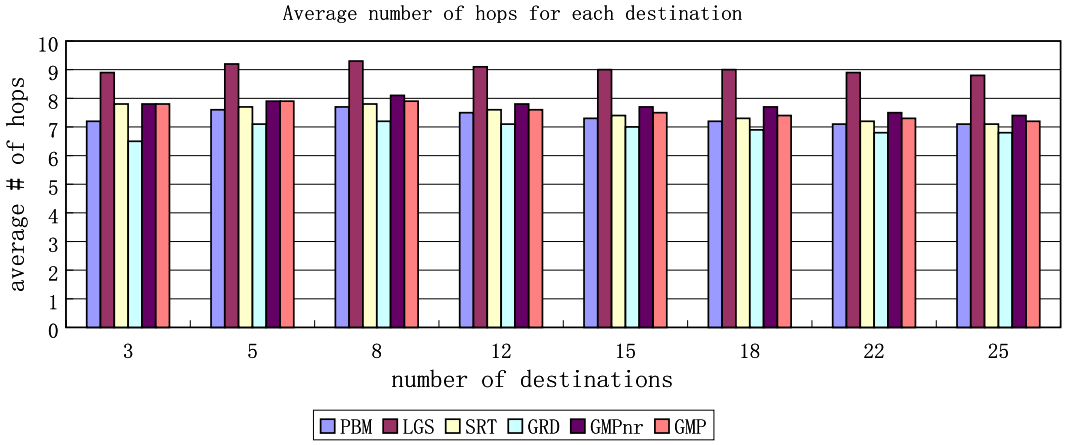

Fig. 17. Per-destination hop count. Naturally, since a greedy solution for reduced per-destination hop-count cannot exploit possible overlaps across different destinations and can result in a large total number of hops and, thus, high energy consumption in the multicast tree. Therefore, it is not acceptable as a multicast tree protocol, but serves well as a lower-bound on the per-destination hop count.

both the total number of hops and the per-destination hop count. PBM takes into account this by using a trade-off parameter, λ. When λ is small, PBM results in a smaller average number of hops and a larger total number of hops. When λ is larger, the average number of hops gets larger but the total number of hops becomes smaller. The value of λ in Fig. 17 is the same as in Fig. 16, i.e., the λ that minimizes the total number of hops.

We see that PBM, SRT, and GMP provide comparable per destination hop counts (close to the greedy solution, GRD). LGS does not match the others in this respect. In LGS, the next hop is chosen to be the neighbor closest to a destination that is closest to current node in the group. Routing in this way tends to reach the destinations sequentially and prevents the destinations from getting divided into groups at intermediate nodes. To see this, consider in Fig. 18. Let us assume that the packet is at node c and the remaining destinations are $\{u, v, d\}$. The MST created by LGS will consist of the following three edges: $\{\overline{cu}, \overline{uv}, \overline{vd}\}$. Thus, node c does not split the destinations into groups and forwards the packet towards u since u is closest to c. Consequently, the path taken by the LGS will eventually be $c \rightarrow \ldots \rightarrow u \rightarrow v \rightarrow \ldots \rightarrow d$. Contrast this with the tree-structured multicast route computed by GMP as illustrated in Fig. 1. Consequently, when LGS is used, destinations that are reached later may take a significantly larger number of hops and this increases the overall average number of hops.

7.1.3. Energy consumption

Fig. 19 shows the energy consumption during multicast routing.[6] We see that since the number of hops used by GMP is less than the number for all of PBM, LGS, and SRT, the energy consumption of GMP is significantly smaller than consumptions of PBM, LGS, and SRT in all configurations we tested. In these experiments, the energy savings compared to PBM and LGS is up to 25% as shown in Fig. 19.

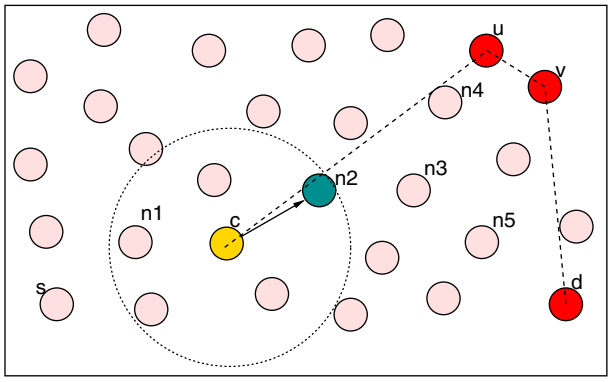

Fig. 18. An example where LGS results in a large per-destination hop count.

7.1.4. Routing delay

Fig. 20 shows the routing delay, defined as the maximum delay of all paths for all destinations, for different multicast algorithms. We see that the significant energy gains of GMP does not result in any disadvantage in terms of the delay performance; the delay characteristic of GMP is similar to those of PBM and SRT. As expected, GRD has the smallest delay; this is because in GRD, a copy of the packet forwarded separately is in a greedy fashion for each destination. On the other hand, since (as discussed in Section 7.1.2) LGS tends to reach destinations one by one, it results in the largest delay among all the experimented algorithms.

7.1.5. Failure ratio due to perimeter routing

When the network density is lower, the probability that there is no neighbor closer to the destination and that the

[6] The energy consumption reported in this paper includes the transmission power of senders as well as the receiving power of all listening nodes within the transmission radio range of the senders.

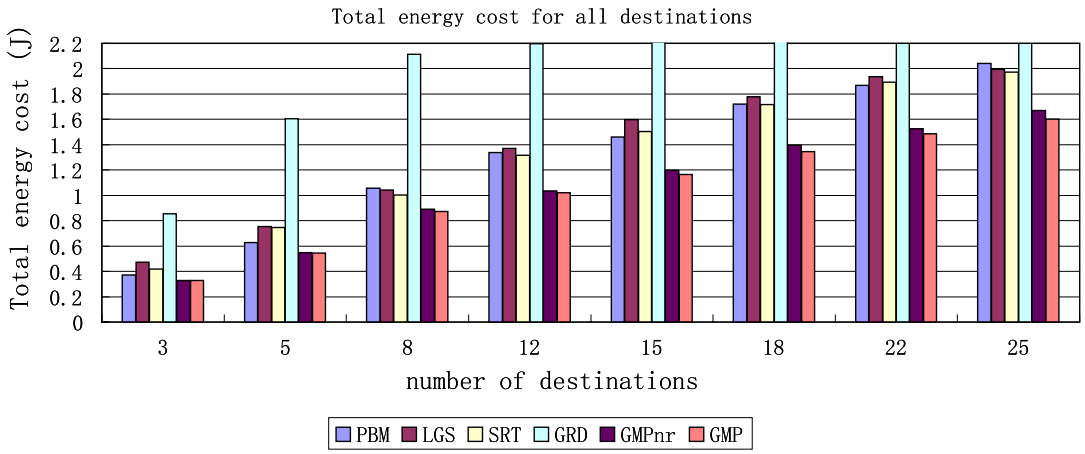

Fig. 19. Total energy cost of the multicast tree (For large number of destinations, values of GRD are not shown completely since the energy consumptions using GRD are too high compared to others).

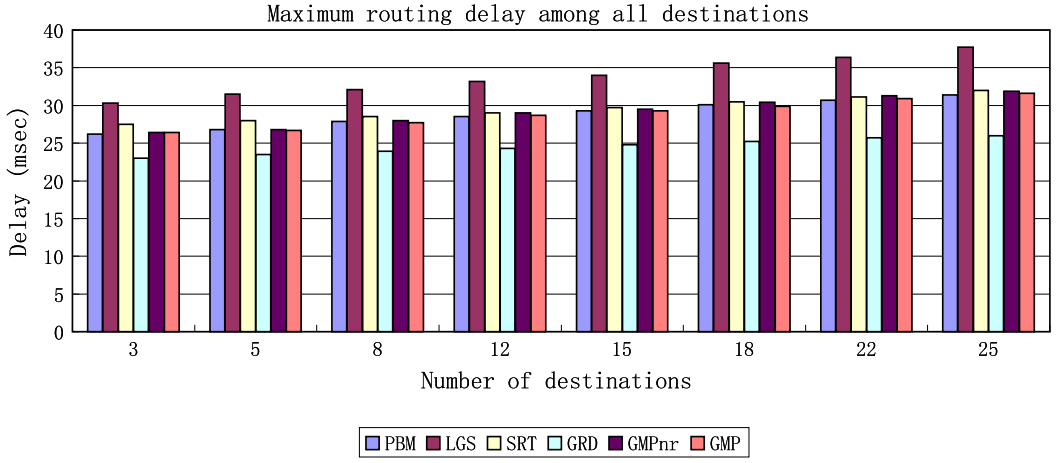

Fig. 20. Maximum delay for all destinations.

packet enters the perimeter mode becomes higher. Because traversing the void area follows the right hand rule, the length of the path may become large. Furthermore, this may result in failed deliveries, especially when there is an upper bound associated with the total number of hops allowed for each packet. To observe how GMP scales against low density networks, we also did experiments to test the performance of GMP in networks with different densities. In these experiments, the number of nodes in the network are 1000, 800, 600, and 400. We set the maximum path length for each destination to be 100, that is a packet is dropped if its hop count reaches 100. Each task in these experiments routes a message from a random source to 12 randomly generated destinations. A task fails if not all destinations are reached. For each network size, we generated 10 networks and ran 100 tasks for each network. Fig. 21 shows the total number of failed tasks in the total of 1000 tasks using three protocols PBM, LGS, GMP. (The other protocols do not use perimeter routing) We see that GMP has the least number of failures. LGS has the largest number of failures because it assumes a valid next hop can always be found and it fails when a void destination is identified even if the hop count is less than 100. PBM will group all the *void* destinations and always mark the packet to be in perimeter mode for these destinations. GMP, on the other hand, may group some *void* destinations with other destinations *without* setting the packet to perimeter mode for this group assuming a valid next hop can be found for this group as described in Section 5.

7.1.6. Location errors

In the above experiments, we assumed that each node knows the exact location of itself. However, in reality, the coordinates obtained through GPS or the calibration processes could be different from the actual coordinates of the sensor nodes. To study the effect of location errors, we have re-run the experiments while introducing random errors in each node's knowledge of its own coordinates. Since each node would broadcast its own coordinates, the node's coordinates maintained by its neighbors will also have error. Similarly, the destination coordinations maintained by the source node will also reflect these errors. In

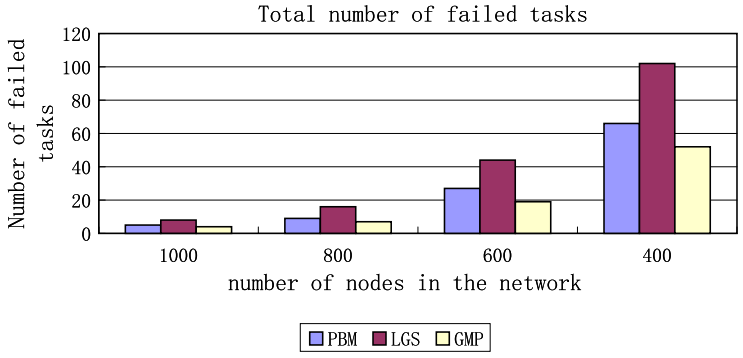

Fig. 21. Number of failed tasks for different network densities.

our experiments, the error is uniformly distributed within the range $[-E, E]$ along each dimension, where E was set to 10 m and 20 m for two different set of experiments.

Fig. 22 shows the corresponding results for total number of hops, average number of hops and total energy consumptions. Location error level 0 means that each node has the perfect location information. As expected, the performance of all routing algorithms suffer slightly when there are location errors. Yet, the comparison of performance between the multicast routing algorithms under location errors highly resembles the comparison between routing algorithms with perfect location information, illustrating that the same comparisons hold even when there are location errors.

7.2. Comparing demand-scalable multicasting protocols to their un-weighted versions

In this section, we present the results of the experiments we carried to evaluate and compare the performance of the proposed weighted multicast protocols against un-weighted version of these algorithms (described in detail in Section 3) as well as against each other. In each experiment, we generated 100 tasks. For each task, we randomly pick a node as the source node and randomly pick k nodes as the destination nodes. The value of k varied from 5 to 15. Each destination was assigned a packet demand ratio, corresponding to the

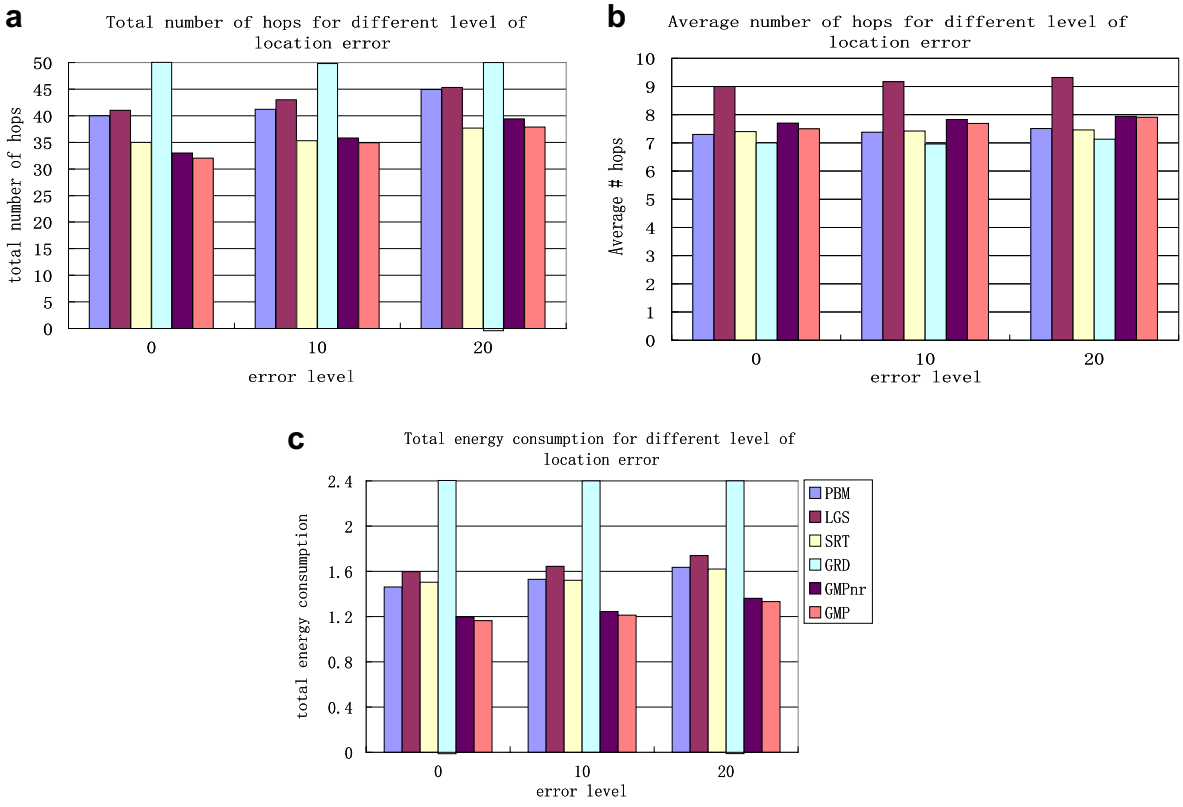

Fig. 22. Routing performance of multicast protocols when coordinates maintained by sensor nodes have errors. (a) Total number hops for all destinations; (b) Average number of hops for each destination; (c) Total energy consumption for all destinations.

average amount of data it would consume to match its corresponding quality demand.

Each experiment is run on 10 different networks and results are averaged. For each routing task, we ran all multicast routing algorithms discussed in this paper, including three un-weighted and three corresponding weighted multicast algorithms. We ran total of four set of experiments. In the first two sets of experiments, the packet demand ratio was randomly generated within a range. The ranges of the packet demand ratio were [0.5...1] and [0.1...1] for these two sets of experiments, respectively. In the last two sets of experiments, the packet demands ratio are randomly selected from two values. The two values are 0.5 and 1 for the third experiment set and 0.1 and 1 for the last experiment set. This second set of experiments highlight the case where, the variation between demands are larger.

Before we present and discuss the experiment results in detail, in Table 3, we first present the number of transmissions and energy consumption obtained by various algorithms, when there is no differences of demands among destinations. As can be seen in this table, the original and weighted versions of the algorithms have the same behavior when all demands are equivalent. This shows that the proposed modifications do not add additional overhead to the algorithms.

7.2.1. Total number of transmissions

Table 4 shows the total number of transmissions required in the experiments when using different geographic multicast routing algorithms. We can see that compared to the un-weighted versions, the weighted version of the multicast algorithms in general result in better performance. When we look carefully though, we see that for various scenarios wPBM actually performed worse than PBM. There are several reasons for this. First, as described in Section 3, PBM uses a constant parameter λ to balance the bandwidth usage on the current node and the total distance to the remaining destinations. The optimal value of λ is difficult to be pre-calculated because it depends on the specifics of each task as well as the node and demand distributions. To be fair for the comparison of PBM and wPBM, in the experiments, we fixed the λ to be 0.5 for both PBM and wPBM. Since the performance of PBM and wPBM depends on this constant parameter, it will similarly affect both protocols. The second factor that negatively affect the performance of wPBM is the way the weighted total distance to the remaining destinations is calculated. Although wPBM intuitively multiplies the distance with the demand ratio, the errors for such estimations are still too large, because the distance considered is the direct distance between a neighbor node and a destination despite the fact that some destinations can share subpaths. The

Table 3
Total number of transmissions and total energy cost when all destinations have the same demand; the number of destinations is 15

	PBM	LGS	GMP	wPBM	wLGS	wGMP
Total # of transmissions	2458	2437	2141	2458	2437	2141
Total energy cost (*J*)	86.27	86.42	76.20	86.27	86.42	76.20

Table 4
Total number of transmissions

Demand	# Dest.	PBM	LGS	GMP	wPBM	wLGS	wGMP
Number of Transmissions (100 tasks)							
[0.5...1]	5	775	760	679	752	746	631
	8	1216	1208	1079	1232	1178	987
	12	1488	1454	1306	1438	1418	1208
	15	1898	1850	1670	1887	1796	1531
[0.1...1]	5	610	599	536	608	580	502
	8	996	980	886	987	958	808
	12	1229	1199	1070	1217	1164	993
	15	1472	1436	1280	1464	1376	1170
{0.5,1}	5	792	781	707	762	763	656
	8	1294	1274	1152	1265	1228	1043
	12	1638	1569	1413	1633	1514	1267
	15	2024	1987	1793	2046	1927	1609
{0.1,1}	5	688	644	576	656	632	513
	8	1058	1029	922	1049	988	825
	12	1263	1258	1140	1273	1214	1062
	15	1553	1515	1367	1545	1458	1233

result is that any possible savings due to weighted adaptation is largely lost.

wLGS, on the other hand provide some reduction as opposed to LGS. This is because wLGS tries to route to destinations with larger demand ratio first, and effectively avoids the cases that routing packets of less demand destinations through larger demand destinations when the larger demand destinations are further away than the less demand destinations. Yet the reduction is only around 3%. This is partly because wLGS may adversely overly split the destinations into multiple groups. To see this, consider the case in Fig. 23. In Fig. 23(a), by wLGS, node *s* will split the two destinations into two groups. However, it is more desirable to consider two destinations as one group and split them at node *z* as shown in Fig. 23(b).

wGMP on the other hand provide up to 10% reduction in the number of transmissions. Compared to GMP, wGMP performed better because with the help of weighted Steiner point, in the tree constructed by wGMP the length of paths to those destinations with larger demand ratio is shorter than that by GMP, required by the fact that more packets will be routed to these destinations. This is achieved through shorter paths to destinations with higher demand ratio. In fact, wGMP achieves the best performance for all routing tasks among all the multicast routing algorithms. Compared to wLGS, the tree structure constructed by wGMP is more effective. As discussed above, wLGS tends to overly divide the destinations.

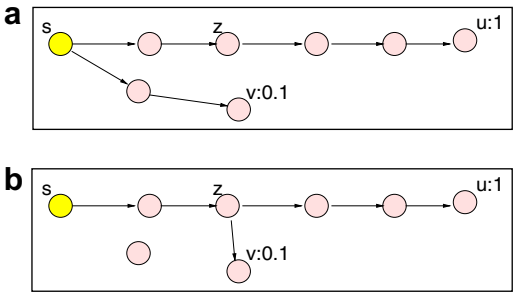

Fig. 23. wLGS adversely splits the destinations into groups.

Note that (interestingly) the total amount of transmissions required by wPBM and wLGS are still larger than the unweighted GMP algorithm. The weight adaptivity achieved by wPBM and wLGS are not enough to overcome the advantages of the GMP algorithm: guided by the reduction ratio and the *rrSTR* algorithm, GMP can split the destinations effectively. In addition, the virtual destinations also help GMP split the destinations at more appropriate intermediate relay node.

Finally, note also that, as expected, the experiments with lower data demands require less transmissions than the experiments with higher demands. To see this, please compare results for sets [0.1...1] and {0.1,1} against sets [0.5...1] and {0.5,1}. Also, when we compare experiment sets {0.1,1} and {0.5,1} against experiment sets [0.1...1] and [0.5...1], we see that larger variability in demand require more transmission.

7.2.2. Total energy cost

Although, we have shown that wGMP provides the best reduction in the number of transmissions, it is still possible that in terms of the actual energy consumption, one of the other algorithms can provide better savings. For this purpose, we also computed the transmission energy consumption by all algorithms. Table 5 shows the energy consumption during the multicast routing. We see that, once again, when considered all variations in the number of receiving nodes in different tasks and the number of destinations in each group chosen by different routing protocols, the energy consumption of wGMP is still the smallest among all multicast protocols in all configurations we tested. Further-

Table 5
Total energy cost (expressed in terms of Joules, *J*)

Demand	# Dest.	PBM	LGS	GMP	wPBM	wLGS	wGMP
Energy Consumption (100 tasks)							
[0.5...1]	5	28.04	27.63	24.19	27.31	26.67	22.60
	8	43.45	43.87	39.50	44.87	42.31	35.47
	12	54.08	52.92	47.11	51.50	51.03	43.99
	15	69.38	66.79	60.79	68.15	64.20	55.05
[0.1...1]	5	21.80	21.51	19.37	21.92	20.67	18.11
	8	36.22	35.19	32.12	35.36	34.08	29.21
	12	43.77	43.83	38.79	44.23	42.43	36.09
	15	52.62	51.99	46.45	53.53	50.25	42.77
{0.5,1}	5	28.56	27.85	25.16	27.17	27.65	23.80
	8	47.39	45.53	41.52	46.06	44.29	37.54
	12	59.96	57.00	50.57	59.40	54.72	46.17
	15	72.36	72.22	63.80	72.87	69.33	58.69
{0.1,1}	5	24.88	23.23	20.54	23.34	23.04	18.34
	8	38.54	37.05	33.71	38.08	35.22	30.02
	12	45.84	45.90	41.44	46.52	43.80	38.11
	15	55.39	54.75	49.21	56.36	52.23	44.87

more, the same pattern of observations also apply when individual geographic multicasting algorithms are compared.

8. Conclusion and future work

In this paper, we first presented a *geographic multicast routing protocol,* GMP, built on a novel and efficient heuristic, rrSTR, for constructing radio-range aware Euclidean Steiner trees. The computational complexity of GMP is significantly lower than that of PBM and comparable to that of LGS. Furthermore, simulation results on NS2 showed that the average number of hops on the multicast trees created by GMP is significantly lower than that of LGS and comparable to that of PBM. Therefore, GMP provides the best features of both alternatives. In fact, in terms of *total number of hops and energy consumption*, GMP is around 25% better than both of these alternatives. We also proposed intuitive mechanisms (wGMP, wLGS, wPBM) to adapt these multicast routing schemes when the data needs to scale based on the consumer demand. The results show that the wGMP algorithm provides the best opportunities for scalability due to its flexible, self-correcting decision making process. Other schemes are not directly suitable for scalable multicasting, due to their greedy structures not amenable for self correction.

In all the multicast protocols discussed in this paper, the group membership information is assumed known by the source node prior to the data dissemination. In our future work, we seek to address the problem of efficiently maintaining and communicating the group membership information. We are also investigating ways for extending our protocol to enable adaptations to dynamic topology changes due to sleep-and-wake-up based energy conservation of the nodes and for explicit power optimizations by leveraging GPER-like radio range adaptations [62,64].

References

[1] S. Basagni, I. Chlamatac, V. Syrotiuk, B. Woodward, A distance routing effect algorithm for mobility(dream), in: MOBICOM, 1998.

[2] J.E. Beasley, A heuristic for euclidean and rectilinear steiner problems, European Journal of Operational Research 58 (1992) 284–292.

[3] J. Boleng, T. Camp, V. Tolety, Mesh-based geocast routing protocols in an ad hoc network, IPDPS (2001) 184–193.

[4] J.-C. Bolot, T. Turletti, I. Wakeman, Scalable feedback control for multicast video distribution in the internet, in: SIGCOMM '94: Proceedings of the conference on Communications architectures, protocols and applications, 1994, 58–67.

[5] E. Bommaih, M. Liu, A. McAuley, R. Talpade, Amroute: Ad hoc multicast routing protocl, in: Internet-Draft, 1998.

[6] P. Bose, P. Morin, I. Stojmenovic, J. Urrutia, Routing with guaranteed delivery in ad hoc wireless networks, in: DIAL-M, 1999.

[7] Q. Cao, T. He, T.F. AbdelZaher, ucast: Unified connectionless multicast for energy efficient content distribution in sensor networks, TPDS (2006).

[8] K. Chen, K. Nahrstedt, Effective location-guided tree construction algorithms for small group multicast in manet, in: INFOCOM, 2002, 1180–1189.

[9] C. Chiang, M. Gerla, L. Zhang, Forwarding group multicast protocol (fgmp) for multihop, mobile wireless networks, Cluster Computing 1 (2) (1998) 187–196.

[10] I. Chlamtac, S. Basagni, V. Syrotiuk, Location aware, dependable multicast for mobile ad hoc networks, Computer Networks 36 (5-6) (2001) 659–670.

[11] M.S. Corson, S.G. Batsell, A reservation-based multicast (rmb) routing protocol for mobile networks, in: INFOCOM, 1995.

[12] S. Das, B. Manoj, C. Murthy, A dynamic core based multicast routing protocol for ad hoc wireless networks, in: MOBIHOC, 2002.

[13] W.-H.L. et al., "geogrid": A geocasting protocol for mobile ad hoc networks based on grid, Journal of Internet Technology 1 (2) (2000) 192–213.

[14] K. Gabriel, R. Sokal, A new statistical approach to geographic variation analysis, Systematic Zoology 18 (1969) 259–278.

[15] W. Gander, J.H. Rebcek, Solving problems in scientific computing using Maple and Matlab, Springer Verlag, New York, 1993.

[16] J. Garcia-Luna-Aceves, E. Madruga, the core-assisted mesh protocol, IEEE Journal on Selected Areas in Communications 17 (8) (1999) 1380–1394.

[17] P.E. Gill, W. Murray, M.H. Wright, Practical optimization, Academic Press, New York, 1981.

[18] C. Gröpl, S. Hougardy, T. Nierhoff, H.J. Prömel, Approximation algorithms for the steiner tree problem in graphs, in:Technical Report, 2000.

[19] Q. Huang, C. Lu, G. Roman, Spatiotemporal multicast in sensor networks, in: SenSys, 2003.

[20] Q. Huang, C. Lu, G.-C. Roman, Spatiotemporal multicast in sensor networks, in: SenSys, 2003.

[21] J. Jetcheva, D.B. Johnson, Adaptive demand-driven multicast routing in multi-hop wireless ad hoc networks, in: MOBIHOC, (2001) 33–44.

[22] L. Ji, M.S. Corson, Differential destination multicast – a manet multicast routing protocol for small groups, in: INFOCOM, 2001.

[23] B. Karp, H.T. Kung, GPSR: greedy perimeter stateless routing for wireless networks, in: MobiCom (2000) 243–254.

[24] R. Karp, Reducibility among combinatorial problems, Complexity of Computer Computations (1972) 85–103..

[25] Y. Ko, N.H. Vaidya, Geocasting in mobile ad hoc networks: Location-based multicast algorithms, in: WMCSA (1999) 101–110.

[26] Y.B. Ko, N.H. Vaidya, Location-aided routing (LAR) in mobile ad hoc networks, in: MobiCom (1998) 66–75.

[27] Y.-B. Ko, N.H. Vaidya, Geotora: A protocol for geocasting in mobile ad hoc networks, in: ICNP (2000) 240–250.

[28] Y.B. Ko, N.H. Vaidya, Flooding-based geocasting protocols for mobile ad hoc networks, Mobile Networks and Applications 7 (6) (2002) 471–480.

[29] L. Kou, G. Markowsky, L. Berman, A fast algorithm for steiner trees, Acta Informatica 15 (1981) 141–145.

[30] E. Kranakis, H. Singh, J. Urrutia, Compass routing on geometric networks, in: CCCG, (1999).

[31] F. Kuhn, R. Wattenhofer, Y. Zhang, A. Zollinger, Geometric ad-hoc routing: of theory and practice, in: PODC (2003) 63–72.

[32] S. Lee, M. Gerla, C. Chiang, On demand multicast routing protocol, in: WCNC (1999) 1298–1302.

[33] S. Lee, C. Kim, Neighbor supproting ad hoc multicast routing protocol, in: MOBIHOC, (2000) 37–50.

[35] J. Liu, J. Liu, J. Reich, P. Cheung, F. Zhao, Distributed group management in sensor networks: Algorithms and applications to localization and tracking, Telecommunication Systems 26 (2-4) (2004) 235–251.

[36] M. Mauve, H. Fübler, J. Widmer, T. Lang, Position-based multicast routing for mobile ad-hoc networks, in: MOBIHOC Poster, 2003.

[37] S. McCanne, V. Jacobson, M. Vetterli, Receiver-driven layered multicast, in: ACM SIGCOMM (1996) 117–130.

[38] I.I. Mändoiu, V.V. Vazirani, J.L. Ganley, A new heuristic for rectilinear steiner trees, in: ICCAD (1999) 157–162.

[39] M. Minoux, Efficient greedy heuristics for steiner tree problems using reoptimization and supermodularity, INFOR 28 (1990) 221–233.
[40] J. Neuberg, Sur le point de steiner, in: Journal de mathématiques spéciales (1886) 29.
[41] Q. Ni, Q. Zhang, W. Zhu, Sarlm: Sender-adaptive and receiver-driven layered multicasting for scalable video, in: ICME, 2001.
[42] T. Ozaki, J. Kim, T. Suda, Bandwidth efficient multicast routing protocol for ad hoc networks, in: ICCCN (1999) 10–17.
[43] E.M. Palmer, Graphical evaolutions: An introduction to the theory of random graphs, WI Series in Discrete Mathematics (1985).
[44] T.S. Rappaport, Wireless Communications: Principles and Practice, Prentice Hall, 1996.
[45] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, S. Shenker, Ght: A geographic hash table for data-centric storage in sensornets, in: WSNA (2002).
[46] G. Robins, A. Zelikovsky, Improved steiner tree approximation in graphs, in: Proc. Symposium on Discrete Algorithms (2000).
[47] V. Rodoplu, T. Meng, Minimum energy mobile wireless networks, in: ICC (1998) 1633–1639.
[48] E. Royer, C. Perkins, Multicasting using ad hoc on demand distance vector routing, in: MOBICOM (1999) 207–218.
[49] J. Sanchez, P. Ruiz, X. Liu, I. Stojmenovic, Gmr: Geographic multicast routing for wireless sensor networks, in: SECON 2006.
[50] C. Schurgers, V. Tsiatsis, S. Ganeriwal, M. Srivastava, Topology management for sensor networks: exploiting latency and density, in: MobiHoc'02: Proceedings of the 3rd ACM International Symposium on Mobile ad hoc Networking & Computing (2002) 135–145.
[51] K. Seada, M. Zuniga, A. Helmy, B. Krishnamachari, Energy-efficient forwarding strategies for geographic routing in lossy wireless sensor networks, in: Sensys (2004).
[52] N. Shacham, Multipoint communication by hierarchically encoded data, IEEE Infocom 92 (1992) 2107–2114.
[53] A. Sheth, B. Shucker, R. Han, Vlm: A very lightweight mobile multicast system for wireless sensor networks, in: WCNC (2003).
[54] P. Sinha, S. Sivakumar, V. Bharghavan, Mcedar: Multicast core extraction distributed ad hoc routing, in: WCNC (1999) 1313–1317.
[55] I. Stojmenovic, X. Lin, Power aware localized routing in wireless networks, IEEE Transactions on Parallel and Distributed Systems 12 (11) (2001) 1122–1133.
[56] I. Stojmenovic, A.P. Ruhil, D.K. Lobiyal, Voronoi diagram and convex hull-based geocasting and routing in wireless networks, in: ISCC (2001) 51–56.
[57] H. Takagi, L. Kleinrock, Optimal transmission ranges for randomly distributed packet radio terminals, IEEE Transactions on Communications 32 (2) (1984) 246–257.
[58] G. Toussaint, The relative neighborhood graph of a finite planar set, Pattern Recognition 12 (4) (1980) 261–268.
[59] B.J. Vickers, C. Albuquerque, T. Suda, Adaptive multicast of multi-layered video: Rate-based and credit-based approaches, in: INFOCOM (3) (1998) 1073–1083.
[60] B.J. Vickers, M. Lee, T. Suda, Feedback control mechanisms for real-time multipoint video services, IEEE Journal of Selected Areas in Communications 15 (3) (1997) 512–530.
[61] C. Wu, Y. Tay, C. Toh, Ad hoc multicast routing protocol utilizing increasing id-numbers functional specification, Intenet-Draft (1998).
[62] S. Wu, K.S. Candan, Gper: Geographic power efficient routing in sensor networks, in: ICNP (2004) 161–172.
[63] S. Wu, K.S. Candan, Gmp: Distributed geographic multicast routing in wireless sensor networks, in: ICDCS (2006).
[64] S. Wu, K.S. Candan, Power-aware single- and multipath geographic routing in sensor networks, Ad Hoc Networks (2006).
[65] Y. Xu, J.S. Heidemann, D. Estrin, Geography-informed energy conservation for ad hoc routing, in: MobiCom (2001) 70–84.
[66] Y. Xue, B. Li, A location-aided power-aware routing protocol in mobile ad hoc networks, in: IEEE Globecom (2001) 25–29.
[67] M. Zachariasen, P. Winter, Concatenation-based greedy heuristics for the euclidean steiner tree problem, Algorithmica 25 (1999) 418–437.
[68] A. Zelikovsky, Better approximation bounds for the network and euclidean steiner tree problems, in: Technical Report (1996).
[69] M. Zorzi, P.R. Rao, Geographic random forwarding (geraf) for ad hoc and sensor networks: multihop performance, IEEE Trans Mobile Computing 2 (2003) 337–348.

Shibo Wu received his B.S. degree in the department of Precision Instruments and Mechanology, China in 2000. He is currently working on his PhD. Degree in the Department of Computer Science and Engineering at Arizona State University. His current research focus is power aware routing in wireless sensor networks. In particular, he is developing unicast, multicast, and multi-source single destination data aggregation algorithms and protocols for power constrained ad-hoc sensor network. He has published his research at various premier networking and distributed computing venues.

K. Selçuk Candan is an associate professor at the Department of Computer Science and Engineering at the Arizona State University. He joined the department in August 1997, after receiving his Ph.D. from the Computer Science Department at the University of Maryland at College Park. He has worked extensively on the management of distributed information and information sources. His current research interests include quality adaptive data processing and indexing, distributed sensory data management, and technologies for efficient dynamic content delivery over different types of network architectures. He has published various articles in respected journals and conferences in related areas. He received his B.S. degree, first ranked in the department, in computer science from Bilkent University in Turkey in 1993.