Determination of Aggregation Points in Wireless Sensor Networks

Utz Roedig, Andre Barroso and Cormac J. Sreenan
Mobile & Internet Systems Laboratory (MISL), University College Cork, Ireland
Email: {u.roedig|a.barroso|c.sreenan}@cs.ucc.ie

Abstract

A primary goal in the design of wireless sensor networks is lifetime maximization, constrained by the energy capacity of batteries. One well known method to reduce energy consumption in such networks is message aggregation. This method reduces the number of messages transmitted in the network, thus extending its lifetime. In order to be effective, aggregation requires messages to be delayed in their path throughout the network. This technique, therefore, must define where and for how long a message should be delayed. This paper shows how aggregation points and corresponding aggregation delays can be determined so that the overall energy efficiency of the system is improved. An algorithm is presented which allows the efficient computation of aggregation points and delays. The benefits of the algorithm is verified through simulation experiments.

1. Introduction

Wireless sensor networks are collections of autonomous devices with computational, sensing and wireless communication capabilities. Research in such networks has been growing steadily in the past few years given the wide range of applications that can benefit from such a technology.

Several applications, such as precision farming, military field monitoring and seismic activity monitoring require long lived deployments and possibly a very large number of sensors. The lifetime of a sensor field is mainly determined by the nodes battery lifetime. Therefore the lifetime of a sensor field is influenced by the usage pattern of the nodes batteries.

Currently, different methods are under investigation to improve the lifespan of wireless sensor networks. Generally, techniques are proposed for different components and levels of the system to reduce energy consumption. A different approach is to increase lifespan through energy harvesting from the environment. In this paper, we investigate *aggregation of sensor information* as a method to improve energy efficiency.

Messages routed through a sensor network can be combined, thus reducing the overall traffic in the system. Since transmitting/receiving messages in wireless sensor networks are power intensive operations, decreasing the amount of messages reduces the total energy consumed.

In order to be effective, aggregation requires messages to be delayed in their path throughout the network. If a message is delayed for a certain period of time in a sensor node, a second message might arrive and aggregation can take place. Despite its importance, the problem of defining where and for how long messages should be delayed in the network to improve aggregation results has been largely neglected in the literature.

This paper shows that the decision of where a message should be delayed and its duration has a significant influence on the aggregation efficacy. It presents an algorithm which allows the efficient computation of aggregation points and delays. Thus better results than by applying straightforward approaches can be achieved. The benefits of the algorithm is verified through simulation experiments.

The rest of the paper is organized as follows. In the remaining paragraphs of this section the subset of wireless sensor network applications for which this study is valid is described and the source of message delays used for aggregation is discussed. Section 2 defines the terms *Aggregation*, *Total Aggregation Gain* and *Aggregation Delay*. An algorithm to determine aggregation points and delays for messages generated by sensors is presented. In Section 3, practical implementation issues of the referred algorithm are discussed. An instance of the algorithm is proposed which allows efficient implementation. Section 4 evaluates through simulation the algorithm instance described in Section 3. Section 5 presents related work. The paper closes with a summary of the findings of our research.

1.1. Application Scenarios

In this paper, only sensor fields with a single base station (sink) are considered. The base station issues requests to the field and sensors matching the request periodically report data back. Messages containing the sensed data are transported hop by hop towards the sink through relatively stable paths established by some routing mechanism.

Example. A temperature monitoring system where sensors are placed in different rooms of a building is an example of a wireless sensor network application with the aforementioned structure. The base station issues requests of the form "report temperature if it differs from the previously reported value by more than 1 degree". This request is flooded to all temperature sensors in the building. Sensors report the local temperature if significant changes occur. The application collects these messages in the base station and calculates a temperature map of the building.

1.2. Message Delay

Different applications running on the top of a sensor network have different requirements for the information extracted from the field. One possible requirement is the maximum delay T for data delivery. If information extracted from the sensor field needs longer than T to reach the sink, the information is too old and can not be processed by the base station. This allowed message delay can be used within the sensor nodes to perform message aggregation.

Example. Consider the temperature monitoring system just discussed. Certain instances of this system allow message delays of several minutes, as the temperature might not drift significantly during this time and/or inconsistency of few degrees could be tolerated. However, if the message delay gets too high, the sensor information can not be used anymore to determine an acurate temprature map.

2. Data Aggregation

In this section the term aggregation is defined and it is discussed how aggregation points and corresponding aggregation delays for a message can be found.

2.1. Definitions

The term data aggregation can be applied to a range of different operations taking place inside a network. The following definition, based on [1], is used for the term data aggregation throughout the paper:

Data aggregation is the task of merging messages while they are traveling through the sensor network.

There are two basic approaches for eliminating redundancy on the data collected. The first approach operates at the packet level. This can be done by dropping packets that were already seen in the network, by merging two or more packets in order to reduce overhead or by the use of compression techniques. The second approach operates at the application level and uses the network as a platform to compute functions on data [3]. According to this method, data is pre-processed in the sensor field before it is extracted for external analysis. Combinations of both approaches are possible. In the following examples for both aggregation methods are given:

- *Packet level aggregation:* Consider the sensor node used in the D-Systems Project [6] at UC Cork. The node uses a Nordic nRF2401 transceiver that is able to transmit data in bursts of fixed size. The energy spent for transmission is therefore the same for packets fully or partially filled.
- *Application level aggregation:* Consider the sink is interested in the maximum of a sensed value. If two packets containing a measurement value are received by one node at the same time, the maximum can be determined by the node. Thus, only one message containing the maximum has to be forwarded to the next hop.

Naming Convention. For presentation clarity, this paper always refers to the problem of aggregation through an instance case where messages originated at a sensor s_h, distant h routing hops away from the base station, must be combined with other messages generated in the system. A node s_n, with $n \in \{0, 1, 2, \ldots, h\}$, is assumed to belong to the routing path of s_h and to be located n path hops away from the base station.

Total Aggregation Gain G. In sensor networks powered by batteries, an important factor is the lifetime of nodes, constrained by the capacity of the batteries employed. The communication subsystem of sensor nodes is specially power intensive and reducing the number of packets transmitted through aggregation can be used to extend the lifetime of a sensor field significantly.

The total aggregation gain G is a measure of the benefits of applying aggregation to the system in terms of communication traffic reduction. Let t_0 be the number of transmissions that are necessary to perform a given application level task. The number of transmissions to perform the same task when aggregation is applied is represented by t_a. The following expression defines the total aggregation gain:

$$G = 1 - \frac{t_a}{t_0} \tag{1}$$

Aggregation Delay T_n. Aggregation is only possible if messages are routed through common nodes along the path to the base station. In addition, messages have to be delayed along their path. This delay is introduced by holding messages before forwarding them to the next node. During this time, other messages might arrive and aggregation with the already present messages can take place.

As previously mentioned in Section 1.1.2, applications might allow a maximum delay T for data propagation within the network. A portion T_n of this delay can be used within a sensor node s_n to allow aggregation activities. T_n is referred as aggregation delay and must observe the following condition:

$$\sum_{n=1}^{h} T_n \leq T \qquad (2)$$

In other words, the sum of aggregation delays along the path to the base station cannot surpass the maximum message delay allowed by the application.

2.2. Determination of Aggregation Delays

Selecting appropriate aggregation delays for messages routed to the base station is critical for maximizing gains. An algorithm able to compute aggregation delays in order to maximize the total aggregation gain G is therefore desirable. In the following paragraphs, the problem of designing one such an algorithm is discussed and a solution is proposed.

Challenges. First, it is undesirable, given the amount of communication operations involved, to employ solutions making use of global state to perform their activities. This constraint prevents the use of an optimal solution for the problem of determining aggregation points and delays.

Second, for a given injection of packets in the system, aggregation alters their number as packets are moved inside the network. Additionally, delays introduced for aggregation purposes change the network traffic pattern. The resulting "feedback" mechanism makes the problem of determining aggregation points and delays difficult.

Assumptions. This study is conducted in the application scenario introduced in Section 1.1.1 with the further assumptions discussed here. The network is comprised of N sensors organized in a connected graph. No other constraints regarding the structure of the field are imposed. Each message generated in the network has a maximum allowed delay for delivery to the base station of T. The occurrence of events - the generation of new messages - in the field is homogeneously distributed with an average of $m > 1$ messages per T time units. It is further assumed that T is far greater then the time needed to transport a message and therefore the transmission time of messages can be neglected. A data message created by a sensor s_h travels along the path selected by the routing protocol towards the sink ($s_h \rightarrow \ldots \rightarrow s_1 \rightarrow s_0$).

In order to propose a solution to the problem, given the aforementioned challenges, optimality is neglected in favour of feasibility. The algorithm presented in this section attempts to maximize G through a localized approach under the following independence assumptions:

1) The total aggregation gain G is maximized by maximizing the individual contribution of each message to the total gain.

2)The arrival pattern of messages at nodes in the routing path of a message generated at time t is unaltered in the interval [t, t+T]

3) The maximum delay time T of a message is never reduced as a result of aggregation.

These independence assumptions, although not perfectly true, allows the problem to be divided into smaller approachable parts.

Probability. The time T_n a message should be delayed in node s_n for aggregation purposes is dependent on the probability of other messages arriving at the node. If s_n is holding a message when another message arrives, an aggregation event A_n occurs. Clearly, in regions close to the base station, the traffic density will be higher than in the periphery of the sensor field. Therefore, the probability $P_n(A_n)$ of an aggregation event in a sensor s_n is in general higher the closer it is to the base station. In addition, P_n is also dependent on the delay T_n allowed for aggregation in s_n. The longer a message is held in a sensor, the more likely it is for another message to arrive.

Probability P_n can be calculated as follows. Messages are generated uniformly in the interval T, what allows the discretization of time period T into m slots. In each slot, exactly one message is generated on average in the field. The probability of a message arriving at a node s_n during a time slot depends on the number w_n of nodes routing through it. Specifically, for each slot, a message may arrive or not with probability w_n/N. Let m_n be the average amount of messages generated in the field during aggregation delay T_n. The Probability P_n of a message arriving at node s_n in m_n time slots can then be calculated as shown in equation (3):

$$P_n(A_n) = 1 - \left(\frac{N - w_n}{N}\right)^{m_n} \qquad (3)$$

Note that m_n must fulfill the following condition:

$$\sum_{n=1}^{h} m_n \leq m \qquad (4)$$

The overall aggregation probability P for a message is the probability of an aggregation event taking place at any node in the path towards the base station. According to independence assumption 2, aggregation events, which are based on the arrival of messages, are independent. Therefore, if all aggregation probabilities P_n along the routing path of the message are known, the overall aggregation probability P for the message can be computed as follows:

$$P\left(\bigcup_{n=1}^{h} A_n\right) = \sum_{k=1}^{h} P_k - \sum_{k=1}^{h}\sum_{l=1}^{h}(P_k \cdot P_l) + \ldots + (-1)^{n-1} \cdot \prod_{k=1}^{h} P_k \qquad (5)$$

Gain. Different choices of aggregation delays along the path of a message produce different results in terms of energy savings. If a message is aggregated with another message at node s_h, then h transmissions can be saved as a consequence of the operation.

The expected individual aggregation gain of a message, represented as g, is the expected amount of transmissions that will be saved by selecting a specific combination of aggregation delays along the path towards the base station. This expected gain is determined as follows. Given that aggregation events are independent, each term of equation 5 represents the probability of a certain combination of aggregation events. Multiplying the probability of each such a combination with the number of transmission saved results in the expected individual aggregation gain g. Equation 6 shows how g is computed for normalized transmission savings (divided by h):

$$g = \sum_{k=1}^{h} P_k \cdot \frac{k}{h} - \sum_{k=1}^{h}\sum_{l=1}^{h} (P_k \cdot P_l) \cdot \left(\frac{k+l}{2 \cdot h}\right) + \dots + (-1)^{n-1} \cdot \prod_{k=1}^{h} P_k \cdot \sum_{l=i}^{h} \frac{l}{h^2} \quad (6)$$

As stated in independence assumption 1, the total aggregation gain G is maximized if the expected individual aggregation gain g for each message is maximized.

Brute-Force Algorithm. The optimal solution for the aggregation problem in the scenario presented can be obtained by computing, for each message generated, the gain g of all possible delays along the path to the base station. The set of delays T_n, $1 \le n \le h$ with maximum gain should then be used to hold the message in each node of the path. The amount of combinations to be checked is determined by the number of hops h in the path and the number of messages m generated in the field during T. Indeed, considering the discretization of time T_n into m_n time slots, the set X of all possible combinations is:

$$X = \left\{ \langle (m_0, \dots, m_h) \in N^h | \sum_{n=1}^{h} m_n \le m \rangle \right\}$$

The algorithm can thus be formally stated as:

$$\max_{(m_0, \dots, m_h) \in X} \{g(X)\} \quad (7)$$

Although optimal, the brute-force algorithm presented incurs high time complexity since the number of combinations for assigning m time slots to h different nodes is given by:

$$|X| = C(h+m-1, m) = \frac{(h+m-1)!}{m(h-1)!} \quad (8)$$

In case $m \ge h$ then $O(|X|) \ge O(h^{m-h})$. This complexity is prohibitive and proves the algorithm unpractical for general sensor fields. This is true specially considering the limited computation capabilities of wireless sensors. However, the algorithm can still serve as basis for an heuristic if optimality is traded for feasibility as explained next.

Heuristic. In order to apply the algorithm presented in a time efficient way, the problem is modified to include a further constraint on aggregation delays: messages should spend all the available aggregation time T in exactly one sensor.

Thus, the following condition now holds:

$$m_n \in \{0, m\} \quad (9)$$

This condition allows equation (6) to be re-written as:

$$g = \sum_{n=1}^{h} \left(P_n \cdot \frac{n}{h} \right) \quad (10)$$

and the set X of all possible combinations of aggregation delays becomes:

$$X = \left\{ \langle (m_0, \dots, m_h) \in N^h | \sum_{n=1}^{h} m_n \le m; m_n \in \{0, m\} \rangle \right\} \quad (11)$$

The cardinality of set X is now reduced to h and the complexity of the heuristic is $O(h)$.

Example. The following example clarifies the operation of the proposed heuristic. Fig. 1 shows half of a grid shaped sensor field with the base station (sink) located in its center. It is assumed that $N = 81$ and $T = 20s$. Furthermore, the routing protocol always find paths to the sink with a minimum number of hops. A message is generated in s_3 and it must be defined how aggregation delay T should be spent in sensors s_3, s_2 and s_1. As the the routing behavior is known, weight w_n (needed to calculate P_n) can be determined. The black marked sensors of Fig. 1, for example, route through s_1 and thus $w_1 = 16$. Fig. 2 shows the graphs of g for each possible partition of T and $1 < m \le 20$. There are only three ways of distributing T for each m. In case $m = 10$, the maximum gain among the three possible choices is obtained by spending T

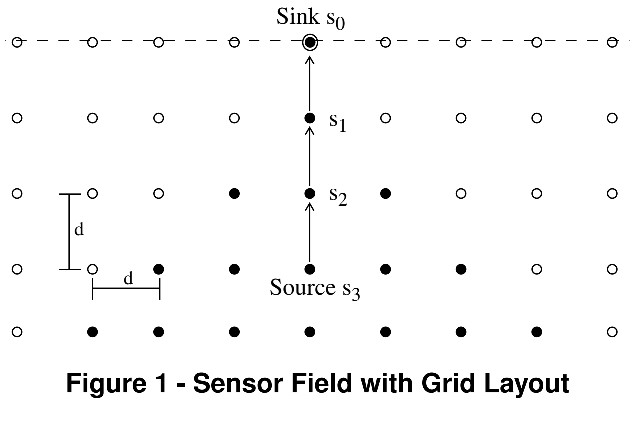

Figure 1 - Sensor Field with Grid Layout

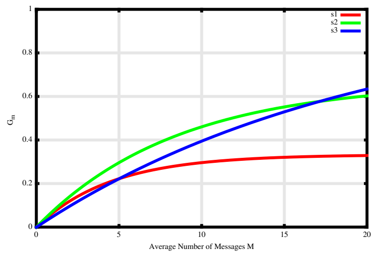

Figure 2 - g

entirely in s_2 ($T_1 = 0, T_2 = T, T_3 = 0$). If $m = 20$ is selected, the aggregation delay should be spent in s_3 ($T_1 = 0, T_2 = 0, T_3 = T$).

3. Aggregation Implementation

In this section it is investigated how the heuristic presented in the previous section can be implemented in a sensor network. A generic method to gather information needed to apply the heuristic is introduced. The section closes with a discussion on how the method can be incorporated to existing routing protocols.

3.1. Determination of Aggregation Delays

The computation of the aggregation delays T_n on the routing path of a message, according to the heuristic introduced in Section 2.2.2, can be performed for sensor fields of arbitrary geometry. Calculating T_n, however, requires the following information to be known:

(i) The probabilities P_n and therefore the weights w_n on the path the message will follow.

(ii) The average amount m of messages generated per T time units in the field.

There are two fundamental different ways of applying the heuristic shown in the previous section based on which nodes possess the above information.

In the first method, each sensor on the routing path knows m, T and the w_n of the other involved sensors. Thus each sensor can apply the heuristic and decide if it should delay a message.

In the second method, the message sender knows m, T and weights w_n for the entire path. The sender alone applies the heuristic and adds information on the selected aggregation point in the message.

The first method has high computational costs, as the heuristic must be calculated at each node in the path. The second method has less computational costs, but the result of the calculation has to be encoded in the packet. To reduce energy usage needed for computation and because of the small amount of information necessary to encode the aggregation point in the packet, the second operation method is chosen.

Basic Operation. When the base station is interested in gathering information from the sensor field, it broadcasts a *REQUEST* message. Sensors reply to requests with *RESPONSE* messages carrying the required information. *REQUEST* messages, besides communicating application-level data, serve an additional purpose in the field: setting up routes for subsequent *RESPONSE* messages from the sensors (see Fig. 3).

In order to apply the aggregation heuristic proposed in Section 2.2.2, *REQUEST* and *RESPONSE* messages are extended with additional data. The maximum allowed delay T for *RESPONSE* messages is included in every *REQUEST* message. Similarly, the average number of messages m generated in the field during T is included by the sink in each *REQUEST*. As these messages move towards the source sensors, weights w_n can be collected and also communicated through them.

Before sending a *RESPONSE* message towards the base station, the source sensor calculates an aggregation point using the information provided by the *REQUEST* message. Together with the application data, the sensor encloses T and the number of hops to the aggregation point in the *RESPONSE* message. At each hop, this count is reduced by one. When this value reaches zero, the message is held for T time units in the sensor for aggregation purposes.

In summary, the following fields are added to application messages:

- REQUEST:
  - T (optional): included if not globally known.
  - m (optional): included if not globally known.
  - w_n: collected weights on the path.
- RESPONSE:
  - T (optional): included if not globally known.
  - Aggregation-hop-count: distance from the source sensor to the aggregation point.

Note that the information carried in a *REQUEST* can be used in the computation of aggregation points for every *RESPONSE* message associated with the request. However, as the information provided by *REQUEST* ages, such messages must be resent periodically.

Weight Determination. A source node must define the weight w_n of each sensor along the path towards the base station in order to compute aggregation probabilities. This can be done in two different ways:

- *Pre-computation of* w_n*:* If the routing paths are known, a pre-computation of the weight factors for each sensor is possible. This method was used in the aforementioned

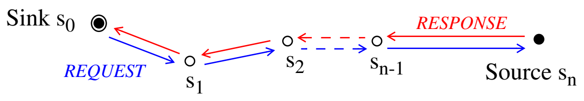

Figure 3 - *REQUEST* **and** *RESPONSE*

example. Obviously, this method might not be practical in real application scenarios with dynamic, unpredictable behaviour.

- *Runtime determination of w_n:* Each node s_n in the field keeps track of the number p_n of RESPONSE messages routed through it. Assuming routes are fixed, the weight factors of s_n can be calculated using $w_n = p_n / p_0$. As time progresses, this value tends towards the correct weight.

Defining weights using the runtime method can be applied in a greater number of real world scenarios. Furthermore, the method is able to dynamically adjust for the reduction of messages caused by aggregation. Runtime determination of the weights, however, requires packet counters to be reset whenever a route change in the field occurs (e.g. a node failed and the route is re-computed).

3.2. Protocol Implementation

In general, every data dissemination protocol based on a *REQUEST* and *RESPONSE* message dynamics in which routing paths are set togheter with *REQUEST* broadcasts can be used with the proposed aggregation scheme. This allows, for example, its usage along with tree based dissemination protocols or direct diffusion.

However, data dissemination protocols exist that do not set routing paths through *REQUEST* messages. Geographic routing protocols (e.g. GPSR [7]) dispense such dynamics. In these cases, the presented aggregation scheme cannot be used.

Tree Based Routing. The simplest method to set up a routing in a sensor network is to build a tree. The sink sends a broadcast message (Interest), containing the event monitoring interest with a unique id, to all sensors in field. The message travels hop-by-hop through the field. Each node receiving the message memorizes the sender id (e.g. MAC address) and then rebroadcasts the message. If a broadcast is received twice, identified by the interest id, the message is discarded. Responses later follow the path set up by the interest.

In this case, the necessary fields for REQUEST and RESPONSE message can be directly included in the interest and response message.

Direct Diffusion. The direct diffusion protocol is described in [9]. As in the previous described protocol, the sink generates an interest, which is distributed as broadcast. For a broadcast message received from multiple senders all senders are memorized. Therefore several path possibilities for the responses exist through the network. In a second stage of the protocol, the sink can enforces the best path that it wants to maintain.

In this case, the necessary fields for REQUEST can be included in the message used to refresh periodically one specific path. The fields for RESPONSE can be included in the normal response messages.

4. Experimental Evaluation

In this Section the heuristic to calculate aggregation points and delays presented in Section 3.3.1 is evaluated by a simulation experiment. Therefore the aggregation gain achieved by applying the heuristic is compared with the aggregation gain that is obtained with simpler algorithms. The evaluation shows that implementation effort necessary to apply the heuristic is rewarded with a high aggregation gain.

4.1. Evaluation Method

For the simulation an own lightweight simulator is used. Packets are passed between nodes without losses. The passing of one packet to the next hop has the cost of exactly one transmission, regardless of the message size. By the usage of aggregation, the necessary amount of transmissions during the experiment can be reduced. The more transmissions are saved, the better is the used aggregation algorithm. To measure the efficiency of an aggregation, the total aggregation gain G defined by equation (1) is determined by the simulator during the experiment.

As described in Section 3, the implementation of the heuristic used to determine aggregation points and delays requires additional information distributed in the *REQUEST* and *RESPONSE* messages. Furthermore, a calculation of the aggregation points and delays is necessary. Simpler algorithms to determine aggregation points and delays should provide a significant worse aggregation gain. If not, the presented heuristic is not useful. In the experiment the presented heuristic is compared with 4 other simpler algorithms (A1-A4).

4.2. Experiment Setup

The sensor field structure that is used for the experiments corresponds to the example given in Section 2.2.2. The sensor field is a grid, the distance between the sensors is $d = 5m$. The size of the field is $8d \times 8d$, containing $N = 81$ sensors with a transmission range of $\sqrt{2} \cdot d$. The base station (sink) is located in the middle of the field.

Routing Protocol. The routing in the field is done by setting up a simple tree. The base station sends periodically a *REQUEST* broadcast. If a *REQUEST* is received by a node, it is checked if the request was already received. If so, the message is silently discarded. Otherwise, the request is forwarded as broadcast and the node memorizes the sender (unique node id) of the message. This way a tree in the field is established, the memorized node ids are used as routing information.

Operation Mode. The sensor field is used in the following way. The simulation duration is 1h. Every 60s a *REQUEST* message is broadcasted by the base station to set up (or refresh) the tree used for routing. In the *REQUEST* message, the base station specifies the allowed delay T for the *RESPONSE* messages. Every 1s a message is randomly generated in the field (A sensor is randomly chosen and then emits a *RESPONSE* message). The response message contains, beside the payload, the aggregation delays T_n that should be spend by each node s_n on the message path. The T_n are calculated using different aggregation algorithms (A1-A5) which are described later.

T, beside the selected algorithm, is used as variable in the experiment. T is varied between 1s and 20s and defines m, the number of messages generated in the field during T.

Algorithm A1. The first algorithm used to determine aggregation points and aggregation delays uses an equal distribution of the maximum allowed message delay among the nodes on the path:

$$T_n = \frac{T}{h} \tag{12}$$

This algorithm is very simple to implement and is proposed in related work on the aggregation (see Section 5).

Algorithm A2. The second algorithm is based on the assumption that the traffic density is higher closer to the sink then far away from the sink. T_n is defined as follows:

$$T_n = \begin{cases} T/2^n & n < h \\ T/2^{n-1} & n = h \end{cases} \tag{13}$$

This algorithm takes traffic pattern into account but does not need the distribution of explicit information about the traffic (e.g. the weight information).

Algorithm A3. This algorithm holds all messages for T one hop before the sink. In this case messages are aggregated close to the sink. The T_n are defined as follows:

$$T_n = \begin{cases} T & n = 1 \\ 0 & n \neq 1 \end{cases} \tag{14}$$

Algorithm A4. This algorithm holds all messages for T in the source. The T_n are defined as follows:

$$T_n = \begin{cases} T & n = h \\ 0 & n \neq h \end{cases} \tag{15}$$

Algorithm A5. This algorithm is the heuristic developed in the previous sections and described by (7). The necessary weights w_n are determined during runtime as described in Section 3.3.1.

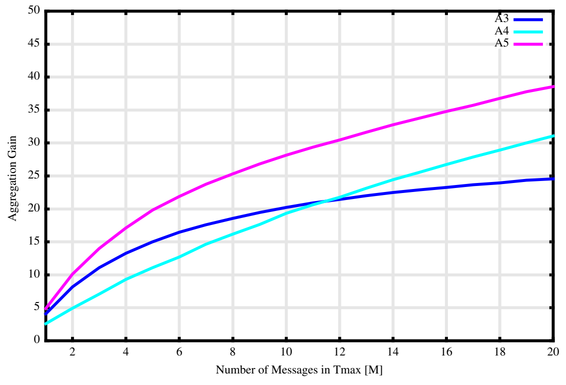

Figure 5 - Aggregation Gain A3, A4, A5

Measurement. The experiment result is shown in Fig. 4 and Fig. 5. The algorithm A5 (The proposed heuristic) achieves always the best aggregation results.

A1 and A2 perform good with increasing activity in the field (more messages are sent during T). In a field with low activity both algorithms do not provide good aggregation results. A5 performs in a field with low activity twice as good.

A3 performs good if the activity in the field is low. The best chance to meet an other message is near the sink. With increasing activity, this strategy can not compete with the others as an aggregation one hop before the sink does not result in a high aggregation gain and the chance for an aggregation far away from the sink gets better.

A4 performs bad if the activity in the field is low. The chance to find an other message for aggregation far away from the sink is not very high. The algorithm performs good when the activity in the field is high.

4.3. Summary

The proposed heuristic performs better than the other, simpler designed algorithms that were tested. As stated in Section 2, the algorithm that the presented heuristic is based on might not provide the optimum solution. There-

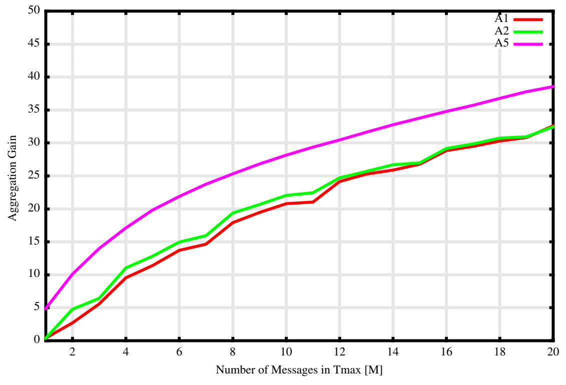

Figure 4 - Aggregation Gain A1, A2, A5

fore algorithms performing better then the presented heuristic might be possible. However, the results show that the heuristic achieves a significant better aggregation then existing aggregation methods. The results of the experiment can be summarized as follows:

(i) The decision where to delay a message and for how long has a significant influence on the aggregation efficiency.

(ii) The algorithm using the heuristic presented in Section 2.2.2 performs significant better than other known algorithms.

5. Related Work

Currently aggregation can be understood and investigated in two slightly different ways. First, aggregation can be focused on the processing capabilities of the sensor field, used to aggregate information (Application-Level aggregation). Second, aggregation can be focused on the concatenation of messages (Packet-Level Aggregation). The presented research of this paper mainly considers the second approach.

In [4], sensor networks that provide the user with periodic estimates of the environment are considered. A distributed estimation algorithm is presented that exploits an energy-accuracy trade-off. The sensing task and the aggregation task are, in difference to the presented work, coupled. The combination of data from different sensors to enhance the quality of the measurement is described in [3]. This form of aggregation takes also place on the application level. Different distributed compression techniques can also be applied to achieve aggregation on an application level (see [2]).

In [5] it is investigated how the routing can be modified, so that messages join a common path. By constructing a greedy aggregation tree path sharing and therefore aggregation can be optimized. The work presented in this paper does not modify the existing routing behavior. Furthermore, [5] does not state how aggregation delays on the greedy aggregation tree should be spent. In [1] different routing protocols and their impact on aggregation are investigated. The resulting aggregation gain and the delay of messages is analyzed. However, no upper bound of T is taken into consideration and aggregation delays are spent with a simple equal distribution among the nodes.

6. Conclusion

It is shown in the paper that the determination of aggregation points and the corresponding aggregation delays has a significant influence on the possible aggregation gain. Therefore, aggregation algorithms delaying messages for a constant time in each node are not sufficient to achieve good aggregation results.

The paper presented an aggregation algorithm which can be implemented without much effort in existing routing protocols. The presented experimental evaluation shows that the aggregation gain can be significantly increased by applying this algorithm.

The presented algorithm assumes a certain operation pattern of the sensor field. More specific, it is assumed that one sink is used to collect information. Furthermore it is assumed that events (the creation of new messages) in the field is homogenous distributed. For other field operation patterns, the presented results might not be valid and other aggregation algorithms might be necessary.

7. Literature

[1] B. Krishanamachari, D. Estrin and S. Wicker, The Impact of Data Aggregation in Wireless Sensor Networks. In International Workshop of Distributed Event Based Systems (DEBS), Vienna, Austria, July 2002.

[2] S. S. Pradhan, J. Kusuma, and K. Ramachandran, Distributed Compression in a Dense Microsensor Network, IEEE Signal Processing Magazine, Mar. 2002.

[3] Kleine-Ostmann, T. Bell, A.E., A data fusion architecture for enhanced position estimation in wireless networks. IEEE Communications Letters. On page(s): 343-345. Volume: 5, Issue: 8, Aug 2001.

[4] A. Boulis, S. Ganeriwal, M. B. Srivastava, "Aggregation in sensor networks: an energy - accuracy tradeoff," Sensor Network Protocols and Applications, Special Issue of Elsevier Ad Hoc Networks Journal, 2003.

[5] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann. Impact of Network Density on Data Aggregation in Wireless Sensor Networks. Technical Report 01-750, University of Southern California, November, 2001.

[6] D-SYSTEMS Project. http://www.cs.ucc.ie/dsystems. 2004.

[7] B. Karp and H. T. Kung, “GPSR: greedy perimeter stateless routing for wireless networks,” in Proc. MOBICOM, 2000, pp. 243-254.

[8] A. Barroso, U. Roedig, C.J. Sreenan. Maintenance Awareness in Wireless Sensor Networks. Proceedings of 1st IEEE European Workshop on Wireless Sensor Networks (EWSN) (Short Papers), Berlin, Germany. January 2004.

[9] C. Intanagonwiwat, R. Govindan and D. Estrin. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks In Proceedings of the 6th International Conference on Mobile Computing and Networks (MobiCOM 2000), August 2000, Boston, Massachusetts.