An Approximation Delay between Consecutive Requests for Congestion Control in Unicast CoAP-based Group Communication

ABSTRACT

This research presents a way to avoid network congestion during unicast CoAP-based group communication using increased delays between consecutive requests (DCR) in LoWPAN border routers to limit request send rates. It also provides a way to determine DCR values that are suitable for various network group sizes with differing node counts. The optimal DCR is obtained using the least squares approximation method and the relative minimum. Results from experimentation shows a positive relation, that is, an increase in group size necessitates an increase in DCR value. Experiments in various group sizes show favorable network performance and support the proposed congestion control method using DCR.

1. INTRODUCTION

Today, the use of constrained devices is becoming commonplace. Their ability to connect to the internet brings about the concept of the Internet of Things (IoT). These devices are typically used for collecting information from sensors into standard networks [1], [2]. Internet of Things come in two categories: Non-IP and IP-based. Non-IP systems do not rely on IP addresses for communication. Such non-IP protocols include the Bluetooth low energy and Zigbee protocols. IP-based systems, which require IP addresses, comprise protocols such as 6LoWPAN and Thread.

To support IP-based communication [3], the Internet Engineering Task Force (IETF) has developed the Constrained Application Protocol (CoAP), a web communication protocol specially designed for constrained devices. It is similar to HTTP [4], but works on top of the User Diagram Protocol (UDP) instead of the Transmission Control Protocol (TCP). This eliminates the need for a three-way handshake and the need to maintain a connected state, making CoAP ideal for constrained devices. The protocol also supports machine-to-machine (M2M) communication. M2M involves a sensor device sending data through a wireless network to an application. The application converts the data into media that is shown to the user, allowing for immediate interaction and decision making.

Typical CoAP communication is one-to-one. A client exchanges messages with a server using the following steps (Figure 1): 1) the user initiates a GET command through a web browser to request data from the CoAP server. 2) The CoAP client node receives this GET command from the browser, and sends a request message to the CoAP server through UDP. 3) When the server receives the request message, it sends a reply message containing the requested data back to the client. 4) The client sends the data to the browser.

In many cases, one-to-many communication is required. Such instances include simultaneous data collection from multiple sensor nodes for verification and processing, and node grouping. CoAP group communication is thus important to IoT, and is one instance of the group concept being incorporated into data communication. Unfortunately, CoAP does not provide a way to handle network congestions that may arise, such as when one client node exchanges messages with multiple server nodes using unicast for basic communication or for resource observation. Such a problem is worth investigating [6], [7]. As a result, a study by I. Ishaq et al [8] proposes the use of a CoAP-based group communication method called unicast where a level of intelligence is introduced to facilitate resource management, making it possible to use a single request message to manage resources.

Figure 1. Fundamental CoAP communication

Group communication through CoAP is highly reliable and effective, but if one client node sends unicasts to multiple server nodes in rapid succession, the response messages from those servers would be equally rapid, and would result in inevitable congestion. If the rate of request messages being sent is not moderated, when the network reaches its capacity threshold, it would start to discard messages, causing data loss, as shown in figure 2. To avoid unicast congestion, a delay between consecutive requests (DCR) is introduced to keep the rate of request messages to within the limits of the network and the recipient nodes. If left uncontrolled, the burden on the network caused by rapid messages could result in the aforementioned data loss and even network collapse.

Figure 2. CoAP-based Group Communication Unicast Congestion

This research aims to provide a solution to the unicast congestion problem by adding DCR into the gateway (LoWPAN border router) to limit the message send rate to within an acceptable tolerance. It also presents a way to determine DCR values for different group sizes with different node densities using the least squares approximation method. Optimum DCR values are obtained from the relative minimum, and tested using the Cooja simulator.

The remainder of this paper is organized as follows: Section 2 provides background information on CoAP, unicast group communication in IoT, and least squares approximation. Section 3 presents the results and analysis, which covers the methodology for finding the DCR, the simulation of the DCR values, and a description of the results. The last section, Section 4, summarizes the research.

2. RESEARCH METHOD

2.1. A background of CoAP

The Constrained Application Protocol (CoAP) is a specialized web transfer protocol designed for constrained nodes and constrained networks. It supports IoT using lightweight messages, and utilizes relatively little server resources and power compared to other protocols. It is able to support requests from a large group of clients. Designed by the Internet Engineering Task Force (IETF), CoAP comprises two layers: 1) Messaging Layer to handle UDP for communication, and 2) Request/Response Layer to handle method and response codes.

CoAP is similar to HTTP in that they both use the RESTful Web Service, a fundamental web technology available on every platform where a server node can create resources for use in the URI, and client nodes can access those server node resources through four types of method codes: GET, PUT, POST, and DELETE [9]. Another similarity is that CoAP can send various types of payloads, and can indicate the type of payload using XML, JSON, and CBOR etc. Amidst the similarities, however, a distinct difference between the two protocols is that CoAP communicates through UDP while HTTP uses TCP. An advantage of UDP, a connectionless protocol, is that data can be sent very quickly without the need for a three-way handshake that is required by TCP.

There are four types of messages defined in CoAP: 1) Confirmable, 2) Non-confirmable, 3) Acknowledgement, and 4) Reset. In the Request layer, Confirmable and Non-confirmable messages can be sent. While in the Response layer, an Acknowledgement message can be piggybacked. The reliability of CoAP messages are defined as Confirmable (CON) and Non-confirmable (NON). CON is used to send reliable messages, with default timeouts and exponential back-offs between retransmissions. The receiving server sends an Acknowledgement message (ACK) using the same message ID as the client's request. The message send/receive diagram is shown in Figure 3 (a). Messages whose reliability is of low importance (unreliable) can be sent as Non-confirmable, in which case the server will not respond with an acknowledgement message but will record the message ID to prevent duplication. In the example in Figure 3 (b), the message ID is [0x7d34]. If the server receives a non-confirmable message but is unable to process it, it would respond with a Reset message (RST) [4].

Figure 3. (a) Reliable message transmission, (b) Unreliable message transmission

2.2. Unicast group communication in IoT

Group communication is essential to many IoT applications, particularly in constrained networks such as low-power and lossy networks (LLN). Typically, in very large networks, constrained devices need to be grouped and controlled using group commands to simplify management and avoid congestion. [7] During unicast communication, one sender node sends messages to multiple receivers, and must wait for an ACK message from each of the receiving nodes. This can lead to congestion, especially when the receiver and sender differ in their message sending and receiving capability.

Ishaq et al [6] suggested a unicast group communication method with the aim of segmenting management tasks to allow easier access to resources. A group of nodes being managed is called an Entity. The group of resources belonging to an Entity is called Entity Members. Resource usage or management commands can be sent to an Entity using a single CoAP request. An Entity Manager (EM) is the collection of management tools for each Entity. An EM is able to manage an Entity and its resources on a CoAP server in LLNs. Clients in the Internet can send commands to the EM to control Entities.

Hou et al [9] suggest in their paper the use of a resource-oriented protocol called SealHttp to solve the issue with unicast-based group communication. Its process utilizes COMBINE and BRANCH instead of the EM, and allows nodes the ability to self-join and leave groups. Comparisons were made between SealHttp and RESTful protocols, and URI performance was improved by adding spatiotemporal attributes in the standard

URI for dynamic group requests of physical resources. The research finds that SealHttp has better average energy consumption during group communication in the web of things (WoT) than CoAP.

A paper by Quakasse et al [10] proposes a way to solve network congestion in CoAP by improving the delay and adapt the behavior of the solution to network conditions. The paper suggests the use of delays between unicast requests depending on the link delay and the estimated group size. The results show improved performance in terms of response time and packet loss.

2.2. Least squares approximation

An assumption made in this research is that there may be discontinuities in the DCR values at the specific range that we focus on, and that there could be data measurement errors during experiments [11]. This research, therefore, uses an approximation function to best construct a graphical line of the DCR values in relation to the size of the communication group, as mentioned in Section 1.

The least squares method produces the best function for data approximation because it averages out data errors to a minimum. When the function is drawn out as a graph, the line will not pass through each data point, but will traverse through the vicinity close to them. The function produces a smooth line that approximates real values. The resulting function is concise and is independent of data size. Differential and integration calculations can easily be applied. The resulting functions are polynomial, and its correctness is dependent on the degree of polynomials [12].

The least squares method of function approximation can be applied in many ways. For our purposes we use one type of least squares called the discrete least squares method. The input data is non-continuous, and the approximation is in the form of the polynomial $p \in \prod_n$ with the highest degree of n that produces minimum mean squared error, shown in Formula 1.

$$\sum_{i=0}^{N} w_i \left(f(x_i) - p(x_i)\right)^2 = \sum_{i=0}^{N} w_i \left(f(x_i) - a_0 - a_1 x_1 - \cdots - a_n x_i^n\right)^2 \quad (1)$$

When $x_0, x_1, \ldots, x_n$ are not continuous, and $w_0, w_1, \ldots, w_n$ represent weighted data, if $N>n$, the error can be eliminated using a polynomial formula. In this case, the number of discontinuous data is more important than the degreee of approximation. For continuous data, Formula 1 can be expanded as follows [13]:

$$a_0 \sum_{i=0}^{N} w_i x_i^j + a_1 \sum_{i=0}^{N} w_i x_i^{j+1} + \cdots + a_n \sum_{i=0}^{N} w_i x_i^{j+n} = \sum_{i=0}^{N} w_i x_i^j f(x_i) \quad (2)$$

when j = 0,1, ... , n or

when j = 0 $a_0 \sum_{i=0}^{N} w_i x_i^0 + a_1 \sum_{i=0}^{N} w_i x_i^1 + \cdots + a_n \sum_{i=0}^{N} w_i x_i^n = \sum_{i=0}^{N} w_i x_i^0 f(x_i)$ (3)

when j = 1 $a_0 \sum_{i=0}^{N} w_i x_i^1 + a_1 \sum_{i=0}^{N} w_i x_i^2 + \cdots + a_n \sum_{i=0}^{N} w_i x_i^{n+1} = \sum_{i=0}^{N} w_i x_i^1 f(x_i)$ (4)

when j = n $a_0 \sum_{i=0}^{N} w_i x_i^n + a_1 \sum_{i=0}^{N} w_i x_i^{n+1} + \cdots + a_n \sum_{i=0}^{N} w_i x_i^{2n} = \sum_{i=0}^{N} w_i x_i^n f(x_i)$ (5)

This can be expressed as polynomial equation 6.

$$f(x) = a_{0+} a_1 x + \cdots + a_n x^n \quad (6)$$

Equation 5 can be expressed as a matrix.

$$Xa = f \quad (7)$$

Matrices X and f, when all weighted data are equal $(w_i = 1)$, can be expressed as

$$\begin{bmatrix} \sum_{i=1}^{m} x_i^0 & \sum_{i=1}^{m} x_i^1 & \sum_{i=1}^{m} x_i^2 & \cdots & \sum_{i=1}^{m} x_i^n \\ \sum_{i=1}^{m} x_i^1 & \sum_{i=1}^{m} x_i^2 & \sum_{i=1}^{m} x_i^3 & \cdots & \sum_{i=1}^{m} x_i^{n+1} \\ & & \vdots & & \\ \sum_{i=1}^{m} x_i^n & \sum_{i=1}^{m} x_i^{n+1} & \sum_{i=1}^{m} x_i^{n+2} & \cdots & \sum_{i=1}^{m} x_i^{2n} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{m} x_i^0 f(x_i) \\ \sum_{i=1}^{m} x_i^1 f(x_i) \\ \vdots \\ \sum_{i=1}^{m} x_i^n f(x_i) \end{bmatrix} \sqrt{b^2 - 4ac} \quad (8)$$

When this is expressed as an equation, we get the linear function $y = a_0 + a_1 x_i$, the quadratic function $y = a_0 + a_1 x_i + a_2 x_i^2$, the cubic function $y = a_0 + a_1 x_i + a_2 x_i^2 + a_3 x_i^3$, the quartic function $y = a_0 + a_1 x_i + a_2 x_i^2 + a_3 x_i^3 + a_4 x_i^4$, and so on. Because the data in this research is obtained from experimentation, the least squares method is better suited to the task than interpolation because experiment data may contain errors caused by the measurement process. The resulting graph line need not pass every data point, but should be a result of the least total error.

3. RESULTS AND ANALYSIS

This research proposes an alternative form of unicast where the LoWPAN border router manages client request messages, as shown in figure 4, and uses an a simple DCR-based solution based on leisure as defined in RFC7252 [5] to lessen congestion. The method sets the limit rate at the border router when unicast messages are continuously sent to servers without waiting for response messages, thus avoiding congestion.

The best DCR value(s) for each server group size is obtained from approximation based on the following process: 1) Finding DCR based on leisure . The results are the DCR values and average response time in different group sizes. 2) The values are then put into a least squares approximation function. 3) The relative minimum that produces the least average response time in each group size is obtained. The details of each step are explained in Sections 3.1. to 3.3.

Figure 4. Request message management through a LoWPAN border router

3.1. DCR calculation based on Leisure

We can find the values of DCR based on leisure [8]. Given the server group size (G), the target data transfer rate (R), and the estimated response size (S), the DCR can be calculated using the formula:

$$DCR = \frac{Leisure_{lowerbound}}{G-1} = \frac{S \cdot G}{R(G-1)} \quad (9)$$

In our experiments, the value of G was between 3 and 23, S was approximately 80 bytes, and R was set to a conservative 8kbits/s (1 kB/s). The lower bound for the leisure was between 120 ms and 83 ms. Using Formula 9, we see a decrease in DCR values starting from 120 ms. We therefore used a DCR range between 0-120 and measured the average response times for different group sizes (G = 3, 5, 10, 15, 20, and 25 nodes) by running simulations using the Cooja simulator. Figure 5 shows the results in Cooja.

Figure 5. Response times for different group sizes as a function of the delay between consecutive requests, evaluated using Cooja simulator

3.2. DCR approximations

To estimate the value of the optimum DCR, we use the data from Section 3.1 to plot a graph using the least squares method. The graph is the best mathematical representation of the relationship between DCR and average response times with minimized effects from data error, as shown in Figure 6.

3.3. Relative minimum for each group size

The relative minimum that produces the least average response time in each group is determined from the graph of DCR values from Section 3.2. Groups comprising 3 and 5 servers yield a DCR of 0.02s. The value increases for group sizes 10, 15, 20, and 25 to 0.03 s, 0.04 s, 0.09 s and 0.10 s, respectively, as shown in Figure 7.

Figures 6 and 7 show the results of those experiments. In every experiment, the LoWPAN border router sends consecutive request messages to server nodes at the same rate. Each experiment yielded a similar pattern for each group size. The initial part of the graphs show a high average response time because the DCR value was low, resulting in request messages being sent too rapidly and causing congestion as response message packets from multiple server nodes overwhelm the border router. But as the DCR value increases, the message send rate becomes more suited to the network's capacity, resulting in lower and lower average response times until a minimum average response time is reached. After this point the response time starts to rise again as the DCR increases due to the higher delay inserted into the border router.

Figure 6. DCR approximation for each group size

Figure 7. Relative minimum for each group size

3.4. DCR estimation

After the relative minimum is found in Section 3.3, it is verified through experimentation. In order to evaluate the performance of our proposed solution, we use simulations. To determine the performance the of the DCR inserted between consecutive unicast requests for group communication between a single gateway and multiple servers, we use the Cooja network simulator.

The following indicators are observed from the simulations: the average response time, the time taken by servers to respond to unicast communication, the hop count (the number of routers each packet goes through to reach its destination), and the packet loss ratio. The settings for these experiments are detailed in Table 1.

Table.1 Evaluation experiments settings.

Node Type	Z1 mote	
Contiki version	Contiki 3.0	
Media Access Control (MAC)	Carrier Sense Multiple Access (CSMA)	
Radio Frequency (RF)	IEEE 802.15.4 channel 15	
Routing	Protocol:	RPL
CoAP	Group size:	DCR:
	3 nodes	0.02 s
	5 nodes	0.02 s
	10 nodes	0.03 s
	15 nodes	0.04 s
	20 nodes	0.09 s
	25 nodes	0.10 s

The experiments involve sending request messages from a LoWPAN border router to a group of server nodes while utilizing the DCR values obtained in Section 3.2 to control the message send rate. For each group size, a total of 50 simulations are carried out. The results confirm the effectiveness of our proposed solution, as shown in Table 2.

In terms of response time, the results show that for smaller group sizes (3 and 5), a DCR of 0.02 s produces an average response time of 0.17 s and 0.26 s, respectively. The average response time increases as the number of neighbors increases, a result of a higher number of collisions in a shared medium.

Regarding packet loss, an important indicator, groups containing less than 20 nodes experience an average packet loss of 0%, or no loss. In groups larger than 20 nodes, the percentage starts to increase to 0.3%, and keeps increasing as the group size grows. This is due to the higher node density causing more collisions between group members. Another possible factor is that in larger groups, there is a higher average hop count. Communication between the LoWPAN border router and the server requires more hops, resulting in more dropped messages before they eventually reach their destination.

Table 2. Summary of the results of the experiments.

		Hop count (h)			Response time (s)			Packet loss (%)		
Group Size	DCR	Min	Average	Max	Min	Average	Max	Min	Average	Max
3	0.02	1	1.00	1	0.17	0.17	0.17	0	0	0
5	0.02	1	1.00	1	0.26	0.26	0.26	0	0	0
10	0.03	1	1.20	2	0.78	0.79	0.79	0	0	0
15	0.04	1	1.73	3	1.57	1.58	1.59	0	0	0
20	0.09	1	1.65	3	2.64	2.67	2.69	0.2	0.3	0.4
25	0.10	1	1.80	3	3.90	3.91	3.93	0.7	0.8	1.0

4. CONCLUSION

This research proposes a means to avoid network congestion in low power and lossy networks during unicast group communication by introducing delays between consecutive requests to limit the message send rate. It also presents a way to obtain the proper DCR value for each network group size by using the default leisure period in the request/response layer of the CoAP protocol, and applying the least squares approximation method to find the relative minimum DCR.

To evaluate the validity and performance of this method, the Cooja simulator is used to simulate different DCR values in networks under various conditions as defined by each experiment. The results are analyzed and evaluated. Experiments confirm that the method proposed in this research is effective for avoiding network congestion caused by unicast group communication in CoAP, with all management being virtually on the sender side. It is able to effectively reduce the communication workload on the server side of the CoAP protocol. The method is suitable for group communication and provides excellent scalability.