Topic-centric and semantic-aware retrieval system for internet of things

Feng Zhao *, Zheng Sun, Hai Jin

Services Computing Technology and System Lab & Cluster and Grid Computing Lab, School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China

Article info

Article history:
Received 30 September 2013
Received in revised form 9 December 2013
Accepted 6 January 2014
Available online 30 January 2014

Keywords:
Internet of things
Information retrieval
Knowledge network
Topic centric
Semantic awareness

Abstract

The Internet of things (IoT) has been considered as one of the promising paradigms that can allow people and objects to seamlessly interact. So far, numerous applications and services have been proposed, such as retrieval service. The retrieval, however, faces a big challenge in IoT because the data belongs to different domains and user interaction with the surrounding environment is constrained. This paper proposes Acrost, a retrieval system based on topic discovery and semantic awareness in IoT environment. The initial contents with interesting information is obtained through the combination of two topic centric collectors. The metadata is extracted by aggregating regular expression-based and conditional random field-based approaches. Moreover, the semantic-aware retrieval is achieved by parsing the query and ranking the relevance of contents. In addition, we present a case study on academic conference retrieval to validate the proposed approaches. Experimental results show that the proposed system can significantly improve the response time and efficiency of topic self-adaptive retrieval manner.

© 2014 Published by Elsevier B.V.

1. Introduction

Internet of things is a novel paradigm that is rapidly gaining ground in the scenario of modern wireless telecommunications, which pervasively presents around us of a variety of things or objects, such as Radio-Frequency Identification (RFID) tags, sensors, actuators and mobile phones [1,2]. The architecture of IoT can be divided into three layers: perception layer, network layer, and application layer. Being involved with a large number of devices, IoT generates enormous data, which are multi-source, real-time, dynamic, sparse, highly heterogeneous and semantically rich. In order to improve processing efficiency and provide advanced intelligence, providing the most relevant and valuable information from unprecedented amount of data according to user’s query becomes a critical issue.

Different from general web search engines, searching in IoT should go beyond keyword-based search, and take into account query-related contexts. Conventional searching technologies fall short of in IoT, such as architecture design of search engines, search locality, scalability, real-time [3]. In recent years, a retrieval model for application-layer of IoT, context-aware information retrieval, appears to provide appropriate results for the user according to the user’s physical state of the surrounding environment [4]. In fact, the technologies used to implement the context-aware retrieval have to adapt to the increasing distributed character of activities both in logical and physical aspects.

Much research work has been conducted to develop the information retrieval in IoT, such as vertical searching engine. The information in IoT can be divided into diverse fields by the content theme, media type or content type. Vertical searching engines usually rely on the focused application-layer crawler, and the scope they crawled is also limited to the IoT content that are related to the predefined theme (or theme collection), while general web search engines use web crawlers to obtain the massive web information (for example, HTML, PDF, MP3, AVI, etc.) from World Wide Web (WWW) and build index for the downloaded data. Compared with general web searching engines, vertical searching engines have significant advantages for searching in IoT, such as higher accuracy, full use of the domain knowledge including different classifications or ontology, and support for some certain services.

Searching engine tries to achieve its goal by associating both users and information items with attributes and by having a service performing information searching based on these attributes. Moreover, the difference between IoT and Internet is that IoT is concerned with physical objects rather than information (e.g., web pages) distributed in far away places. However, most of the current existing searching engines in Internet focused on the manner of query which is much simple, such as single retrieval by keywords. In IoT environment, the search suffers from two problems: Firstly, the existing retrieval systems almost ignore the topic-relevant searching, while the valuable information is hard to get; Secondly,
the retrieval manner is generally organized as the property of document rather than the habit of users, which is not flexible and limited on and user interaction with the list of surrounding environment (e.g., retrieved resources).

To address these issues, this paper proposes Acrost, a retrieval system for application-layer of IoT, which provides users with retrieval services based on theme information. Our system relies on topic adaptation and on knowledge network that is built using semantic mining. The overall data are obtained by two different methods: General Search Engines (GSE)-Based Collector is used to access general information on application-layer and Topic-Focused Collector is used to access data from predefined topic objects whose contents belong to a specific area. When extracting metadata from collected data, Acrost uses two different ways to identify entities: linear Conditional Random Fields (CRF) [5] is used to extract metadata from non-structured data and HTMLParser with regular expression is used to extract metadata from semi-structured data.

Two main contributions of this work is listed as follows. One is to provide topic-relevant information according to users’ demand, which improves the reading efficiency in a pervasive environment (e.g., for mobile users). The other is to provide interactivity between users and the surrounding environment. The topic-relevant information is based on the topic discovery and relevance ranking. The ranking on relevance is used to evaluate the importance of IoT contents, whose inputs are the impact factors of the statistical ranking. The interactivity is achieved by the explanation of how an object (e.g., document) matches IoT resources by knowledge network.

The remainder of the paper is organized as follows. In Section 2, we survey the related work on searching in IoT and scholar searching engines. Section 3 presents the overview about system architecture and system workflow. Section 4 describes two topic centric data collecting and information extraction approaches based on ready data. Section 5 discusses how to achieve retrieval by semantically parsing query and how to rank massive IoT contents. Section 6 presents a case study, academic conference retrieval, and also describes the prototype system and its experimental results to validate proposed approaches. Section 7 concludes this paper.

2. Related work

As searching in Web has become one of its most popular services, searching for real-world entities to be equally important in IoT. OCH system allows users to query the current location of lost real-world objects (such as umbrellas or keys), that objects are tagged with a device holding the identity of the object and mobile object sensors are used to detect the presence and identity of such objects [6]. However, since the sought object cannot be found exactly even if it is actually detected by an object sensor, OCH system is hard to determine whether the objects belong to a user and cannot be extended to other types of sensors. Dyser [7] is a search engine for the Web of things, which allows real-world entities (i.e., people, places, and objects) to be searched by their current state. In Dyser, there are two key elements: sensors and entities. Each sensor and each entity have a virtual counterpart, a Web resource, identified by a URL and accessible using HTTP. Evaluation results indicate that Dyser may be exploited to significantly decrease the communication overhead of a real-world search engine.

Snoogle and Microsearch are two systems based on a two-tiered hierarchy of mediators, i.e., index points that maintains an aggregate view of all sensors in a certain geographical area such as a room. Snoogle [8] is a search engine for the physical world, which uses information retrieval techniques to index information and process user queries, and compression schemes such as Bloom filters to reduce communication overhead. Microsearch [9] is an interface for semantic search when search engines meet small devices. Some of these systems have been explicitly designed with entity discovery, some support the discovery of sensors with certain properties, however their underlying mechanisms are also applied to the entity discovery.

Context-aware approach is widely used for searching in IoT. Covington et al. [10] proposed a role-based access control framework for context-aware applications, that user roles are stored in certificates and access control policies are defined based on these roles as well as on dynamic environmental parameters captured by sensors, such as turned on devices, time, location and other. [4] proposed and implemented a secure and context aware information lookup architecture for the IoT, which uses attributes to define access control policies, as well as, to semantically determine users and information items.

Public scholar search engines and digital libraries are built by universities or organizations. As a pioneer in the area of Autonomous Citation Indexing (ACI) research [11], now CiteSeer [12] has become one of the most popular digital libraries in the field of computer science. As the next generation of CiteSear–CiteSeer [13], a new architecture has been developed to solve the problems existing in CiteSear, such as the weakness in interoperability and scalability. CiteSeer proposes a fully new architecture, including modular-based web services, pluggable service components, distributed object warehouse and transaction security process, and also builds a more comprehensive data model. Subsequently, many other professional academic search engines and digital libraries were developed, such as SmedSearch [14], OAI-Based NCSTRL [15], CoRR [16], Perseus [17]; meanwhile, more general academic search engines and digital libraries which indexed a variety of disciplines of literature appeared timely, such as NSDL [18], ScienceDirect [19]. Whether centralized SmedSearch, NSDL, Rosetta [20], or non-centralized OverCite [21,22], every digital library uses modular-based services to achieve scalability. DBLP [23] is another computer science bibliography online, and it listed more than 2.1 million articles on computer science until November 2012. SemreX [24] also indexes over 2 million publications.

3. System architecture

3.1. System overview

Acrost not only shares many goals as previous searching engines such as high scalability, performance and reliability, but also has goals of its own, such as topic-relevance, topic self-adaptation, high updating efficiency, authoritative ranking. The system architecture is shown in Fig. 1. In this section, we will describe the layers and the components contained in Acrost.

3.1.1. User interactivity layer

User interactivity layer is the open platform for users to access the system. It receives the user’s service request, sends the request to the data process layer for the appropriate response, and receives the retrieval results from the data process layer and displays the results to the user. Specifically, it includes the following modules: Query interface providing an entrance for users to input query and receive retrieval results; Statistics interface providing a list of statistical conditions for retrieval, such as topic areas, time stamp and location; Recommendation interface providing a function for users to upload a query and receive the recommendation. Controller, provides scheduling function in data process layer according to users’ demanding, that itself never process and response to users requests, it is only responsible for receiving the service requests
and deciding to call modular component lying in the data process layer to handle the requests.

3.1.2. Data process layer

Data process layer contains the vital tasks of Acrost, including data pre-process, information extraction, data storage and index, knowledge network management.

Data preparation. Data preparation includes three aspects: The first is data normalization. Since the presentation of IoT information is diverse, the datasets obtained by information extraction have different formats. In order to facilitate the management and organization of the datasets, the datasets need to be standardized in a unified format. The data normalization is conducive to improve the efficiency of data utilization. The second is ranking model generation. In order to evaluate collected data, it needs to calculate the impact factor. In this paper, we propose a computational model based on topic-evaluation. The third is topic discovery. In order to achieve topic self-adaptive retrieval, the theme information hidden in the texts is discovered and parsed out. The detailed will be described in next section.

Information extraction. The goal of information extraction is to get metadata, such as the location and the topics, from the collected contents. Fig. 2 shows the workflow of Acrost how to extract information. Step 1, the classifier divides the collected contents into two categories: one category is the contents collected by Topic Focused Collector from topic websites or related IoT devices, which is define as C1; the other category is the contents collected by GSE-Based Collector, which is define as C2. Step 2, for category C1, we use RE-Based Extractor to extract metadata, known as attribute values, and then define some entities for the attributes. Step 3, for category C2, we use CRF-Based Extractor to extract the metadata, and then define some entities for the attributes as well. Step 4, we write the entities generated from Steps 2 and 3 into files.

Data storage and index. Data storage takes the charge of managing driver, creating a connection instance and executing related SQL statements. Fig. 3 illustrates the entity-relationship in our data model: there are two entities defined in Acrost, named content and topic, and the relationship between them is n–n. As the data is massive and dynamic, the traditionally centralized database structure cannot achieve the high performance in IoT. Hence, an applicable storage structure is used as shown in Fig. 4. Acrost consists of one master server and many chunk servers. The master server manages namespace, access and control message, mapping relation and block location and chunk servers store data from various devices through databases and file systems. The data is partitioned into some blocks in the same size, and every block is identified by a unique 64 bit chunk-handle.

With the help of open source full-text search engine toolkit-Lucene [25], Acrost builds a full indexing engine for metadata stored in the databases. The index files established by Lucene can be applied through cross-platform applications, and Lucene has a good
performance in terms of indexing speed. The inverted index file established here provides the foundation in response to user queries. In order to complete the query operation efficiently, data indexing is a vital procedure. In Acrost, 5 foundational classes are used to index data, named Document, Field, IndexWriter, Analyzer, Directory, which are explained as following:

Document: It is used to describe text files generated by extracting.

Field: It is used to illustrate the attributes of a document. Here, we defined the fields like “abbreviation”, “full name”, “rank”, “location”, “topics”, etc. It is worth noting that the “rank” field should be indexed with un-tokenized.

Analyzer: The child class MMAnalyzer is used to do the task of word segmentation.

IndexWriter: The function of this class is to add each document into the index and optimize the index.

Directory: It allocates the storage path of the index built above. Note that, we used FSDirectory to denote the location that lays at file system.

The structure of inverted index built for academic conference by Lucene can be described as Fig. 5. Three columns are stored as term dictionary files, frequency files, and position files. The term dictionary files not only store each keyword, but also keep the pointers to frequency files and position files.

3.1.3. Knowledge network management

Knowledge network provides organization, navigation, search, share, and innovation to effectively locate knowledge by describing inherent relationship among various knowledge. Acrost builds a conceptual knowledge network based wikipedia to manage knowledge hidden in IoT. Knowledge network management includes following aspects:

Knowledge documents extraction. Acrost extracts knowledge documents from wikipedia: classifies documents into two categories: concept document and category document; stores these documents. To classify documents, Naive Bayes method is used according to the contents of documents. In detail, the content of a document can be parsed as rough term set, Acrost tokens rough terms of document, removes stop words and stems the remaining terms, that rough term set is ordered as term vector \( \{t_1, t_2, \ldots, t_n\} \). Finally, the posterior probability of term vector belonging to each category \( c \) is computed.

Knowledge information extraction. Acrost analyzes knowledge documents to excavate knowledge information, such as conception, category, and relationship between concept and category. The interesting information is stored into database. In Acrost, knowledge information are excavated using semantic analysis-based approach. From semantic view, each knowledge in IoT data consists of lots of parts, which can be described as semantic tuple \( S = \{ID, concept, category, parent – node\} \). Concepts are obtained from Wikipedia and can be classified into five categories: redirect concept, disambiguation concept, bracketed concept, normal concept and others. The relationship among different concepts are organized using synset tree by computing semantic depth, semantic density and semantic superposition.

Knowledge node construction. Acrost translates knowledge information into knowledge node, which describes information in a uniform format and stores information as files. A knowledge node can be described as a record including semantic information \( S \) and geographical information. If two knowledge nodes have relationship on semantic concept in \( S \), including correlation, causation, logical, synonymous relationship, there is an edge between these two nodes.

Knowledge information utilization. Acrost builds index for knowledge information and provides the retrieval function.

3.1.4. Data sources discovery layer

Data sources discovery layer is the base of Acrost system, which locates at lowest level and provides data support for application services and data processing in upper layers. There are two kinds of data sources in Acrost: Web data sources from Internet and topic data sources from IoT devices. For Web data sources, general search engines and web crawler are used to collect possible information. For IoT devices, topic relevance analysis based collecting approach is used to identify of all possible information databases according to a specific theme. The collected data URLs are classified based on content features and filtered. The selected contents are downloaded based on the filtered seeds. Acrost adopts support vector machine (SVM) [26] for feature analysis and classification of contents. For acquiring topic data sources, the collector handles a
special list, which was set manually in advance. To screen out the topic-relevant information from the interference contents, it needs to customize the string matching rules for links of each topic website or device.

### 3.2. System workflow

As shown in Fig. 6, Acrost workflow can be divided into data generation stage and data retrieval stage.

**Data generation stage.** Data generation stage deals with metadata acquisition, data storage, and indexing. In the data generation stage, Acrost obtains the IoT contents, extracts the text information, generates text files to store the plain text information of these contents in the unit of one topic, parses out the metadata from these text files to fill each field of the records, writes these metadata records into the databases through storing procedure, and builds index for these records by using Lucene.

**Data retrieval stage.** In data retrieval stage, Acrost processes the user’s query request with lexical and semantic analysis, generates different query patterns based on the results of query parser of Lucene, including TermQuery, BooleanQuery, FuzzyQuery, etc., and searches in the index file by using the generated query patterns and query keywords and the result array will then be returned by Lucene searcher. When retrieving, Acrost reads entire data information from databases corresponding to the sorted contents and packages them into a result set.

### 4. Topic centric data collecting and extracting

#### 4.1. GSE-based collector

GSE-based collector is an effective and economical approach to find sites and pages containing information from Internet. Focusing on topics, Acrost builds a list of name as retrieval seeds, which are used as keywords to submit searching in a general search engine, such as Google, Baidu. The retrieved results are filtered according to specific rules, and screened out the probable topic-relevant links using SVM classification model. The URLs belonging to topic-relevant information are downloaded finally.

Compared with ordinary websites, topic-relevant homepages have some unique properties in URL signature, web content feature and link characteristic. Taking advantage of these features, SVM classification model is used to determine whether the site is topic-relevant. By calculating word \( \text{TF-IDF} \) [27] and inputting it as the feature vector for SVM classification, Acrost gets a complete classification model based on SVM and classifies a given site with high accuracy. The following formulas present how to calculate \( \text{TF-IDF} \) of every word vector:

\[
\text{TF}_{ij} = \frac{n_{ij}}{\sum_{j} n_{ij}},
\]
\[
\text{IDF}_{i} = \log \frac{|D|}{|\{j : t_i \in d_j\}| + 1},
\]

where \( n_{ij} \) denotes the number of times of term \( t_i \) appeared in document \( d_j \), \( |D| \) denotes the sum account of documents in repository, \( |\{j : t_i \in d_j\}| \) denotes the numbers of documents contained term \( t_i \).

**Algorithm 1. Link Filter Algorithm**

**Input:**
- linkText: The plain text of link.
- linkURL: The address of link.
- depth: The depth of link in one site.
- \( D(s_1,s_2) \): The levenshtein distance between \( s_1 \) and \( s_2 \).
- hLink: The number of homogenous links on page.

**Output:**
- isValid: A Boolean value indicates whether the link should be filtered.

1. If depth is greater than 5 do return isValid equals to false end if
2. If domain name in black list do return isValid equals to false end if
3. If the link is visited do return isValid equals to false end if
4. If linkText or linkURL contain specific keywords do
5. for each pair linkURL on page do
   if \( D(\text{linkURL}_i,\text{linkURL}_j) \) is less than 3 do
   hLink increases by 1 end if
6. end for
7. If hLink is greater than 10 do return isValid equals to false end if
8. set isValid equals to true for all linkURL on page
9. end if

Based on SVM classification, we also use some pattern features and URL features to build a feature vector for describing sample cases, and use “1” or “0” to match these features. Moreover, since Google limits the load of intensive request from one single IP address, a proxy mechanism is also designed to handle this problem. The proxy mechanism can find some proxy sites from Internet automatically, and use the specific proxy sites to execute the retrieval operation. In addition, a series of heuristic rules are applied in Acrost to filter out a majority of irrelevant contents. These rules contain the link text, URL, etc. Meanwhile, the link filter algorithm can detect the similar links and prevent the collector from collecting large number of homogenous sub-sites.

The proposed link filter algorithm is presented in Algorithm 1. When a new link is coming, the depth and domain are checked. If this link is not visited before and specific keywords are contained in linkText or linkURL, each pair linkURL on page is processed and the number of homogenous links on page increased. Finally, the qualified links are selected to GSE-based collector.

#### 4.2. Topic-centric collector

A prime hypothesis is that the information about topic contents is often semi-structured. To discover the topic contents, topic-focused collecting is a practical and efficient way. In Acrost, we have developed a topic-focused collector and collected some data from IoT devices, such as sensor, mobile and alerts. In the process of
collecting, we use java packages to implement the operation of extracting links from downloaded contents and maintain a queue being collected with the help of core module of the topic-focused collector. The detailed workflow is shown as Fig. 7.

Theme relevance analysis also adopts SVM to calculate the topic-relevant value. According to $n$ keywords, Acrost builds a vector space of $n$ dimensions, the weights $w_1$ of each keyword is set as the variable of each dimension vector, it can be written as: $x = (x_1, x_2, \ldots, x_n)$, $i = 1, 2, \ldots, n$, $x_1 = w_1$. Then, we calculate the appearance frequency of each keyword, and that with the highest appearance frequency is used as the base to deduce the ratio of each appearance frequency of keywords, i.e., $f$. Each dimension vector of one page can be written as $f_{w_1}$, the theme of one page can be denoted as: $y = (f_{w_1}, f_{w_2}, \ldots, f_{w_n})$, $i = 1, 2, \ldots, n$. Thus, the topic-relevant value can be calculated by the cosine of these two vectors, as shown in formula (3):

$$\cos(x,y) = \frac{f_{w_1}^1 + \ldots + f_{w_n}^n}{\sqrt{w_1^2 + \ldots + w_n^2} \sqrt{f_{w_1}^2 + \ldots + f_{w_n}^2}}.$$ (3)

Finally, a threshold $t$ is assigned to determine whether a link of an identification is topic-relevant or not.

In order to provide proper links for scheduler and theme relevance analysis, 5 queues are maintained in Acrost that each queue stores a set of URLs in the same state.

Unvisited queue. In this queue, the links on object is waiting to be handled by collector, the new parsed links are pushed into this queue.

Processing queue. When the collector starts to work, the links on object will be sent to this queue. In order to guarantee one link to be handled only at a time, after one link was processed, it should be moved into fault queue, or abandon queue, or visited queue.

Fault queue. When an exception occurred during collecting data from an object, the links would be pushed into this queue. Once a link was moved into the fault queue, it would be discarded forever by the collector.

Abandon queue. In this queue, the link on object is not topic-relevant enough. It means that the value calculated through theme relevance analysis for the link is less than threshold $t$, but the link can be accessed normally. Besides, the links in this queue will be useless afterwards.

Visited queue. This queue stores the links on object that has been handled by collector, and the corresponding content has been downloaded to the local queue.

4.3. RE-based information extraction

Regular expression is an abstracted formula, which is used to match a certain type of strings using a specific pattern. For data collected by GSE-based collector, the contents are semi-structured, and the semantic ontology is located at the specific HTML tags of the DOM tree associated with contents. The RE-based information extraction is used to parse out metadata from the collected web pages, which is implemented by HTMLParser. The entire process of RE-based extraction is organized as follows.

Firstly, initializing an instance of HTMLParser. Then, defining a NodeFilter. Some child classes are used to allocate the specific filter, such as TagNameFilter, HasAttributeFilter, AndFilter and OrFilter. Thirdly, using the method parser () with the parameter of specific filter to get the object NodeList. Fourthly, traversing the NodeList and define proper regular expression to extract out the attribute name and value. Finally, using BufferedWriter to write the attribute name and value into txt files.

For example, a piece of HTML source code of web page in WikiCFP is shown in Fig. 8. In order to obtain the attribute name from this source code, we need to define the regular expression as: $(\text{td align=})/(\text{span})$. The attribute name can be extracted out by selecting the first match. Similarly, we can extract out the attribute value by defining the following regular expression:

$$(\text{td align=}/*\text{v}*/\text{startDate}*/\text{span})$$.

4.4. CRF-based information extraction

Conditional random fields is a conditional probability distribution of a sequence of labels given a sequence of observations, represented as $p(Y|X)$, where $X$ denotes the observation sequence and $Y$ presents the label sequence. The conditional probability is defined as:

$$p(Y|X) = \frac{1}{Z(X)} \exp(\lambda \cdot F(Y,X)),$$ (4)

where $F(Y,X)$ is feature function vectors defined on cliques of vertices and edges in the linear graph; parameter $\lambda$ is feature function weights vector corresponding to each feature function, and is to be estimated from the training data; $Z(X)$ is the normalization factor.

CRF-based extraction is used to parser out metadata from the contents crawled by topic-focused collector and implemented by using CRF. The contents in IoT devices are non-structured, and the semantic ontology is located at the random tags tree associated with contents without specific regular pattern. The features used in

![Fig. 7. Workflow of topic-focused collector.](image1)

![Fig. 8. Example of HTML source code.](image2)
CRF-based extraction include features of elements, features of blocks, and features of pages.

**Element features.** *Word feature* includes words in element and words in context. *Data feature* is like the keywords “2013”, “March”, etc. *Location feature* is got the location list from [http://www.world-gazetteer.com/dataen.zip](http://www.world-gazetteer.com/dataen.zip). If a word matches one item in the list, it is regarded as location information. *Title Feature* is like the keywords “st”, “nd”, “rd”, “th”, “conference”, etc.

**Block features.** *Children number feature* is the number of children of the block. *Position feature* is the center position of current block. *Area feature* is the area of current block. *Block keywords feature* denotes whether the first element before this block contains keywords.

**Page features.** *Page keywords feature* is like the keywords “Topic”, “Theme”, “deadline”, “key”, “committee”, “sponsor”, etc.

### 5. Semantic-aware retrieval

#### 5.1. Relevance ranking

With the help of topic relevance discovery, it is able to get the related contexts. However, being involved with massive retrieval results in IoT, it is essential to find the most relevant contexts for underlying themes. Based on this motivation, we propose the following formula to calculate the relevant rank of contents on a given topic, which is also used to evaluate the importance of retrieval results.

$$IF(s) = \frac{\sum_y \sum_{py} \text{cit}(p_y) / \sum_y \text{pcnt}(y)}{E(\delta)}.$$  \hspace{1cm} (5)

In Acrost system, the most relevant and latest contents are most important. Specially, \(IF(s)\) denotes the impact factor score of a searchable content referred to rank, \(y\) denotes the year of a searchable content, \(p_y\) denotes IoT object with searchable contents related to this topic at year \(y\), \(\text{cit}(p_y)\) denotes the quantity of searchable contents of \(p_y\), \(\text{pcnt}(y)\) denotes the number of IoT objects related to the topic at year \(y\). \(E(\delta)\) is average ratio between searchable contents related to the given topic and searchable contents of IoT objects in year \(y\).

**Algorithm 2.** Semantic-aware Retrieval Algorithm

```
Input:
querySentence: The query words entered by user.

Output:

hits: A sorted set of result docs.
```

1. **if** the length of querySentence equals to 1 **do**
2. **else**
3. **for** each keyword of querySentence **do**
4. **end for**
5. generate a TermQuery||Array of query keyword
6. **for** each TermQuery **do**
7. **end for**
8. generate a syntax tree according to querySentence
9. set the minimum number of keywords should match
10. **if**
11. create RangeFilter|| Array of searching topics
12. create SortField||Array of searchable contents in ascending order
13. create SortField of contents ranking in descending order
14. search the index by using RangeFilter and SortFields
15. **return** the hits by searcher

### 5.2. Retrieval algorithm

In this paper, a novel retrieval algorithm is proposed to search and sort the resulted contents. The retrieval process includes query parsing and result sorting. In query parsing stage, Acrost parses the query sentence, including lexical analysis, syntactic analysis, and linguistic processing. Through the lexical analysis, words and logical keywords like “AND”, “NOT” should be obtained; through the syntactic analysis, a syntax tree should be generated; through the linguistic processing, some words should be normalized. In result sorting stage, the retrieved results are ordered according to the relevance ranking score computed by Eq. (5). Thus, the most interesting information are provided to users.

The semantic-aware retrieval algorithm is described in Algorithm 2.

### 6. A case study: academic conference retrieval services

Academic search engines, a vertical search engines for scholar searching services, contain academic publications, international conferences, journals, patents, standards, researcher profiles, etc. In IoT environment, academic searching engine is designed to be concerned with physical objects rather than web pages. Firstly, IoT contributes to retrieval sources via extension of previously hardly or not possible to digitalize data from physical environments. Secondly, retrieval result in IoT consists not only of correct relevant knowledge in correct representation form with acceptable quality but also of appropriate time and geographical delivery place of information results.

As a case study, this paper describes a call-for-paper (CFP)-Oriented academic conference retrieval system for application-layer of IoT, which provides users with conference retrieval services and personalized submission recommend services based on conference theme information. This prototype system also relies on topic adaptation and on knowledge network that is built using semantic mining. The overall data of academic conferences are obtained by two different methods: GSE-Based Crawler is used to access general information on application-layer and CFP-Oriented Focused Crawler is used to access data sources from predefined topic objects whose contents belong to a specific area. When extracting metadata from crawled data, the case also uses two different ways to identify entities: linear CRF is used to extract metadata from non-structured pages and HTMLParser with regular expression is used to extract metadata from semi-structured pages.

#### 6.1. Experiment setup

Our experiments evaluate the performance of academic conference retrieval system based on Acrost and compare it with SemreX, CiteSeer\(^2\), and DBLP in terms of response time to simultaneous online requests and the scale of data records returned by using the same query keywords. The experimental datasets are collected through GSE-based crawler and the CFP-oriented crawler mentioned above. We also download academic conference web pages and divide them into two categories: conference lists webpages (CLW) and conference details webpages (CDW). The datasets are summarized as in Table 1.

Compared with ordinary websites, academic conference homepages have some unique properties in URL signature, web content feature and link characteristic, as shown in Table 2. Taking advantage of these features, SVM classification model is used to determine whether the site is academic conference homepage. The most common operation among Acrost, SemreX, CiteSeer\(^2\), and DBLP is query request. We use virtual clients to send their query requests to these 4 systems. For each query request, it carried a random key-
word selected from a predefined set which contains the popular term vocabulary. Taking some characteristics of information retrieval in IoT into consideration, some comparisons are done during experiments: For real-time retrieval in IoT, we test the response time under simultaneous online requests; For big data retrieval in IoT, we test the scale of data records returned by using the same domain-specific concepts as query keywords; For high-relevance retrieval in IoT, we test the scale of data records returned by using the same domain-specific concepts as query keywords; For push-service retrieval in IoT, we carry out some experiments to evaluate the functionalities of personalized submission recommendation services and the effect of academic conference ranking in Acrost.

6.2. Simultaneous requests

In order to observe the performance of Acrost and other 3 systems under different workloads, we varied the number of simultaneous online requests from 10 to 10,000 to calculate the corresponding response time. The results are shown in Fig. 10. It can be seen that the response time of Acrost is less than the other 3 systems when dealing with the same number of requests. The overall performance of both 4 systems decreases with the increase of simultaneous queries. As shown in Fig. 9, as the number of users reaches 2,000, the response time of each system increases sharply.

6.3. Query by abbreviations

As shown in Fig. 10, we use 80 different academic conference abbreviations keywords according to the list of important conferences on http://grid.hust.edu.cn/hjin/cfp.htm to search among Acrost, SemreX, and DBLP, the number of results returned by Acrost is more than that of SemreX and less than that of DBLP when using the same abbreviations keywords.

6.4. Query by topics

We used 10 different topic keywords (operating systems (OS), security, databases (DB), information retrieval (IR), architecture, network, artificial intelligence (AI), theory, graphics and multimedia) to search among Acrost, SemreX, CiteSeer, and DBLP. The experimental result in Fig. 11 shows that the number of results returned by Acrost is much more than those of SemreX, CiteSeer, and DBLP when using the same topic keywords. Acrost achieves more retrieval results since semantic-aware retrieval approach can obtain more related contents.

6.5. Submission recommendation services

For the experiment, we uploaded papers to Acrost randomly, and observed the parsing time and recommended academic conferences. For convenience, three samples were selected to present the results shown in Table 3. From the table, we can see that the overall parsing time is less than 1 s, and the topics of recommended conferences are close to the topics of uploaded papers mostly.

6.6. Conference ranking

The effect of academic conference rank was investigated through the rank attribute shown on result webpages of the prototype. Right now, we ranked 600 academic conferences of 9 research fields in computer science. As shown in Table 4, we compared our ranking results with the lists on http://www.cs-conference-ranking.org/. From the table, we can see that the precision rate of our conference rank according to perceived quality is perfectly acceptable.

Generally, from the experimental results of query by abbreviations, we can deduce that the scale of datasets in DBLP is larger than that of Acrost. But when we use topic keywords to search at DBLP, there is a few number of records returned. On the contrary, the returned results are very abundant when the same operation.

<table>
<thead>
<tr>
<th>Table 1</th>
<th>Datasets used in the case experiment.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
</tr>
<tr>
<td>CLW</td>
<td>HTML</td>
</tr>
<tr>
<td>CDW</td>
<td>HTML</td>
</tr>
<tr>
<td>CFP unit</td>
<td>TXT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 2</th>
<th>Unique properties of academic conference homepages.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Categories</td>
<td>Properties</td>
</tr>
<tr>
<td>URL signature</td>
<td>The sub sites of organizations or universities, e.g. (+ academic conference abbreviation + year)</td>
</tr>
<tr>
<td>Web content feature</td>
<td>Containing certain texts, e.g. call for papers, important date, etc.</td>
</tr>
<tr>
<td>Link characteristic</td>
<td>Having the links to sponsors, travel information, etc.</td>
</tr>
</tbody>
</table>
is conducted at Acrost. Thus, Acrost has the obvious ascendency on topic self-adaptive retrieval.

7. Conclusion

This paper presents Acrost, a retrieval system based on topic discovery and semantic awareness in IoT environment. Two topic centric collectors are combined to obtain the initial contents with interesting information. Moreover, the regular expression-based and conditional random fields-based approaches are also aggregated to extract the metadata. Besides, the query and ranking the relevance of contents are parsed to achieve the semantic-aware retrieval as well. The proposed approaches are validated through a case study on academic conference retrieval. The experimental results show that the proposed approaches can significantly improve the response time and efficiency of topic self-adaptive retrieval manner. Our future work will consider the automatic and dynamic algorithms on knowledge network searching in IoT and providing an informative overview to enhance the performance for searching in IoT.

Acknowledgments

This work is partially supported by National Natural Science Foundation of China under Grant 61272411 and Fundamental Research Funds for the Central Universities of HUST under Grant 2013TS099.

References


On-Demand Information Retrieval in Sensor Networks with Localised Query and Energy-Balanced Data Collection

Rui Teng * and Bing Zhang

National Institute of Information and Communications Technology, Kyoto 619-0288, Japan; E-Mail: zhang@nict.go.jp

* Author to whom correspondence should be addressed; E-Mail: teng@nict.go.jp.

Received: 29 October 2010; in revised form: 8 December 2010 / Accepted: 27 December 2010 / Published: 30 December 2010

Abstract: On-demand information retrieval enables users to query and collect up-to-date sensing information from sensor nodes. Since high energy efficiency is required in a sensor network, it is desirable to disseminate query messages with small traffic overhead and to collect sensing data with low energy consumption. However, on-demand query messages are generally forwarded to sensor nodes in network-wide broadcasts, which create large traffic overhead. In addition, since on-demand information retrieval may introduce intermittent and spatial data collections, the construction and maintenance of conventional aggregation structures such as clusters and chains will be at high cost. In this paper, we propose an on-demand information retrieval approach that exploits the name resolution of data queries according to the attribute and location of each sensor node. The proposed approach localises each query dissemination and enable localised data collection with maximised aggregation. To illustrate the effectiveness of the proposed approach, an analytical model that describes the criteria of sink proxy selection is provided. The evaluation results reveal that the proposed scheme significantly reduces energy consumption and improves the balance of energy consumption among sensor nodes by alleviating heavy traffic near the sink.

Keywords: on-demand information retrievals; localised data query; data collection; attribute-object name; sink proxy; balancing of energy consumption
1. Introduction

Recent progress in wireless sensor networks has revealed that multifarious sensors can be used to monitor various environment objects, such as plants, farm soil, factory instruments, and bird nests. To support ubiquitous computing, sensor networks should be integrated with the Internet, enabling people around the world to ubiquitously access information about the physical world [1]. In wireless sensor networks, sensor nodes can either periodically report sensing data to a server in a proactive manner, or deliver sensing data on-demand, namely when a user queries the sensor nodes only. On-demand information retrieval enables interaction between Internet users and sensor nodes, and lets a user retrieve up-to-date sensing information (such as the up-to-date landscape of a region) [1-4].

On-demand information retrieval consists of a data query phase and a data collection phase. A user initiates the process by sending a query message to a sensor network through the sink node, which broadcasts the query message to nodes in the network. The destination of a user’s query is not described by the sensors’ identifiers (IDs) but by the users’ interest, as represented by the name of data attributes [2]. For instance, a typical query would not be destined to a certain sensor node for its temperature data, but rather to the corresponding sensor nodes for ‘the temperature in the north-west quadrant’. Upon receiving the query message, the sensor nodes that match the query deliver sensing data to the sink node. The sink node then sends to the user the collected sensing data via the Internet.

Because of the power constraints of battery-powered sensor nodes, energy efficiency is a substantial requirement of information retrieval in sensor networks. For this reason, each query message should be sent to sensor nodes with small traffic overhead and sensing data should be collected with low and balanced energy consumption. However, on-demand query messages are generally forwarded to sensor nodes in network-wide broadcasts, which create a large traffic overhead. Although geocasting can be used to localise a query dissemination area, the success of conventional query geocasting relies on two assumptions: the user specifies a region as the query’s designated area and the specified query region includes all sensors that are corresponding to the query [5,6]. In fact, a user who issues a query to sensor networks often knows only the attributes of his/her interests, such as temperature information, and the name or location of relevant query objects such as a room. The location of a query object that the user knows may also be different from the location where the relevant sensors are deployed.

Sensing data collection has meanwhile been the subject of extensive research. Data aggregation is a key solution to achieve energy efficiency. To aggregate sensing data in the sensor network, various network structures such as clusters and chains are utilised. Most of these data collection structures are applied to the proactive data collection, in which sensors periodically report data to a data server [7-11]. Note that on-demand information retrieval initiates query and data collection at arbitrary times from users with diverse interests. This makes it difficult to utilise data aggregation structures of clusters or chains, because of the complexity of setup and maintenance of the network structures of data aggregation.

Many conventional on-demand data collection approaches adopt reverse trees, which are built based on the query dissemination. Sensing data are delivered in the tree towards the sink node, which is the root of the tree. However, this on-demand data collection is inefficient because of the large and unbalanced energy consumption in the network. Because most data are accumulated at a sink node with multihop relays, nodes near the sink node are likely to consume more energy than nodes remote to the sink node.
This paper addresses the problem of how to localise query dissemination and maximise data aggregation in an on-demand manner. The goal of the proposed approach is not only to reduce total energy consumption of information retrieval in sensor networks, but also to balance energy consumption among individual sensor nodes. We propose an approach of localised on-demand information retrieval that explores the name resolution for each query before the query message is forwarded to sensor nodes. The name resolution resolves a query name to the addresses of sensors that are corresponding to the query, leading to node-wise information retrieval in sensor networks. According to the location of each sensor that corresponds to the query, the area where the resolved sensors reside can be specified. To localise query dissemination and maximise in-network data aggregation, we propose the use of sink proxies as local query broadcasters and local data collectors. Each sink proxy is selected among sensor nodes according to the name resolution result.

The specific features of this paper are as follows: first, an attribute-object naming system and query resolution operation are adopted in the on-demand information retrieval. The query resolution, which could be considered analogous to the function of the Domain Name System (DNS) in the Internet, resolves a query’s name to local node IDs, the locations of a collective of corresponding sensors, and sink proxies, before the query message is disseminated to the sensor network. Second, we adopt a localised on-demand information routing scheme that consists of two parts: Localised Data Query Distribution (LDQD) and Localised Sensing Data Collection (LSDC). LDQD and LSDC significantly reduce the energy consumption of information retrieval. Given consideration of the temporal-spatial nature of user queries, LDQD and LSDC adopt dynamic sink proxies based on query content to improve the balance of energy consumption among sensor nodes. Third, to effectively select a sink proxy for each query, an analytical model is provided to describe the criteria of sink proxy selection. Both energy efficiency and energy balance are considered to effectively choose and utilise sink proxies. Fourth, this paper extensively studies the performance of the proposed approach for localised data query and collection through simulation evaluations of the NS-2 simulator integrated with the IEEE 802.15.4 module [12,13]. In the simulation, we investigate protocol performance in terms of both total energy consumption and the distribution of energy consumption among individual sensor nodes.

The remainder of this paper is organised as follows. Section 2 introduces the related work. Section 3 introduces the procedures for a proposed approach to localised information retrieval. Section 4 describes the system analysis of the proposed approach. Section 5 describes the evaluation and numerical results of the proposed approach and Section 6 concludes.

2. Related Works

Information retrieval in sensor networks can be broadly categorised into two types: proactive and on-demand. In proactive information retrieval, sensor nodes periodically report data with a predefined data rate and data delivery infrastructures. Meanwhile, in on-demand information retrieval, a query is delivered to the sensor network and data are collected according to the query’s requirements.

Existing approaches for proactive information retrieval mainly focus on the energy-efficient collection of sensor data [7-11,14,15]. To aggregate sensing data, sensor nodes are organised into infrastructures, such as clusters, chains or optimised trees. For example, LEACH (Low-Energy Adaptive Clustering Hierarchy) is a prevailing scheme of proactive data collection based on a dynamic
clustering approach [7]. In LEACH, data can be efficiently collected and aggregated in a hierarchical way. EEDC (Energy Efficient Data Collection) is another proactive data collection approach based on a cluster infrastructure [9]. It introduces an energy-efficient data collection approach by exploiting spatiotemporal correlation of sensing data. Energy efficiency is achieved by reducing the spatial sampling rate of sensor nodes. PEGASIS (Power-Efficient Gathering in Sensor Information Systems) organises sensor nodes into a chain structure rather than clusters [15]. The network structures of data aggregation, such as clusters and chains, significantly improve the energy efficiency of data collection. However, generating and maintaining a network structure entail much additional cost. In addition, a fixed network structure for data collection creates an energy imbalance. That is, the nodes near the sink node or cluster head generally consume much more energy than other nodes do.

Unlike proactive data collection, query-based on-demand information retrieval generally has a temporary organization of sensor nodes to collect data. A straightforward approach for on-demand information retrieval generally utilises a broadcast-reverse tree based model. A query is broadcast from the sink to the network and then a tree structure is constructed together with the task allocation of query. Sensors report their data back to the sink using the tree structure. Collected data can be aggregated on the tree. Most on-demand information retrieval approaches follow this model, but they differ in the mechanisms they use for query propagation and route selection for data collection [16-19].

Efficient query propagation is considered in [16]. The target is to propagate query message to a limited number of nodes so as to save energy. The ideal case is the minimized tree that includes no redundant broadcast of query message. However, this tree-based query requires repetition of sink queries and is not suitable to the arbitrary queries from users.

If a query specifies a query area, a spatial query broadcast can be archived based on geocasting or multicasting, which reduce redundant traffic overhead and achieve energy efficiency [5,20]. A typical approach Location based multicast (LBM) can be used to efficiently broadcast query messages in on-demand information retrieval in the case that there is a predefined region as the query destination [20]. LBM uses a multicast zone that is rectangular in shape and contains both the sender and all destination nodes. LBM assumes a roughly predefined region in which destination nodes reside. But the predefined region of destined sensor nodes might be either large or not available to a user who initiates on-demand information retrieval.

Although geocasting can be used to localise a query dissemination area, these operations rely on the assumptions that the user specifies a region as a query designated area. In fact, a user-defined region of destined sensor nodes may not always be available to a user who initiates on-demand information retrieval, or it may differ from the area that contains the sensor nodes corresponding to the query. There are other location based approaches that attempt to achieve localised routing operations in ad-hoc networks, however, most of these approaches focus either on end-to-end routing issues or only on the dissemination of a message to a predetermined area [5].

The route(s) for a sensor node to reply with data can be selected based on various criteria, for example, using a small hopcount in [21-23], minimum energy cost of data delivery in [17], adopting reliable routes from multiple candidates to avoid route failure in [3], or using an alternative path obtained from neighbouring nodes to improve security [24]. The reverse-tree-based data collection is simple to construct and imposes little additional overhead, as it is built on the basis of query dissemination. In contrast, a data collection tree with a root at the sink causes most of the collected
data to be aggregated at the sink, and nodes near the sink experience heavier traffic than nodes remote from the sink, draining their energy faster and leading to network faults even though there are many nodes with substantial residual energy.

A tour based data collection approaches is proposed in [25]. The aim of this approach is to save energy consumption in data collection and query propagation. The sink builds a source route as a tour for data collection, and the source route tour guides the delivery of query together with data collection from sensors. In the data collection from small sensors, this approach is good to reduce energy. However, in case of data collection from a large number of nodes, the source route is a high cost of overhead for each sensor to handle. Further, the efficiency of source route is low and complex [26].

Mobile sink based data collection has been studied in recent years. The support of mobile sinks in sensor networks increases the flexibility of user interaction with sensor networks. The mobile sink can be a user’s mobile phone. It can be applied in case a user is in the sensor networks. The mobile sink alleviates the energy overuse at a certain place as that in the sensor networks with static sink [27,28]. On the other hand, the mobility of mobile sink introduces the routing complexity. Using mobile sink to fetch sensing data was considered in [29]. When sensing data is required, the mobile sink will go to the place near the sensor node, and direct collect sensing data by one-hop communication. This requires the high intelligent sink which can move automatically. This approach might be applied to the intelligent robot network.

Recent researches have tried to analyze the phenomenon of uneven energy consumption in sensor network [30-32]. One attempt to alleviate the problem of energy uneven consumption by using specialized node deployment is proposed in [30]. More nodes are deployed near the sink node to share the energy consumption. The demerit of this approach is that the specialized deployment also causes deployment redundancy.

This paper discusses the problems in on-demand information retrieval. The disadvantages of conventional on-demand information retrieval are summarized as follows: first, query messages are often broadcast throughout the network in a data-centric manner, without distinguishing the corresponding individual sensor node ID. These on-demand data queries generate an excessive amount of redundant network traffic and energy consumption in large sensor networks. Second, sensor nodes reply to queries by sending data back towards the sink node via various paths (such as the reversed paths obtained from query broadcasting), and most of the data are accumulated at a sink node. This causes high and unbalanced energy consumption, since nodes near the sink node are likely to consume more energy than nodes remote from the sink node.

3. Localised Information Retrieval with Query Resolution

3.1. Query Resolution Mechanism

A query resolution resolves the name of a query to the corresponding sensor nodes before the query message is forwarded to sensor nodes. A query resolution is somewhat analogous to the name resolution at the Domain Name System (DNS) widely used on the Internet. Domain name resolution is a process of resolving a host name to an IP address before a user sends the initial IP packet. A main function of domain name resolution is to facilitate the IP routing from the source node to the
destination. Similarly, the query resolution mechanism in our scheme attempts to resolve each query’s name to facilitate efficient query dissemination and data aggregation in sensor networks.

Figure 1 illustrates an example of a query resolution process. We use an Attribute-Objects name (AO name) based on low-level naming, that describes the query’s name as well as the properties of sensor nodes. A user’s query can be specified by the interested objects and attributes of sensor data. For instance, in a building automation application, a query might be what temperature (attribute) status is in the storage room (object). In a factory application, a query could concern the vibration (attribute) status around the robot in a manufacturing line (object). In the same way, a sensor node can be described by the object that it is monitoring, and the physical attribute such as temperature or humidity that describes a sensor’s type.

**Figure 1.** Query to IDs resolution.

A resolution table is adopted in the query resolution mechanism to discover the locality of sensor nodes corresponding to a query. The resolution table maps the attribute name of each query and sensing IDs. The query resolution table is implemented at the sink node of a sensor network. This is because the sink node generally has greater power and memory capacity than sensor nodes have. The query resolution table is initially achieved by the following two operations:

(a) Each sensor node registers its ID, location, sensing attributes and monitoring object (if the sensor knows it) with the sink node.

(b) A sink node constructs a table that maps sensing attributes and monitoring objects to sensor IDs.

The registration can operate in conjunction with the network configuration in the phase of network formation, minimising additional overhead. Because the sink node is generally a powerful node, it can maintain a large resolution table. Although the use of location information may increase the complexity of the system, the development of localization in sensor networks in the past decades has provided more and more low cost, easily-maintained locating system [19,33,34]. Many applications of sensor networks use location of sensors to know the network deployment, obtain sensing context, analyze sensing data, and to perform network maintenance, recovery, and task management, etc. Because a general static sensor node is used in a sensor network, the requirements for updating the
location of sensor node are minimal. The object that a sensor is monitoring refers to static objects that the sensors are monitoring over the long term, instead of the dynamic results that a sensor detected.

The sink node processes each query before it is disseminated to sensor nodes. As shown in Figure 1, when the sink node receives a query message, it resolves the name of the query (i.e., AT1, MO1 in the example of Figure 1) to the corresponding sensor IDs (i.e., node 1, 3, 9, 16, 27), according to the resolution table. After a query’s name is translated into an ID group corresponding to sensor nodes, the sink node calculates the query area by deriving a rectangle in which all corresponding sensor nodes reside. A rectangle area is calculated as follows, based on the locations \((x_i, y_i)\) of each sensor node \(i\) that is in the ID list obtained from the query resolution:

\[
\text{Rect} = [\min(x_1,x_2,\ldots,x_n), \min(y_1,y_2,\ldots,y_n), \max(x_1,x_2,\ldots,x_n), \max(y_1,y_2,\ldots,y_n)]
\]

where \(n\) is the number of corresponding sensor nodes.

A sink proxy that will be used for localised query broadcasts of queries and local data collections is then selected according to the query resolution result. As show in Figure 2, there are two types of sink proxy selection schemes. One is fixed selection scheme, in which an identical sink proxy (such as node 9 in the Figure 2 A) will be selected at different times for the queries (such as Q1, Q2, Q3 in Figure 2 A) that have the same query resolution result; the other is dynamic selection scheme, in which various sink proxies will be selected at different times for the queries that have the same query resolution result, as shown in Figure 2 (B).

**Figure 2.** Two Types of Sink Proxy Selections.

As for the fixed selection scheme, a simple way adopted in this paper is to select a sink proxy that is the “centre node” of the query area. The centre node is defined as the node that is nearest the centre location of the query area. If the closest nodes are equidistant from the centre of the query area, the node with the smallest ID is chosen as the centre node. In Section 3.4 we discuss the dynamic sink proxy selection process in detail.
Consequently, the sink proxy provides the node information at which the query message starts to be broadcasted and the query area gives a region in which a query message can be disseminated. According to the calculated query area and sink proxy, the distribution of a query message is performed by two steps: unicast distribution and geocast distribution. These are introduced in the following subsection.

3.2. Localised Data Query Distribution with the Sink Proxy

Localised Data Query Distribution (LDQD) consists of two steps: Query unicast, and Query geocast. The unicast distribution is used to deliver query messages from a sink node to the sink proxy. The sink node calculates a source route to a sink proxy based on the nodes’ location. There are a number of potential approaches to building a source route. For example, the sink node first selects the node that is nearest the sink proxy among one-hop neighbours of the sink node. The selected node is included in the source route. In the same way, the sink node continues to calculate the next node in the source route, by selecting the node nearest to the sink proxy among the neighbours of the previous selected node in the source route. This process continues until a source route is found to the sink proxy. Similar approaches were elaborated in [35].

After the query message is delivered to the sink proxy in the query area, it is geographically broadcast to all sensor nodes inside the query area. The sink proxy forwards the query message to its one-hop neighbours. As illustrated in algorithm1, upon receiving the query message, each sensor node checks whether it should relay the packet to its neighbours by the following rules: a) Has not received this packet before; b) In the query area. If both a) and b) are satisfied, a sensor node will relay the packet to its neighbouring nodes. Otherwise, the sensor node just discards the packet.

3.3. Localised Sensing Data Collection with the Sink Proxy

Localised Sensing Data Collection (LSDC) consists of three steps: local data delivery, data aggregation, and aggregated data delivery to the sink node. An example of the LSDC scheme is illustrated in Figure 3. On receiving a query message, the corresponding sensor nodes send the sensing data back to the sink proxy in a local region. This is achieved using the reverse path obtained from the query geocast initiated by the sink proxy. The sink proxy, such as node 9 shown in Figure 3, collects the sensing data locally before forwarding it to the sink node. The sink proxy aggregates the sensing data by placing multiple sensing data into one packet. To efficiently aggregate sensing data and deliver the aggregated data to the sink node, the sink proxy sends the aggregated data according to the following two conditions: (a) The number of aggregated sensing data equals the largest number that the packet can contain; (b) The number of aggregated sensing data equals the number of the total corresponding nodes, which is calculated from the sink node. If either of these two conditions is satisfied, the sink proxy delivers the aggregated packet to the sink node. The sink proxy sends aggregated data to the sink node by unicasting, which can be obtained from the query dissemination. Finally, the sink node collects all data that correspond to the query. The collected data are then delivered to the user who initiated the query.
3.4. Balancing of Energy Consumption and Rotations of a Sink Proxy

Query content varies according to users’ interests, and the group of sensor nodes responding to a query varies with the query contents. This query diversity generally enables localised data collection to not only alleviate the overuse of energy at nodes near the sink node, but also to avoid overuse of energy at particular sensor nodes, such as sink proxies, achieving distributed energy consumption among sensor nodes.

In the case many users have same query interest at different times, a fixed sink proxy will result in an energy overuse at the sink proxy corresponding to these queries. This is because that the multihop relay features of wireless sensor networks require nodes near (in hop-distance) the sink proxy relay packets from nodes further away from the sink proxy [30,31]. To avoid such overuse of energy at a sink proxy, a rotation operation on the sink proxy is utilised. When a sink node receives a query that asks for the same content as that of the previous query, it will assign a new sink proxy randomly in the query area, in order to distribute the energy among the local sensor nodes corresponding to the query. We use LDQD-random and LSDC-random to generally represent the approaches that use randomly selected nodes as sink proxies for LDQD and LSDC.

4. System Analysis

In this section, we describe an analytical model of sink proxy selection. Furthermore, we analyze the impact of the proposed approach on the energy bottleneck in data collection. Here, we call the information retrieval model in our proposed approach the S3 (Sink-Sink_Proxy_Sensors) scheme, and call the conventional on-demand information retrieval model as S2 (Sink-Sensors).
4.1. Selection of Effective Sink Proxies

The effective selection of sink proxies is expected to result in both balanced and decreased energy consumption of compared with the conventional S2 retrieval model. Because the use of sink proxies enables maximum data aggregation among various sensor nodes, it alleviates the heavy energy consumption near the sink. Hence, if the use of each selected sink proxy also causes a lower energy cost compared with the conventional S2 information retrieval approach, the proposed approach then achieves both balanced and decreased energy consumption. Therefore, the basic criterion for the selection of a successful sink proxy is that both the following two conditions are satisfied: (a) The energy cost of the S3 query is no larger than that of conventional S2 query; (b) The energy cost of the S3 data collection is not larger than that of the conventional S2 data collection.

A. Data query cost and effective sink-proxies for the data query.

For analytic tractability, we analyze S2 and S3 schemes in a simple setting. We assume the query area is a square consisting of N nodes. The average energy consumption of a one-hop transmission of a packet is assumed to be $E_0$.

In the one-to-many model of an S2 query, flooding is typically adopted for disseminating a query to nodes in the network. The energy cost $E_{Q2}$ of a data query is given by:

$$ E_{Q2} = N \times E_0 $$

(2)

In S3 information retrieval, in contrast, the operation consists of unicasting from sink to sink_proxy, and geocasting from sink_proxy to sensors in the geocast area, which is a combined rectangle area that contains both sink proxy and the query area. Given that $b$ is the ratio of the node number in the geocast to the total node number N, and $0 < b \leq 1$. Therefore, $b \times N$ is the node number in the query geocast area of S3. Let $R_u$ be the hops of transmission in the unicast from the sink to the sink proxy. Then, the energy cost of S3 query can be computed as:

$$ E_{Q3} = b \times N \times E_0 + R_u \times E_0 $$

(3)

As a result, the ratio $R_Q$ of query cost of S3 to S2 is given as follows:

$$ R_Q = \frac{E_{Q3}}{E_{Q2}} = \frac{b \times N \times E_0 + R_u \times E_0}{N \times E_0} = \frac{b \times N + R_u}{N} $$

(4)

The smaller $b$ and $R_u$ are, the smaller $R_Q$ is. This means greater energy savings.

When a S3 query saves energy compared with the conventional S2 approach, the ratio of the S3 query cost to the S2 query cost is no larger than 1. That is:

$$ R_Q \leq 1 $$

(5)

When $R_u \leq N(1-b)$, we have $R_Q \leq 1$. Therefore, an effective sink proxy should enable $R_u \leq N(1-b)$. If a query interest area is defined, the value of $b$ and $R_u$ can be determined by the position of a sink proxy. Given a sink proxy candidate with location coordinates of $(x,y)$, and an resolved query area $(X_{min}, X_{max}, Y_{min}, Y_{max})$, as shown in Figure 4, we can determine whether the sink proxy candidate is a suitable sink proxy for S3 query geocasting by calculating $b$ and $R_u$ if the sink proxy candidate is used. The query geocast area in S3 model can be calculated as $(X'_{min}, X'_{max}, Y'_{min}, Y'_{max})$, where $X'_{min} = \min(x, X_{min})$, $X'_{max} = \max(x, X_{max})$, $Y'_{min} = \min(y, Y_{min})$, $Y'_{max} = \max(y, Y_{max})$. 
Y’\text{max} = \max(y, \text{Ymax}). We can then determine the ratio \( R \) of query geocast area to the network area. \( R \) equals \( b \), assuming nodes are uniformly deployed. Further, since \( R_u \) is the shortest hopcount from the Sink to the sink proxy candidate, it can be calculated according to the Greedy Perimeter Stateless Routing (GPSR) hop-by-hop routing process [26]. Consequently, \( RQ \) can be obtained.

If:
\[
R_Q = \frac{E_{Q2}}{E_{Q3}} = \frac{b \ast N + R_u}{N} \leq 1,
\]
then the sink proxy candidate can be selected as a sink proxy for S3 query geocasting.

\textbf{Figure 4.} Selection of effective sink proxy for data query.

B. Data collection cost and effective sink-proxies for the data query.

In the many-to-one model of S2 data collection, the energy cost can be approximately given by the sum cost of the replied data unicast from each sensor node to the sink. Note \( \alpha \) is the ratio of sensor nodes that will reply a query message, and \( \alpha \ast N \) is the node number of sensor nodes corresponding to the query of S3. Let \( \bar{R}_0 \) be the average route length from each corresponding sensor node to the sink. Thus, the energy cost of S2 data collection, denoted by \( E_{C2} \), can be written as:
\[
E_{C2} = \alpha \ast N \ast E_0 \ast \bar{R}_0
\]  

In S3 information retrieval, meanwhile, the data collection cost is the sum cost of local data collection and delivery of the aggregated data. Given \( \bar{R}_a \) as the average route length from each corresponding sensor node to the sink proxy, let \( \beta \) be the ratio of aggregated data size to the un-aggregated data size. Thus, the data collection cost in S3 information retrieval \( E_{C3} \) can be denoted by:
\[
E_{C3} = \alpha \ast N \ast E_0 \ast \bar{R_a} + \beta \ast (\alpha \ast N) \ast R_u \ast E_0
\]  

Consequently, the ratio of data-collection cost of S3 to S2 is computed as:
\[
R_C = \frac{E_{C3}}{E_{C2}} = \frac{\alpha \ast N \ast E_0 \ast \bar{R_a} + \beta \ast (\alpha \ast N) \ast R_u \ast E_0}{\alpha \ast N \ast E_0 \ast \bar{R}_0} = \frac{\bar{R_a} + \beta \ast R_u}{\bar{R}_0}
\]  

In data collection operations, adopting an appropriate sink proxy consumes no larger energy than the conventional S2 approach. That is:
\[
R_C \leq 1
\]
when \( r_C \leq 1 \) (\( R_0 > R_u + \beta R_u \)), there is energy saving in S3 data collection compared with S2 data collection.

This result indicates that the smaller \( \beta \) is, the smaller the relative cost of S3 becomes. The use of a sink proxy enables maximum aggregation of small sensor data and the smallest \( \beta \), since all data are collected at the sink proxy and are aggregated before being transmitted to the sink.

\( R_0 \), \( R_u \) and \( R_u \) can be calculated at the sink node based on the GPSR route discovery protocol that is described in [36], given the location of the sink proxy and resolved nodes corresponding to the query. \( R_0 \) is a parameter that depends on the position of the corresponding sensor when using the shortest path routing. The values of \( R_u \) and \( R_u \) depend on the location of the sink proxy. As a result, the ratio of data-collection cost of S3 to S2 depends on the aggregation ratio and the position of sink proxy.

4.2. Impact on the Energy Bottleneck

With conventional S2 data collection, there is an energy bottleneck at the sink node. This occurs in multihop wireless sensor networks because most of the data are accumulated at the sink node, and nodes near the sink node are likely to consume more energy than nodes far removed from the sink node. The impact of the S3 model is that data are aggregated to the maximum degree before being delivered to the sink node and the sink proxy is selected from various sensor nodes, so that energy consumption is much balanced.

We analyze the impact of the S3 model on the energy bottleneck, with a specific focus on the energy reduction at the one-hop neighbour nodes of the sink node. Given a sensor network consisting of \( N \) nodes, and average energy consumption of a one-hop transmission of a packet assumed to be \( E_0 \). Note \( \alpha \) is the ratio of sensor nodes that will respond to a query message. Then, energy consumption \( E_{S2}(\text{one-hop}) \) at one-hop neighbour nodes of the sink in the conventional S2 model can be denoted as:

\[
E_{S2}(\text{one-hop}) = \alpha * N * E_0
\]  

(10)

where \( \alpha * N \) is the number of sensor nodes corresponding to a query. The energy consumption at the one hop neighbour nodes of sink in S3 model is denoted as:

\[
E_{S3}(\text{one-hop}) = \alpha * N * E_0 * \beta + \gamma * E_0
\]  

(11)

where \( \beta \) is the ratio of aggregated data size to un-aggregated data size, \( \gamma * E_0 \) is the energy consumed at one-hop nodes for relaying a packet from a sensor to the sink proxy before collecting data at the sink. When the network is large, the number of one-hop nodes is small compared with the number of nodes in the network, the relay probability is very small, and \( \gamma * E_0 \) can be ignored, and so:

\[
E_{S3}(\text{one-hop}) = \alpha * N * E_0 * \beta
\]  

(12)

This means that energy consumption is \( \beta \) times the S2 model (\( \beta \leq 1 \)). The greater the aggregation capability of the network, the smaller the \( \beta \) is, leading to more energy savings. For example, when 10 packets each with 5 bytes of payload data are aggregated into one packet with a 10 × 5 payload, sharing one header with 28 bytes, \( \beta \) is \((10 \times 5 + 28)/(5 + 28) \times 10 = 25\%\).
5. Simulation Evaluation and Numerical Results

5.1. Evaluation Metric, Objects, and Simulation Setup

We evaluate our proposed protocols using an NS-2 simulator, which is integrated with the IEEE 802.15.4 based MAC module [12,13]. In the simulation description, we call conventional on-demand approaches S2 (Sink-sensors), and our proposed approach S3 (sink-proxy-sensors).

The following protocols are evaluated in the simulation:

- Flooding: A S2 approach with flooding based query dissemination and many-to-one data collection from reverse paths of the flooding [3].
- LBM based information retrieval: A S2 approach with location-based multicasting for query dissemination and data collection from reverse paths of multicasting [20].
- LDQD: Localised Data Query Distribution in S3 approach.
- LSDC: Localised Sensing Data Collection in S3 approach.

LDQD and LSDC schemes are classified according to the position of the sink proxy. For instance, schemes that choose the centre node in the query area as a sink proxy are called LDQD centre and LSDC centre; schemes that use the nearest node in the query area as a sink proxy are called LDQD sink; schemes that use the random selected sink proxy in the query area are called LDQD limited random and LSDC limited random, schemes that use the network-wide random selected sink proxy are called LDQD network random and LSDC network random, etc.

The protocols are evaluated using the following metrics:

- Ratio of energy consumption: The ratio of energy consumption for the proposed protocol S3 compared with conventional approaches S2, which includes flooding-based and LBM based protocols. It is defined as: \( R_{ratio} = \frac{E_{S3}}{E_{S2}} \); where \( E_{S2} \) is the energy consumption of conventional approaches of on-demand information retrieval (including Flooding and LBM); and \( E_{S3} \) is the energy consumption of proposed approaches of on-demand information retrieval.

- Energy consumption at each node: The energy consumption at each sensor node during the simulation.

Unless otherwise stated, the simulation is set up as follows. There are 100 sensor nodes and 1 sink node in the network. The sensor nodes are distributed over a square of 100 m × 100 m, and the detailed topology of the sensor network is shown in Figure 5. The network topology is static and all nodes are connected to each other by either single-hop or multi-hop links. Each sensor node is equipped with an IEEE 802.15.4 radio module, and has a radio range of 15 m. The maximum number of sensing data that can be aggregated in a packet is set to 10.
5.2. Numerical Results

We first study the total energy consumption and data delivery ratio in sensor networks. To evaluate the information retrieval operation with various queries from users, in the simulation we adopt the variable of square areas in which users query sensors for data. The selection of the area being queried is defined by squares of different dimensions. The position of each square in relation to the corresponding nodes is randomly selected among the network. The total energy consumption in a round of data query and collection is the average of 10 cycles of data query and collection.

Figures 6 (A) and (B) present the ratio of energy consumption in the proposed S3 based approaches to the conventional S2 based protocols, which include Flooding and LBM. Most of S3 based approaches, except for LDQD + LSDC Network Random approach, highly reduce the energy consumption compared with S2 approaches, especially when the length of the square being queried is small. In the S3 based approaches, the LDQD + LSCD Center approach that adopts the centre node of the query area as the sink proxy achieves optimised performance when the length of the square being queried is larger than 20, with a ratio of 25–90% energy consumption compared to the flooding-based protocol and 50–90% compared to the LBM-based protocol. The smaller the length of the square being queried, the larger the ratio of energy savings for the proposed approach. The exception is the LDQD + LSDC Network Random, which has greater energy consumption than LBM when the length of square is small because the network-wide random selection of a sink proxy cannot achieve a localised query and data collection for small query areas.

In addition to the evaluation of total energy consumption in the network, we studied the distribution of energy consumption at individual sensor nodes. To evaluate the impact of dynamic sink proxy rotation on the energy consumption among individual sensor nodes, we set up a simulation consisting of 40 identical queries with and each query was followed by a data collection operation. The length of
query square is set to 40 m and the maximum number of sensing data that can be aggregated in a packet is set to 20.

**Figure 6.** Ratio of energy consumption for approaches using S2 and S3 models.

![Energy Consumption Ratio](image)

**(A)** Energy consumption ratio of S3 to S2 with Flooding.

**(B)** Energy consumption ratio of S3 to S2 with LBM.

Figures 7 (A) and (B) shows an example of the distribution of energy consumption at each sensor node in the data collection operation. Flooding-query based data collection and LBM based data collections have high and non-uniform energy consumption with respect to both sending and receiving data. LSDC-center and LSDC-sink based data collections have much lower energy consumption at sensor nodes. LSDC-Random, in which the sink proxy is random selected in every query and rotates among sensor nodes, has the lowest value of the max energy consumption among sensor nodes. Hence, it has most uniform energy distribution among sensor nodes.

We also set up a simulation consisting of three query objects, which are selected in the network. Sensor nodes monitoring each object are located within a 20 m × 20 m square, in which effective sink proxy can be obtained. For each query object, 40 rounds of queries and data collections are performed. In the simulation S2 approach adopts LDQD + LSDC Limited Random with effective sink proxies.
Figure 7. Energy consumption at each sensor node in data collection.

(A) Energy consumption of sending packet at each node.

(B) Energy consumption of receiving packets at each node.

Figure 8 (A) illustrates the energy consumption at nodes with respect to their hop-distances to the sink. In the conventional S2 model, nodes of one-hop neighbours of the sink have the largest energy consumption, far larger than that of nodes with a long distance to the sink. In the S3 model, the energy consumption was about 40% of that of the S2 model at nodes of one-hop neighbours of the sink, leading to significant alleviation of the energy bottleneck. Figure 8 (B) shows the energy consumption ratio of the S3 model to the S2 model. Energy consumption at nodes near the sink is significantly reduced using the S3 model. On the other hand, energy consumption at nodes located at a far distance is larger using the S3 model, leading to a more balanced energy distribution compared with the conventional S2 model. This is because the S3 model utilises dynamic sink proxies to balance energy consumption, and the sink node’s operations are distributed to sensor nodes.
Figure 8. Energy consumption with regard to hop-distance.

(A) Hop-wise energy distribution.

(B) Energy consumption ratio.

Figures 9 (A) and (B) show the energy consumption of data queries at each sensor node. S2 queries have high energy consumption at most sensor nodes. The proposed S3 based query has much lower energy consumption for most sensor nodes. The largest energy consumption of the S3 query among sensor nodes is about 0.13 J, which is much smaller than that of S2 (0.2 J).

Figure 9. Energy consumption distribution among nodes.

(A) Conventional S2 Query energy distribution.
Figure 9. Cont.

(B) Proposed S3 query energy distribution.

(C) Conventional S2 data reply energy distribution.

(D) Proposed S3 data reply energy distribution.
Figures 9 (C) and (D) show the distribution of energy consumption among sensor nodes in the data collection (reply) operation. S2-based data collection has high and non-uniform energy consumption at sensor nodes. Nodes near the sink node have the highest energy consumption: up to 0.42 J. In contrast, S3-based data collection has much lower energy consumption at sensor nodes. The highest energy consumption for S3 is 0.17 J, which is about 40% of that in S2-based data collection. Further, S3 data collection has more uniform energy distribution among sensor nodes.

6. Conclusions

This paper has proposed an on-demand localised information retrieval scheme for sensor networks with awareness of a user query’s content. In the proposed scheme, a query’s name is resolved into the IDs and locations of corresponding sensor nodes before being distributed to the network. According to the location of sensor nodes, query distribution and data collection are performed in a corresponding local area. The query message is efficiently unicasted to the sink proxy in a query area, and is then forwarded to a localised area of the network. Sensing data are collected at a sink proxy, at which data are aggregated and sent to the sink node. We provided an analytical model to describe the criteria of sink proxy selection and we analyzed the impact of the proposed approach on alleviating the energy bottleneck in data collection. The simulation results show that the proposed scheme significantly reduces the energy consumption of data query and data collection in the network. And the energy consumption is more uniformly distributed among sensor nodes than in conventional approaches of on-demand information retrieval. The proposed scheme achieves 60% of energy reduction at the neighbouring nodes of the sink in the simulation.

From the evaluation results, we know that the proposed scheme is promising for applications where sensing data are queried on-demand from users with diverse interests. As for future work, the in-network and high-level context abstracting of sensing data are considered. The maximised data aggregation in the proposed information retrieval model is expected to provide high effectiveness of context abstracting at sensor nodes.

References


13. Department of Electrical Engineering, the City University of New York, New York, NY, USA; Available online: http://ees2cy.engr.ccny.cuny.edu/zheng/pub/ (accessed on 1 December 2010).


