Research on Improving, Evaluating and Applying the Ternary Search Tree and Binary Search for Storing and Searching Content - Based Address for Forwarding Technique in Service-Oriented Routing

Nguyen Thanh Long, Nguyen Duc Thuy, Pham Huy Hoang, and Tran Dinh Chien

[1] Informatics Center of Ha noi Telecommunications
75 Dinh Tien Hoang, Hoan Kiem, Ha Noi, Viet Nam
{longptpm,chientd}@vnpt-hanoi.com.vn
[2] Research Institute of Posts and Telecommunications
122 Hoang Quoc Viet, Nghia Tan, Cau Giay, Hanoi, Viet Nam
thuynd@ptit.edu.vn
[3] Hanoi University of Science and Technology
1 Dai Co Viet Road, Hanoi, Viet Nam
hoangph@soict.hut.edu.vn

Abstract. Service-based network infrastructure is a new network interface in which the flow of messages is controlled by class of services that generated it. Next is its content, improved shipping address specified by the sender and attached to the message. Networks based on services complement for networks based on traditional unicast and multicast addresses, which provides support for communication patterns based on the service class of large-scale applications, loose connections, multiple partitions and scattered like auctions, information sharing, distributed according to personal information.

With Service Based Routing (SBR), the sender does not indicate message receiver by the unicast or multicast use. Instead it simply pushes messages to the network. It defines the routing based on the messages it cares. It determines the appropriate message class based on message content based on its key-value pairs or regular expressions. Therefore, in SBR routing the receiver determines the transmission of messages, not the sender. Communication based on content services increases the independence, flexibility in the distributed architecture. In SBR the routing table consist of content based addresses, We have to find the structure to store and organize routing table efficiently and save memory and time to search all its items match a content message. In this paper We introduce and improve *Ternary Search Tree* structure to use for storing and process the SBR routing table.

Keywords: Ad hoc network, MANET, TST, ternary search tree, binary, forwarding, service, content.

H. Qian and K. Kang (Eds.): WICON 2013, LNICST 121, pp. 189–202, 2013.

1 Introduction of Service-Oriented Routing

1.1 Service-Oriented Routing Protocol Model

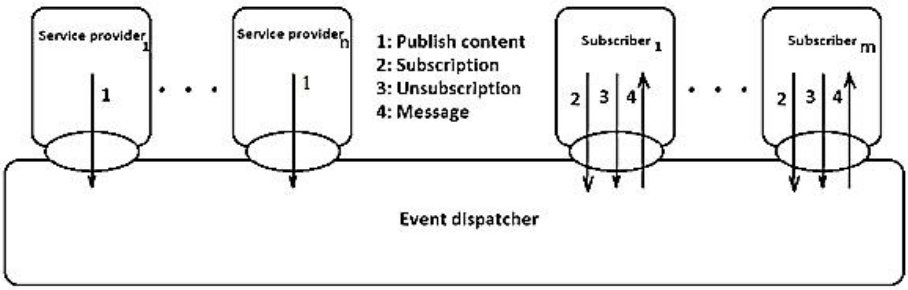

Fig. 1. Model of service-oriented routing network

Interaction model of service delivery or content publishing and content subscription (abbreviated by P / S) is the interaction model is done asynchronously in the service based routing (SBR)system.

Most of the SBR routing system is based on the P / S model.

1.2 Overview of Service Oriented Architecture

Service Oriented Architecture is the architecture based on the patterns.

There are three types of Pattern:

a) Architecture;
b) Design;
c) Enforcement;

In Pattern-oriented software, schema structured platform is shown for software systems. It provides a set of subsystems defined, indicating the ability to perform, including the rules and guidelines for organizing the relationship between them.

Overlaying the traditional point - to - point: modern communication based on Virtual Endpoint / Message Broker.

To allow flexible communication, the ability to provide location transparency for the transmission of information between applications. Position transparency is defined as to avoid the point - to - point connection because the application is separated with the data transmission services below, will undertake the implementation when the applications require communicating with each other.

1.3 Characteristics of Service-Oriented Routing

Service-based network infrastructure is a new network interface in which the flow of messages is controlled by class of service that generated it. Next is its content, improved shipping address specified by the sender and attached to the message.

Networks based on services complement for networks based on traditional unicast and multicast addresses, which provides support for communication patterns based on the service class of large-scale applications, loose connections, multiple partitions and scattered like auctions, information sharing, combined, distributed, sensor networks, distributed according to personal information, service discovery, multi-player game.

In the SBR routing, the sender does not indicate message receiver by the unicast or multicast use. Instead it simply pushing messages to the network. It defines the routing based on the messages it cares. It determines the appropriate message class based on message content based on its key-value pairs or regular expressions. Therefore, in SBR routing the receiver determines the transmission of messages, not the sender. Communication based on content services increases the independence, flexibility in the distributed architecture.

2 Organization of the Message Types in SBR Network [7]

2.1 The Subscription / Unsubscription Message

Subscription / unsubscription message is emitted from the application service classes to subscribe / unsubscribe content requests. The message is structured: the address of the subscriber and binding on the list of services and content requirements (constraints). In particular, each constraint is a set of 3 components, has the form: (key, operator, value). For example, the contents of the registration message: [service_class = "Network monitor" ∧ alert-type = "instrusion" ∧ severity> 2] or [service_class = "Network monitor" ∧ class = "alert" ∧ device-type = "web-server"], these are 2 request messages of Network monitor service class.

2.2 The Content Message

Content messages are transmitted from the host service provider. These messages will be transmitted to the network, it will be transmitted to the machine based on the subscription request message received from that machine.

2.3 The Advertisement Message

Advertisement message is the message advertises the basic content that certain services provide for applications. To direct the subscription requests to the right offer places. Prevents spread the subscription messages throughout the network.

Advertisement message is very useful for establishing multicast tree from the root that is destination node to a group of matched content providing nodes. Then matched content will be sent simultaneously from leaf nodes to root node by this multicast tree.

Establish multicast tree by a pair of messages Route Request (RREQ) and Route Reply (RREP): i) the node has subscription request will make RREQ, broadcasts it to network; ii) any node that has advertisement messages that are matched this subscription message, will make RREP that stores the path from the content publishing node to the content requesting node and forward back to this node; iii)

after a time interval, all paths from received RREP messages will make the multicast tree from requesting node to content publishing nodes.

The advertising message is passed under the minimum spanning tree from source node.

Advertisement message is also structurally similar content message, including a set of attributes. So the content message is to expand the content of advertising messages.

Advertisement message is transmitted from the service delivery system.

2.4 The Message Pair of Update Request and Reply Sender

Use this pair of messages to update source node routing table on demand. Sender request: request from the router, the structure includes 3 fields. Two fields: sender address and request number determine the uniqueness of the message. Timeout determines the longest time the sender waits for an answer. This message is transmitted by the minimum spanning tree (Broadcast) starting from the source router (formed by Prim algorithm) to the other routers in the network.

When the leaf router nodes of Broadcast tree receive the sender request message, it will respond by reply update (UR) message. The message consists of three fields, two fields from the sender request, the field no.3 contains its content based address. The UR message spreads back the sender router.

On the way to the sender router, the intermediate router will incorporate its content based address and of the message then push it to the sender router.

When the sender router receives UR message, it updates its routing table. End of the implementation process.

2.5 Route Request and Route Reply Message Pair

This pair of messages is used to detect routes on the network to a set of destination addresses from a source node for building multicast trees with root is source node and leaf nodes are this set of destination nodes.

Route request message is used to make a request from the source router, the message structure consists of eight fields. Two fields: 1) Source Address and 2) Request Number determine the uniqueness of the message. 3) The Type field identifies of the kind of message, is set to 1; 4) the Timeout field determines the longest time the sender node waits for a response. 5) The Time To Live (TTL) field determines the maximum number of HOPs of the route that message is passed on. 6) The Route field records addresses of the hops on the route the message passes through. 7) The Free Time Slots field records free bandwidth at the nodes of the route that message is transmitted on. 8) The Destination List field saves address list that contains addresses of the set of destination nodes that are the leaf nodes of multicast trees that we need to build. The message is transmitted by broadcast protocol to other routers in the MANET. When a node receives RREQ, it checks whether it's address is in destination field, if it is true, this node makes Route Reply message and forwards back it to the source node on the detected route. RREQ message contains all fields as RREQ message except the destination field.

3 The Concepts and Terminology

3.1 Service Based Forwarding Table Concept

Each service has a separate forwarding table.

The table includes predicates, also called content-based addresses.

Forwarding table is mapping 1-1 between content-based address and the identity of the subscriber.

Predicate is logical disjunction of filters are received from the subscriber.

Forwarding algorithm is the algorithm finding in the forwarding table the filters match message content received.

Filter is required content (subscription) from a subscriber. Each filter is a combination of subscriber identification and conjunction of the constraints.

3.2 Routing Technology in Service Oriented Routing

A. Find relevant content and communications

Upon receiving a content message from certain application services on the network. The router compares the received data with predicate list to determine compliance Predicates. If found, the message will be pushed to the requested nodes respectively.

The algorithm finds out set of routers have predicates match message received.

How to find the shortest path from source node to the nodes with compliance requirements:

Option 1: Find the shortest path from source node to the appropriate node on the request graph is weighted according to **Dijkstra**'s algorithm. Synthesis of the routes forms the tree for transmitting messages from the source node.

Option 2: Find the minimum spanning tree by **Prim**'s algorithm, using the nearest neighbor method. The tree is found to satisfy the conditions of having root is the original source node, the leave nodes are the requesting nodes.

At each router has appropriate request, when receiving a message, it sends the message to the request nodes in its own network.

3.3 Messages Forwarding Technique in Service-Oriented Routing

A. Concepts and terminology:

Forward algorithm is finding algorithm in the forwarding table the filter matches message content received.

The transition Table is the 1-1 relation between Predicate and the identity of the subscriber.

Filter is required content subscription from the subscriber. Each filter is a combination of subscriber identification and conjunction of the constraints.

Predicate is disjunction of filters is received from the subscriber.

Forwarding table includes predicates also called content-based addresses.

B. Find relevant content and communications

1. Find matched predicates

Upon receiving a content message from certain application services of the network. Router will compare the received data with stored Predicate list on it to see which predicate matches. If found, the message will be pushed to the requested node.

The algorithm will find all routers have matched predicates with the message received. Divide the set of all constraints into two subsets: i) the first subset includes constraints which have value component of string type; ii) the second subset includes constraints which have value component of number type. Build improved ternary search tree from the first subset of constraints. Build a dynamic array with each element for each separated key, an element of the array is the root of a multi branches tree, a branch for one of operators that stores all values of all constraints with this key and operator.

2. Routing requested content to the nodes that have matched subscription

How to find the shortest path from source node to the node with compliance requirements:

a) Method 1:

Find the shortest path from source router to other routers in the network by **Dijkstra**'s algorithm.

The algorithm ends when finding the shortest path to all routers have consistent requirements. So we find the tree of transmitting messages from the source router to routers with the matched requirements by removing the leaf nodes don't have matched requests.

On the paths from the source router, the intermediate routers don't have matched requests will relay the message to the next routers on the tree.

When the message has reaches the leaf routers, will pass the message to its child nodes in their own network and don't relay message to other routers on the network.

b) Method 2:

Find the minimum spanning tree by **Prim** algorithm, using the nearest neighbor method. The tree found to satisfy the conditions of having root is the original source node, the leave nodes are the request nodes.

Description of algorithm: The graph consists of n vertices or routers form the network, the weights of links between routers is given by the weighted matrix C. Conventional weighting of the link between not directly connected routers is a very large number.

At the tree T and a node v, called the minimum distance from v to T is the smallest weights of the edges connecting v to a vertex point in T: $d[v] = \min \{c[u, v] \mid u \in T\}$.

1. Initially started skeleton tree T consisting of the source node s: $T = \{(s,)\}$.

2. Then just select from the set of nodes outside of the tree T a node v has link e with the smallest weight, admitting it to T, simultaneously admitting that link e into skeleton tree T: $T = T \cup \{(v, e)\}$.

3. Perform step 2 until:

Or has admitted all of the requested nodes, we have T is the smallest skeleton tree from source node to all the request nodes.

Or not admit all of the request nodes when the node outside T have links to any node of T has extreme large weight. When it will only transmit the message to a subset of request nodes.

At each router has appropriate request, when receiving the message it sends to the nodes have appropriate requests on its own network.

4 Storing Predicates Having String Values in Ternary Search Tree

4.1 Overview Ternary Search Tree

In computer science, ternary search tree (TST) is a tree data structure in which the child nodes of a tree are arranged as a standard binary search tree. Denote TST by recursive formula:

$$\mathbf{TST = \{TST_{left}, Root, TST_{right}\}} \quad (1)$$

In which Root stores the first character X of the first string, it has left and right pointers that point to left and right child TSTs denoted by $\mathbf{TST_{left}}$ and $\mathbf{TST_{right}}$ respectively**.** $\mathbf{TST_{left}}$ has root stores a character which stands before X in Alphabet table. $\mathbf{TST_{right}}$ has root stores a character which stands after X in Alphabet table. Root has middle child that store next character in current string. End of string is detected by storing a special character for example '#'.

The search for a string in a Ternary Search Tree consists of a series of binary search steps, each step for finding a character in a string. Current character in the string value compares with the character of the current node, one of the three options can happen:

i) If character in string value is less than then continues the search with the left child node.

ii) If character in string value is more than then continues the search with the right child node.

iii) If equal, the search continued with the middle node and move to the next character in the string value.

The image below illustrates a ternary search tree with the strings "as", "at", "cup", "cute", "he", "i" and "us":

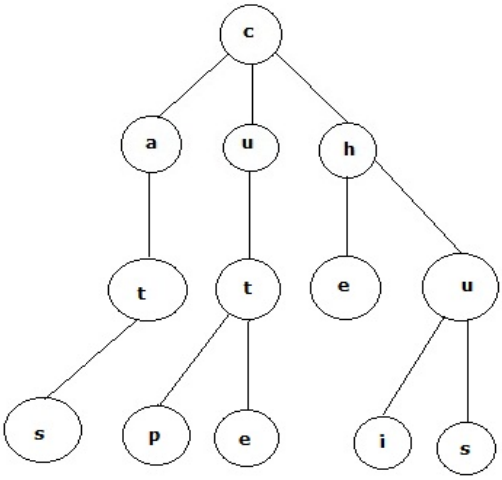

Fig. 2. Ternary Search Tree stores "as", "at", "cup", "cute", "he", "i" and "us"

As with the structure of another ternary data, each node in a ternary search tree represents a prefix of the string is stored. All the strings are in the middle of a tree that begin with that prefix.

Unlike a binary search tree, a ternary search tree can be balanced or unbalanced, based on the order of strings are added to the tree. The search for a string length m in a balanced ternary search tree is to use: $[m + \log_2(n)]$ character comparisons. In particular, each of a comparison or a character is found in the tree or the search area is divided into 2 parts.

A ternary search tree can store data in compressed format to save storage space in which the redundant nodes are removed.

4.2 Improve Ternary Search Tree to Store the Predicates (Content - Based Addresses)

For data of string type has more data comparisons, use the modified Ternary Search Tree structure (Ternary Search Tree: TST).

The end node of string value in TST has a pointer to the dynamic list of pointers, each pointer points to the dynamic list of operators, which are described in the diagram below:

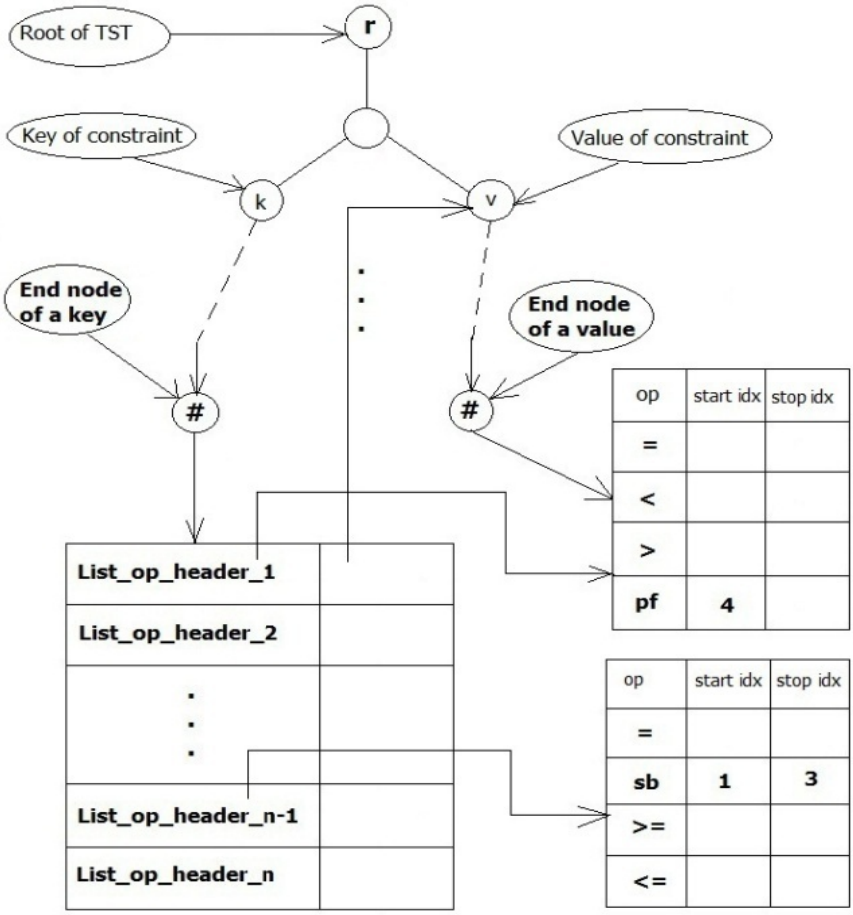

Fig. 3. Improved Ternary Search Tree stores predicates in Routing Table

Assuming the value domain of each key of the constraints is independent.

In case of a key is associated with multiple constraints will be organized into groups. In which the operators associated with a given value are put into a dynamic array, a pointer to the first element is List_op_header_i. The pointers that consist of List_op_header_1, List_op_header_2, …, List_op_header_n are stored in a dynamic array.

The set of the operators that are belonged to a group of constraints related to one key by the way: (denote a constraint by: (key, op, value)).

Archive list of operators associated with a value in a dynamic array:

With each list of the operators, maintain a pointer to the head of this list: List_op_header.

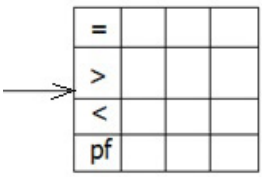

Fig. 4. Dynamic array of the operators of a group by a pair of a value and a key

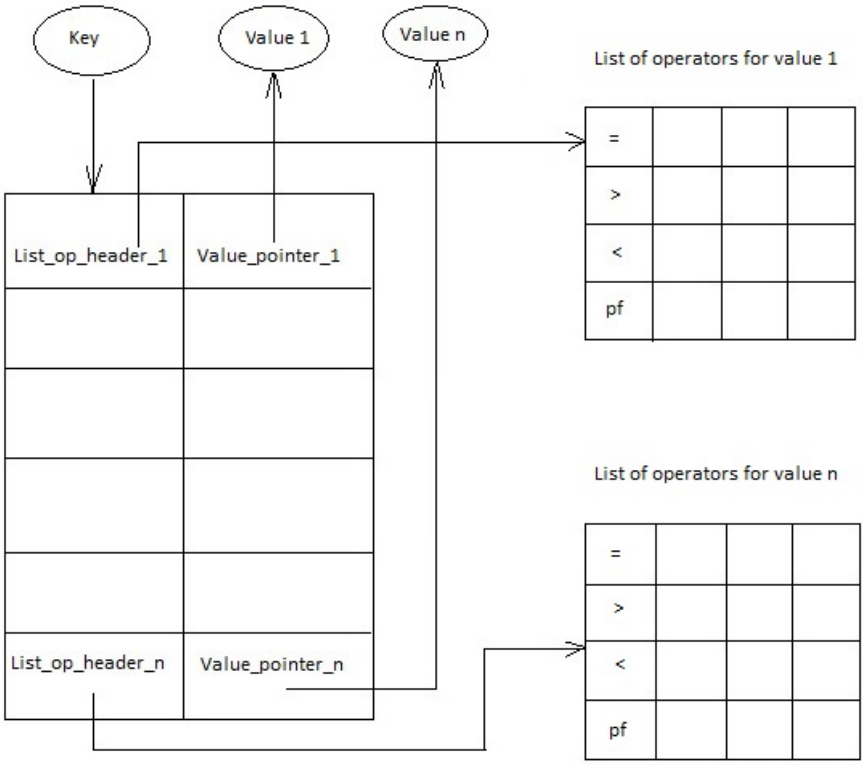

Fig. 5. Linked list of operators of a group of constraints have a same key

A set of pointers that point to head of the lists are put into a dynamic array.

The end node of a key in the ternary search tree will store a pointer that point to this dynamic array.

The dynamic array stores a list of pointers, each pointer points to the first element of the dynamic list of corresponding operators of the constraints of a value.

The algorithm checks existence and adds a constraint to a Ternary Search Tree, the constraint has the form of (**key, op, value**) has the following steps:

1, Check whether the key exists in the ternary search tree or not.

2, If it doesn't already exist, add the key in the ternary search tree. We create a list of pointers (List_of_operator) that has one pointer List_op_header. The List_op_header point to the list of storage components for storing the constraint operators (at first having op). The leaf node in the Ternary Search Tree is belonged to a part of Tree stores key that will store a pointer to List_of_operator.

3, Check if the value part of this constraint exists in the Tree.

4, If it doesn't already exist: add this value to the tree with Value_pointer_i points to the starting node of this value in the tree. Create a dynamic list, at first with only one item for storing the operator (op), with the pointer List_op_header_i points to this list. Two components (List_op_header_i, Value_pointer_i) belong to an item Key_group_i. Add Key_group_i to the list of pointers.

5, If exists, get the pointer to the list of operators corresponding value. Update this list of operators so that this list contains the operator op of the current constraint.

6, Check if the operator op has been stored in the dynamic list pointed by List_op_header_i.

If it doesn't exist, add an item to store op at the end of the list.

4.3 The Algorithm for Checking Existence and Adding a Constraint to Ternary Search Tree

Assume a constraint is denoted by **{key, op, value}:**

1. Check ∃ key in TST.
2. If ∄ key in TST then

a. TST = TST ∪ {key};

b. Create operator list: List_of_operator with only one item: List_of_operator_item={List_op_header, Pointer_to_node_value}. In which List_op_header points to the operator list with one element op, Pointer_to_Node_value points to entrance node in TST belongs to a part of TST that stores value of this constraint. We have to set a pointer in leaf node of TST belongs to a part of TST that stores the key which points to List_of_operator.

3. Check ∃ value in TST:
4. If ∄ value in TST then:

a. TST = TST ∪ {value}

b. Create dynamic list with one element op, and set List_op_headerpoints to this list.

5. If value in TST then:

a. Set: List_op_header points to the list of operators of value.

b. Check ∃ op in the list of operators is pointed by List_op_header. If ∄ then add new element stores op and relative information to this list.

4.4 The Algorithm for Finding All Nodes That Have Suitable Subscriptions with a Content Message

1) Build TST tree from all constraints in routing table if this tree doesn't exist.

2) Use SBR_Couting algorithm to find all constraints are suitable with every property of content message and finally get list of nodes to transfer content message to depending on addresses (predicates) of content based routing table:

a) For each key that exists in current content message: find List_of_operator which is pointed by a pointer of leaf node in TST which belongs to a part of TST that stores key of current property as indicated in above algorithm. For each item in List_of_operator, get List_op_headerelement from List_of_operator_itemthat is the pointer to list of operators relating to the value of current property. Associate this list of operators and other information such as key and value of constraints, replace key with the value from content message. Evaluate the given expressions, We can get all constraints that accommodate current property.

```
public List<cContraint> find_constraint(_message msg)
{
    List<cContraint> kq = new List<cContraint>();
    foreach (_property pt in msg.msgcontent)
    {
        op_1 = Contains(pt.ten);
        if (op_1 != null)
        {
                kq.AddRange(cCommon.find_constraint_match(pt,
op_1.List_op_header, msg.service_name));
         }
    }
    return kq;
}
```

b) Check condition 1: For each constraint found We check if all constraints of the filter accommodate the content message. If this condition is true, add this filter to result list simultaneously add address of node that emit the filter to list of nodes to transfer content message to.

c) Check condition 2: if list of nodes contain all nodes of the network or loop all executions of operations then stop the algorithm.

5 How To Use Binary Search to Store and Search Constraints with Values of Number Type

5.1 How to Implement

Each constraint is a set of 3 components: {key, operator, value}, the set of constraints are divided into 2 types of constraints are: number or string depend on the type of component value.

With type of number:

Step 1: key-based indexing;

Step 2: set up sub-index based on values of each key.

For example, to index the values of the constraints that all have key is name: {name, <, k_1}, {name, <, k_2}, ..., {name, <, k_n}, we sort array of values {k_1, k_2, ..., k_n} in increasing order. Then find all the relevant constraints by the modified binary search to find out range [k_i, k_{i+1}] which attribute value x belongs to. So all the constraints have value from the k_i value backwards satisfied.

5.2 Algorithm Diagram

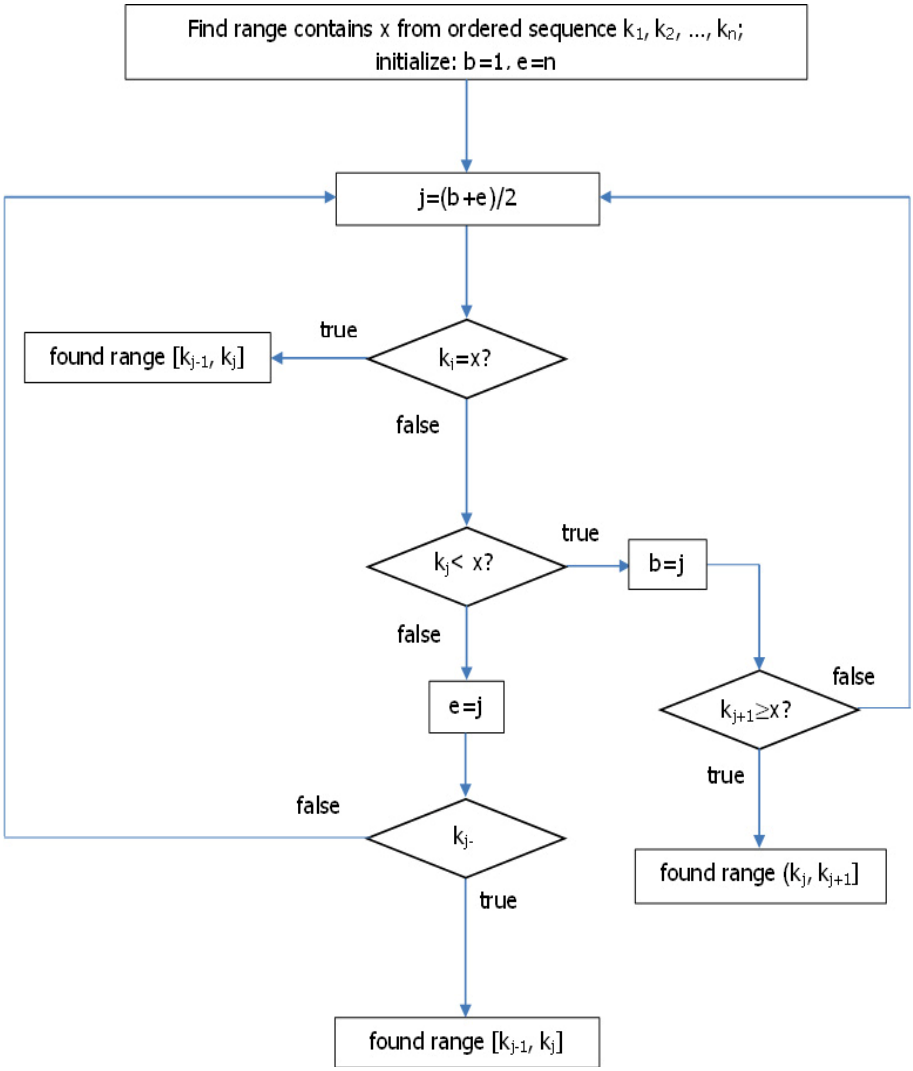

Fig. 6. Algorithm diagram finds out constraints satisfy an attribute of a content message with type of its value is number

6 Simulate the Techniques Presented

Write an application in C# on .NET 4.0 to implement the following tasks:

1) Create subscription message.

2) Create content message.

3) Transmit subscription message to the network using the broadcast method of transmission.

4) Transmit content messages to the network using broadcast method of transmission.

5) When a router receives subscription messages will update the forwarding table.

6) Making ternary search tree based on the transition table for constraints have string type values.

7) Make a list of binary search for constraints have number type values.

8) When a router receives a content message will look in the ternary search trees and binary search list to get a list of constraints have matched this content message.

9) We measure time to find routers by SBR_Couting Algorithm that accommodate a content message when it is happened on network. Transmit to each router in the list using unicast transmission or optimized transmission method. With the amount of predicates change over time We can draw a diagram that represent all results of the test. Following is the diagram that show that results are rather change evenly.

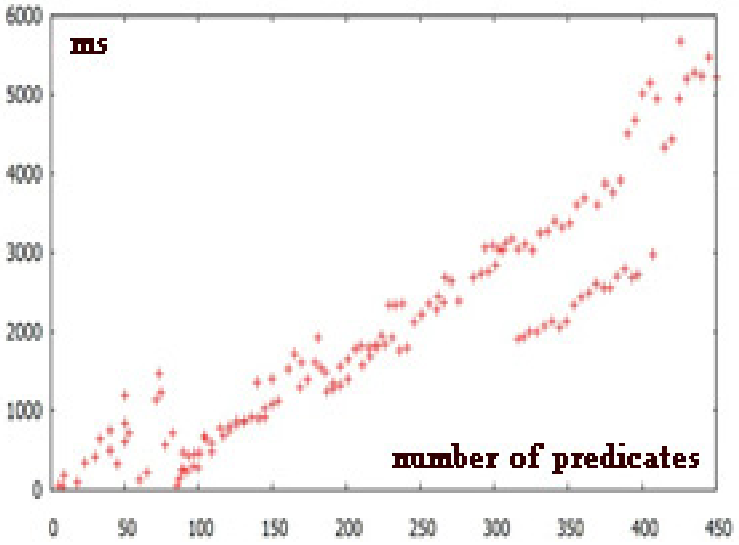

Fig. 7. CBR_Couting Algorithm result diagram

7 Conclusions and Further Research

Ternary search tree has been used in many fields especially in computer science for searching quickly and saving memory. For example, in dictionary searching, because

it can be used for approximately search. In one searching time, it can get all items in TST tree that have approximate values with condition value. In association with dynamic arrays and pointers to improve standard TST tree for storing and searching content based addresses in service oriented routing effectively and reliably. Because of using dynamic data structures, so the memory resources are easily to allocate and suppress so that according with computation in MANET. According to simulation results, the time is increased non-linearly when the number of predicates increases.

The use of ternary search trees and binary search list is very effective in the storage, making quick and accurate search.

1) Ternary search tree has the computational complexity is **O[l*(log2(N) + l)] + | result |**, where l is the length of input data, N is the number of strings in TST, result is the result set returned.

2) Binary search on a list of numbers has computational complexity is **O[log2(N)*log2(K)]**, where N is the number of attributes (key), K is the number of corresponding values of the attribute (value).

In near future will research and apply some techniques such as technique to find route that satisfy some quality of service requirements. Search techniques to organize MANET network for SBR routing effectively and safely.

Search a number of other techniques for storing and looking up predicates, such as using hash function. For example, use hash function for hashing all predicates in Routing Table before sending it to other routers.

References

[1] Cao, F., Singh, J.P.: Efficient Event Routing in Content-based Publish-Subscribe Service Networks. In: Proc. IEEE Infocom (2004)

[2] Kalaiarasi, R., Sara, G.S., Pari, S.N., Sridharan, D.: Performance Analysis of Contention Window Cheating Misbehaviors in Mobile Ad Hoc Networks. International Journal of Computer Science & Information Technology (IJCSIT) 2(5) (October 2010)

[3] Carzaniga, A., Wolf, A.: Forwarding in a Content-Based Network. In: Proc. SIGCOMM (2003)

[4] Carzaniga, A., Rutherford, M.J., Wolf, A.L.: A Routing Scheme for Content-Based Networking. In: Proc. IEEE Infocom 2004 (2004)

[5] Li, J.: Time Slot Assignment for Maximum Bandwidth in a Mobile Ad Hoc Network. Journal of Communications 2(6) (November 2007)

[6] Wu, H., Jia, X.: QoS multicast routing by using multiple paths / trees in wireless ad hoc networks, Research supported by a grant FFCSA 2006. Elsevier BV (2006)

[7] Long, N.T., Thuy, N.D., Hoang, P.H.: Research on Innovating, Evaluating and Applying Multicast Routing Technique for Routing messages in Service-oriented Routing. In: Vinh, P.C., Hung, N.M., Tung, N.T., Suzuki, J. (eds.) ICCASA 2012. LNICST, vol. 109, pp. 212–228. Springer, Heidelberg (2013)