

CLAIMS

1. A query enhancement system for constructing an elastic field based on time delay, wherein comprising: network nodes are subjected to logical division to obtain a group of containers composed of a plurality of containers, thereof the plurality of containers have a nested relationship, that is, the plurality of containers constitute a group of containers, and each container further comprises: a management node used for node organization, neighbor maintenance and query service in the container;

the query enhancement system for constructing an elastic field based on time delay specifically comprises: based on a time delay index set $\{T_i\}$, network nodes are logically divided, thereof each time delay index T_i corresponds to a layer of C_i level container with no intersection and full coverage, the C_i level containers have a nested relationship therebetween, that is, multiple C_i level containers constitute a C_{i+1} level container, thereof $T_i < T_{i+1}$, $1 \leq i \leq I-1$, I is the number of layers of nested containers, and T_i is an actual query time delay index;

a management topology of a query enhancement system for constructing an elastic field based on time delay is a tree structure, and a root node is a management node of a C_1 level container; a local query node accessed by any query request is denoted as LQN_j , and forms an elastic field region together with a C_i level container containing a local query node LQN_j and neighbors of the container, so as to satisfy requirements of a query that the time delay index $\{T_i\}$ is smaller than any given time delay upper limit T_s ; specially,

a) when $T_s < T_1$, $i=1$, querying in an LQN_j receiving a query request, a C_1 level container containing LQN_j , and neighbors of the container;

b) when $T_i \leq T_s < T_{i+1}$, querying in a C_i level container containing LQN_j and neighbors of the container; and

c) when $T_s \geq T_I$ and $i=I$, querying in a C_I level container containing LQN_j .

2. A query enhancement method based on the query enhancement system for constructing an elastic field based on time delay of a claim 1, comprising: network nodes are subjected to logical division with no intersection and full coverage on each layer, to obtain a group of containers composed of a plurality of containers, the plurality of containers have a nested relationship, and respectively perform a query flow without given low time delay requirements and a nearby query flow with given low time delay requirements;

the query enhancement system for constructing an elastic field based on time delay specifically comprises: based on a time delay index set $\{T_i\}$, network nodes are logically divided, thereof each time delay index T_i corresponds to a layer of C_i level container with no intersection and full coverage, the C_i level containers have a nested relationship therebetween, that is, multiple C_i level containers constitute a C_{i+1} level container, thereof $T_i < T_{i+1}$, $1 \leq i \leq I-1$, I is the number of layers of nested containers, and T_i is an actual query time delay index;

a management topology of a query enhancement system for constructing an elastic field based on time delay is a tree structure, and a root node is a management node of a C_1 level container; a local query node accessed by any query request is denoted as LQN_j , and forms an elastic field region together with a C_i level container containing a local query node LQN_j and neighbors of the container, so as to satisfy requirements of a query that the time delay index $\{T_i\}$ is smaller than any given time delay upper limit T_s ; specially,

a) when $T_s < T_1$, $i=1$, querying in an LQN_j receiving a query request, a C_1 level container containing LQN_j , and neighbors of the container;

b) when $T_i \leq T_s < T_{i+1}$, querying in a C_i level container containing LQN_j and neighbors of the container; and

c) when $T_s \geq T_I$ and $i=I$, querying in a C_I level container containing LQN_j ;

thereof

the query flow without given low time delay requirements adopts existing query technologies, including direct mapping, aggregation query, Skyline query, and Top-k query, to realize query, so as to obtain a network address(es) corresponding to an identifier;

as to a query flow with given low time delay requirements, a distributed nearby query method is adopted for nearby query, such that an actual query time delay index T_i is smaller than the requirement of a certain time delay upper limit T_s .

3. The query enhancement method of claim 2, wherein as to the nearby query, the requirement that an actual query time delay index T_i is smaller than a certain time delay upper limit T_s specifically comprises:

step 1)、when $T_i < T_s$, receiving, by a certain local query node LQN_j , a query request with given low time delay requirements aiming at a certain unique entity identifier ID, thereof T_i is an actual query time delay index, and T_s is an upper limit of time delay;

step 2)、when $i=1$, judging whether $T_s < T_1$ is satisfied, if $T_s < T_1$, then going to the next step; if T_s is greater than or equal to T_1 , then skipping to step 4);

step 3)、 when $T_s < T_1$ and $i=1$, querying in an LQN_j receiving a query request, a $C1$ level container containing LQN_j and neighbors of the container; if successfully querying network address(es) corresponding to an identifier, then directly returning query results; if failing to query a network address corresponding to an identifier, then returning failure of query;

step 4)、 for any i satisfying $1 \leq i \leq I-1$, checking whether $T_i \leq T_s < T_{i+1}$ is satisfied: if satisfying $T_i \leq T_s < T_{i+1}$, then going to the next step, if not satisfying $T_i \leq T_s < T_{i+1}$, then skipping to step 7);

step 5)、 when $T_i \leq T_s < T_{i+1}$, forwarding the query request to the query node of the C_i level container containing LQN_j , and denoting the container as C_{ik} , and performing preliminary query in the C_{ik} and its neighbor list $\{C_{im}\}$; if successfully querying network address(es) corresponding to an identifier, then going to the next step; if failing to query a network address corresponding to an identifier, then returning failure of query;

step 6)、 querying in the C_{ik} and its neighbor list $\{C_{im}\}$; if successfully querying network address(es) corresponding to an identifier, then directly returning query results, if failing to query a network address corresponding to an identifier, then returning failure of query;

step 7)、 performing preliminary query in a CI level container containing LQN_j ; if successfully querying network address(es) corresponding to an identifier, then going to the next step; if failing to query a network address corresponding to an identifier, then skipping to step 9);

step 8)、 querying in a CI level container containing LQN_j ; if successfully querying network address(es) corresponding to an identifier, then directly returning query results; if failing to query a network address corresponding to an identifier, then going to the next step;

step 9)、 querying a neighbor list $\{C_{Im}\}$ of a CI level container containing LQN_j through inter-domain exchange; if successfully querying network address(es) corresponding to an identifier in a certain CI level container, then denoting the container as C_{Ip} , and going to the next step; if failing to query a network address corresponding to an identifier, then returning failure of query; and

step 10)、 forwarding a query request to C_{Ip} , and querying in the C_{Ip} ; if successfully querying network address(es) corresponding to an identifier, then directly returning query results; if failing to query a network address corresponding to an identifier, then returning failure of query.

4. The query enhancement method for constructing an elastic field based on time delay of claim 2, wherein in step 3), query is performed in an LQN_j receiving a query request, a C1 level container containing LQN_j, and local hash table HT in neighbors.

5. The query enhancement method for constructing an elastic field based on time delay of claim 2, wherein in step 5), query is performed in a bloom filter BF in the C_{ik} and its neighbor list {C_{im}}.

6. The query enhancement method for constructing an elastic field based on time delay of claim 2, wherein in step 6), query is performed in a distributed hash table DHT in the C_{ik} and its neighbor list {C_{im}}.

7. The query enhancement method for constructing an elastic field based on time delay of claim 2, wherein in step 7), preliminary query is performed in a bloom filter BF in a C1 level container containing LQN_j.

8. The query enhancement method for constructing an elastic field based on time delay of claim 2, wherein in step 8), query is performed in a distributed hash table DHT in a C1 level container containing LQN_j.

9. The query enhancement method for constructing an elastic field based on time delay of claim 2, wherein in step 9), query is performed in a bloom filter BF in a neighbor list {C_{im}} of a C1 level container containing LQN_j.

10. The query enhancement method for constructing an elastic field based on time delay of claim 2, wherein in step 10), query is performed in a distributed hash table DHT in the C_{ip}.

11. The query enhancement method for constructing an elastic field based on time delay of claim 2, wherein as to a nearby query flow with given low time delay requirements, a neighbor list {C_{1m}} of C_{1k} can be degraded into a C1 level container without neighbors, and a specific query flow comprises:

step 1)、when $T_i < T_s$, receiving, by a certain local query node LQN_j, a query request with given low time delay requirements aiming at a certain unique entity identifier ID, thereof T_i is an actual query time delay index, and T_s is an upper limit of time delay;

step 2)、when $i=1$, judging whether $T_s < T_1$ is satisfied, if $T_s < T_1$, then going to the next step; if T_s is greater than or equal to T_1 , then skipping to step 4);

step 3)、when $T_s < T_1$ and $i=1$, querying in an LQN_j receiving a query request, a C1 level container containing LQN_j and neighbors of the container; if successfully querying network address(es)

corresponding to an identifier, then directly returning query results; if failing to query a network address corresponding to an identifier, then returning failure of query;

step 4)、for any i satisfying $1 \leq i \leq I-1$, checking whether $T_i \leq T_s < T_{i+1}$ is satisfied: if satisfying $T_i \leq T_s < T_{i+1}$, then going to the next step, if not satisfying $T_i \leq T_s < T_{i+1}$, then skipping to step 7);

step 5)、when $T_i \leq T_s < T_{i+1}$, forwarding a query request to a query node of a C_i level container which contains LQN_j and is denoted as C_{ik} , and performing preliminary query in the C_{ik} and its neighbor list $\{C_{im}\}$; if successfully querying network address(es) corresponding to an identifier, then going to the next step; if failing to query a network address corresponding to an identifier, then returning failure of query; preferably, query is performed in a bloom filter BF in the C_{ik} and its neighbor list $\{C_{im}\}$;

step 6)、querying in the C_{ik} and its neighbor list $\{C_{im}\}$; if successfully querying a network address(es) corresponding to an identifier, then directly returning query results, if failing to query a network address corresponding to an identifier, then returning failure of query; preferably, query is performed in a distributed hash table DHT in the C_{ik} and its neighbor list $\{C_{im}\}$;

step 7)、performing preliminary query in a CI level container containing LQN_j ; if successfully querying network address(es) corresponding to an identifier, then going to the next step; if failing to query a network address corresponding to an identifier, then skipping to step 9); preferably, in step 7), preliminary query is performed in a bloom filter BF in a CI level container containing LQN_j ; and

step 8)、querying in a CI level container containing LQN_j ; if successfully querying network address(es) corresponding to an identifier, then directly returning query results; if failing to query a network address corresponding to an identifier, then going to the next step; preferably, in step 8), query is performed in a distributed hash table DHT in a CI level container containing LQN_j .