Available online at www.sciencedirect.com

SCIENCE @ DIRECT®

Computer Standards & Interfaces 27 (2005) 285–301

COMPUTER STANDARDS & INTERFACES

www.elsevier.com/locate/csi

Constructing agents blackboard communication architecture based on graph theory

Y.C. Jiang*, P. Yi, S.Y. Zhang, Y.P. Zhong

Department of Computing and Information Technology, Fudan University, Shanghai 200433, P.R. China

Received 23 April 2004; received in revised form 22 August 2004; accepted 3 September 2004
Available online 2 October 2004

Abstract

In the agent blackboard communication architecture, agents do not interact with each other directly but through blackboard. The blackboard architecture includes central blackboard architecture and distributed one. In blackboard communication architecture, the location of central blackboard (or distributed sub-blackboards) and communication topology among sub-blackboards are two important issues that can influence the agent communication performance very much. However, there are few works about such issues; and in the existing agent systems, the central blackboard (or distributed sub-blackboards) is (or are) usually randomly located in the underlying network. To solve such problem, this paper presents a model for constructing agent blackboard communication architecture based on graph theory. The model computes the location of central blackboard or sub-blackboards based on *median location method*, and computes the communication topology among sub-blackboards based on *Steiner Tree method*; the model also applies graph theory to the construction of blackboard architecture's adaptation mechanism for dynamic topology and the realization of the blackboard architecture's fault-tolerance ability. At last, several case studies and simulation experiments are conducted, which prove that the presented model can construct the effective agent blackboard communication architecture.
© 2004 Elsevier B.V. All rights reserved.

Keywords: Multi agents; Agents interaction; Agents communication; Blackboard architecture; Graph theory; Network topology

1. Introduction

In multi-agent systems, interaction will enable agents to solve the problems that cannot be solved by individual one. To implement interaction among agents, there is a significant demand for agents to communicate with each other effectively. Nowadays, *blackboard communication architecture* [1,2] is one of the commonly used communication architectures.

In the blackboard communication architecture, agents do not interact with each other directly; and information is made available to all agents in the system through a common information space and there is no direct communication between agents. Obviously, the message overheads and implementa-

* Corresponding author. Tel.: +86 21 65643235; fax: +86 21 65647894.
E-mail address: jiangyichuan@yahoo.com.cn (Y.C. Jiang).

0920-5489/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.csi.2004.09.003

tion complexity of blackboard architecture are relative low. Blackboard communication architecture can be well suited for dynamic and large agent systems [4].

Blackboard communication architecture includes central blackboard architecture and distributed one, as shown in Fig. 1. Central blackboard architecture is simple. However, in this architecture, the blackboard is subject to become the "performance bottleneck" of agent system. A popular way of enhancing communication architecture is to implement distributed blackboard architecture, in which some sub-blackboards are set in the system and each sub-blackboard takes charge of the communications of some agents [3]. Here agents are organized into some federated systems where agents do not communicate directly with each other but through their respective sub-blackboards. The agents in a federated system surrender their communication autonomy to the sub-blackboard and the sub-blackboard takes full responsibility for their needs. Fig. 1 shows a simple federated multi-agents system in which there are three multi-agent sub-systems (i.e. federated systems) with agents in each sub-system controlled by a sub-blackboard. The sub-blackboards communicate among themselves to express the needs of their respective agents.

In the agent blackboard architecture, the key is the communication cost that is mainly influenced by the communication distance from the agents to the blackboard and the communication distance among sub-blackboards. Therefore, the location of blackboard (or sub-blackboards) and the communication path among sub-blackboards should be attached much importance so as to minimize the sum of all communication distances. However, there are few researches on such issue; and central blackboard or sub-blackboards are always located randomly, or are located on some management nodes. Among the relative works on agent blackboard architecture [1,2], they often focus on the information sharing and information communication of blackboard.

To solve the above problems, based on our original work [5], this paper presents a comprehensive model for constructing agent blackboard communication architecture based on graph theory. According to the current underlying network topology, our model can compute the location of central blackboard or sub-blackboards based on *median location method*, and compute the communication topology among sub-blackboards based on *Steiner tree method*. We also construct the adaptation mechanism for dynamic topology and the fault-tolerance mechanism of blackboard architecture.

Our constructed architecture can perform better than the architecture that blackboard or sub-blackboards are located randomly; the constructed architecture can also adapt for dynamic topology and have fault-tolerance ability, which are testified by our simulation experiments.

The rest of the paper is organized as follows. Section 2 addresses how to locate central blackboard based on graph theory. Section 3 addresses how to

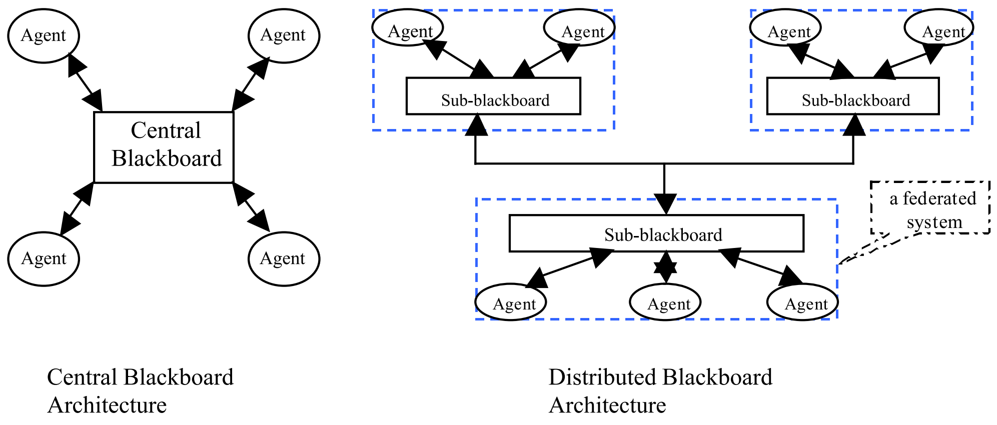


Fig. 1. Blackboard communication architecture.

construct distributed blackboard communication architecture based on graph theory. Section 4 presents the agent blackboard architecture's adaptation mechanism for dynamic topology. Section 5 describes the fault-tolerance and self-healing of blackboard architecture. Section 6 gives the case studies and simulation experiments. Then the conclusions are summarized in Section 7.

2. Location of central blackboard

In central blackboard communication architecture, the key is how to construct the central blackboard effectively. It is required to locate the central blackboard in such a way so as to minimize the sum of all shortest distances from the blackboard to the nodes of the underlying network.

For a given network $G=(X, E)$, where X denotes the nodes and E denotes the links among nodes, we define a *communication sum number* for every node $x_i \in X$, as follows:

$$\sigma(x_i) = \sum_{x_j \in X} d(x_i, x_j) \tag{1}$$

where $d(x_i, x_j)$ is the shortest distance from node x_i to x_j.

Minimizing the sum of total communication cost, the central blackboard location should be the node x_o of the network G that:

$$\sigma(\overline{x_o}) = \min_{x_i \in X} [\sigma(x_i)] \tag{2}$$

Let Fig. 2 be an underlying network topology of agent system, we can compute the distances among nodes, shown as the distance matrix in Fig. 3. In Fig. 3, n_5 is the node with the minimum transmission number; therefore, central backboard should be located on n_5.

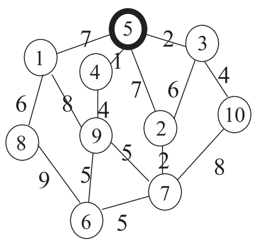


Fig. 2. A network example.

	n_1	n_2	n_3	n_4	n_5	n_6	n_7	n_8	n_9	n_{10}	$\sigma(n_i)$
n_1	0	14	9	8	7	13	13	6	8	13	91
n_2	14	0	6	8	7	7	2	16	7	10	77
n_3	9	6	0	3	2	12	8	15	7	4	66
n_4	8	8	3	0	1	9	9	14	4	7	63
n_5	7	7	2	1	0	10	9	13	5	6	60*
n_6	13	7	12	9	10	0	5	9	5	13	83
n_7	13	2	8	9	9	5	0	14	5	8	73
n_8	6	16	15	14	13	9	14	0	14	19	120
n_9	8	7	7	4	5	5	5	14	0	11	66
n_{10}	13	10	4	7	6	13	8	19	11	0	91

Fig. 3. The distance matrix of Fig 2.

The algorithm that computes the central blackboard is shown as Algorithm 1.

Algorithm 1**.** Locate the central blackboard

```
Main ( )
{
for (int $i$=1; $i \le n$; $i$++)
  for (int $j$=1; $j \le n$; $j$++)
    $d(i, j)$=0;
for ($i$=1; $i \le n$; $i$++)
  for (int $j=i$; $j \le n$; $j$++)
    $d(i, j)$=the shortest distance between nodes $i$ and $j$;
for ($i$=1; $i \le n$; $i$++)
        {$\sigma(n_i)$=0;
    for ($j$=1; $j \le n$; $j$++)
    {if $d(i, j)$<>0
      $\sigma(n_i) = \sigma(n_i) + d(i, j)$;
    else $\sigma(n_i)$=$\sigma(n_i)$+$d(j, i)$;
    }
   }
 $\sigma_o$ = max number; center = 0;
 for ($i$=1; $i \le n$; $i$++)
 if $\sigma(n_i) < \sigma_o$
 {$\sigma_o$=$\sigma(n_i)$; center=$i$}
 output (center).
 }
```

Obviously, the time complexity of Algorithm 1 is $O(n^2)$.

Therefore, after locating the central blackboard, all agents can communicate through the central blackboard. Agents do not interact with each other directly but with the blackboard.

3. Distributed blackboard communication architecture construction

In distributed blackboard communication architecture, the key is how to construct the distributed sub-blackboards effectively. It is required to fix the sub-blackboards number and locate the sub-blackboards in such a way so as to minimize the total communication cost between sub-blackboards and their allocated agent. The communication cost is a function of the distance between sub-blackboards and their allocated agents, so the number and location of sub-blackboards should minimize the total communication distances.

Otherwise, the communication paths among sub-blackboards are also critical for efficient communication of agent system. Therefore, it is also required to compute the communication topology among sub-blackboards so as to minimize the communication distance sum.

Now we compute the number and location of sub-blackboards on the base of *multi-medians location method* [6], and compute the communication topology among sub-blackboards on the base of *Steiner tree* [14].

3.1. Formal description of the problem

Problem of finding the "best" location of facilities in network abound in practical situations. The problem can be separated by two situations [6]: (a) In some location problems, the objective is to minimize the largest travel distance to any vertex from its nearest facility, are, obvious reasons, called *minimax location problems*. The resulting locations are then called the *centers* of a graph. (b) In other location problems, however, a more appropriate objective would be to minimize the total sum of the distances from vertices of the graph to the nearest facility. Problems of this type are generally referred to as *minisum location problems*, although the objective function is often not simply the sum of distances but the sum of various functions of distance. The facility locations resulting from the solution to a minisum problem are called the *medians* of a graph. The problem of finding the p-median of a graph is the central problem in a general class studied in the literature under the name of "facility location and allocations" [8].

On the base of the multi-medians problem description in [6], now we give a formal description of our problem.

In the agent system, we should minimize the sum of communication cost from agents to the sub-blackboards. Therefore, in particular, we discuss the problem of finding the p sub-blackboards location in the underlying network G; that is the problem of locating a given number (p say) of sub-blackboards optimally so that the sum of the shortest distances that from their nearest sub-blackboard to the agent on the node of G is minimized.

Firstly, let $G=(X, E)$ be a network topology with X the set of nodes and E the set of links. Let X_p be a subset of the set X and let X_p contain p nodes. Now we write

$$d(X_p, x_j) = \min_{x_i \in X_p} [d(x_i, x_j)] \tag{3}$$

where $d(x_i, x_j)$ denotes the shortest path distance between x_i and x_j.

If X'_i is the node of X_p which produces the minimum in Eq. (3), we will say that node x_j is *allocated* to the blackboard on x'_i. The *transmission numbers* for the set X_p of nodes are defined as Eq. (4).

$$\sigma(X_p) = \sum_{x_j \in X} d(X_p, x_j) \tag{4}$$

A set $\bar{X}_{po}$ for which

$$\sigma(\bar{X}_{po}) = \min_{X_p \subseteq X} [\sigma(X_p)] \tag{5}$$

is now called the *sub-blackboards location* in the network G. The search process is called *sub-blackboards location problem*. The aim of such problem is to select p nodes to construct sub-blackboards and assign each of the other nodes to its closest sub-blackboard so as to minimize the total distance between the sub-blackboard and other nodes. Therefore, the communication cost between the sub-blackboards and agents can be minimized accordingly.

To describe the allocation relation between nodes and sub-blackboards, we presented the concept of *allocation matrix*. Let $[\xi_{ij}]$ be an allocation matrix so that:

$$\xi_{ij} = \begin{cases} 1 \text{ implies that node } x_j \text{ is allocated to sub-blackboard on } x_i \\ 0 \text{ otherwise.} \end{cases} \tag{6}$$

If node x_j is allocated to the sub-blackboard on x_i, then the agent on x_j should interact with the sub-blackboard on x_i.

Further, we will take ξ_{ij}=1 to imply that node x_i is a sub-blackboard location and ξ_{ij}=0 otherwise. Therefore, we should locate the sub-blackboards so as to minimize:

$$z = \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}\xi_{ij} \tag{7}$$

where d_{ij} denotes the shortest path distance between x_i and x_j.

Obviously, in the agent distributed blackboard communication architecture, we have:

$$\sum_{i=1}^{n} \xi_{ij} = 1 \text{ for } j = 1,\ldots,n \tag{8}$$

$$\sum_{i=1}^{n} \xi_{ii} = p \tag{9}$$

$$\xi_{ij} \leq \xi_{ii} \text{ for all } i,j = 1,\ldots,n \tag{10}$$

$$\xi_{ij} = 0 \text{ or } 1 \tag{11}$$

where $[d_{ij}]$ is assumed to be the communication distance matrix of the network. Eq. (8) ensures that any given x_j is allocated to one and only one sub-blackboard x_j. Eq. (9) ensures that there are exactly p sub-blackboard locations, and constraint (10) guarantees that ξ_{ij}=1 only if ξ_{ii}=1, i.e. allocations are made only to the sub-blackboard location nodes.

Therefore, if $[\bar{\xi}_{ij}]$ is the optimal solution to the sub-blackboards distribution, then the sub-blackboards set is:

$$\overline{X_{p_o}} = \left\{ x_i \middle| \overline{\xi_{ii}} = 1 \right\} \tag{12}$$

Therefore, the aim of agents distributed blackboard architecture construction is to find such $[\bar{\xi}_{ij}]$ so as to the $\bar{X}_{p_o}$ can satisfy Eq. (5).

3.2. Compute the sub-blackboards number and locality

3.2.1. Fix the sub-blackboards number

There are many methods to compute the number of medians in graph theory [9], which can be referred for fixing sub-blackboards number. Firstly, we can set an initial test scope of sub-blackboards number, and then we can use the well-known *binary search* method to search the optimal number.

Binary search is searching a *sorted array* by repeatedly dividing the search interval in half. Begin with an interval covering the whole array. If the value of the search *key* is less than the item in the middle of the interval, narrow the interval to the lower half. Otherwise narrow it to the upper half. Repeatedly check until the value is found or the interval is empty.

A possible improvement in binary search is not to use the middle element at each step, but to guess more precisely where the key being sought falls within the current interval of interest. This improved version is called *Fibonacci search*, which can search the optimal number more rapidly than binary search [23]. Instead of splitting the array in the middle, this implementation splits the array corresponding to the *Fibonacci numbers*. Therefore, to compute the sub-blackboards number rapidly, we can adopt the Fibonacci number to search the optimal number of sub-blackboards.

Fibonacci number is formed by starting with 0 and 1 and then *adding the latest two numbers* to get the next one, i.e. 1, 1, 2, 3, 5, 8,. . .

Let F_1=1, F_2=1, F_3=2, F_4=3,. . ., then we can conclude:

$$F_n = F_{n-1} + F_{n-2}, n \geq 2 \tag{13}$$

We can set the test scope of sub-blackboards number, and reduce the scope step by step based on Fibonacci series. The explicit steps can be seen in Algorithm 3.

3.2.2. Compute sub-blackboards location

On the base of the approximate algorithm of multi-medias problem [6,7,9], we design the sub-blackboards location-computing algorithm.

Let the sub-blackboards number is p, the method proceeds by choosing any p nodes in the network at random to form the initial sub-blackboards set S, which is assumed to be an approximation to the optimal sub-blackboard locations set $\bar{X}_p$. The method then tests if any node $x_j \in X-S$ could replace a node $x_i \in S$ as a sub-blackboard location node and so produce a new set $S' = S \cup \{x_j\} - \{x_i\}$ whose transmission number $\sigma(S')$ is less than $\sigma(S)$. If so, the substitution of node x_i by x_j is performed thus obtaining a set S' that is a better approximation to the p-location nodes set $\overline{X_p}$. The same tests are now performed on the new set S' and so on, until a set $\bar{S}$ is finally obtained for which no substitution of a vertex in $\bar{S}$ by another node in $X-\bar{S}$ produces a set with transmission less than $\sigma(\bar{S})$. This final set $\bar{S}$ is then taken to be the required approximation to $\overline{X_p}$.

Algorithm 2. Sub-blackboards Location-Computing (*int p*).

```
Step 1. Select a set $S$ of $p$ nodes to form the initial approximation to the sub-blackboards set. Call all nodes $x_j \notin S$ "untried".
Step 2. Select some "untried" node $x_j \notin S$ and for each node $x_i \in S$, compute the "reduction" $\Delta_{ij}$ in the set transmission if $x_j$ is substituted for $x_i$, i.e. compute:
        $\Delta_{ij} = \sigma(S) - \sigma(S \cup \{x_j\} - \{x_i\})$
Step 3. Find $\Delta_{i_o j} = \max_{x_i \in S} [\Delta_{ij}]$.
        I. If $\Delta_{i_o} j \le 0$ call $x_j$ "tried" and go to step 2.
        II. If $\Delta_{i_o} j > 0$ set $S \leftarrow S \cup \{x_j\} - \{x_i\}$ call $x_j$ "tried" and go to step 2.
Step 4. Repeat steps 2 and 3 until all nodes in $X-S$ have been tried. This is referred to as a cycle. If, during the last cycle no node substitution at all has been made at step 3 (i), go to step 5. Otherwise, if some node substitution has been made, call all nodes "untried" and return to step 2.
Step 5. Stop. The current set $S$ is the estimated sub-blackboards location nodes set $\overline{X_p}$.
```

3.2.3. The combined algorithm

In the construction of distributed blackboard communication architecture, the target function should include the cost of building sub-blackboards and the communication distance. Therefore, we can define the target function as follows.

$$t\ \arg\ et = \min(\rho_1 p \times B_{\text{cost}} + \rho_2 \sigma(S)) \qquad (14)$$

where B_{cost} denotes the cost of building a sub-blackboard, p denotes the number of sub-blackboards, ρ_1 and ρ_2 denote the weight.

By combining the method of Sections 3.2.1 and 3.2.2, now we design the algorithm for computing sub-blackboards number and locations, shown as Algorithm 3.

Algorithm 3. Computing the sub-blackboards number and location

```
Step 1. Let $a$=1, $b$=$n$, where $n$ is the underlying network nodes number of the agent system.
Step 2. Set the sub-blackboards number test scope as $[a, b]$.
Step 3. Let $F_k$ is the least Fibonacci number that $\ge b-a$. Select $P_1$ and $P_2$ as the test sub-blackboards, where $P_1 = a + F_{k-2}$, $P_2 = a + F_{k-1}$.
Step 4. Sub-blackboards Location-Computing ($P_1$); /*the 1st scenario */
        Sub-blackboards Location-Computing ($P_2$). /*the 2nd scenario */
Step 5. Compute the results of the target functions of the two scenarios, respectively.
Step 6. If the result of *the 1st scenario* is more optimal than the 2nd one, then $b$=$P_2$; Else $a$=$P_1$.
Step 7. If there exists any test numbers between $a$ and $b$, then go to step 2.
Step 8. Output the approximate optimal sub-blackboards number and locations.
```

After the number and location of sub-blackboards are fixed, then each node can select the nearest blackboard to form a federated system; then the agent on a node should interact with the nearest blackboard. The sub-blackboards communicate among themselves to explain the agents' need. Therefore, the communication topology among sub-blackboards is important.

3.3. Compute the communication topology among sub-blackboards

3.3.1. Steiner tree

The Steiner tree problem is one of the fundamental topological network design problems. The problem is to interconnect (a subset of) the nodes such that there is a path between every pair of nodes while minimizing the total cost of selected edges [10–12].

A minimum Steiner tree is defined to be the minimal cost sub-graph spanning a given set of nodes in the graph [13,14]. Formally, it can be formulated as follows: Given a weighted, undirected graph $G=(V, E, w)$, V denotes the set of nodes in the graph and E is the set of edges (or links). Let w: $E\rightarrow R$ be a positive edge weight function, and designate a non-empty set of terminal nodes M, where $\phi \subset M \subset V$. The nodes that belong to the complementary subset $\bar{M}$, where $\bar{M}=V-M$, are called non-terminals. A Steiner tree for M in G is a tree that meets all nodes in M. The MST problem is to find a Steiner tree of minimum total edge cost. The solution to this problem is a *minimum Steiner tree T*. Non-terminal nodes that end up in a minimum Steiner tree T are called *Steiner Nodes*.

Among various sub-blackboard communication topologies, one basic question is how to achieve the connectivity with lest communication cost. Since in our agent system, the communication cost is mainly influenced by communication distance among nodes, we should compute the communication topology among sub-blackboards with the least total communication distances. Note that a communication topology of sub-blackboards may contain some nodes that are not sub-blackboard. These nodes are referred to as *forwarding nodes*. Therefore, we can apply Steiner tree method in the topology computation.

3.3.2. Compute the communication topology among sub-blackboards

Based on the KMB algorithm [14,15], now we compute the communication topology among sub-blackboards. Given a weighted undirected graph $G=(V, E, w)$ which denotes the underlying network topology, and a set of sub-blackboard nodes $M\subseteq V$, consider the complete undirected graph $G'=(V', E', w')$ constructed from G and M in such a way that $V'=M$, and for every edge $(i,j)\in E'$, weight $w(i,j)$ is set equal to the weight sum of the shortest path from node i to node j in graph G. For each edge in G', there corresponds a shortest path in G. Given any spanning tree in G', we can construct a subgraph G by replacing each edge in the tree by its corresponding shortest path in G.

Fig. 4 shows a network G and sub-blackboard set $M=\{1, 3, 9, 7\}$ (shaded nodes). We first calculate the shortest distance between every two sub-blackboards in G. They are 8, 9, 13, 7, 8, 5 respective to $a\langle 1, 9\rangle$, $b\langle 1, 3\rangle$, $c\langle 1, 7\rangle$, $d\langle 3, 9\rangle$, $e\langle 3, 7\rangle$, $f\langle 7, 9\rangle$. Let a, b, c, d, e, f form a graph G', shown as Fig. 4(b). The minimum spanning tree of G' is shown with thick lines in Fig. 4(b), and then we construct the communication topology among sub-blackboards by replacing each edge in the tree by its corresponding shortest path in the network G. The communication topology among sub-blackboards is shown as thick lines in Fig. 4(c).

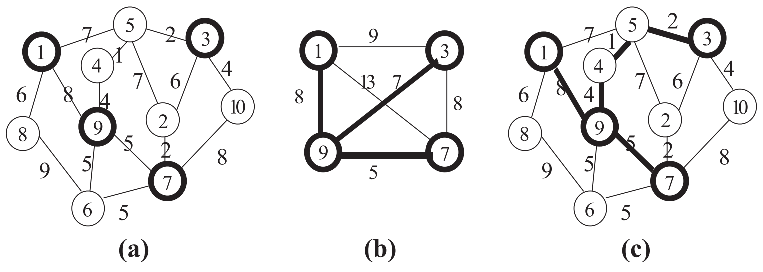


Fig. 4. An example of a Steiner tree construction in network.

4. Adaptation mechanism for dynamic topology

Nowadays, the underlying network topology is always dynamic, old links may fail and new links may be formed over time [16].

In static networks, the blackboards location may be pre-computed and keep static afterwards. However, in dynamic topology network, this may not be optimal since the topology often changes. Therefore, dynamic selection of blackboard locations is important for good performance.

In agent blackboard construction, the key issue is the location of blackboard (or sub-blackboards). Therefore, we should re-select the blackboard (or sub-blackboards) location while network topology changes.

About the blackboard location re-selection, we can consider the problem under two conditions: (a) If the topology changes *locally* only within a federated system, then we only re-select the sub-blackboard location within the underlying network of the federated system. (b) If the topology changes *globally* in the whole underlying network, we should re-select all of the sub-blackboard locations.

Selecting blackboard location each time the network topology changes in the network is expensive in terms of overhead and resource utilization. Therefore, we do not re-select blackboard location each time the network topology changes, and we do it only when the network topology changes more than a certain degree.

To describe how much the network topology changes, we present the concept of *topology variation degree*.

Let G=(V, E) be the network topology before changing, and G' =(V' , E') be the network topology after changing, and N=|V∪V' |.

Let P=|p_{ij}|, where 1≤i,j≤N, and

$$p_{ij} = \begin{cases} 1 & \text{if there exists a link } (v_i, v_j) \text{ in } G \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

Let P' =|p'_{ij}|, where 1≤i,j≤N, and

$$p'_{ij} = \begin{cases} 1 & \text{if there exists a link } (v_i, v_j) \text{ in } G' \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

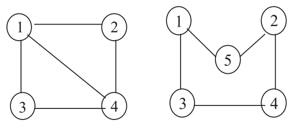
Let D be the difference sum between P and P'. Firstly D=0; and we compare each element of P and P' , if $p_{ij} \neq p'_{ij}$, we add D by 1. Let m be the number of the links of the initial network topology, i.e. m=|E|. Now we define the topology variation degree as follows:

$$\text{topology variation degree} = \frac{D}{2m} \tag{17}$$

For example, Fig. 5(a) is a network topology before changing, Fig. 5(b) is the network topology after changing; and (c) and (d) in Fig. 5 are the resulted P and P' , respectively, from (a) and (b).

By comparing (c) and (d) in Fig. 5, we can get the D, and D=8. Therefore, topology variation degree=8/(2×5)=0.8.

Therefore, when the network topology changes, we firstly compute the global topology variation degree. If the global variation degree is more than the predefined



(a) **(b)**

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \qquad \begin{pmatrix} 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

(c) **(d)**

Fig. 5. An example of changing network topology.

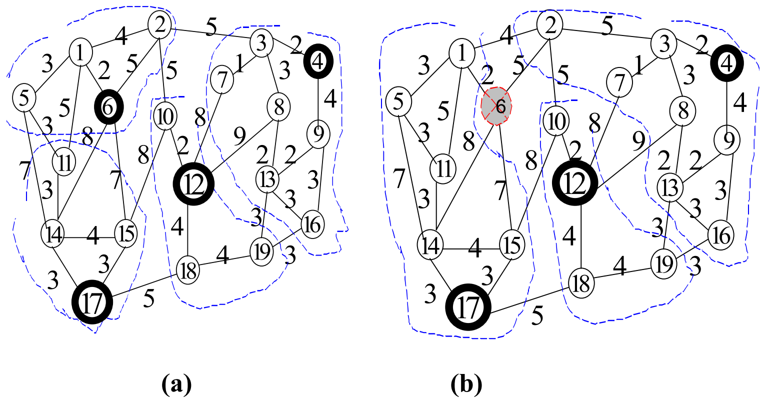


Fig. 6. Illustration of the fault-tolerance and self-healing of distributed blackboard architecture.

number, we should re-select all of the sub-blackboard locations. If the global variation degree is less than the predefined number, we should compute the topology variation degree of each federated system, and decide whether the sub-blackboard location needs to be re-selected within each federated system.

5. Fault-tolerance and self-healing of blackboard architecture

The solutions based on blackboards approach lend themselves to a single point of failure problem and bring fault-tolerance issues in networks. The blackboard (sub-blackboard) also acts as a hub for all traffic in its federated system, and therefore, if it is serving large number of groups, the congestion at the blackboard will cause unnecessary delays in the real-time traffic. Otherwise, in the dynamic topology network, the blackboard location maybe fails to work.

Therefore, we should make the system have fault-tolerance ability so that single point of failure of the blackboard (or sub-blackboard) in the agent system can be handled efficiently. So, every node (i.e. the agent on the node) is served by r_j sub-blackboards instead of just one. The sub-blackboards other than the closest one are "*backup*" sub-blackboards for that node (i.e. the agent on the node), and will be used only if the closer sub-blackboard fails.

While we select the backup sub-blackboards, the goal is to optimize the sum of the cost of sub-blackboards and the weighted sum of the communication costs of each node to the closest sub-blackboard. On the base of [17], we design the fault-tolerance mechanism of blackboard architecture.

Let $[\xi_{ij}]$ be an allocation matrix, and sub-blackboard i is the rth closest one to j. So our goal is to:

$$\text{Minimize}\left(\sum_i \sum_j \sum_r d_{ij}^{(r)} \xi_{ij}^{(r)} + \sum_i f_i\right)$$

$$\sum_i \xi_{ij}^{(r)} \geq 1 \;\; \forall j, r$$

$$\xi_{ij}^{(r)} \in \{0, 1\} \;\; \forall i, j, r \tag{18}$$

where f_i denotes the cost of the sub-blackboard i.

In our agent blackboard architecture construction, supposing the costs of all sub-blackboards are the same, so our goal can be simplified as follows:

$$\text{Minimize} \sum_i \sum_j \sum_r d_{ij}^{(r)} \xi_{ij}^{(r)} \tag{19}$$

Therefore, each node can select the $(r-1)$ closest sub-blackboards as the "backup" ones.

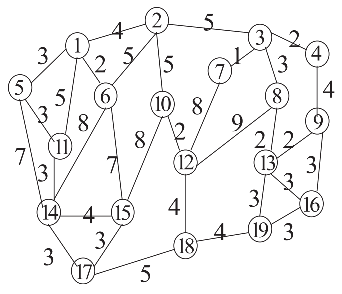


Fig. 7. Simulation network topology.

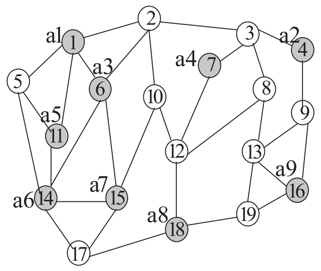


Fig. 8. Agent distribution.

For example, there is a simulated agent blackboard communication architecture, shown as Fig. 6(a). Now we illustrate the fault-tolerance and self-healing of agent sub-blackboard architecture.

Let r=2, i.e. each node can select one backup sub-blackboard, that is the 2nd closest to it. Therefore, in Fig. 6(a), the backup sub-blackboard of *N2* is *N4*, the one of *N1* is *N17*, and the one of *N5* is also *N17*.

If the sub-blackboard on *N6* fails, then *N2* should select the *N4* as its main sub-blackboard, *N1* and *N5* should select the *N17* as their sub-blackboard. Therefore, with the fault-tolerance ability, the communication architecture can be self-healed. The new federated system is shown as Fig. 6(b).

6. Case studies and simulations experiment

For the purpose of our experiment, we have developed a minimal platform that provides the basic functions required to program agents. We have implemented a prototype that is developed with Tcl/Tk, Tclx, Tix and Binprolog [18–21]. Also, the prototype was also partly based on the work of Aglets Software Development Kit v2 (Open Source release) [22].

In order to show how effectively our proposed agent blackboard communication architecture construction model can work, we made four kinds of case studies and simulation experiments: (1) case studies and test of central blackboard architecture; (2) case studies and test of distributed blackboard architecture; (3) case studies and test of the performance when network topology changes; (4) case studies and test of the fault-tolerance performance when some sub-

$$\underset{\textbf{(a)}}{\begin{pmatrix}0&1&0&1&0&1&0&0&0\\1&0&1&0&0&0&1&0&0\\0&1&0&0&0&0&0&0&0\\1&0&0&0&1&1&0&1&0\\0&1&0&1&0&0&0&0&0\\1&0&1&0&1&0&0&1&0\\0&1&0&0&0&0&0&0&0\\1&0&0&1&0&0&0&0&0\\0&0&1&0&1&0&1&0&0\end{pmatrix}}\quad\underset{\textbf{(b)}}{\begin{pmatrix}0&1&1&0&0&1&0&0&0\\0&0&1&0&1&0&1&0&0\\0&0&0&0&0&0&0&0&0\\1&0&0&0&1&1&0&1&0\\0&1&0&1&0&0&0&0&1\\1&0&1&0&1&0&0&1&1\\0&0&0&0&1&0&0&0&0\\1&1&0&0&1&0&0&0&0\\0&1&1&1&0&0&1&1&0\end{pmatrix}}\quad\underset{\textbf{(c)}}{\begin{pmatrix}0&1&0&1&0&0&0&1&1\\0&0&1&0&0&0&1&0&0\\0&0&0&0&0&0&0&0&0\\1&0&0&0&1&1&0&1&0\\0&1&0&1&0&0&0&0&1\\1&0&1&0&1&0&0&0&1\\1&1&0&0&0&1&0&0&0\\0&0&1&1&1&0&0&0&0\\0&0&1&1&1&0&1&0&0\end{pmatrix}}$$

$$\underset{\textbf{(d)}}{\begin{pmatrix}0&0&0&0&0&0&0&1&1\\0&0&1&0&0&0&1&1&1\\0&0&0&0&0&0&0&0&0\\1&0&0&0&1&1&0&1&0\\1&1&0&1&0&0&0&0&0\\1&0&1&0&1&0&0&1&0\\0&1&1&0&0&0&0&0&0\\1&0&1&1&0&1&1&0&0\\0&0&1&0&1&1&1&0&0\end{pmatrix}}\quad\underset{\textbf{(e)}}{\begin{pmatrix}0&0&0&1&0&1&0&0&0\\0&0&1&0&0&0&1&0&0\\1&1&0&0&0&0&0&0&0\\1&0&0&0&1&1&0&1&0\\0&1&0&1&0&0&1&1&1\\1&0&1&0&1&0&0&1&0\\0&1&0&0&0&0&0&0&0\\1&0&1&1&1&0&0&0&0\\0&0&0&0&0&0&1&0&0\end{pmatrix}}\quad\underset{\textbf{(f)}}{\begin{pmatrix}1&0&0&1&0&1&0&0&1\\0&0&1&0&0&0&1&0&1\\0&0&1&0&0&0&0&0&0\\1&0&0&0&1&1&0&1&0\\0&1&0&1&1&0&0&0&0\\1&1&1&0&1&1&0&1&0\\0&0&0&0&0&0&1&0&0\\0&0&0&1&0&0&0&0&0\\1&0&1&0&1&0&1&0&0\end{pmatrix}}$$

Fig. 9. Agents' communication relations.

	n_1	n_2	n_3	n_4	n_5	n_6	n_7	n_8	n_9	n_{10}	n_{11}	n_{12}	n_{13}	n_{14}	n_{15}	n_{16}	n_{17}	n_{18}	n_{19}	$\sigma(n_i)$
n_1	0	4	9	11	3	2	10	12	15	9	5	11	14	8	9	17	11	15	17	182
n_2	4	0	5	7	7	5	6	8	11	5	9	7	10	12	12	13	15	11	13	160*
n_3	9	5	0	2	12	10	1	3	6	10	14	9	5	17	17	8	17	12	8	165
n_4	11	7	2	0	14	12	3	5	4	12	16	11	6	20	19	7	18	13	9	189
n_5	3	7	12	14	0	5	13	15	18	12	3	14	17	6	10	20	9	14	18	210
n_6	2	5	10	12	5	0	11	13	16	10	7	12	15	8	7	18	10	15	18	194
n_7	10	6	1	3	13	11	0	4	7	10	15	8	6	18	18	9	17	12	9	177
n_8	12	8	3	5	15	13	4	0	4	11	17	9	2	17	17	5	14	9	5	170
n_9	15	11	6	4	18	16	7	4	0	16	20	13	2	17	17	3	14	9	5	197
n_{10}	9	5	10	12	12	10	10	11	16	0	14	2	13	12	8	13	11	6	10	184
n_{11}	5	9	14	16	3	7	15	17	20	14	0	15	18	3	7	18	6	11	15	213
n_{12}	11	7	9	11	14	12	8	9	13	2	15	0	11	12	10	11	9	4	8	176
n_{13}	14	10	5	6	17	15	6	2	2	13	18	11	0	15	15	3	12	7	3	174
n_{14}	8	12	17	20	6	8	18	17	17	12	3	12	15	0	4	15	3	8	12	207
n_{15}	9	12	17	19	10	7	18	17	17	8	7	10	15	4	0	15	3	8	12	208
n_{16}	17	13	8	7	20	18	9	5	3	13	18	11	3	15	15	0	12	7	3	197
n_{17}	11	15	17	18	9	10	17	14	14	11	6	9	12	3	3	12	0	5	9	195
n_{18}	15	11	12	13	14	15	12	9	9	6	11	4	7	8	8	7	5	0	4	170
n_{19}	17	13	8	9	18	18	9	5	5	10	15	8	3	12	12	3	9	4	0	178

Fig. 10. Distance matrix of the simulated network topology.

blackboards are compromised in distributed blackboard architecture.

Fig. 7 is a simulation network topology for our tests, and Fig. 8 is the agent distribution. Let there be six kinds of agents communication relations, the communication relation matrix $[r_{ij}]$ is shown as Fig. 9. In the matrix, if a_i communicate to a_j, then r_{ij}=1, else r_{ij}=0.

Now we make the four kinds of case studies and simulation experiments.

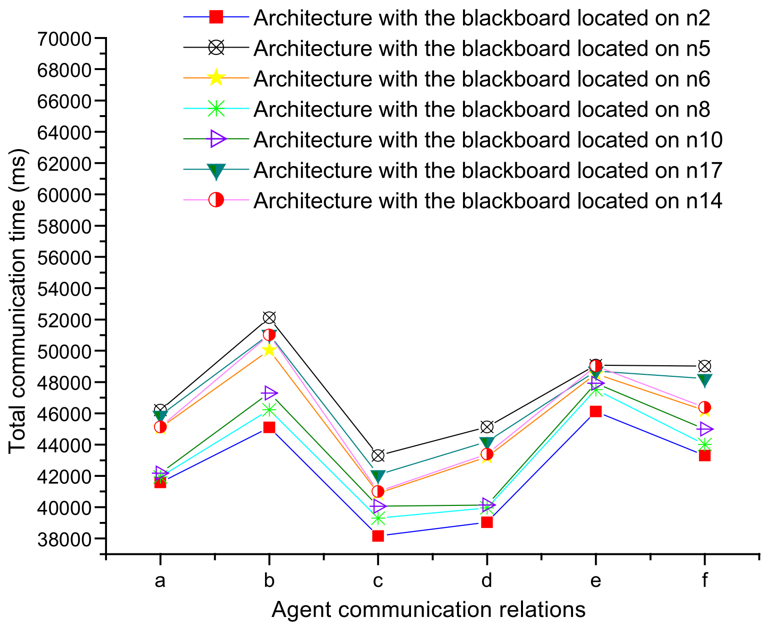


Fig. 11. The performance comparison between the constructed central blackboard architecture and the random ones.

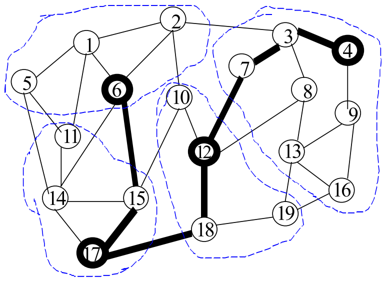


Fig. 12. Constructed blackboard architecture.

6.1. Case studies and simulation test for central blackboard architecture

Now we take the network topology in Fig. 7 and the agent system in Fig. 8 as an example. According to the method in Section 2, the distance matrix is shown as Fig. 10. Therefore, we should select n_2 as the central blackboard location. Otherwise, we make some random central blackboard architectures where the blackboards are located randomly, such as n_5, n_6, n_8, n_{10}, n_{14}, and n_{17}.

Now we make simulated agent communications according to Fig. 9 through the constructed central blackboard architectures and the ones of random central blackboard. The results are shown as Fig. 11. Obviously, we can see that our constructed architecture is the most efficient. Therefore, our constructing model for central blackboard architecture is effective.

6.2. Case studies and simulation test for distributed blackboard architecture construction

In order to show how effectively our proposed model can work, we compare the performance of (i) the distributed blackboard architecture that sub-blackboards are randomly located and (ii) the one that applies multi-medians location and Steiner tree method.

We also adopt the simulated network topology in Fig. 7, the agent distribution in Fig. 8, and the agent communication relations in Fig. 9.

It is not convenient to simulate the cost of building sub-blackboard accurately. Therefore, for simplicity, commonly in our simulation experiment, we can pre-assign the number of sub-blackboards as $\lceil\sqrt{n}\rceil$, where n is the number of nodes in the network.

Therefore, in the simulated network topology in Fig. 7, we set the number of sub-blackboards as 4. Now we use Algorithm 2 to construct the sub-blackboards locality, and each node (i.e. the agent on the node) selects the nearest sub-blackboard as its *master sub-blackboard*. Finally, we compute the communication topology among sub-blackboards.

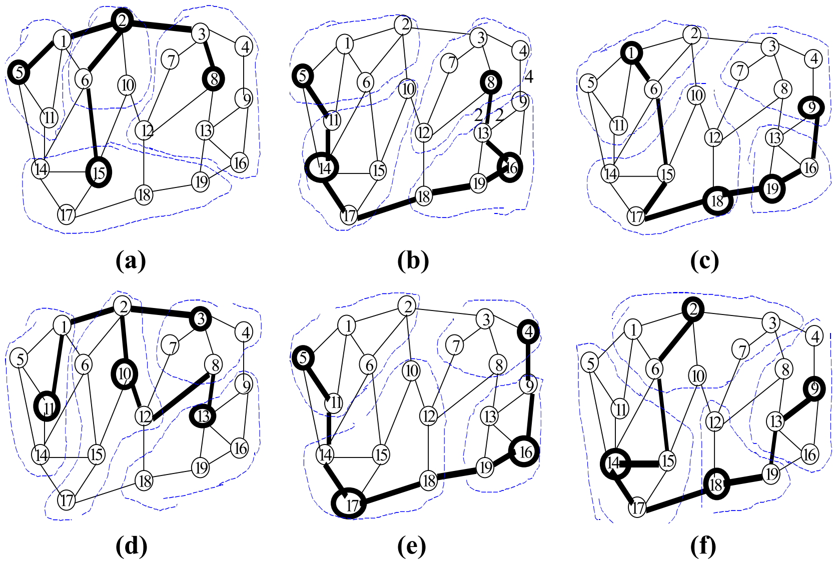


Fig. 13. Randomly constructed distributed blackboard architecture.

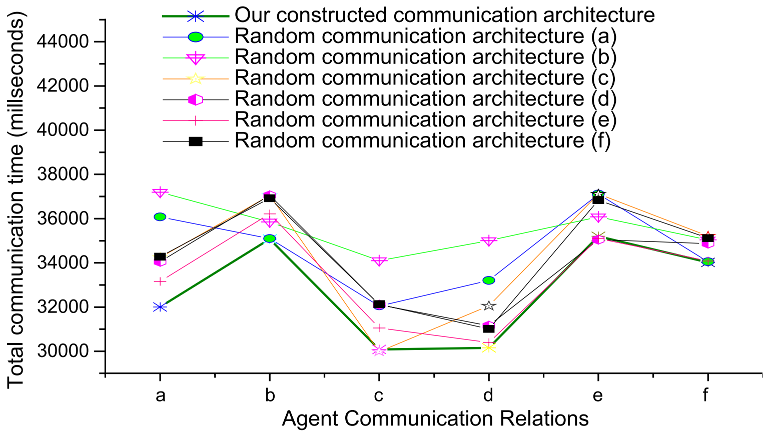
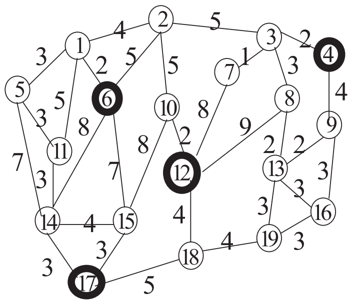
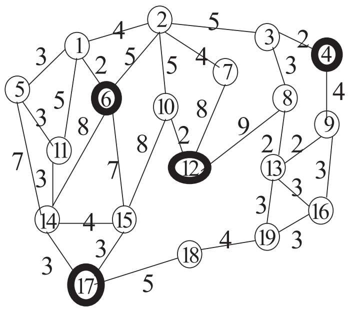


Fig. 14. The performance comparison between the constructed distributed blackboard architecture and the random one.



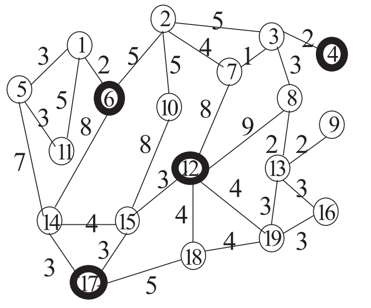
$$\begin{pmatrix}
0&1&0&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
1&0&1&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0\\
0&1&0&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&1&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&1&0&0&1\\
0&0&0&0&1&1&0&0&0&0&1&0&0&0&1&0&1&0&0\\
0&0&0&0&0&1&0&0&0&1&0&0&0&1&0&0&1&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&1&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&1&0&1&0
\end{pmatrix}$$

Fig. 15. Initial network topology and the matrix P.



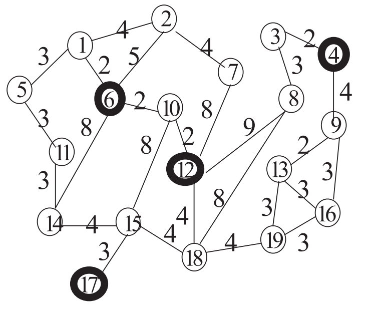
$$\begin{pmatrix}
0&1&0&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
1&0&1&0&0&1&1&0&0&1&0&0&0&0&0&0&0&0&0\\
0&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0\\
0&2&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&1&0&0&1\\
0&0&0&0&1&1&0&0&0&0&1&0&0&0&1&0&1&0&0\\
0&0&0&0&0&1&0&0&0&1&0&0&0&1&0&0&1&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&1&0&1&0
\end{pmatrix}$$

Fig. 16. Topology variation degree=3/32.



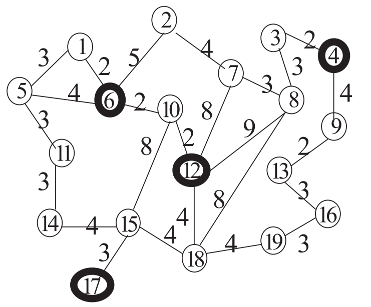
$$\begin{pmatrix}
0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&1&0&0&1&1&0&0&1&0&0&0&0&0&0&0&0&0\\
0&1&0&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&1&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&1&1\\
0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&1&0&0&1\\
0&0&0&0&1&1&0&0&0&0&0&0&0&0&1&0&1&0&0\\
0&0&0&0&0&0&0&0&0&1&0&1&0&1&0&0&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&1&0&1\\
0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&1&0&1&0
\end{pmatrix}$$

Fig. 17. Topology variation degree=9/32.



$$\begin{pmatrix}
0&1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&1&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&1&0&0&1\\
0&0&0&0&0&1&0&0&0&0&1&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&1&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&1&0&1&0
\end{pmatrix}$$

Fig. 18. Topology variation degree=14/32.



$$\begin{pmatrix}
0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&0&0\\
1&1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0\\
0&0&1&0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&1&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&1\\
0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&1&0&1&0
\end{pmatrix}$$

Fig. 19. Topology variation degree=19/32.

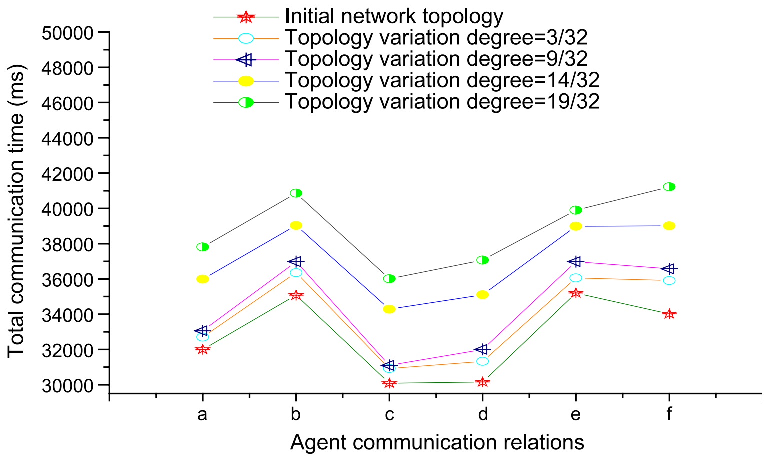


Fig. 20. Performance comparisons among different network topology.

The final agent distributed blackboard communication architecture is shown in Fig. 12.

Now we make simulation for the agents communication by the architecture of Fig. 12 and other architectures where sub-blackboards are randomly located of Fig. 13.

Fig. 14 is the simulation test results, from which we can see that the architecture in Fig. 12 is the most efficient. Therefore, our model that applies multi-medians location and Steiner tree methods into agents distributed blackboard architecture construction is correct and effective.

6.3. Case studies and simulation test of the performance when network topology changes

Now we make the case studies and simulation test of the performance when network topology changes in the distributed blackboard architecture. The initial topology is shown as Fig. 15. Now the network

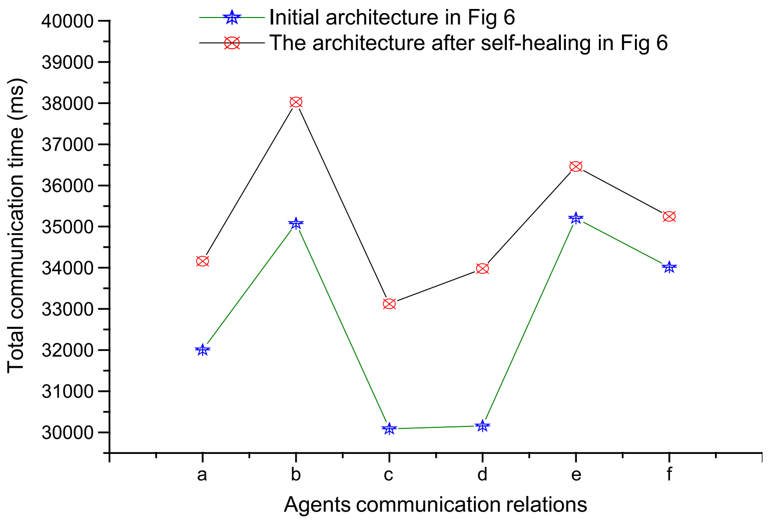


Fig. 21. Performance comparisons between the initial architecture and the self-healed one.

topology changes, the topology and the matrix *Ps* are shown as Figs. 16–19. The *topology variation degrees* are 3/32, 9/32, 14/32 and 19/32, respectively.

Now we make the simulated agent communication according to Fig. 9. The result is shown as Fig. 20. From Fig. 20, we can see that the more the topology variation degree be, the more difference the performance of the initial blackboard architecture is. Therefore, if the underlying network topology changes much more than a certain degree, we should adapt the blackboard architecture for current topology.

6.4. Case studies and simulation test of the fault-tolerance of distributed blackboard architecture

In Section 5, we have made case study for the fault-tolerance of distributed blackboard architecture. Now we take the architectures in Fig. 6 as examples. There is an agent blackboard constructed by our approach, shown as Fig. 6(a). Now if the sub-blackboard on N6 fails, then according to our self-healing mechanism, the healed architecture can be seen as Fig. 6(b). Now we make simulation test for Fig. 6, the result is shown as Fig. 21. From the result, we can see that the distributed blackboard architecture after self-healing can perform the communications among agents. Therefore, our fault-tolerance and self-healing mechanism can get a new architecture which can operate well. However, we can see that the performance of the healed architecture is bad than the one of the initial constructed architecture, which also proves that the architecture where blackboard locations and communication topology are computed by our approach is optimal.

7. Conclusion

For constructing effective agent blackboard communication architecture, this paper presents an all-around model based on graph theory. Blackboard communication architecture includes the central blackboard architecture and the distributed one; and the central blackboard or distributed sub-blackboards locality is critical in the architecture. Therefore, we mainly address the problem of blackboard (or sub-blackboards) locality.

In central blackboard architecture, we design the blackboard locality-computing algorithm based on the single-median method of graph theory. We can use the algorithm to select the optimal locality for constructing central blackboard, and all agents in the system can communicate through the blackboard. The case studies and simulation experiment prove that our constructed agent central blackboard communication architecture outperforms than the other ones that central blackboard is randomly constructed.

In distributed blackboard architecture, we design the sub-blackboards locality and number-computing algorithm based on the multi-median method of graph and *Fibonacci search* method. After constructing the sub-blackboards, each node can select the nearest sub-blackboard to construct the federated system. Also, the agents can communicate through their respective sub-blackboard. Moreover, we apply the Steiner tree method to compute the communication topology among sub-blackboards. The case studies and simulation experiment prove that our constructed agent distributed blackboard architecture outperforms the other ones that sub-blackboards are randomly constructed.

At last, we also address the agent blackboard communication architecture's adaptation mechanism for dynamic topology and blackboard failure. If the network topology changes more than a certain degree, we should re-construct the architecture; if one or some blackboard (or sub-blackboard) goes wrong, the system can select the "backup" blackboard and realize self-healing.

References

[1] Francois Charpillet, Philippe Lalanda, Jean-Paul Haton, A real time blackboard based architecture, ECAI92 10th European Conference on Artificial Intelligence, ECAI, August, 1992, pp. 262–266.

[2] V. Botti, F. Barber, A temporal blackboard for a multi-agent environment, Data and Knowledge Engineering 15 (1995) 189–211.

[3] Cyprian Foinjong Ngolah. A Tutorial on Agent Communication and Knowledge Sharing. http://www.enel.ucalgary.ca/People/far/Lectures/SENG60922/PDF/tutorials/2002/Agent_Communication_and_Knowledge_Sharing.pdf.

[4] Y.C. Jiang, Z.Y. Xia, Y.P. Zhong, S.Y. Zhang, An adaptive adjusting mechanism for agent distributed blackboard architecture. Microprocessors and Microsystems, in press.

[5] Yichuan Jiang, Yiping Zhong, Shiyong Zhang, Applying *multi-medians location and Steiner tree methods* into agents distributed blackboard architecture construction, Proceeding of the 17th Australian Joint Conference on Artificial Intelligence (AI2004). Cairns, Australian, 2004.
[6] Nicos Christofides, Graph Theory: An Algorithmic Approach, Academic Press, London, 1975, pp. 79–120.
[7] M.B. Teitz, P. Bart, Heuristic methods for estimating the generalized vertex median of a weighted graph, Operations Research 16 (1968) 955–961.
[8] L. Cooper, Location allocation problems, Operations Research 11 (1963) 331–343.
[9] Chen Senfa, Network Model and Optimization, South-East University Press, Nanjing, China, 1992, pp. 91–112.
[10] S.E. Dreyfus, R.A. Wagner, The Steiner problem in graphs, Networks 1 (1972) 195–207.
[11] F.K. Hwang, D.S. Richards, Steiner tree problems, IEEE/ACM Transactions on Networking 22 (1992 January) 55–89.
[12] Harris, Jr., C. Frederick, Steiner Minimal Trees: An Introduction, Parallel Computation, and Future Work. University of Nevada. http://www.cs.unr.edu/~fredh/papers/books/handbook/paper.ps.gz.
[13] Martini Zachariasen. Algorithms for Plane Steiner Tree Problems. PhD thesis [1998]. University of Copenhagen, Denmark, 1998.
[14] Brian Dazheng Zhou. Steiner Tree Optimization in Multicast Routing. MS thesis [2002]. University of Guelph. July, 2002.
[15] L. Kou, G. Markowsky, L. Berman, A fast algorithm for Steiner trees, Acta Informatica 15 (1981) 141–145.
[16] S.K.S. Gupta, P.K. Srimani, Adaptive core selection and migration method for multicast routing in mobile ad hoc networks, IEEE Transactions on Parallel and Distributed Systems 14 (1) (2003 January) 27–38.
[17] Sudipto Guha, Adam Meyerson, Kamesh Munagala, Improved algorithm for fault-tolerant facility location, Proceedings of 12th ACM-SIAM Symposium on Discrete Algorithms, 2001.
[18] Tcl Developer Xchange: The Tcl and Tk Toolkit, Tcl /Tk Version 8.4.4. URL: http://www.tcl.tk/software/tcltk/8.4.html.
[19] Tcl Contributed Archive: Extended Tcl (TclX), Version 7.0a. URL: http://www.neosoft.com/tcl/. Neosoft, 2003.
[20] Home Page for Tix: Tk Interface eXtension: Tix-Programming Library for the Tk Toolkit, Version 4.0. URL: http://tix.mne.com/. 2003.
[21] Internet Programming Toolkit: Binprolog Version 1.99. URL: http://www.binnetcorp.com/BinProlog/, 1995.
[22] Aglets Software Development Kit: Aglets Software Development Kit v2 (Open Source). URL: http://www.trl.ibm.com/aglets/. 2002.
[23] K. Mehlhorn, Data Structures and Algorithms: 1. Sorting and Searching, Springer-Verlag Berlin, Heidelberg, 1984.

Y.C. Jiang was born in 1975. He received his M.S. degree in computer science from Northern JiaoTong University (now Beijing JiaoTong University), China in 2002. He is currently a PhD candidate in computer science with the Department of Computing and Information Technology, Fudan University, China. His research interests include mobile agent system, artificial intelligence and network security.

P. Yi was born in 1969. He received the BSc degree in department of computer science and engineering from the PLA University of Science and Technology, Nanjing, China, in 1991. He received the MSc degree in computer science from the Tongji University, China, in 2003. He is currently a Ph.D. candidate at the department of Computing and Information Technology, Fudan University, China. His research interests include mobile computing and ad hoc networks security.

S.Y. Zhang was born in 1950. He is now a professor, and also the director of the Center of Networking & Information Engineering of Fudan University, China. His research interests include network system, mobile agent system and network security.

Y.P. Zhong was born in 1953. She is now an associate professor, and also the associate director of the Department of Computing & Information Technology of Fudan University, China. Her research interests include distributed system and data communication.