EURASIP Journal on Image and Video Processing

RESEARCH Open Access

Design and analysis of an effective graphics collaborative editing system

Chunxue Wu[1], Langfeng Li[1], Changwei Peng[1], Yan Wu[2], Naixue Xiong[3*] and Changhoon Lee[4]

Abstract

With the rapid development of computer-supported cooperative work (CSCW) technology, graphical collaborative editing plays an increasingly important role in CSCW. The most important technique in graphics co-editing is the consistency of graphics co-editing, which mainly includes causality consistency, consistency of results, and consistency of intention. Most of the previous research was abstract and ineffective, lacking theoretical depth and scalability. However, because the algorithm proposed in this paper can solve the contradictions in the consistency of graphical collaborative editing, the research in this paper has particularity, and the results will be proven by the experiment described in the paper. In order to solve the consistency conflict problem of graphic collaborative editing, the common graphics collaborative editing algorithm (CGCE algorithm) is proposed. It is proposed not only to perfect and expand the definition of graphics collaborative editing but also to merge with HTML5 Canvas, WebSocket, jQuery, Node.js and other network programming languages and technologies. The graphic collaborative editing based on the design and implementation of this paper can effectively solve the consistency conflict problem of many users during the collaborative editing of graphics, which ensures that the graphics of each graphical collaborative editing interface is consistent and the collaborative work can achieve the desired effect.

Keywords: CSCW, Graphics co-editing, Consistency, CGCE algorithm

1 Introduction

Object-based graphical editing systems are a particular class of collaborative editing systems where shared objects subject to concurrent accesses are graphic objects such as lines, rectangles, circles, and text boxes [1]. Graphic collaborative editing brings great convenience; for example, many artists in different regions can work together to edit and complete a picture. Graphic collaborative editing can also have a positive impact on education. For example, an art teacher can monitor students' work online and modify the work at any time. These were unimaginable before. With the increase of collaborative users, the structure of the network is more and more complicated, and collaborative editing is deeply influenced by network congestion. Traditional routing mechanisms and load-balancing approaches cannot make efficient use of the network, owing to lack of network status information and flexible ways to perform dynamic controlling operations [2]. Disadvantages caused by network congestion include increased packet loss rate, end-to-end delay, and reduced system throughput. If finances permit, improving physical hardware facilities is also a good choice to shorten the delay. The newest technologies, including an optical amplifier, dispersion compensation, and forward error correction [3], may alleviate congestion problems. High data rates, low cost, and collision reduction with the full-duplex approach and the elimination of chaining limits inherent in hub-bed Ethernet networks have made the switched Ethernet a dominant network technology [4].

Because full-duplex communication has so many advantages, the graphic editing system is generally used in full-duplex communication, and at the same time, it is

* Correspondence: 276682959@qq.com
[3]College of Intelligence and Computing, Tianjin University, Tianjin 300350, China
Full list of author information is available at the end of the article

also extremely important to solve the distribution of different regions of the collaborative user site synchronization between the computers. Realistic network applications often require multiple computers with a high clock consistency-clock synchronization. The Network Time Protocol (NTP) is widely used to synchronize computer clocks on the Internet [5]. But traditional time synchronization technology such as NTP has been unable to meet this precision requirement, and the cost of a GPS system is prohibitively expensive. However, IEEE 1588 protocol development and maturity provide a low-cost, high-precision network clock synchronization program. Software time stamping may deliver acceptable results for a certain range of applications [6]. Furthermore, the shown architecture will be able to support the upcoming PTP version 2 in hardware as well as in software [7]. Taking into account the economic and practical needs related to these factors, the IEEE 1588 protocol is adopted to solve problems with synchronization among the collaborative sites, and this paper is based on the B/S architecture. B/S architecture is adopted because of the following strengths of the B/S architecture software:

1. *Simple maintenance and upgrades*: For facilitation of the frequent improvements and upgrades of the software system, B/S architecture is more advantageous than C/S architecture. With a slightly larger unit, system administrators of C/S architecture have to manage thousands of computers, but with B/S architecture, system administrators only need to manage the server online.
2. *Independence of the system*: Based on the B/S structure, software can be simply installed on a Linux server, Windows server, Unix server, and so forth. Therefore, the choice of operating system is diversified. No matter what kind of system the user chooses, the computer will not be affected.
3. *Running heavy load of data*: Administrators only need to manage the server online through the Internet browser on the server, and the key is that logic settings are generally relatively simple on the browser so that managers only need to do hardware maintenance on the browser.

In this paper, we describe how computer graphics co-editing is communicated between all sites using WebSocket technology. WebSocket is a specification developed as part of the HTML5 initiative, and it only establishes a TCP socket connection after the first request for connection, which can save server resources and network bandwidth and achieve real-time communication [8]. The WebSocket protocol supports full-duplex communication between the client and the remote host. By using an existing server, we can focus on learning about the easy-to-use API that enables creation of WebSocket applications [9]. New networking approaches have recently been introduced that are based on repurposed techniques for delivering web pages (Comet) or integration of real-time communication directly into the browser (HTML5 WebSockets) [10–12].

This technology is simplifying the work of programmers, harmonizing access to diverse devices and applications, and giving users amazing new capabilities [13]. HTML5 and the Canvas element have real potential in many useful applications, but the rest of this paper just focuses on Cartagen, a vector-based, client-side framework for rendering maps in native HTML5, and its potential application [14, 15]. In addition to the use of HTML5 Canvas technology, jQuery- and Node.js-related technology are also used. jQuery is a JavaScript library that is fast, small, and feature-rich. jQuery strikes a completely different balance between cost and flexibility of its configuration interface [16]. Node.js is one of the more interesting developments that has recently gained popularity in the server-side JavaScript space, and it is a framework for developing high-performance concurrent programs that do not rely on the mainstream multithreading approach but use asynchronous I/O with an event-driven programming model [17]. With the HTML5 Canvas technology, jQuery, Node.js, WebSocket, and other technical support for the graphics collaborative editing system with software to achieve protection, the next step is to achieve the consistency of graphic editing computer-supported cooperative work (CSCW) algorithm research [18, 19]. Within the CSCW field, collaborative editing systems have been developed to support a group of people sharing editing documents from different sites. Object-based graphic editing systems are a special type of graphic editing system [20]. Consistency of key technologies needed to achieve these are described next.

1.1 Computer synchronization

Computer synchronization refers to the distribution of the different geographical sites of the collaborative computer to achieve clock synchronization between locations. This is the most basic requirement for graphics collaborative editing, because if the collaboration between the site computers is not synchronized, a very complex algorithm to maintain the consistency between different sites is needed. As the number of collaborators increases, the number of computers in the collaborative site increases, and it is more difficult to maintain the consistency of the graphic editing system between different sites [21–23]. Therefore, the synchronization of computers between sites ensures graphics co-editing consistency.

1.2 Consistency algorithm

A good algorithm can play a key role in the coherence of graphics collaborative editing. For example, the algorithm used to achieve the required software and hardware resources generally refers to the algorithm's time complexity and spatial complexity being as small as possible, and the efficiency of the algorithm being very high, which can be expressed as the time required to reach the end of the operation [24–28]. There is no best algorithm in the world, and only only the improved algorithms in this paper can be continually effective to the updated applications. Any algorithm has its own limitations, and the limitations of the algorithm over time will become more and more obvious, so this paper studies a relatively efficient algorithm for consistency. This paper mainly describes a common algorithm for graphics collaborative editing (CGCE) [29–31].

1.3 Programming language selection

If a good system has a good algorithm, then to implement the algorithm, a programming language is needed. This paper mainly uses HTLM5's Canvas core technology, WebSocket, jQuery, and Node.js-related technology as the method, with the graphics collaborative editing algorithm as the core, according to the reality of the situation, the use of C# language in front of the background, and the preparation of graphics collaborative editing system, so as to complete the subject graphics editor consistency research [32].

2 Method

2.1 Some problems of graphics collaborative editing

Graphic editing refers to the computer's ability to edit some of the graphics, the package on the point, line, surface additions and deletions, as well as their movement, copy, rotation, and other operations. Common graphic editing software are Adobe's AI and Photoshop, Corel's CorelDRAW, Autodesk's AutoCAD, Discreet's 3D Studio Max, and so on. With scientific progress, many people are required to collaborate sometimes to complete tasks, and graphics collaborative editing can provide people with great convenience. Collaborative editors can be distributed in different areas, or they can be structurally diverse and complex WANs [33], just requiring sitting in an office to collaborate and complete projects. Therefore, the prospect of applying collaborative editing technology is very exciting. Although the graphic editing software is so abundant and graphic editing technology is very mature, graphics collaborative editing software is rare. Graphics collaborative editing systems have many practical problems:

1. *Robustness*: Many graphics collaborative editing techniques are hindered by technical problems of poor robustness. Different theories of graphics collaborative editing research regarding the increased problems faced in the real environment include the different network structure, network congestion, network routing, and so forth [34, 35] The robustness of the environment fluctuates poorly, resulting in inconsistent results.
2. *High responsiveness*: In the Internet environment, the response to the local user's actions must be quick, even as collaborating users reside on different machines connected via the Internet with a long and nondeterministic communication latency [19]. The graphics collaborative editing operation response speed depends on a lot of conditions:
   a. *Hardware performance of the computers in the site*: There are significant differences in performance between different prices, different models, and different systems. The price is high, the model is new, and the system is upgraded. The computer generally has a good processor and fast response.
   b. *Spatial complexity and time complexity of the algorithm*: When a few graphics operations are performed on the graphics co-editor, the computer will run the corresponding graphical co-editing algorithm code after each step [36, 37]. The greater the time complexity and spatial complexity of the algorithm, the more time it takes for the computer to run the code, and the slower the response of the graphical collaborative editor.
   c. *The transmission delay of data*: Computer data in the physical layer are transmitted in binary form, which raises the following problems:

i. *Bit error rate*: Data in the channel will be the cause of data distortion, which results in bit error rate. The data length of the data transmission needs to be consistent with Shannon's theorem:

$$C = B\ log_2^{\left(1+\frac{S}{N}\right)}$$

Then the transmission process code length can be as long as possible, and the bit error rate can be reduced to a relatively low level.

ii. *Network congestion*: With the development of science and technology, computer users are increasing very rapidly at the same time, the computer network-related hardware facilities have also been greatly improved, but network congestion is still inevitable. When the computer network is congested, the spread of communication between the different computers through the Internet will become very slow, and the response between the cooperative operations will also slow. This may lead to the following results:

- When the network is in normal recovery, collaborative application confusion or even collapse can occur. Because the collaboration among users, if they do not know the network has been congested, is still constantly in cooperation, when the network returns to normal, the collaborative operation may be invalid.
- When the network is recovered, collaborative editing consistency will have become damaged. After the network is restored, the response between the cooperative users is normal, and the cooperative information of the network congestion may be transmitted to the different cooperative computers or lost. The result may be inconsistent owing to the lag of the response, resulting in inconsistent computer collaboration among the collaborative users [38].

2. *High concurrency*: Multiple users are allowed to concurrently edit any part of the shared document at any time by facilitating natural information flow among collaborating users [19]. High concurrency is the basis for computer collaboration requirements. The higher the concurrency, the better the synergistic effect [39]. However, the higher the concurrency will have a problem that affects the performance of the entire graphics co-editing [40, 41], launching consistency issues. The traditional consistency maintenance methods, such as lock and serialization methods, are not suitable for real-time editing. In this paper, we will study the consistency of graphics co-editing.

2.2 Related concepts involved in coherence of graphics co-editing

Definition 1 Minimal Editing Unit "⊕". Minimal Editing Unit means that the easiest graphics can be edited at once. The smallest editing unit is a point. If the operation is to draw a point, it can be expressed as ⊕point.

Definition 2 The Operational Specification Representation: the Intent Consistency Model. The graphical representation of graphical co-editing in the Model of Intention Consistency is more complex than other conformance models, and each operation needs to add the intent of the co-editor. The operation of graphic editing, includes increase, delete, turn, move, copy, and so on. The operations in these operational intent conformances are shown in Table 1.

Definition 3 Simple Graphics "Δ". Point, line, equilateral triangle, regular quadrilateral, regular polygon (more than four edges), and circle are called simple shapes in graphic collaborative editing. In addition to simple graphics, all others are called complex graphs. A simple graphic can itself be minimal editing units, or it can be made up of many minimal editing units. If the operation is to draw an equilateral triangle, it can be expressed as ΔEquilateral triangle.

Definition 4 The Center of Simple Graphics. A simple graphics center is center of gravity in this paper according to Definition 3, and it can be more convenient for graphic editing. Complex graphics for these operations can be divided into a number of simple graphics, and then the center of these simple graphics can be found.

Definition 5 Delayed Deadline " ρ". Delay time is an operation from one site to another site specified by the maximum delay. In order to prevent the occurrence of a loss of operation due to network congestion or failure, we set a delay cutoff time in the graphic editing system. If there is a delay in the operation of the graphic editing system, the operation has not reached the destination site at the deadline, and the destination site issues a retransmission request. Regarding delay time, depending on the computer network environment, you can set a different delay cutoff time. This scheme requires a station to send an operation to another station, but an operation also needs to be sent to receive confirmation information if the other destination site in the delay deadline to receive the operation, the source site to send a received operation confirmation of information.

Definition 6 Causal Ordering Relation "→". Given two operations O_1 and O_2, generated at sites i and j, O_1 is causally ordered before O_2, expressed as $O_1 \to O_2$, if and only if: (1)i = j and the generation of O_1 happened before the generation of O_2; or (2) i ≠ j and the execution of O_1 at site j happened before the generation of O_2; or (3) there exists an operation O_x, such $O_1 \to O_x$ and $O_x \to O_2$ [22].

Table 1 Expression of Operations

Operations	Expression
Add	Add[Object, Data, xPos, yPos, PreOperation]
Delete	Delete[Object, Data, xPos, yPos, PreOperation]
Move	Move[Object, xPos, yPos, PreOperation]
Rotate	Rotate[Object,Radian, PreOperation]
Copy	Copy[Object, xPos, yPos, PreOperation]

Definition 7 Dependent and Independent Relations "‖". Given any two O_1 and O_2, (1) O_1 is dependent on O_2 if and only if $O_1 \rightarrow O_2$; (2)O_1 and O_2 are independent (or concurrent), expressed as $O_1 \| O_2$, if and only if neither $O_1 \rightarrow O_2$, nor $O_2 \rightarrow O_1$ [22].

Definition 8 Conflict Relation "⊗". Operations O1and O2 conflict with each other, expressed as $O_1 \otimes O_2$, if and only if (1) $O_m \| O_n$; (2) $Target(O_1) = Target(O_2)$; (3)$Att.Key(O_1) = Att.Key(O_2)$; and (4) $Att.Value(O_1) \neq Att.Value(O_2)$ [22].

Definition 9 Compatibility Relation "⊙". Two operations O_1 and O_2 are compatible, expressed as $O_1 \odot O_2$, if and only if they do not conflict with each other [22].

Definition 10 Compatible Group Set (CGS). Given a group of operations GO, the conflict relationships among these operations can be expressed as a CGS [22].

Definition 11 Different Graphics Coverage (DGC). When the users are editing the different objects because the different objects are compatible with each other. If two or more different objects appear in the same coordinate position, the later object will overlay the object with the editing time earlier.

Definition 12 Concurrent Group Operations (CGO). In graphic editing, if a site in a few short periods of time continues to operate, then we can treat the operations of this site as concurrent group operations. In this paper, concurrent group operations are whole and not indivisible.

Definition 13 Absolute Differences in Coordinates. Assume that the coordinate of the operation intention is (x_1, y_1) and the actual coordinate position is (x_2, y_2), $\partial = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$, and if the value of ∂ is greater than a certain number, the operation will be revoked.

Definition 14 Graphical collaborative editing uncertainty. Graphical collaborative editing uncertainty refers to the final results of graphics co-editing being uncertain.

Property 1. Given a group of operations GO targeting the same object, there is a unique *MCGS* for this GO [22].

3 Results and discussion

The graphical editor consistency model mainly includes three categories as follows: (1) Causal Consistency Model, (2) Results Consistency Model, and (3) Intention Consistency Model.

3.1 Causal Consistency Model

Causal consistency, given any two operations O_1 and O_2, if $O_1 \rightarrow O_2$, O_1 happened before O_1 at all the sites. Graphics co-editing causal consistency is shown in Fig. 1. Assume that only the site $Site_1$ and $Site_2$ are graphically edited, and the collaborative user at $Site_1$ launches an operation O_1, which draws the graphic of the pentagonal star on Canvas. The operation O_1 contains the basic information of the five-pointed star

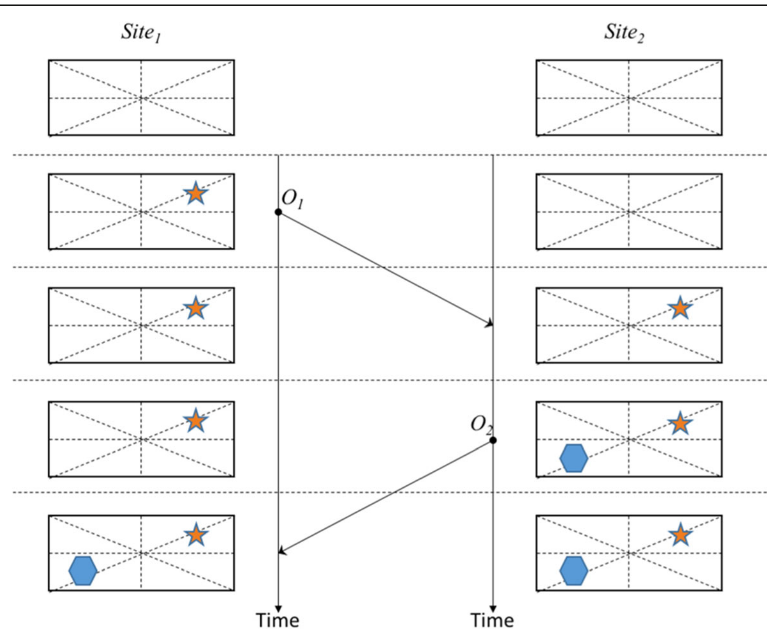

Fig. 1 Causal consistency

graphic object: the ID of the graphic, shape, color, and the position coordinates of the graphic on canvas. After the O_1 operation is performed at the site, a five-pointed star appears at the top right of the site's canvas. At the same time, the O_1 is sent from $Site_1$ to $Site_2$, and $Site_2$ checks whether there are any other operations that have not been performed in the cache. If no other operations are waiting in the cache, O_1 is to be put in the queue of waiting for execution. If not, the other operations to be executed are checked, and then $Site_2$ directly implements the operation O_1. The canvas would appear at $Site_2$ the same as at $Site_1$. In a certain period of time, the $Site_2$ produced O_2, when O_2 satisfies the execution condition at $Site_2$, and then the pentagonal star will appear in the lower right corner on the canvas. O_2 was passed from $Site_2$ to $Site_1$ through the Internet. If the execution condition at $Site_1$ is satisfied, O_2 is to be executed at $Site_1$.

There is an important problem in that $Site_1$, $Site_2$,⋯,$Site_n$, according to our experiment summary, because the number of sites approaches nearly 40, which will increasingly become quite complex, as shown in Fig. 2.

3.2 Results Consistency Model

Results consistency is present when all sites share a copy of the document when a collaborative session is in silent state [21, 42, 43]. There is a great difference between the Results Consistency Model and the Causal Consistency Model. Results consistency between operations does not necessarily require a certain relationship, and the consistency of the results only needs cooperation between sites after the completion of all operations. Ultimately, the results between the stations are the same. The Results Consistency Model is shown in Fig. 3.

The $Site_1$ creates O_1 and then executes the operation of O_1. There is a five-pointed star in the upper right corner on canvas at $Site_1$. At the same time, O_1 was passed from $Site_1$ to $Site_2$ through the Internet, and the transmission needs a certain delay. If the operation O_1 has not yet reached the $Site_2$, the $Site_2$ have creates an operation O_2, that is, five-pointed star is in the lower left corner of the canvas at $Site_2$. After it executes O_2 at $Site_2$, immediately O_2 was passed from $Site_2$ to the $Site_1$ through the Internet. When the operation O_1 and O_2 respectively arrive at $Site_1$ and $Site_2$. What's more, if both satisfy the execution condition, O_1 and O_2 would be executed at $Site_1$ and $Site_2$. A five-pointed star appears in the lower left corner of the Canvas at $Site_1$, and a five-pointed star appears in the upper right corner on Canvas at $Site_2$. We will find that the final result is the same at $Site_1$ and $Site_2$, although the operation O_1 and operation O_2 are created in order, but the execution results of $O_1$and O_2 are independent of their execution order. According to Property 1, it is not difficult to

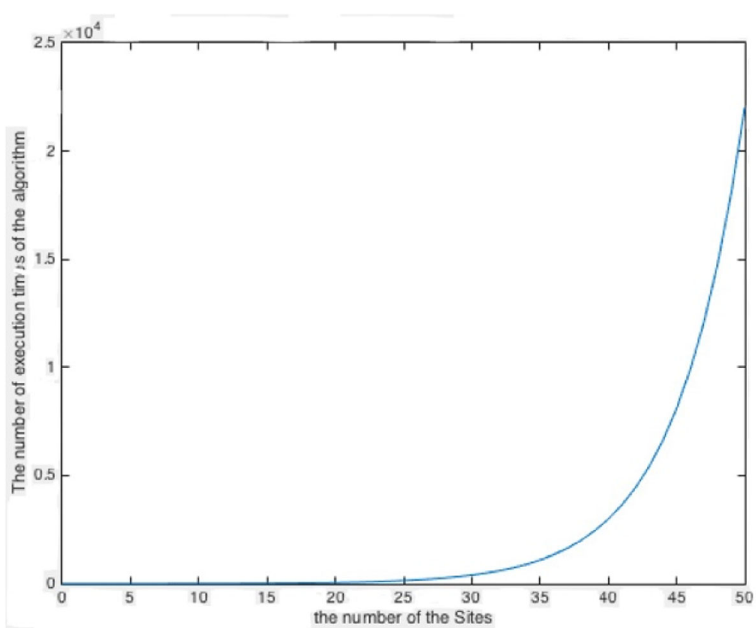

Fig. 2 Complexity analysis of causal consistency

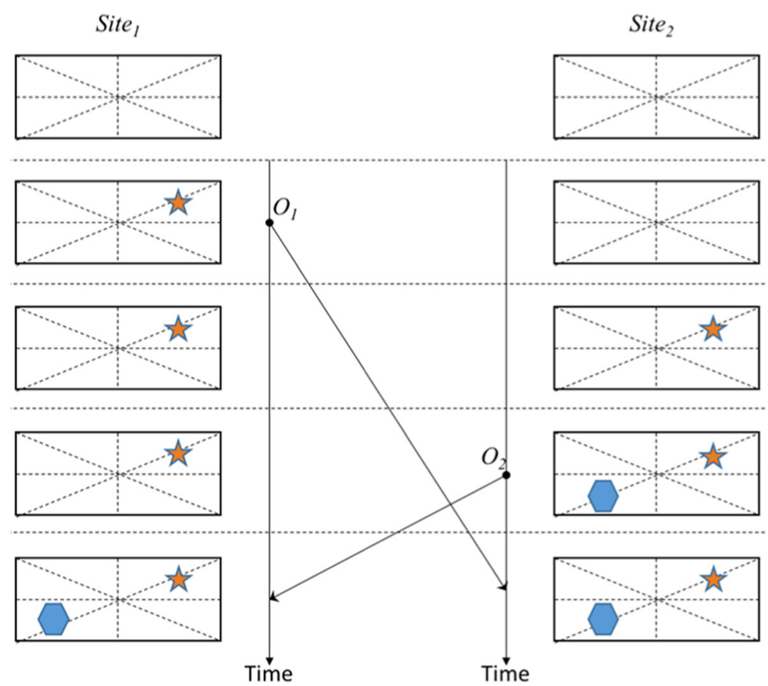

Fig. 3 Results consistency

know. The relationship between O_1 and O_2 in Fig. 3 is $O_1 \odot O_2$. In the graphics collaborative editing, assume that $MCGS_1$, $MCGS_2$, $\cdots$, $MCGS_n(n\epsilon N^*)$, $GO = \{MCGS_1, MCGS_2, \cdots, MCGS_n\}$, and $MCGS_1 = \{O_{p_1}, O_{p_2}, \cdots, O_{p_n}\}$, $MCGS_2 = \{O_{k_1}, O_{k_2}, \cdots, O_{k_n}\}, \cdots, MCGS_n = \{O_{r_1}, O_{r_2}, \cdots, O_{r_n}\}$.

The operation in the *MCGS* are compatible with each other, is that the operation in the *MCGS* no matter what kind of implementation order, the last site can maintain consistency. This conclusion is given in detail in the paper, Consistency Maintenance In Real-time Collaborative Image Editing Systems [22, 54, 55], it has been given a detailed proof, this article is not cumbersome.

There is an important problem that the $Site_1$, $Site_2, \cdots$, $Site_n$, according to our experiment summary: the number of sites approaches 25 and will become increasingly complex quickly, as shown in Fig. 4.

3.3 Intention Consistency Model

Intention Consistency Model is the operation of each site, the operation of the establishment of a certain operational effect among the orders, through the implementation of the scheduling algorithm, through the implementation of scheduling algorithms and conversion functions. As long as the implementation of each step needs to ensure that the implementation does not violate the established operation sequence, and then it is ensured that when all the operations are performed at each site, the internal data structure of each site has the same operational effect order from the operation object; that is, the consistency of the operational intention is maintained [21, 44].

Thus, among the Causal Consistency Model, Results Consistency Model, and Intention Consistency Model, the most complex is the Intention Consistency Model. The efficiency of the Intention Consistency Model can reflect the efficiency of the algorithm of graphic collaborative editing system [45, 46]. In graphic collaborative editing, while satisfying causal consistency or consistency of results, it may not be able to achieve the real intention of the graphical collaborative editing system.

Suppose there are three sites $Site_1$, $Site_2$, and $Site_3$, and then there are four operations, namely O_1,O_2,O_3 and O_4. $O_1 \| O_2$, $(O_1 \| O_2) \rightarrow O_3$, $O_1 \rightarrow O_4$, $O_2 \| O_4$, $O_3 \| O_4$, and the operation O_3 is lost during processing from $Site_1$ to $Site_2$, as shown in Fig. 5.

When there are many sites in the collaborative graphic editing, the relationship among the operations is complex. The Causal Consistency Model can only solve the problems about graphics collaborative editing in a relatively simple situation, such as when the operations among the sites are compatible with each other. The Results Consistency Model can

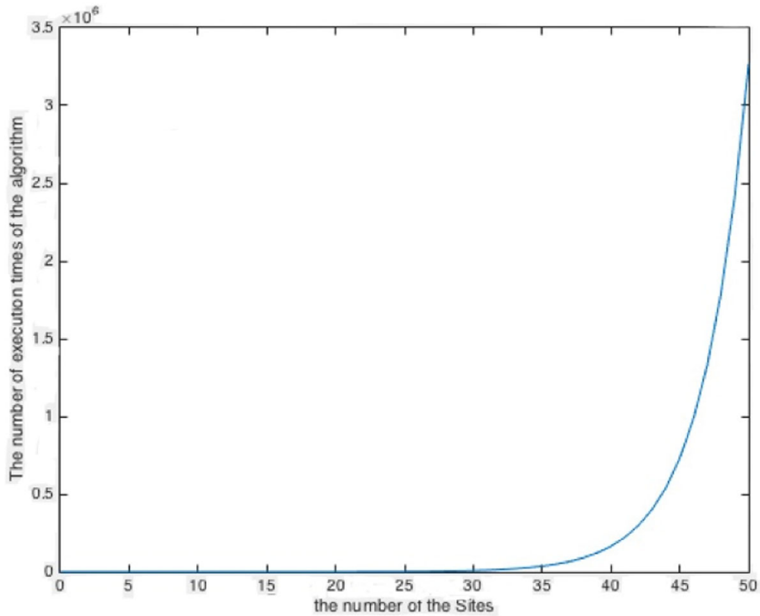

Fig. 4 Complexity analysis of causal consistency

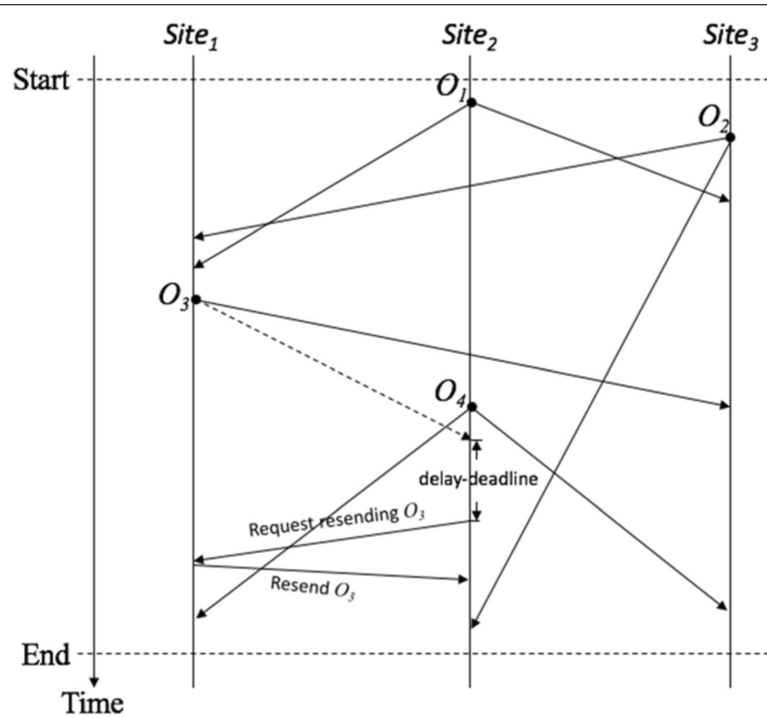

Fig. 5 Transfer of operations between different sites

ensure that the final result of graphics co-editing is right, but it cannot guarantee that the final result can be expected for collaborative editing users [47, 48]. For example, there is a concurrency model in Fig. 5, and the Causal Consistency Model cannot be used; thus, two models can be used: one is the Result Consistency Model, and the other is the Intention Consistency Model. Use the Results Consistency Model to handle the situation, as shown in Fig. 6. A group of operations (GO) represents the set of operations creates by all sites. *MCGS* (Max Compatible Groups Set) represents a set of the largest compatible sets of operations that operate in GO. At the beginning of the graphics co-editing, $GO = \{\}$ and $MCGS = \{\}$, indicating that the operation in GO and MCGS is empty. The process of results consistency is as follows: (1) When $O_1$and O_2 arrive at $Site_1$, variables are updated, $GO = \{O_1, O_2\}$. Because of $O_1 \| O_2$, $MCGS_1 = \{O_1\}$, $MCGS_2 = \{O_2\}$. When the $Site_1$ creates O_3, $GO = \{O_1, O_2, O_3\}$, $(O_1 \| O_2) \rightarrow O_3$, $MCGS_1 = \{O_1, O_3\}$, $MCGS_2 = \{O_2, O_3\}$. When O_4 arrive at $Site_1$, variables updated, $GO = \{O_1, O_2, O_3, O_4\}$. Because of $O_1 \rightarrow O_4$, $O_2 \| O_4$, $O_3 \| O_4$, so $MCGS_1 = \{O_1, O_3\}$, $MCGS_2 = \{O_2, O_3\}$, $MCGS_3 = \{O_1, O_4\}$. Finally, $GO = \{\{O_1, O_3\}, \{O_2, O_3\}, \{O_1, O_4\}\}$. (2) When the $Site_2$ creates O_1, variables updated, $GO = \{O_1\}$, $MCGS = \{O_1\}$, when $Site_2$ creates O_1 creates O_2, variables updated, $GO = \{O_1, O_4\}$, because of $O_1 \rightarrow O_4$, $MCGS = \{O_1, O_4\}$. Due to network congestion [49–53], because of the cutoff time delay, $Site_2$ has not received O_3, which has passed $Site_1$, and $Site_2$ requires $Site_1$ resend. $Site_1$ receives a resending request from $Site_1$, and then resends O_3. After $Site_2$ has received O_3, the variables are updated, $GO = \{O_1, O_3, O_4\}$. Because of $O_1 \odot O_3$, $O_1 \| O_4$, $O_3 \| O_4$, $MCGS_1 = \{O_1, O_3\}$, $MCGS_2 = \{O_4\}$. When O_2 arrives at $Site_2$, the variables are updated, $GO = \{O_1, O_2, O_3, O_4\}$, $O_2 \odot O_3$, $O_2 \| O_1$, $O_2 \| O_4$, and $MCGS_1 = \{O_1, O_3\}$, $MCGS_2 = \{O_2, O_3\}$, $MCGS_3 = \{O_1, O_4\}$. $GO = \{\{O_1, O_3\}, \{O_2, O_3\}, \{O_1, O_4\}\}$. (3) When $Site_3$ creates O_2, variables are updated, $GO = \{O_2\}$, $MCGS = \{O_2\}$. When O_1arrive at $Site_3$, variables are updated: $GO = \{O_1, O_2\}$, $O_1 \| O_2$, so $MCGS_1 = \{O_1\}$, $MCGS_2 = \{O_2\}$. When O_3 arrives at $Site_3$, variables are updated: $GO = \{O_1, O_2, O_3\}$, $O_3 \odot O_2$, $O_3 \odot$.

O_1, $MCGS_1 = \{O_1, O_3\}$, $MCGS_2 = \{O_2, O_3\}$. When O_4 arrives at $Site_3$, variables are updated: $GO = \{O_1, O_2, O_3, O_4\}$, $O_4 \odot O_1$, $O_4 \| O_2$, $O_4 \| O_3$, $MCGS_1 = \{O_1, O_3\}$, $MCGS_2 = \{O_2, O_3\}$, $MCGS_3 = \{O_1, O_4\}$, $GO = \{\{O_1, O_3\}, \{O_2, O_3\}, \{O_1, O_4\}\}$. To sum up, we can know the

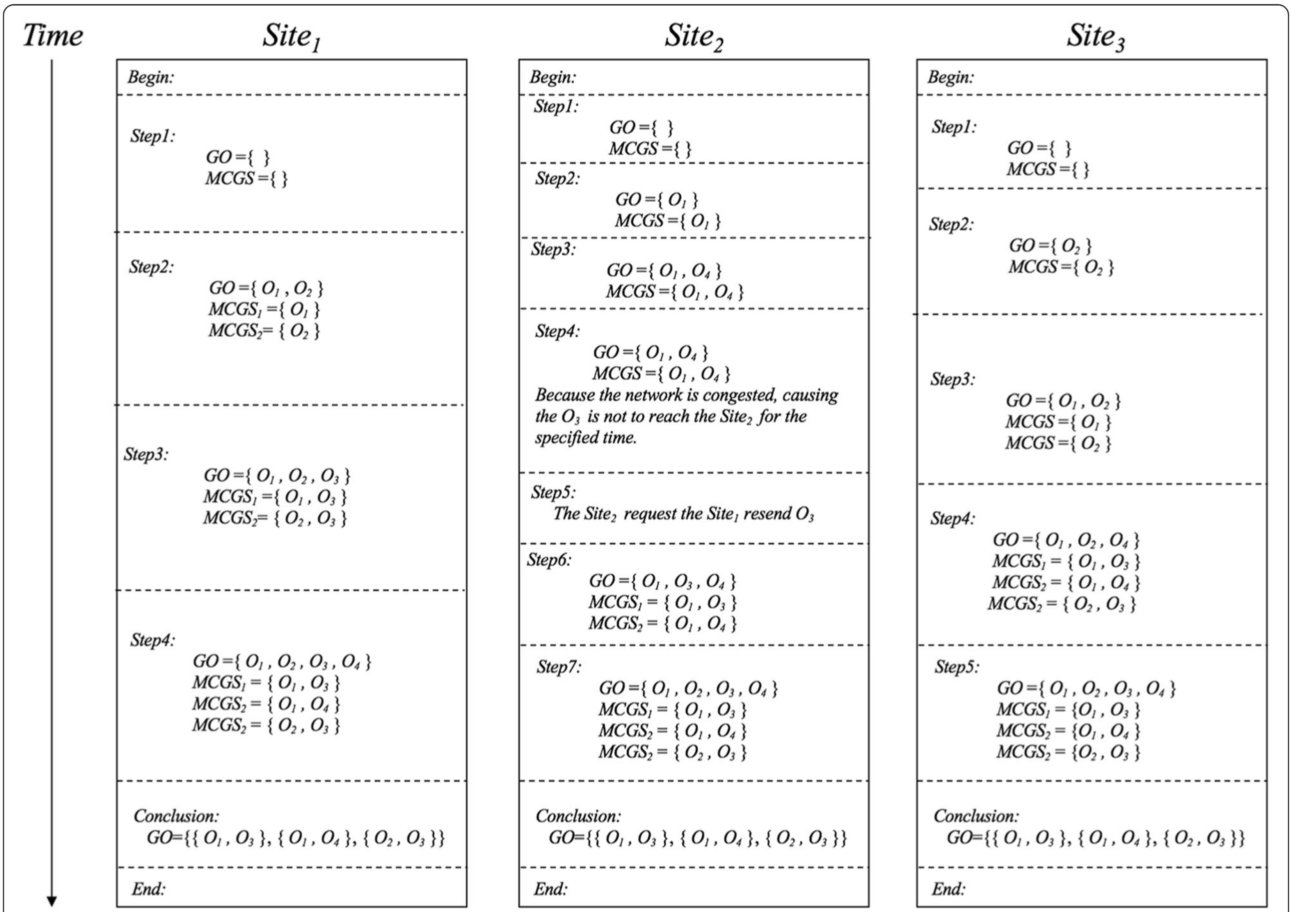

Fig. 6 Theoretical realization of results consistency

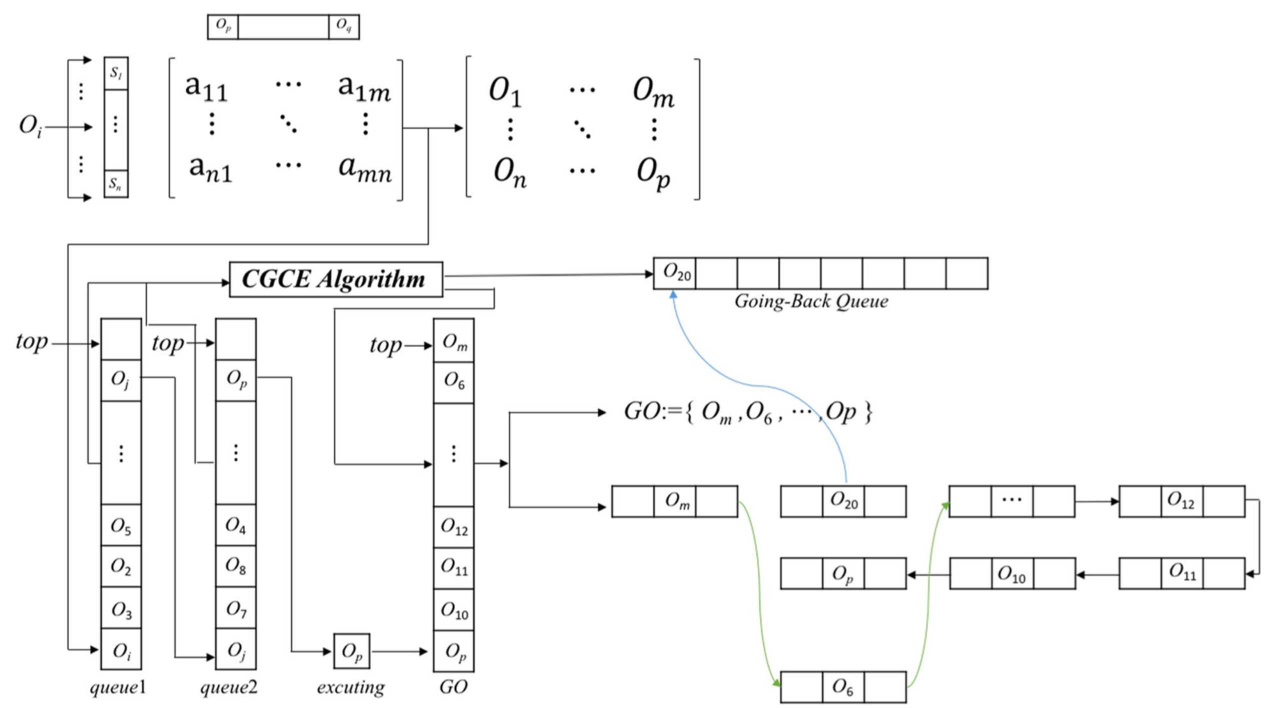

Fig. 7 Schematic diagram of intention consistency

basic principle of the Results Consistency Model, and the advantages and disadvantages of the Results Consistency Model are obvious. Advantages of the Results Consistency Model are as follows. In a set of operations, the operations in MCGS do not take into account the order of execution between them, which is more efficient for graphics co-editing. However, the shortcomings of the Results Consistency Model are also very prominent, according to the graphical collaborative editing uncertainly (Definition 14). The result of the Results Consistency Model is uncertain, as shown in Fig. 5, although three sites get the consistency of results finally. The collaborative users only expect that the result of collaborative co-editing is unique so that we have to further improve the coherence of graphics collaborative editing. We need a better consistency model than the Results Consistency Model, so the Intention Consistency Model will be a good choice, and next the common graphics collaborative editing algorithm (CGCE algorithm) for the Intention Consistency Model will be described in detail.

4 CGCE algorithm

4.1 Flowchart schematic diagram of intention consistency

A flowchart schematic diagram of the Intention Consistency Model is shown in Fig. 7.

4.2 Details of the CGCE algorithm

The core technology of editing consistency in the above collaborative graphics is about researching the intention consistency model and its algorithm for the complex. Because intention consistency is the most important consistency in collaborative graphic editing. The graphics collaborative editing consistency key technology CGCE algorithm is shown in Algorithm 1–8.

Algorithm 1 CGCE Algorithm

```
if ∂ ≤ ∂0 then
  ConflictResolution(Object[] GO, O)
  if O.id==Add_ID then
      Add( Object, Data, xPos, yPos, PreOperation )
  else if O.id==Delete_ID then
      Del(Object, Data, PreOperation );
  else if O.id==Move_ID then
      Move(Object, Data, xPos, yPos, PreOperation)
  else if O.id==Rotate_ID then
      Rotate(Object, Radian, PreOperation)
  else if O.id==Copy_ID then
      Copy(Object, xPos, yPos, PreOperation)
  else
      alert(" Error!")
  end if
else
   Undo( );
end if
```

Algorithm 2 Add Algorithm

```
if Data is Point then
    Object:=Object+⊕Point(xPos, yPos)
    if ξ[i + 1] is not empty then
        PreOperation← ξ[i + 1]
    else
        PreOperation← empty
    end if
else if Data is Line then
    Object:=Object+⊕Line{Head(xPos_1, yPos_1), End(xPos_2, yPos_2)}
    if ξ[i + 1] is not empty then
        PreOperation← ξ[i + 1]
    else
        PreOperation← empty
    end if
else if Object is Simple graphic then
    Object:= ⊕Δ{Head(xPos_1, yPos_1), Center(x, y), End(xPos_2, yPos_2)}
    if ξ[i + 1] is not empty then
        PreOperation← ξ[i + 1]
    else
 ⊕ # ← ⊕ # +{{⊕Point_1, ⊕Point_2, ⋯ , ⊕Point_k} ∪
            {⊕Line_1, ⊕Line_2, ⋯ , ⊕Line_i}∪ ⋯ ∪
{⊕Δ_1, ⊕Δ_2, ⋯ , ⊕Δ_n} }
    end if
    if ξ[i + 1] is not empt then
        PreOperation← ξ[i + 1]
    else
        PreOperation← empty
    end if
end if
```

Algorithm 3 Delete Algorithm

```
if Object is not empty then
    if Data is point then
        Object ← Object+⊕Poin
    else if Data is Line then
        Object ← Object+⊖ Line;
    else if Data is simple graphic then
        Object ← Object+⊖ Δ
    else
        Object ← Object+⊖#
    end if
    if ξ[i + 1] is not empty then
        PreOperation← ξ[i + 1]
    else
        PreOperation← empty
    end if
else
    alert("the Object is empty! The deleting operation is error!")
end if
```

Algorithm 4 Move Algorithm

```
if Object is not empty then
    if Data is point then
        ⇆ Point(xPos, yPos)
    else if Data is Line then
        ⇆ Line{Head(xPos_1, yPos_1), End(xPos_2, yPos_2)}
    else if Data is simple graphic then
        ⇆ Δ{Head(xPos_1, yPos_1), Center(x, y), End(xPos_2, yPos_2)}
    else
        ⇆#⇔ {{⇆ Point_1, ⇆ Point_2, ⋯ ,
             ⇆ Point_k}
 ∪⇆ Line_1, ⇆ Line_2, ⋯ , ⇆ Line_i} ∪ ⋯ ∪ {⇆ Δ_1, ⇆ Δ_2, ⋯ , ⇆ Δ_n}
    end if
    if ξ[i + 1] is not empty then
        PreOperation← ξ[i + 1]
    else
        PreOperation← empty
    end if
else
    alert("the Object is empty! The deleting operation is error!")
end if
```

Algorithm 5 Rotate Algorithm

```
if Object is not empty then
    if Data is Line then
        ↺ Line{Center(x, y), radian}
    else if Data is simple graphic then
        ↺ Δ{Center(x, y), radian}
    else ↺#← {{↺ Point_1, ↺ Point_2, ⋯ , ↺ Point_k} ∪
↺ Line_1, ↺ Line_2, ⋯ , ↺ Line_i} ∪ ⋯ ∪{↺ Δ_1, ↺ Δ_2, ⋯ , ↺ Δ_n}
    end if
    if ξ[i + 1] is not empty then
        PreOperation← ξ[i + 1]
    else
        PreOperation← empty
end if
else
    alert("the Object is empty! The deleting operation is error!");
end if
```

Algorithm 6 Copy Algorithm

if Object is not empty **then**
  if Data is point **then**
    ⊾Point(xPos, yPos)
  else if Data is Line **then**
    ⊾Line $\begin{cases} \text{Head}(xPos_1, yPos_1) \\ \text{End}(xPos_2, yPos_2) \end{cases}$
  else if Data is simple graphic **then**
    ⊾Δ $\begin{cases} \text{Head}(xPos_1, yPos_1) \\ \text{Center}(x, y) \\ \text{End}(xPos_2, yPos_2) \end{cases}$
⊾ ♯ ← {{⊾$Point_1$, ⊾$Point_2$, ⋯, ⊾$Point_k$}
    ∪ ⊾$Line_1$, ⊾$Line_2$, ⋯, ⊾$Line_i$} ∪ ⋯ ∪{⊾Δ_1, ⊾Δ_2, ⋯, ⊾Δ_n} }
  end if
  if $\xi[i+1]$ is not empty **then**
    PreOperation← $\xi[i+1]$
  else
    PreOperation← empty
  end if
else
  alert("the Object is empty! The deleting operation is error!")
end if

Algorithm 7 Undo Algorithm

while i < GO.length do
  if OperationNext (O) is not null **then**
    if isFind(GO, O) then
      if isRelation(OperationNext(O), O) **then**
        Undo(OperationNext(O))
      else
        Delete(GO, O)
        i++
      end if
    else
      alert("Do not find the Operation");
      Delete(GO, O)
    end if
end while

Algorithm 8 Undo Algorithm ConflictionResolution Algorithm

InStack(object, μ[n], ξ[n], t, ρ, i)
if $t_0 < t \le \rho_0$ **then**
  if μ[n] **is empty** & ξ[n] **is empty then**
    $\xi[i] \leftarrow 0$
  else if μ[n] **is not empty** & ξ[n] **is full then**
    $\xi[i] \leftarrow \mu[n]$
    $\mu[n-i] \leftarrow \mu[n-i-1]$;
    $\mu[0] \leftarrow 0$
  else if μ[n] **is empty** & ξ[n] **is full then**
    $\mu[i] \leftarrow 0$
  else
    alert(" the two buffer is full")
  end if
else if $t \le t_0$ **then**
  $\varphi[k] \leftarrow O_1, O_2 \cdots, O_k$
  if μ[n] **is empty** & ξ[n] **is empty then**
    $\xi[i, i+1, \cdots i+k-1] \leftarrow \varphi[1, 2, \cdots, k]$
  else if μ[n] **is not empty** &ξ[n] **is full then**
  $\xi[i, i+1, \cdots i+k-1] \leftarrow \mu[n-k+1, n-k+2, \cdots, n]$
      $\mu[n-i+1] \leftarrow \mu[n-k-i+1]$
      $\mu[i, i+1, \cdots, i+k-1] \leftarrow \varphi[1, 2, \cdots, k]$
  else if μ[n] **is empty** & ξ[n] **is full then**
      $\mu[i, i+1, \cdots, i+k-1] \leftarrow \varphi[1, 2, \cdots, k]$
  else
    alert(" the two buffer is full");
  end if
else
  alter("The operation has not been received, please senagain!")
end if
while isNotConflict(GO,O) **do**
  InStack(GO, μ[n], ξ[n], t, ρ, *i*--)
end while
while isConflict(GO,O) **do**
  if $O_i \| O$ **then**
    if $\partial \le \partial_0$ **then**
      InStack(GO, μ[n], ξ[n], t, ρ, *i*);
    else
      Delete(O);
    end if
  end if
end while

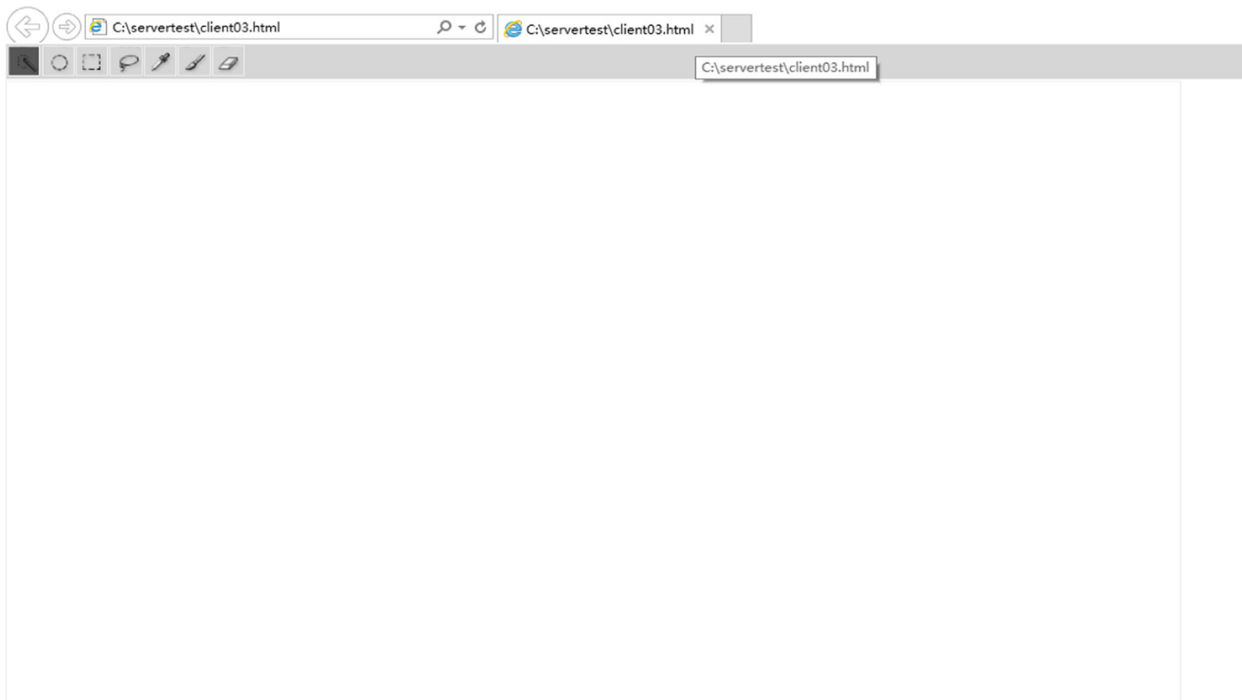

Fig. 8 Interface of graphic collaborative editing software

5 Realization of the key technology of consistency

5.1 Graphical collaborative editing software

Our study refers to the realization of the algorithm the common graphics collaborative editing algorithm (CGCE algorithm). This is done to perfect and expand not only the definition of graphics collaborative editing but also HTML5 Canvas, WebSocket, jQuery and Node.js and other network programming language and technology in order to realize the system of the key technology of graphics co-editing. The interface of graphic collaborative editing software is as shown in Fig. 8. As shown in Fig. 9, the main functions of the code are graphic drawing, including the drawing of arbitrary points, lines, and surfaces, and the drawing of some basic figures as shown in Fig. 10.

5.2 Experimental results

The network conditions of graphics collaborative editing and drawing are mainly composed of three modes:

1. 1. *The local mode*: This mode is used to open multiple pages on a computer. Multiple pages can be co-edited to the graphics, and each page in the open four local pages is like a drawing board and can be drawn on each page. The content they display on the pages is exactly the same.
2. *The LAN mode*: In the same LAN, two users can open graphics and edit software at the same time and also can produce the same effect, as shown in Fig. 11.
3. *The WAN mode*: The wide area network mode is more complex. For example, the high concurrency and packet loss cases in this paper are nearly all in WAN.

The consistency of graphics collaborative editing system on MacBook Pro is shown in Fig. 12. The consistency of graphics collaborative editing system on the Windows system is shown in Fig. 13.

Fig. 9 Graphic collaborative editing software code file

Fig. 10 Drawing graphics

5.3 Graphical collaborative editing software port monitoring

The server monitoring information of monitoring graphics collaborative editing is shown in Fig. 14. The server monitoring information of graphical collaborative editing can be used by each user to edit each operation of the graphics together and generate the corresponding log files.

There are no issues of high concurrency and packet loss in graphics collaborative editing software running on local and LAN networks, but there may be high concurrency or packet loss problems in WAN networks. If the routing path of two cooperative editors is long, the time delay is great. It is difficult to match the real-time situation; what's more, it is easy to lead to concurrent problems. However, the algorithm in the paper, which is called CGCE algorithm, is excellent and ensures that the graphics collaborative editing will be good. The cooperative multiuser operation of WAN networks is shown in Fig. 15.

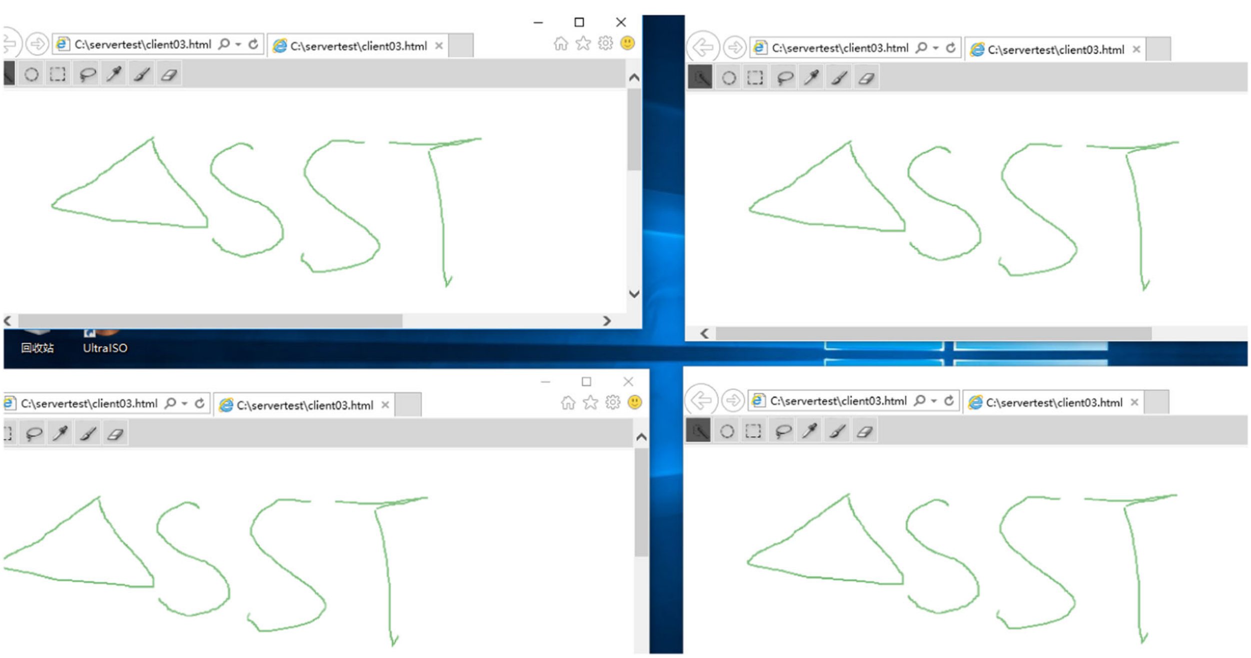

Fig. 11 Co-editing of local and LAN multiweb pages

Fig. 12 Graphics co-editing system on Macintosh

5.4 Analysis of time complexity and space complexity of the algorithm

The analysis of time complexity and space complexity of the algorithm.

(1) *Time complexity*: The time complexity of the best one is O_1. For example, the operations of inserting, deleting, rotating, and copying in the best case of graphical collaborative editing can be completed at one time. However, the worst is that all operations of multiusers exist concurrently, then the time complexity is $O(n^m)$, sufficient data from experiments shows the average time complexity of the algorithm is $O(n\log n)$.

(2) *Space complexity*: In order to maintain the normal operation of the algorithm, the experiments in this

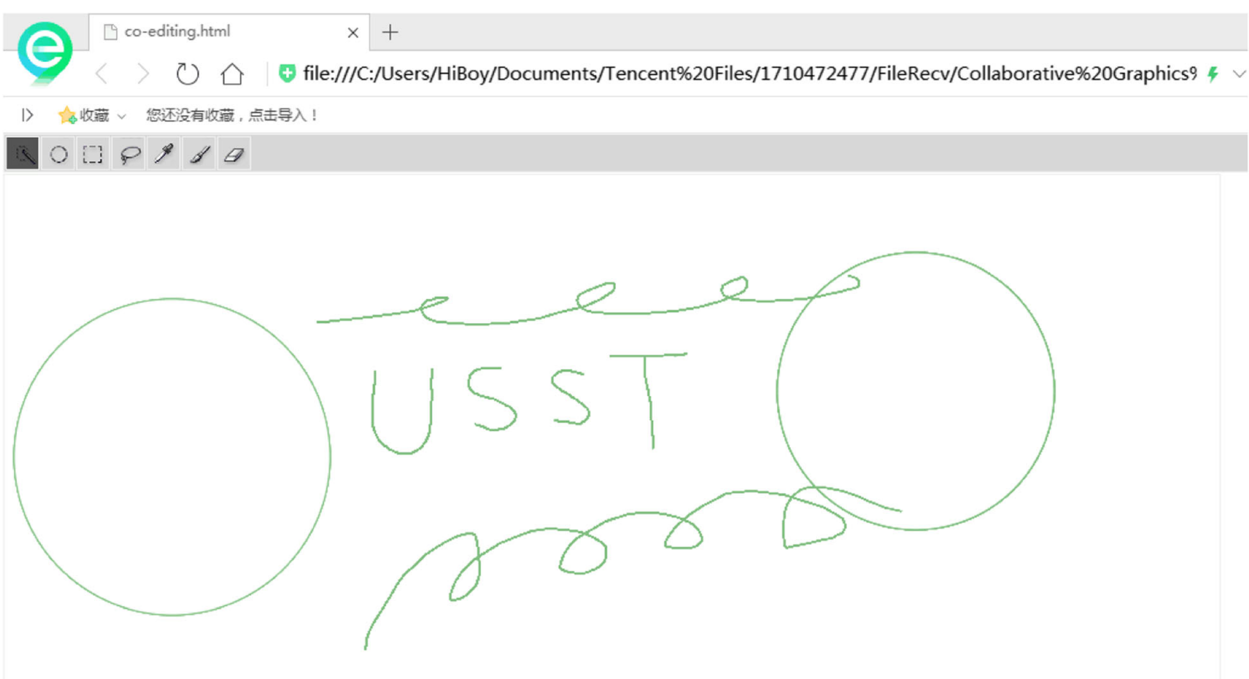

Fig. 13 Graphics co-editing system in Windows

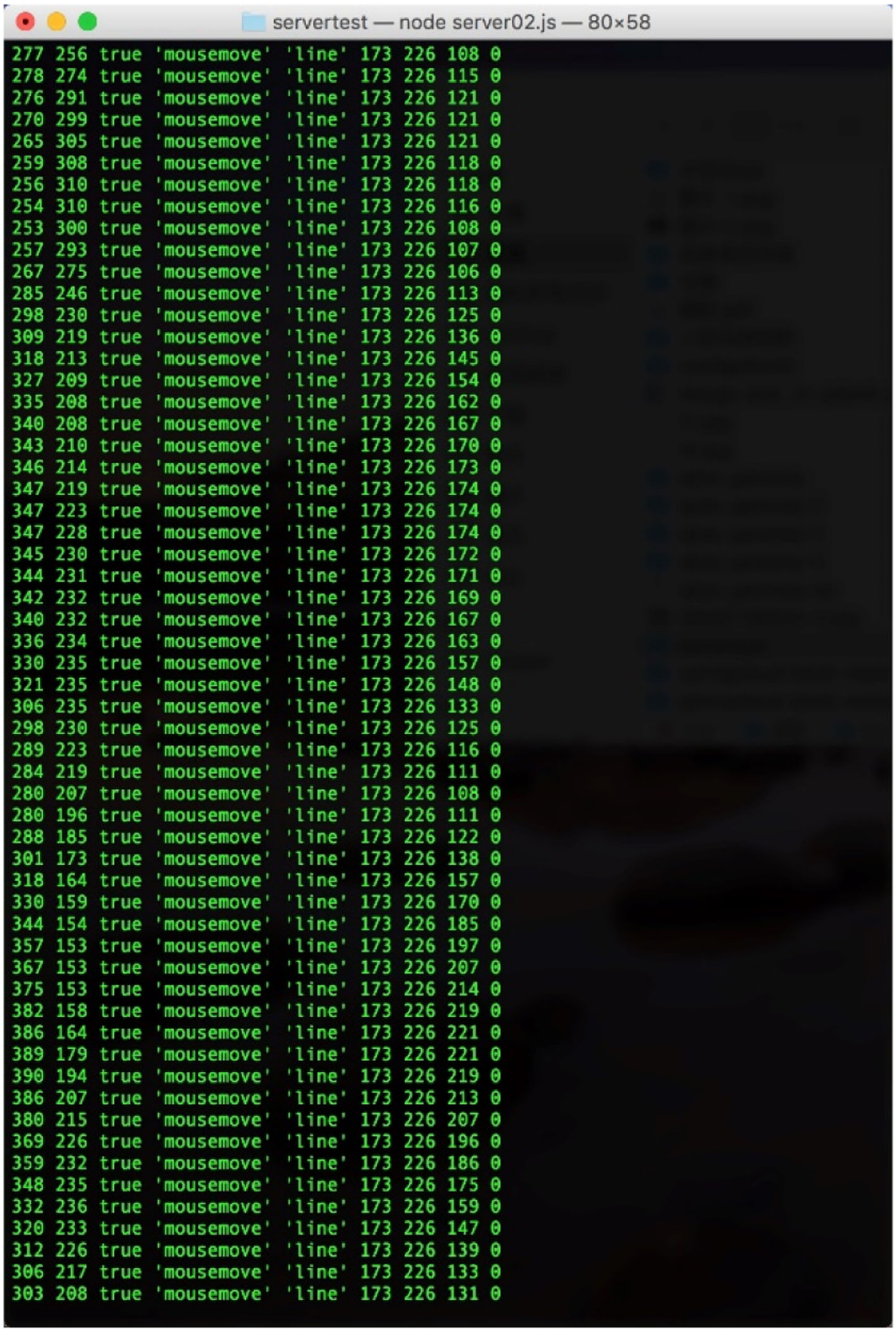

Fig. 14 Graphics cooperative editing server monitoring

paper at least open up three cache queues in the memory. If each queue has n unit space, the algorithm has a space complexity of $O(3n)$. For the queue, then the algorithm's spatial complexity is $O(nm)$. Assuming that there is a queue of length m, and then the required space complexity is $O(nm)$.

5.5 Time delay, packet loss, and bit error rate of software data transmission

This paper is used to test the software data transmission delay, packet loss rate, and bit error rate. The software is ATKKPING, and the network packet loss testing tool (ATKKPING) is a flat enhancement program network packet loss rate testing software. ATKKPING is mainly

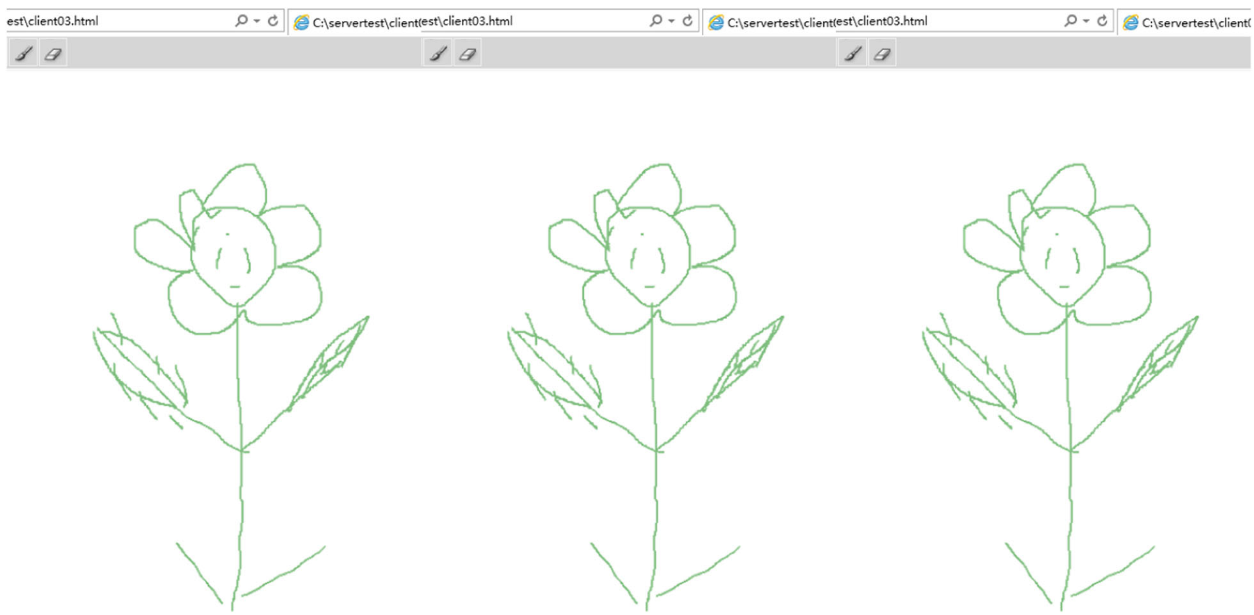

Fig. 15 Cooperative operation of multi-user in WAN

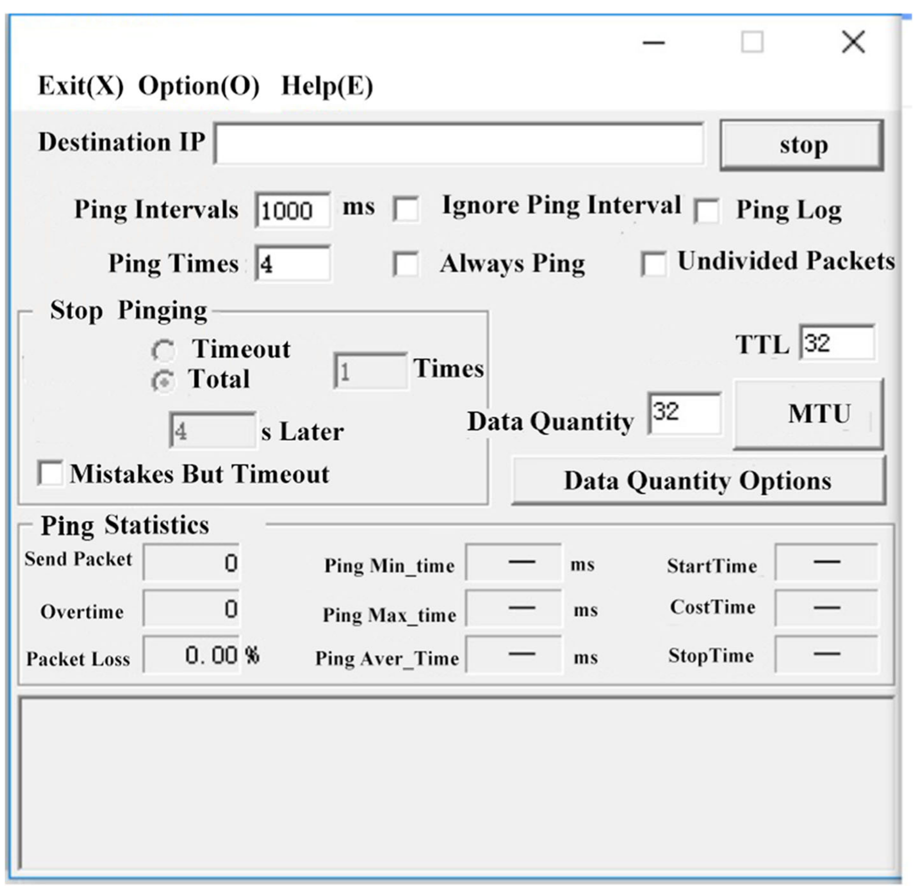

Fig. 16 Time delay, packet loss and bit error rate of software

used to perform packet loss test. You can test the packet loss situation of the intranet or extranet, thus providing an important reference for solving a series of packet loss and BER issues. Test the level of the network environment and how much is lost. The network drop test tool (ATKKPING) interface is shown in Fig. 16.

From the network test tool interface as shown in Fig. 16, we can know that, when the "Ping interval" is selected, the Ping interval is only used as a timeout, and Ping is performed fastest. If the data size value is set greater than the MTU value, the data packet must be partitioned. The MTU depends on the physical layer. Therefore, special attention should be paid to the size of the MTU, especially when pinging to the Internet. First, the relevant algorithm of this paper is not added in the graphical collaborative editing software of Fig. 17, and then the relevant parameters of the network testing software are configured. The "target host" is 192.168.1.163, the "ping interval" is 1 millisecond, "ping log" is selected, "Ping Times" is 4, and then click the "Start" button, as shown in Fig. 17.

From Fig. 18, we can know that the packet loss rate is 25% for the graphical collaboration software without CGCA algorithm in this paper. The minimum transmission delay is 2 milliseconds about data, the maximum value is 14 milliseconds, and the average value is 8 milliseconds. The packet loss rate is relatively much larger than post-CGCA Algorithm. Configure the same parameters in the network test tool and click the "Start" button. As shown in Fig. 16, the number of sent packets is 1659, 18 of which are timeouts; thus, the packet loss rate is 1.08%. The minimum transmission delay is 1 millisecond, the maximum delay is 214 milliseconds, and the average delay is 6.14 milliseconds. Through analysis of the data, it can be clearly known that the maximum delay of transmission is 214 milliseconds. If it exceeds 214 milliseconds, the packet will be lost. Above all, after the CGCE algorithm is added in the graphics co-editing software, the packet loss rate is reduced to 1.08%, which is nearly 24% lower than the comparison, and the average delay of data transmission has also been reduced.

6 Conclusions

Graphical collaborative editing plays an increasingly important role in CSCW. The most important technique in graphics co-editing is the consistency of

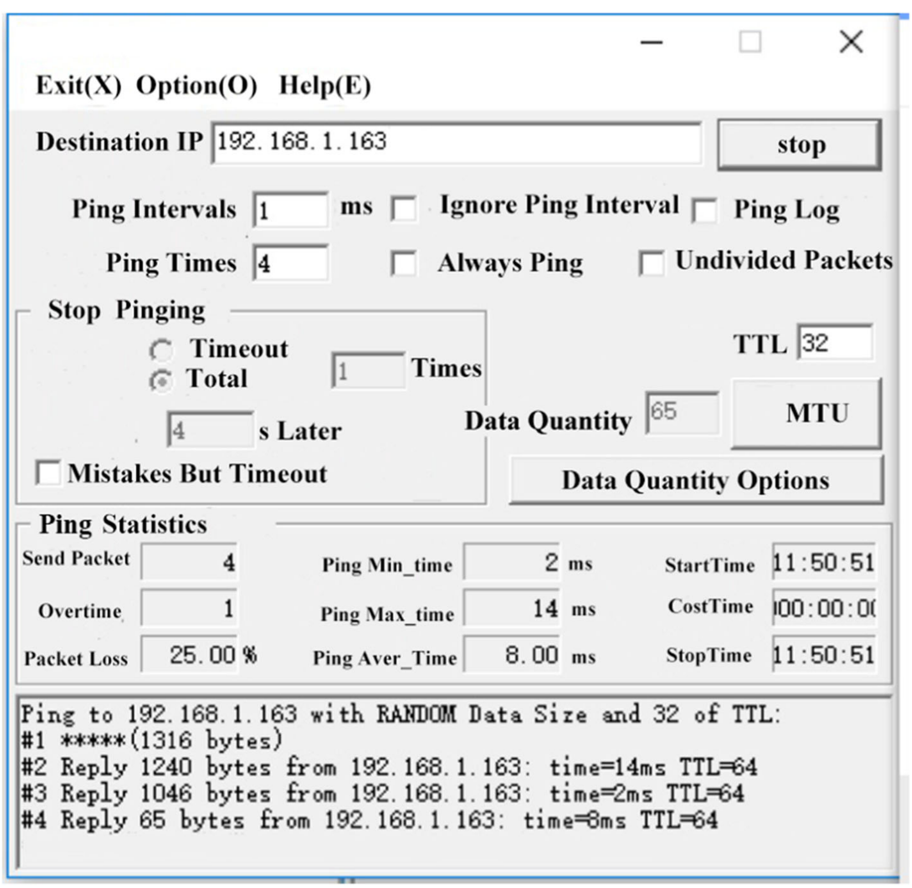

Fig. 17 Pre-CGCA testing

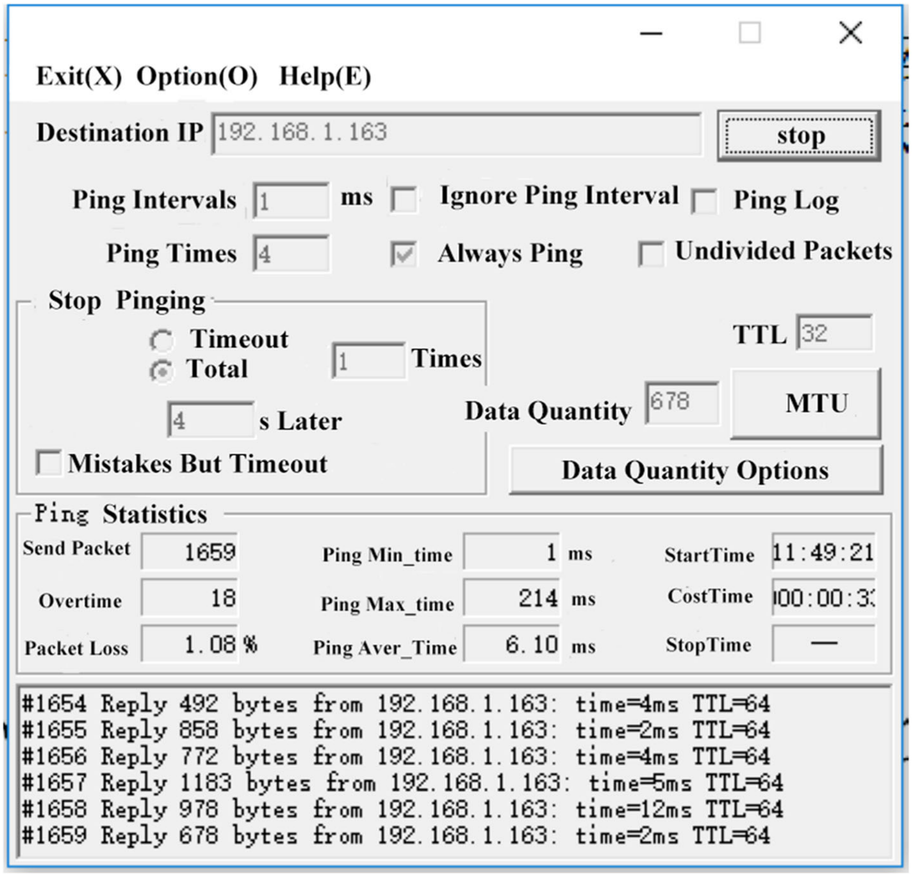

Fig. 18 Post-CGCA testing

graphics co-editing, which mainly includes causality consistency, consistency of results, and consistency of intention. In order to solve the consistency conflict problem of graph collaborative editing, the CGCE algorithm is proposed in this paper, and a large number of experiments show that this algorithm plays an irreplaceable role in performance optimization. The CGCE algorithm in this paper can solve the contradictions in the consistency of graphical collaborative editing, the research in this paper has particularity and results, and it will proved by the experiment. However, due to the relationship of time, the algorithm of the paper is more or less inadequate, whic is the key import that some scholars can point to as the shortcoming in the paper, especially the optimization and improvement of CGCE algorithm. A mathematical definition of some graphic cooperative editors can be easily followed. The scholars are going deeper and deeper. This paper has a strict mathematical definition of some basic operations of graphic collaborative editing, which can facilitate the follow-up scholars to carry out more in-depth and complex academic research.

Acknowledgements

The authors appreciate all anonymous reviewers for their insightful comments and constructive suggestions to polish this paper in high quality. This research was supported by the National Key Research and Development Program of China (No. 2018YFC0810204), National Natural Science Foundation of China (No. 61872242, 61502220), Shanghai Science and Technology Innovation Action Plan Project (17511107203, 16111107502) and Shanghai key lab of modern optical system.

Availability of data and materials

We can provide the data.

About the authors

Chunxue Wu (1964-) received the Ph.D. degree in Control Theory and Control Engineering from China University of mining and technology, Beijing, China, in 2006. He is a Professor with the Computer Science and Engineering and software engineering Division, School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, China. His research interests include, wireless sensor networks, distributed and embedded systems, wireless and mobile systems, networked control systems.
Langfeng Li is a student of University of Shanghai for Science and Technology, and E-mail is 1710472477@qq.com.
Changwei Peng (1994-) received the B.E. degree from Hubei Polytechnic University. HuangShi, Hubei, China, in 2013, He is currently pursuing the master degree in computer technology with the University of Shanghai for science and technology, China. His research interests include Mechanical learning and Data analysis.
Email: 740546437@qq.com
YAN WU is currently a postdoctoral associate at the school of public and environmental affairs, Indiana University Bloomington. He obtained his PhD

degree in Southern Illinois University Carbondale, with concentrations in environmental chemistry and ecotoxicology. His research involves elucidations of environmental fate of contaminants using chemical and computational techniques, as well as predictions of their associated effects on wildlife and public health. Data Processing and Analysis in Environmental Related Fields. E-mail is wuyan8910@126.com and ORCID is 0000-0001-7876-261X

Naixue Xiong is currently an Associate Professor (3rd year) at School of Computer Science and Technology, Tianjin University, Tianjin, China, 300350. He received his both PhD degrees in Wuhan University (about software engineering), and Japan Advanced Institute of Science and Technology (about dependable networks), respectively. Before he attends Colorado Technical University, he worked in Wentworth Technology Institution, Georgia State University for many years. His research interests include Cloud Computing, Security and Dependability, Parallel and Distributed Computing, Networks, and Optimization Theory. Dr./Prof. Xiong published over 100 international journal papers and over 100 international conference papers. Some of his works were published in IEEE JSAC, IEEE or ACM transactions, ACM Sigcomm workshop, IEEE INFOCOM, ICDCS, and IPDPS. He has been a General Chair, Program Chair, Publicity Chair, PC member and OC member of over 100 international conferences, and as a reviewer of about 100 international journals, including IEEE JSAC, IEEE SMC (Park: A/B/C), IEEE Transactions on Communications, IEEE Transactions on Mobile Computing, IEEE Trans. on Parallel and Distributed Systems. He is serving as an Editor-in-Chief, Associate editor or Editor member for over 10 international journals (including Associate Editor for IEEE Tran. on Systems, Man & Cybernetics: Systems, and Editor-in-Chief for Journal of Parallel & Cloud Computing (PCC)), and a guest editor for over 10 international journals, including Sensor Journal, WINET and MONET.

Prof. Changhoon Lee, Department of Computer Science and Engineering, Seoul National University of Science and Technology (SeoulTech), Republic of Kornbea. Email: cryptograph1@gmail.com

Authors' contributions
CW conceived and designed the study. LL, CP, YW, NX and CL participated in experiments and data processing and edited the manuscript. All authors read and approved the final manuscript.

Competing interests
The authors declare that they have no competing interests.

Publisher's Note

Author details
[1]School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China. [2]School of Public and Environmental Affairs, Indiana University, Bloomington, IN 47405, USA. [3]College of Intelligence and Computing, Tianjin University, Tianjin 300350, China. [4]Department of Computer Science and Engineering, Seoul National University of Science and Technology (SeoulTech), Seoul, Republic of Korea.

Received: 26 September 2018 Accepted: 13 December 2018
Published online: 04 March 2019

References
1. C.L. Ignat, M.C. Norrie, in *CSCW '04: Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work*. Grouping in collaborative graphical editors (Association for Computing Machinery, New York, 2004), pp. 447–456
2. Z. Wu et al., Alleviating network congestion for HPC clusters with fat-tree interconnection leveraging software-defined networking. In: 3rd International Conference on Systems and Informatics (ICSAI), Shanghai, China, 19–21 Nov 2016 (2017)
3. L.H. Dang et al., in *International Electronic Elements*. New technology of high-speed and ultra long haul fiber-optic communication system (2008)
4. J. Hillebrand et al., in *International Symposium on Industrial Embedded Systems*. Coexistence of time-triggered and event-triggered traffic in switched full-duplex ethernet networks (2007)
5. X. Liu, S. Zhao, A. Liu, N. N. Xiong, A. V. Vasilakos, *Knowledge-aware proactive nodes selection approach for energy management in Internet of Things, Future generation computer systems,*. https://doi.org/10.1016/j.future.2017.07.022. Accessed 5 Aug 2017.
6. P.H. Weibel, D. Béchaz, *IEEE 1588 implementation and performance of time stamping techniques* (2004)
7. P. Loschmidt et al., *IEEE 1588 hardware for fault tolerance and high precision* (2005), pp. 323–332
8. F.Y. Jiang, H.C. Duan, in *International Symposium on Information Technology in Medicine and Education*. Application research of WebSocket technology on Web tree component (2012)
9. V. Wang et al., *The WebSocket API* (2013), pp. 13–32
10. R. He, N. Xiong, L.T. Yang, J.H. Park, Usingmulti-modal semantic association rules to fuse keywords and visual features automatically for web image retrieval. Information Fusion. **12**(3), 223–230 (2011)
11. H. Liu et al., Consistency maintenance in collaborative CAD systems. Chin. J. Electron. **22**(1), 15–20 (2013)
12. C. Sun et al., A generic operation transformation scheme for consistency maintenance in real-time cooperative editing systems. Leadersh. Org. Dev. J. **35**(35), 425–434 (1997)
13. Z. Wan, N.N. Xiong, N. Ghani, A. Vasilakos, L. Zhou, *Adaptive unequal protection for wireless video transmission over IEEE 802.11 e networks, Multimedia Tools and Applications*. **72**(1), 541–571 (2014)
14. M.N. Boulos et al., Web GIS in practice VIII: HTML5 and the canvas element for interactive online mapping. Int. J. Health Geogr. **9**, 1 (2010)
15. S.A. Asghar et al., An open source modelica graphic editor integrated with electronic notebooks and interactive simulation (2011)
16. K.D. Volder, in *International Symposium on Practical Aspects of Declarative Languages*. JQuery: A generic code browser with a declarative configuration language (2006)
17. S. Tilkov, S. Vinoski, Node.js: using JavaScript to build high-performance network programs. IEEE Internet Comput. **14**(6), 80–83 (2010)
18. T. Bellmann, in *Modelica Conference*. Interactive simulations and advanced visualization with modelica (2009)
19. G.P. Liu et al., Design and stability criteria of networked predictive control systems with random network delay in the feedback channel. IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. **37**(2), 173–184 (2007)
20. S. Xia et al., Collaborative object grouping in graphics editing systems (2005)
21. L. Chen et al., A negotiation methodology for multidisciplinary collaborative product design. Adv. Eng. Inform. **28**(4), 469–478 (2014)
22. X. Xu et al., in *IEEE International Conference on Systems, Man and Cybernetics*. Consistency maintenance in real-time collaborative image editing systems (2003)
23. M. Ressel, D. Nitsche-Ruhland, in *ACM Conference on Computer Supported Cooperative Work*. An integrating, transformation-oriented approach to concurrency control and undo in group editors (1996)
24. S Xi and C. Wu, Super resolution reconstruction algorithm of video image based on deep self encoding learning. Multimed Tools Appl., 2018: 1–18 (2018)
25. C. Liu, A. Zhou, C. Wu, Image segmentation framework based on multiple feature spaces. IET Image Process. **9**, 271–279 (2015)
26. H. Chen, C. Wu, W. Huang, Design and application of system with dual-control of water and electricity based on wireless sensor network and video recognition technology. Int. J. Distrib. Sens. Netw. **14**(9), 1550147718795951 (2018)
27. X. Xue, C. Wu, Z. Sun, Y. Wu, *Vegetation greening for winter oblique photography using cycle-consistence adversarial networks. Symmetry* (2018), pp. 1–14. https://doi.org/10.3390/syml0070294
28. C. Wu, J. Du, Y. Liu, An energy balancing routing algorithm for mobile sensor networks. Sens. Lett. **11**, 1783–1791 (2013)
29. H. Li, X. Mao, C. Chun, F. Yang, Design and analysis of a general data evaluation system based on social networks. EURASIP J. Wirel. Commun. Netw. **2018**, 109 (2018)
30. L. Gao, F. Yu, Consistency maintenance of group/ungroup operations in object-based collaborative graphical editing systems. Int. J. Signal Process. Image Process. Pattern Recogn. **8**(9), 59–68 (2015)
31. S. Greenberg et al., in *Hawaii International Conference on System Sciences*. Issues and experiences designing and implementing two group drawing tools (1992)
32. H Gu et al., AnchorMF: towards effective event context identification. pp. 629–638 (2013)
33. C.L. Ignat, M.C. Norrie, in *ACM Conference on Computer Supported Cooperative Work*. Grouping in collaborative graphical editors (2004)
34. B. Jiang et al., Semantic consistency maintenance in collaborative graphics design systems (2008)

35. C. Liang, Dynamic coordination principle and method of multidisciplinary collaborative product design based on semantics reasoning. Trans. Chin. Soc. Agric. Machinery **42**(5), 208–213 (2011)
36. C. Ramstein, in *Conference of the Centre for Advanced Studies on Collaborative Research, October 24–28, 1993, Toronto, Ontario, Canada, 2 Volumes*. COOPDRAW: a multiagent architecture for a shared graphical editor (1993)
37. J.X. Wang et al., Design and implementation of an agent-based collaborative product design system. Comput. Ind. **60**(7), 520–535 (2009)
38. H. Si, W. Sun, J. Zhang, J. Wan, N.N. Xiong, L. Zhou, Y. Ren, An Effective Identification Technology for Online News Comment Spammers in Internet Media, IEEE Access, Digital Object Identifier: https://doi.org/10.1109/ACCESS.2019.2900474, On page(s): 1–3, Print ISSN: 2169-3536, Online ISSN: 2169–3536
39. W. Wei, Consistency maintenance of multidisciplinary collaborative product design. China Mech. Eng. **21**, 12 (2010)
40. P. Wu, F. Xiao, C. Sha, H. Huang, R. Wang, N. Xiong, Node scheduling strategies for achieving full-view area coverage in camera sensor networks. Sensors **17**(6), 1303 ((2017). https://doi.org/10.3390/s17061303
41. Y. Wang, K. Chen, J. Yu, N.X. Xiong, H. Leung, H. Zhou, L. Zhu, Dynamic propagation characteristics estimation and tracking based on an EM-EKF algorithm in time-variant MIMO channel. Inform. Sci. **408**, 70–83 (2017)
42. J. Gui, L. Hui, N.X. Xiong, A game-based localized multi-objective topology control scheme in heterogeneous wireless networks. IEEE Access **5**(1), 2396–2416 (2017)
43. H. Zhang, R.W. Liu, D. Wu, Y. Liu, N.N. Xiong, Non-convex total generalized variation with spatially adaptive regularization parameters for edge-preserving image restoration. J. Internet Technol. **17**(7), 1391–1403 (2016)
44. Z. Xia, X. Wang, X. Sun, Q. Liu, N. Xiong, Steganalysis of LSB matching using differences between nonadjacent pixels. Multimed. Tools Appl. **75**(4), 1947–1962 (2016)
45. Z. Xia, N.N. Xiong, A.V. Vasilakos, X. Sun, EPCBIR: an efficient and privacy-preserving content-based image retrieval scheme in cloud computing. Inform. Sci. **387**, 195–204 (2017). https://doi.org/10.1016/j.ins.2016.12.030
46. Z. Lu, Y.-R. Lin, X. Huang, N. Xiong, Z. Fang, Video topic discovering, tracking and summarization from social media streams. Multimed. Tools Appl. **76**(8), 10855–10879 (2017)
47. L. Shu, Y. Fang, Z. Fang, Y. Yang, F. Fei, N. Xiong, A novel objective quality assessment for super-resolution images. Int. J. Signal Process. Image Process. Pattern Recogn. **9**(5), 297–308 (2016)
48. W. Fang, Y. Li, H. Zhang, N. Xiong, J. Lai, A.V. Vasilakos, On the throughput-energy tradeoff for data transmission between cloud and mobile devices. Inf. Sci. **283**, 79–93 (2014)
49. N. Xiong, A.V. Vasilakos, L.T. Yang, L. Song, Y. Pan, R. Kannan, Y. Li, Comparative analysis of quality of service and memory usage for adaptive failure detectors in healthcare systems. IEEE J. Sel. Areas Commun. **27**(4), 495–509 (2009)
50. Y. Fang, Z. Fang, F. Yuan, Y. Yang, S. Yang, N.N. Xiong, Optimized multi-operator image retargeting based on perceptual similarity measure. IEEE Trans. Syst. Man Cybern. Syst. **99**, 1–11 (2016)
51. X. Lu, S. Chen, N. Xiong, ViMediaNet: an emulation system for interactive multimedia based telepresence services. J. Supercomput. **73**, 3562–3578 (2017)
52. C. Zhang, D. Wu, R.W. Liu, N. Xiong, Non-local regularized variational model for image deblurring under mixed gaussian-impulse noise. J. Internet Technol. **16**(7), 1301–1320 (2015)
53. Y. Yang, S. Tong, S. Huang, P. Lin, Dual-tree complex wavelet transform and image block residual-based multi-focus image fusion in visual sensor networks. Sensors **14**(12), 22408–22430 (2014)
54. N. Xiong, R.W. Liu, M. Liang, W. D., Z. Liu, W. H., Effective alternating direction optimization methods for sparsity-constrained blind image deblurring. Sensors **17**(1), 174 (2017). https://doi.org/10.3390/s17010174
55. L. Gao, F. Yu, Q. Chen, N. Xiong, Consistency maintenance of do and undo/redo operations in real-time collaborative bitmap editing systems. Clust. Comput. **19**(1), 255–267 (2016)