Research on the Function Consistency of the Network Element based on Atomic Capability Architecture

Jianqing Li
Institute of Networks and Communication Engineering, Hangzhou 310018, P. R. China
Email: lijianqing@pop.zjgsu.edu.cn

Weiming Wang 1, Chuanhuang Li 1, and Julong Lan 2
1. Institute of Networks and Communication Engineering, Hangzhou 310018, P. R. China
Email: {wmwang, chuanhuang_li}@mail.zjgsu.edu.cn
2. National Digital Switching System Engineering & Technological Research Center, Zhengzhou 450002, P. R. China

Abstract—When the functions of a network element (such as router, switch, etc.) based on Atomic Capability (ATC) architecture are reconstructed dynamically, the topology of ATC will be changed. The process of this reconstruction usually results in the fact that the ATCs being reconfigured cannot process data, and a large parts of the system services suspended and packets discarded. To solve this problem, from a theoretical point of view and under the definition of ‘ATC port status’ and ‘ATC work status’, two different methods to guarantee the consistency of the system during dynamic reconstruction are presented. Meanwhile, the issues of data packet loss and latency are analyzed through some examples. And the result shows that the methods can reduce data packet loss and latency during dynamic reconstruction.

Index Terms—Dynamic Reconstruction; Consistency; Atomic Capability; Open Programmable

I. INTRODUCTION

Today’s network relies on the basis interconnected transmission capacity which constrains the entire information network. On the other hand, the inherent ability and structure of the network have poor adaptability to business needs. It not only results in low capacity in integration, ubiquitous, quality, safety, expandability, management and controllability, mobility and so on, but also makes the gap between business requirements and network infrastructure capabilities increasingly obvious. To solve this problem, institutions both at home and abroad present their future network development plans. Stanford Clean Slate [1] Study Group proposed a new network architecture SDN(Software Defined Network). Its core technology is OpenFlow that divides the network devices into the control plane and data plane, enabling flexible control of network traffic. Based on the OpenFlow technologies it gives businesses and consumers a more flexible network management and control, a more efficient resource utilization and a more flexible resource scheduling [2]. In China, 973 project group proposed a "Flexible Architecture of Reconfigurable Infrastructure (FARI) [3-5]", its core idea is creation of Atomic Capability network theory which can establish reconfigurable and scalable infrastructure network. It makes the internal functional structure of the network adjust businesses’s characteristics and requirements adaptively, and realize effective configuration to the varied businesses.

Basing on Atomic Capability network theory, the network is divided into the business layer-Atomic, Service Layer-Atomic and Capability layer-Atomic, shown in Figure 1. According to the specific implementation of Atomic Service, Atomic Capability may be queue space, scheduling, routing protocols, encryption algorithms etc. Its physical distribution, Atomic Capabilities’ connection relations and inherent attributes constitute the system configuration. Configuring system when system is running is called dynamic configuration.

For long-running business-critical systems, such as satellite positioning and navigation systems, aviation navigation systems, financial systems and national communications infrastructure spin, etc. In terms of the perspective for the interests of national defense or national interests, it is intolerable to interrupt the services
provided by these systems. Therefore, in order to meet changes of customers’ needs, adapt to the complex and ever-changing environment and achieve QoS control, these systems can use the technology of dynamic configuration in the process of long-term continuous operation, to realize the Atomic Capabilities’ online evolution, specifically including Atomic Capabilities’ updates, Atomic Capabilities’ migration for load balancing. If there is an Atomic Capability removed in the topology, but the system has no effective mechanism to record the new status of Atomic Capability which results in packet loss during packet transmission. I call this behavior—“the damage of system consistency [6]” that the system record the state of Atomic Capability is inconsistent with the actual Atomic Capability’s behavior because of Atomic Capability configuration.

Based on the above background, this paper will study about the problem of the loss and delay of packet when the consistency of the system is destroyed. The service provided by the network element topology will be forced to suspend during the process of reconstruction, but if packets received continuously that will lead to the loss and delay of packets. This paper aims to design a method based on network elements ensuring system consistency. It will avoid packet loss or delay when system consistency is destroyed.

II. RELEVANT WORK

In the domestic there is some research on reconfigurable network, such as PLA Information Engineering University. They studied an open reconfigurable routing and switching nodes platform [7] based on "Structural Unit-Component-Platform", and its core strategy is web hosting service. Reconfigurable routing and switching platform is key supported technology. At the same time, they designed a service-oriented network architecture: Universal Carrying Network (UCN) model [8]. Tsinghua University proposed a reconfigurable, service-oriented network model [9]. National Defense University proposed a reconfigurable router control software model based on virtualization technology, which can provide the same operating environment for different components through the kernel virtualization technology of operating system. Therefore it increases the openness and security of the entire reconfigurable router controlled software system [10]. Zhejiang Gongshang University studied mapping service of reconfigurable routing node, and proposed a universal service model ForCES. It contains four elements: role, protocol, quality and control [11]. In 2009, ZTE launched a "modular components, software reconfigurable" router ZXR10 M6000.

On research of system consistency, Jeff Kramer and Jeff Magee [12] from Imperial College London thought that the target component must have no interactive relationship with the other components during dynamic configuration. The process of full interaction between components is defined as transaction. That components cannot start a transaction but just can participate in the transaction is defined as a passive state. If components can initiate the transaction with the target component, then, these components and the target component itself will be driven into a passive state, before dynamic configuration of the target component. So that the target component turns to the resting state that it will not participate in any transactions and start any transaction, which is called safe state of Jeff method.

System assertion may exist in application system, which restrains a group of members’ state. For example, the system is based on token ring structure that all components can only have a token requirement that is such assertion. Dynamic configuration cannot destroy this constraint of global state, which is named application state consistency [13-14] by Goudarzi from Imperial College London. When a component is joining or leaving the system, it will call ‘link’ or ‘unlink’ method achieved by required components to interact with other components and set the status of target component and others through which to ensure the consistency of application state.

WebFrame is a J2EE server which is developed by Institute of Software Chinese Academy of Sciences independently. It achieves dynamic migration of EJB components, and ensures citation consistency, status consistency and message consistency [15] in this process. Citation consistency means solving the problems of referential integrity in the process of dynamic configuration. Message consistency means that the components are not dealing with data or waiting for other components’ requests when system deletes components. Message consistency guarantees requests from clients will not be lost because of component migration. Status Consistency highlights the component states during the component migration correctly.

It may call the target components before and after migration old and new target ones respectively. When a component is going to migrate, WebFrame will create a new target component firstly, and wait until the old target component completes request which is processed by the old target component. After that system forwards the request to the new target component, and deletes old target components to complete the component migration. It ensures consistency of message at last. In this process, to ensure reference consistency and state consistency, JNDI’s name rebinding, component state transferring and other operations also will be put into effect.

To ensure behavior consistency of components during dynamic evolution and avoid illegal evolution, Ma Chuan and Shen Limin [16] proposed a component model based on process algebra. It introduces initiate request port and internal connection into the model to formally describe the components and their interaction protocol. This component model can verify the component interface compatibility, exterior interactive behavioral consistency and run-time behavioral integrity in dynamic evolution.

Zhou Xueyao [17] from Guangxi University used ADarwin language which extends Darwin, to describe the aspect-oriented dynamic software architecture (AOA). And basing on semantic foundation Pi-calculus and
ADarwin, he proposed the semantic model of AOA to verify consistency.

III. BEHAVIORAL CONSISTENCY OF ATOMIC CAPABILITY’S DYNAMIC RECONSTRUCTION

This chapter is going to study the consistency of Atomic Capability’s dynamic reconstruction (ATC is short for Atomic Capability). The main part studies about the definitions of ATC’s concept and its dynamic reconstruction ensures consistency of global guaranteed methods. I will focus on the related definitions and assumptions of the dynamic reconstruction for ATC, as preparations for studying ATC global guaranteed methods.

Before the study of dynamic reconstruction method, we need to define the behavior of ATC. ATC as a fine-grained software module, whose behavior is not closed, but needs to constantly communicate with the outside world. There are two types of ATC behavior--mutual behavior (communications between ATCs) and the local behavior (communication with local resources).

1) Mutual behavior: Entity of communications between ATCs in the system is actually a packet object. Packet objects after each ATC’s processing will forward to the next ATC along the ATC topology. The behavior between ATCs is called Mutual behavior. For a simple example, in a switching process, the packet will be processed by four steps: checking framing and buffer packet(ATC1), checking router table(ATC2), re-encapsulation layer 2 header(ATC3) and forwarding from one local interface(ATC4), shown in Figure 2.

![Figure 2. Mutual behavior](image)

2) Local Behavior: ATC communicates with local resources including reading and writing the local hard disk and system memory. For example, ATC1’s capability is checking router table; however, it doesn’t have the router table (recorded in local router memory in physical layer). Therefore ATC1 needs to ask for router request to local router, shown in Figure 3.

![Figure 3. Local behavior](image)

When ATC actively triggers a behavior, the rest ATCs in the topology are triggered by a certain sequence, until all of the ATCs are triggered, such a process in the dynamic reconfigurable system is defined as a transaction.

3) ATC behavioral consistency: The status of ATC recorded in the system is same with the actual status of ATC.

There are a lot of behavior when dynamic reconfigurable systems run, including the local behavior and the mutual behavior. Any interruptions may change the status of ATC and break off the system consistency. Therefore, to ensure ATC behavioral consistency, the system must know the new status of ATC when interruptions occur and rewrite the status of ATC in the system status record.

IV. RESEARCH ON METHODS OF ATOMIC CAPABILITY BEHAVIORAL CONSISTENCY

Through the introduction of behavioral consistency and its related definitions, this chapter focuses on methods of ATC behavioral consistency.

A. Preliminary Works

To ensure the ATC behavioral consistency when ATC reconfiguration occurs, the system should timely get the status of relevant ATC, rewrite the system status record and process the relevant data. Hereinafter we will define the status of ATC and related properties.

1) Definition 1 ATC Role: According to the location of ATC in the topology, ATC is defined as two kinds of roles. They are head ATC and retinue ATC. When the ATC can initiate a transaction that means it locates at the data input port of topology, it is called the head ATC; Remninder ATC in the topology is called retinue ATC. In a network node, there is a Memory ATC that every ATC has an output port connected to it, shown in Figure 4.

![Figure 4. Memory ATC](image)

2) Definition 2 ATC topology Processing Time: The average processing time that a packet through the entire topology is called topology processing time.

3) Definition 3 ATC Port Status: Status is used to reflect the current ATC port status in the process of reconfiguration. The status of the ATC input port is divided into ATC resting port status and ATC receiving port status; The status of the ATC output port is divided into ATC normal forwarding port status and ATC forwarding and storing port status. The ATC resting port indicates that the port doesn’t receive any packets during an ATC topology processing time, then the status of ATC port will turn to resting; The ATC receiving port state indicates that the input port is receiving data; The ATC normal forwarding port state means the output port sends...
the packets which has been processed to the next ATC in the topology; The ATC forwarding and storing port status indicates the output port sends the packets which has been processed to the memory ATC instead to the next ATC.

The status of ATC port determines the status of ATC.

4) Definition 4 ATC status: When the target ATC request reconstruction, and all input ports of target ATC are into the ATC resting port status, the target ATC will turn into reconfigurable status. If there is one input port’s status is receiving port state, then the status of ATC is working status.

B. ATC Topology Systems:

1) a four-tuple ⟨S, I, R, IC⟩: ATC connection interface as follows:
   - S (Send Packet Port Set): The set of the ATC output ports.
   - I (Initiate Request Port Set): The set of the head ATC input ports. Apparently, When ATC is not the head ATC, I = ∅.
   - R (Receive Packet Port Set): The set of the ATC receive packet ports.
   - IC (Implicit Connection): The connections between output ports and input ports in the inner of ATC, IC ∈ R × S. Such as ATC receives package from port p, then sent out from the port q, where p ∈ R, q ∈ S ∥ s < p, q > ∈ IC.

2) a two-tuple ⟨N, EC⟩: The application system which is Connected together by the ATC as follows:
   - N: The set of the ATC.
   - EC (Explicit Connection): The connections between different ATCs’ output ports and input ports. EC ∈ ∪ S × ∪ R, where Sn represents the set of ATC_n’s output ports, Rn represents the set of ATC_m’s input ports. Such as ATC_n directly through the output port p send data packets to ATC_m input port q when n, m ∈ N and n ≠ p, p ∈ Rn, q ∈ Sn, then s, q > ∈ EC.

3) Definition 5 Connection on structural level: If there is a line ⟨s1, r2, s2, ………, sm−1, r_m⟩ which is made up by ATCs’ ports between ATC_A’s output port p and ATC_B’s input port q, and the following conditions are met, then we call the port p connecting to port q on structural level, ATC_A connects to ATC_B on structural level:
   - m > 1
   - p = s1, sl ∈ SA
   - q = rm, rm ∈ RB
   - ∃ j ∈ {1, ..., m−1} such that s_j, r_j+1 > ∈ EC

4) Definition 6 Connection on semantic level: If ATC_A’s output port p connects to ATC_B’s input port q on structural level, and the line⟨s1, r2, s2, ………, sm−1, r_m⟩ between port p and port q satisfies the following conditions, then the port p connects to the port q on semantic level, ATC_A connects to ATC_B on semantic level:
   - m = 2, or m > 2 and ∀ j ∈ {2, ..., m−1} ∀ n ∈ N < r_j, s_j, > ∈ IC.

In this paper, RPS (A) is ATC_A’s Resting Port Set which is made up by the ATC_A’s input ports. RRPS (A) is the Response Resting Port Set, which is made up by other ATCs’ output ports that connect with the target ATC’s input ports. RS (A) is the Reconfigurable ATC Set which is made up by the target ATC and head ATC that connect to target ATC on semantic level.

5) Definition 7: The ports in RPS are relative to the target of ATC (RRPS cannot send data to the target ATC), but the ports can send data to the Memory ATC. When remodeling occurs, the data that is sent to the target ATC will dump to a Memory ATC.

In addition to individual object in dynamic configuration, but also can be multiple ATCs. For the TS (target ATC set), RRPS(TS) = ∪ iTS RRPS(i), RPS(TS) = ∪ iTS RPS(i), only when all of ports in RRPS (TS) have entered ATC forwarding and storing port status, after some time, RPS (TS) turn into the ATC resting port status. Then the the target ATCs turn into reconfigurable status.

C. Theorem:

1) When other ATCs’ output ports which connect to the target ATC input ports all be driven to the ATC forwarding and storing port status, after an ATC topology processing time, the target ATC turns into reconfigurable status.

Proof: When other ATCs’ output ports which connect to the target ATC’s input ports all be driven to the ATC forwarding and storing port status, after an ATC topology processing time, there are no data into the target ATC’s input ports. By definition 3 all of the target ATC’s input ports come into the ATC resting port status, and by definition 4 the target ATC comes into the reconfigurable status.

2) When the head ATC’s input ports are driven to the ATC resting port status, after an ATC topology processing time, the target ATC connect to the head ATCs on semantic level turns into reconfigurable status.

Proof: The input ports of head ATC which connect to the target ATC on semantic level provide the source of data in this link, when head ATC’s input ports turn into the ATC resting port status, there is no new data enter this link. After an ATC topology processing time, the old original data on this link have been processed. And after another ATC topology processing time, according to the Definition 3 the target ATC ports turn into the ATC resting port status. Because of the reconstruction request has been issued, according to Definition 4, the target ATC into reconfigurable status.

D. System Hypothesis:

1) All transactions must be completed within a limited time.

If this condition is not satisfied in reconfigurable system, the system can not provide the timing for the
implementation of dynamic configuration. Because the transaction doesn’t end while ATC maybe enter the unlimited waiting and thus unable to complete any dynamic configuration.

2) All transactions must be completed independently, there is no wait or dependencies between transactions.

Each transaction is initiated by the head ATC for the completion of a data packet processed. If there are waiting or dependencies between transactions the in the dynamic reconfiguration process, the system may deadlock or die into an infinite wait status. Therefore each transaction is independent, each packet processing does not rely on other packet processing results.

E. ATC Consistency Preservation Methods During Reconstruction Process

In the above argument, we find that ATC port-based reconstruction has a good reaction about ATC status and promptly informs the system about the current ATC status to avoid system errors caused by the changes of ATC status. But in the actual implementation process, there may exist the port state can not be changed, leading to the target ATC’s port connects the port can not enter the ATC resting port status, which the target ATC cannot enter the reconfigurable status. Therefore, here we will divide ATC consistency preservation methods into the ATC port-based consistency preservation method and the ATC-based consistency preservation method.

1) ATC port-based consistency preservation method

In the ATC’s reconstruction process, in order to real-time feedback about ATC status information, we set up steps of ATC port-based consistency preservation method.

The method is as follows:

1. Knowing the target ATC from the system requests.
2. Invoking the target ATC’s four-tuple information from system records, thus obtaining the target ATC’s RPS.
3. Knowing the other ATCs’ two-tuple information from system records where the other ATCs’ ports connect to the RPS.
4. System’s Command drives RRPS come to the ATC forwarding and storing port status.
5. According to Theorem 1, we learned that all of RRPS enter the ATC forwarding and storing port status, after an ATC topology processing time, the target ATC turns into reconfigurable status. By Definition 7, we know the data RRPS send to the target ATC will dump to the Memory ATC.
7. After reconstruction, firstly, driving the target ATC’s input port set RPS into the ATC receiving port status, system records new status of the target ATC (working status). The data stored in Memory ATC forwards to the target ATC. Secondly, driving RRPS into the ATC normal forwarding port status.

According to the ATC port-based consistency preservation method, we reconstruct the following example.

Shown in Figure 5, if the system is turning to reconstruct the ATC_D, firstly, finding ATC_D’s four-tuple information:

\[ \text{ATC}_D = \{(\text{Send}_3, \text{Send}_4, \text{Send}_5), \varnothing, \{\text{Get}_0, \text{Get}_1, \text{Get}_2\}, \{\{\text{Get}_0, \text{Send}_3\}, \{\text{Get}_0, \text{Send}_4\}, \{\text{Get}_0, \text{Send}_5\}, \{\text{Get}_1, \text{Send}_3\}\} \]

Secondly, the two-tuple information:

\[ \text{System} = \{\{\text{ATC}_A, \text{ATC}_B, \text{ATC}_C, \text{ATC}_D, \text{ATC}_E, \text{ATC}_F, \text{ATC}_G\}, \{\{\text{Send}_0, \text{Get}_0\}, \{\text{Send}_3, \text{Get}_5\}, \{\text{Send}_1, \text{Get}_2\}, \{\text{Send}_2, \text{Get}_1\}, \{\text{Send}_3, \text{Get}_5\}, \{\text{Send}_4, \text{Get}_4\}, \{\text{Send}_5, \text{Get}_3\}\} \]

According to the four-tuple, we know the ATC_D’s input ports are \{Get_0, Get_1, Get_2\}, depositing in the RPS(D). Also finding the ports from the two-tuple connect to RPS(D) are \{Send_0, Send_1, Send_2\}, depositing in RRPS(D). System drives RRPS(D) into the ATC resting port status, and dumping the data which is sent from the RRPS(D) to the Memory ATC. Because there is no data sent to RPS(D), after a topological processing time, ATC_D turns into the reconfigurable status. At this point, ATC_D’s status is no longer the original system records, the system consistency is destroyed. But ATC_D’s status will timely feedback to the system, so the system can quickly restore consistency. After reconstruction, driving the target ATC_D’s input port set RPS \{Get_0, Get_1, Get_2\} into the ATC passive port status, system records new status of the target ATC (working status). The data stored in Memory ATC is forwarded to ATC_D, then driving RRPS into the ATC passive port status. So far, the system has returned to full working condition.

When the target ATC’s reconsititution occurs, this method has the advantage that parts of the system function can still be used, only those function that is related to the target ATC can’t be used and the packet loss rate also reduces. We will verify it in the next chapter.

2) ATC-based consistency preservation method

When the reconstition occurs, there are some ports in RRPS(A) cannot be driven into the ATC resting port status, leading to the target ATC can’t turn into the reconfigurable status, then you need another reconstruction method. This method will drive the input ports of head ATC which connects to the target ATC on
semantic level into the ATC resting port status, leading to no data over the link. So A’s input port will enter the ATC resting port status, A enters reconfigurable status.

The method is as follows:
1. Knowing the target ATC from the system requests.
2. Knowing the target ATC’s four-tuple information from system records, thus obtaining the target ATC input ports set RPS.
3. Knowing the other ATCs’ ports set RRPS(1) connect to RPS from two-tuple.
4. Through the IC (1) Information in four-tuple find the ports set I(1) connected to the RRPS (1).
5. Through two-tuple finding the ports set S(2) connect to I(1), and so on, until you find the head ATC (head ATC’s second property is \( \varphi \)).
6. The system drives each head ATC’s input ports into the ATC resting port status, then records the new status of head ATC. The data sent to head ATC will be forwarded to other network nodes for processing.
7. According to Theorem 2, we know that when the head ATC’s input ports are on the ATC resting port status, after two ATC topology processing time, the target ATC and the head ATC turn into reconfigurable status. By Definition 7, the data sent to each head ATC dumps to Memory ATC or other network for dispose.
8. System records the new status of target ATC.
9. After the reconstruction, firstly, the system drives each head ATC’s input ports into the ATC receiving port status, then the packet will enter into the link and the status of target ATC’s input ports will turn to ATC receiving port status. Then, the system records the new status (working status) of target ATC and each head ATC.

According to the ATC-based consistency preservation method, we reconstruct the following example.

Shown in Figure 6, if the system is going to reconstruct on ATC_D, assuming ATC_C’s port Send2 can not be driven to the ATC resting port status.

\[
\text{ATC}_A = \langle \{\text{Send0}, \text{Get0}, \text{Get1}\}, \varphi \rangle, \text{ATC}_C = \langle \{\text{Send1}, \text{Send2}, \text{Send3}\}, \varphi \rangle, \text{ATC}_D = \langle \{\text{Send0}, \text{Get0}, \text{Send6}\}, \text{Get1}, \text{Send3}\rangle
\]

The two-tuple information:

\[
\text{System} = \langle \text{ATC}_A, \text{ATC}_B, \text{ATC}_C, \text{ATC}_D, \text{ATC}_E, \text{ATC}_F \rangle = \langle \{\text{Send0}, \text{Get0}, \text{Send4}, \text{Get1}\}, \{\text{Send1}, \text{Get2}, \text{Send2}, \text{Get3}\} \rangle.
\]

The previous work is same with the ATC port-based consistency preservation method. Firstly, through the four-tuple finding ATC_D’s input port is \( \text{Get3} \), and depositing it in RPS(D). Also basing on two-tuple we find ports set \{Send2\} connect to RPS(D), and depositing it in RRP(D). However, the system can not drive Send2 enter the ATC resting port status, so using the ATC-based consistency preservation method. Through the four-tuple finding Send2 is ATC_C’s output port, and through the IC we find Send2 connect to Get0 in ATC_C’s inner connection. Followed by two-tuple finding port Get0 connect to Send0, and querying four-tuple we find Send0 is ATC_A’s output port, and that ATC_A is the head ATC (because ATC_A’s I is not empty). So that the system knows the target ATC_D connect to the head ATC is ATC_A, then drives ATC_A’s input port Get6 enter the ATC resting port status, so that the data will no longer flow through ATC_D. After two ATC topology processing time, ATC_D’s RPS (D) turns into the ATC resting port status, then ATC_D turns into reconfigurable status. At this point, the system records the new status of ATC_D and ATC_A. The data sent to ATC_A will be forwarded to other network nodes for processing. After reconstruction, driving Get6 into the ATC active status (record ATC_A new status), and driving Get3 into the ATC passive port status.

When reconstitution occurs, the whole of link which is relevant with the target ATC will be not on working status. However, in a distributed system, the system knows that some links in the network node don’t work that means the system can’t provide some certain functions, then it will forward the data to other network nodes for processing, avoiding loss and delay.

V. ANALYSIS OF SYSTEM’S PROPERTIES

According to Chapter 4, in the reconstruction process, there are two ways to ensure consistency, they are the ATC-based consistency preservation method and the ATC port-based consistency preservation method, respectively. This chapter will analyze performance of the system using both methods when the ATC reconstitution occurs.

So let’s test the problem of system’s packet loss rate and delay under ATC consistency preservation methods during the ATC reconstitution.

According to Figure 7, it is assumed that the average input data rate of this topology is 100M/s, each ATC cache is 200M. If implicit connections in ATC are not shown, indicating there is only one path between output port and input port. There are two paths from ATC_A output port, the average transmission rates are 40M/s and 60M/s respectively. Implicit connections have existed in
ATC_B, the average rates of data coming from the port p and sent to the output port a, b, c are 10M/s, 20M/s, 30M/s. Now each ATC reconfiguration occurs, Supposing that the time for reconfiguration is 10s, each network node has a Memory ATC and its storage capacity is 1T. Now we will get system packet loss rate and delay by using and not using ATC consistency preservation methods.

The system’s packet loss rate is LA when ATC_A is reconfigured, other ATCs’ are LB, LC, LD, LE. And the system’s delay is DA when ATC_A is reconfigured, other ATCs’ are DB, DC, DD, DE. The total data traffic sent to this topology is 100*10=1000M in 10s.1)

1) When not using ATC consistency preservation methods, each ATC’s packet loss rate:

\[
LA = \frac{100 \times 10 - 200}{1000} = 80\%
\]

\[
LB = \frac{10 \times (40 + 60) - 200}{1000} = 80\%
\]

\[
LC = \frac{(40 + 10) \times 10 - 200}{1000} = 30\%
\]

\[
LD = \frac{10 \times 20 - 200}{1000} = 0\%
\]

\[
LE = \frac{10 \times 30 - 200}{1000} = 10\%
\]

each ATC’s delay: Clearly, there is no delay because all data will be delayed are discarded.

2) When using ATC consistency preservation methods,

1. ATC port-based consistency preservation method, each ATC’s packet loss rate:

When ATC_A turns into the ATC resting port status, the data transferred to the topology will be forwarded to the Memory ATC. The Memory ATC is big enough to store the data, so it did not produce any packet loss, but data delay exists. So LA = 0%, Similarly, LA=LB=LC=LD=LE=0%. Each ATC’s delay: Obviously, DA=DB=DC=DD=DE=10s.

2. ATC-based consistency preservation method, each ATC’s packet loss rate:

When ATC_A turns into the ATC resting port status, the data transferred to the topology will be forwarded to other network nodes for processing. So it did not produce any packet loss and data delay exists. So LA=LB=LC=LD=LE=0%, DA=DB=DC=DD=DE=0s.

Results shown in Figure 8,

The above analysis shows under ideal conditions not using ATC consistency preservation methods, the system’s packet loss rate has something to do the size of the cache capacity and the time of reconstitution, whereas using this method, the system will not drop data packets as long as the Memory ATC’s capacity is large enough. What’s more, ATC-based consistency preservation method will not cause any data delay.

VI. CONCLUSIONS

This paper presents the system consistency problems during the reconstruction process, and cites the current domestic and international consistency researches during dynamic reconstruction process, and proposing the inadequacies. The third and fourth part based on Atomic Capability system propose ATC consistency preservation methods. Firstly, defining four states about ATC’s port status and two states about ATC’s status. Only driving target ATC into reconfigurable status the system can implement dynamic reconstruction. Under these definitions and theorems, this paper proposes ATC port-based consistency preservation method and ATC-based consistency preservation method. Using these two methods the system solves two problems--ATC’s status changes don’t timely feedback to the system during the dynamic reconstruction and facing the status changes the system doesn’t make a reasonable treatment options. Finally, the analysis shows using of ATC consistency preservation methods will greatly reduce system’s packet loss rate and delay during dynamic reconfiguration process.

ACKNOWLEDGMENT

This work was supported in part by a grant from the National Basic Research Program of China (973 Program) (No. 2012CB315902), the National Natural Science Foundation of China (No.61379120, 61170215), Zhejiang Provincial Key Laboratory of New Network Standards and Technologies (NNST) (No. 2013E10012).

REFERENCES


[16] MA Chuan, SHEN Li-min, WANG Tao, Behavior Consistency Verification Method Based on Component Dynamic Evolution, Computer Engineering, 2010(6)


Jianqing Li is currently studying at Zhejiang Gongshang University as a postgraduate. He earned his B.E. Degree in Communication Engineering from Zhejiang Gongshang University, China, in 2012. His research interests in open-arch networks and analysis model.

Chuanhuang Li is currently working at Zhejiang Gongshang University as a lecturer. He earned his Ph.D. degree in Computer Science and Technology from Zhejiang University, China, in 2013, M.A. degree in Electrical Engineering from Zhejiang Gongshang University, China, in 2007, and B.E. degree in Mechanical and Electronic Engineering from Nanchang University, China, in 2001. His research interests include open-arch networks, system performance prediction and analysis model, stochastic network calculus.

Weiming Wang is a professor in the College of Information and Electrical Engineering at Zhejiang Gongshang University. He earned his Ph.D. degree in Electrical Engineering from Zhejiang University, China, in 1992, M.A. degree in Electrical Engineering from National University of Defence Technology, China, in 1989, and B.E. degree in Telecommunications from Zhenzhou Institute of Information Engineering, China, in 1983. He worked as a Research Engineer in the Internet Technology Lab, The University of Arizona, from March 2001 to May 2002. His active research area is on open-arch networks and its applications in Internet, specializing in IETF ForCES (Forwarding and Control Element Separation) technology.

Julong Lan is the Director of the Broad-band Network R&D Department of the National Digital Switching System Engineering & Technological R&D Center and professor of Information and Communications Systems. His research contributions encompass aspects of information theory and security, network communication, parallel computation and signal processing.