Electronic Preprint for Journal of Information Processing Vol.26

Regular Paper

Silk Road: A Framework for Distributed Collaborative Simulation

Jiachao Zhang[1,a)] Shinji Fukuma[1,b)] Shin-ichiro Mori[1,c)]

Received: June 11, 2017, Accepted: September 5, 2017

Abstract: Diverse simulations on centralized high performance computers are the driving force behind innovation in nearly all fields. Higher resolution and real-time interactions of users are expected to reveal valuable phenomena by providing more precise details and efficient dynamic steering inside simulations. If simulations are performed at a centralized server, such as supercomputer or cloud system, it is difficult to satisfy the requirements of both real-time interaction and high-resolution for many of those user's domain of interest (DoI). To solve this problem, we utilize distributed regional servers closer to users to perform simulations for each independent DoI region. However, this may introduce inaccuracies in the boundary condition for each DoI region. In order to improve the boundary conditions for better regional simulations, the servers must cooperate to exchange necessary information. For this reason, this paper proposes a general purpose framework named Silk Road, to help application users realize a distributed collaborative simulation which utilizes regional servers to perform high-resolution simulation with low network delay, with help from exchanging boundary conditions through collaborations with each other via a central server which performs a low-resolution but wider area simulation to couple the regional simulations to propagate the higher resolution simulation results. The most notable feature of Silk Road Framework is the bi-directional refinement of on-going simulation by exchanging an adequate amount of simulation results occasionally without inducing unacceptable network delays. Through a case study, with a 2D diffusion simulation, we show the framework can achieve the distributed collaborative simulation and our model can help refine simulations on both the regional and central server side.

Keywords: real-time interactive simulation, distributed collaborative simulation, edge/fog computing, IoT, diffusion simulation

1. Introduction

Simulation has been widely used in various fields for academic and industrial value generation [1]. Instead of post analysis in common batch processing, real-time simulation and visualization provides experts with an on-the-fly analysis of the phenomenon that is occurring. This in situ methodology can also avoid the expensive storage and transfer of the expanding big data along with the increase of computing ability [2]. Further, interactive simulation and visualization adds human-in-the-loop steering of the simulation and visualization so that real-time interaction can participate in computing which allows more immediate, effective, flexible and valuable processing in simulations [3]. And indeed with the explosive emergence of IoT [4], [5] devices such as sensors, different kinds of mobile devices in wider distributed geographical areas, the boundary of man and machine has already been blurring, so that user interaction in the simulation will not only be from humans but also an increasingly large number of machines.

We assume that different users (men or machines) may have different domains of interest and they need real-time interaction and high-resolution simulation result on their interested domain. Facing this requirement, we believe collaborative simulation by distributed users can play important roles in future [6], [7]. To be specific, each user in wider geographical distributions may have separate domain of interest where they expect to know more details about the simulation result, while they have no such interest and indeed no way to know what is happening outside of their focused limited area. If computing is performed on a centralized high performance server which is usually known as cloud computing, the network delay and jitter will hamper the real-time effectiveness. Even without network delay and we consider the definition of real-time ranges from few milliseconds to days or weeks according to specific applications, it's still impossible for even powerful centralized servers to provide a real-time interactive simulation which can satisfy every user's requirement in their desired simulation resolution. So to allow high-resolution simulation and achieve real-time interaction, we utilize distributed regional servers to perform high-resolution simulation on each interested domain and achieve real-time effectiveness by avoiding high network delay.

However, without collaboration, none of the regional servers can perform a correct simulation since they have no way to know what is happening outside their own region which serves as the mandatory boundary condition for simulation. For simplicity of understanding, here we take the weather forecast as an example to clarify this situation. For instance, people living in Tokyo area care about the details of weather change at Tokyo such as strong

[1] University of Fukui, Fukui 910–8507, Japan
[a)] zhang@sylph.fuis.u-fukui.ac.jp
[b)] fukuma@u-fukui.ac.jp
[c)] moris@u-fukui.ac.jp

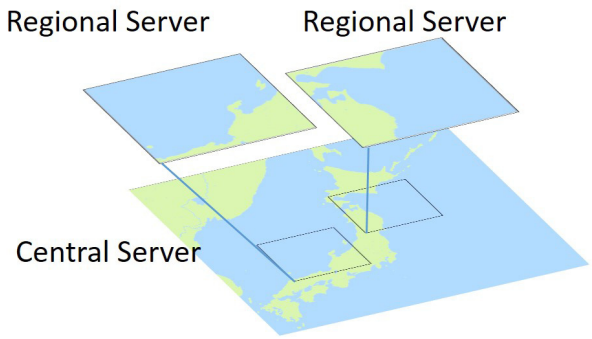

Fig. 1 Multi-scale bi-directional refinement model for Distributed Collaborative Simulation.

wind, temperature change, the diffusion of air pollution particles and so on. If possible, high resolution information on a specific district or street is useful and necessary for users. Intuitively, they do not have to pay any attention to the weather information of Kyoto area, but the current weather phenomenon in the Kyoto area may approach and influence the Tokyo area sooner or later. Vice-versa, the influence from Tokyo area to Kyoto area may also exist. This is also true for other simulations since the phenomena we are simulating are inherently correlative at continuous space and time in the real world. So in order to perform accurate simulation, each regional server which performs high-resolution simulation on limited area must collaborate with others to get the necessary boundary conditions.

Aiming at the above issues, this paper proposes a general-purpose framework to help application developers to realize collaborative simulation with real-time interactions in a distributed environment[*1]. **Figure 1** shows the basic multi-scale bi-directional refinement model for distributed collaborative simulation in our framework. It utilizes both decentralized distributed regional servers and a centralized server to realize a distributed collaborative simulation environment, where several regional servers help to improve their simulation results by performing higher resolution simulation for the area of each interest, working with the central server which covers the whole area but at a lower resolution. Rather than direct communication between regional servers, we use their results through a central server on another layer to integrate each regional server's interactions and propagate a global simulation, so that the central server could provide each regional server with their required boundary conditions via the global simulation result. The benefits are the fact that it can remove the dependency of regional servers, reduce the complexity, and let simulation propagate on global space. Otherwise, even if a regional server can receive some data from the other one, it still don't know when and how the phenomena happening at an outside region will approach its own focused area. Although Fig. 1 just shows two regional servers as examples, since the communication is between regional server and a central server, the increase of regional servers will not bring about more complexity in design. And although this figure and the current proposal only consider two-layer regional/central design, it's possible to extend more hierarchy of servers to suit user requirements in the future.

[*1] In this paper, we focus on time-series simulations which solve boundary condition problems for stable and continuous near field phenomena.

As a result, we expect that regional server can provide satisfactory high-resolution simulation with the help of the boundary conditon from a central server, and inversely each regional simulation result can help refine the global simulation on a central server side. That's why we call this model a multi-scale bi-directional refinement model.

In ancient China, in order to sell China's own rich products such as silk, tea, and china, and meanwhile to obtain required products from other countries such as furs, jewelry, and wine, people built an ancient network of trade roads covering a wide area known as the Silk Road [8]. As a result, both China and other countries can get benefits from the multilateral trade, and they can help each other to improve the quality and diversity of their own domestic marketing. In some sense, our proposed framework is trying to build an initial message highway to achieve bi-directional data exchanging between independent yet correlated regional areas. Because of the common grounds with the ancient Silk Road as follows, we name our framework the Silk Road Framework.

- Distributed Collaboration: Distributed regional servers know quite well about their own interested region, but due to the lack of boundary conditions, they need to collaborate with each other to exchange necessary data for performing a correct simulation. In the ancient Silk Road, each country is best at its own goods and culture; to improve the quality and diversity of goods for living, they have to help each other by trading to get the necessary goods. The Silk Road Framework and the ancient Silk Road make this exchange possible.
- Bi-directional Refinement: Our framework doesn't assume that the central server is a high performance server or supercomputer, and nor that one supercomputer can perform high-resolution simulation with real-time interactions for every user. Instead, by collaboration among each regional servers, the requirements can be fulfilled and results can be bi-directionally refined. In ancient Silk Road, there never exists a supercountry which can produce everything due to limitations imposed by nature and human resources. However by trading, countries can help each other improve their goods, and the final result is bi-directional win-win situation.
- Reliability: Each regional server must be able to perform its simulation independently, so that it can continue working and the whole system is reliable in case of any failure of another regional server. It is similar to the situation where each country is independent to produce their own products, and they may join or quit the multilateral trade relationship freely as they wish regardless of the situation of entire marketplace.

2. Related Work

Fog computing is gathering the attention as a future computing architecture in the coming IoT era to support applications such as connected vehicles, and smart city [9], [10], [11], [12]. Due to the explosive increase of IoT devices in these eras, it becomes impossible for the central server to gather all the information from these devices at once and process them to perform the necessary

reaction to the devices interactively. So, the recently proposed fog computing or edge computing decentralizes the workload of the remote cloud server by distributing a part of their work to the edge nodes which are closer to the end-users or the leaf IoT devices. By utilizing the capacity of computing, storage and network of edge servers, both centralized clouds and IoT devices can offload their computing tasks to edge servers and also reduce the huge amount of long distance network traffic sent back and forward between devices and the central server.

The framework we proposed in this paper can be thought as a sort of fog computing, however, the purpose of using regional servers as edge servers is not for offloading the work of the central server but for performing high-resolution simulation, real-time interaction, and filtering the data sent to the central server. The central server, which may not necessarily be a powerful HPC server like supercomputer, is used to serve as a bridge to help realize the collaboration of regional servers, which is expected to realize a bi-directional refinement distributed collaborative simulation system.

In this direction, JAMSTEC and NTT are conducting research which utilizes edge computing and the Earth Simulator which is a supercomputer for earth science, to perform multi-scale weather forecast simulation for smart society of the future [13], [14]. Although it is quite similar to our research on performing multi-scale simulation using edge computing, there still exist some essential differences. In their research, there is a powerful supercomputer, so users can always trust the result on the Earth Simulator; and the purpose of using edge servers is to integrate and filter the measured data to reduce the observed noises in order to finally help refine the global simulation on the Earth Simulator. In our case, we don't expect the central server to have extreme reliability nor high performance like a supercomputer and the purpose of the central server is to exchange boundary conditions for regional servers, and we always expect the result on regional server is more reliable. As a result, the system in our proposal could be more reliable in case of any failure of another regional server or central server, since each regional server performs simulations independently so that any failure at one regional server will not make other regional servers stop or restart. Even the central server might stop.

Multi-scale simulation in various resolutions utilizes techniques such as nested grid and adaptive mesh refinement [15], [16]. For example, the research [15] improves the resolution of tsunami simulation by using multi-scale nested grid, and the research [16] dynamically outstands regional phenomena in wind-speed visualization by nested finer grid. Firstly, in the nested grid simulation, multiple-levels of finer-grid simulations are performed on the region of interests, and their results are repeatedly embedded into their upper-level, or parent, coarser-grid simulations. All of these simulations are usually performed on a same centralized server. In contrast to this, the fine-grid simulation on regional servers and the coarse-grid simulation on a central sever makes up a similar multi-scale simulation in our research, however, these simulations are separately performed at the distributed computing servers and the computing initially starts from regional servers which are loosely coupled via the coarse-grid simulation on the central server. Secondly, the usual nested grid in Ref. [15] computes the coarse-grid parent domain and fine-grid child domain alternatively, since in each computing loop, the computing on fine-grid child domain has to wait for completion of computing on parent domain and then generate boundary condition, after that the fine-grid result is used to refine the coarse-grid result. In our case, we can perform both fine-grid simulation on regional server and coarse-grid simulation on a central server at the same time in parallel, and help refine each other by bi-directional communications. Thirdly, papers [15], [16] don't consider errors due to various resolutions inherently, but during our investigation, we found that these errors are inevitable and this issue has already been reported in two-way embedding algorithms of nested grid [17], [18], which is used in ocean dynamics or weather forecast field, and our related consideration on such errors will be discussed in Section 3.6.

3. Silk Road Framework for Distributed Collaborative Simulation

In previous section, we have already explained the distributed collaborative simulation and its basic multi-scale bi-directional refinement model used in our framework. This section will introduce the details of our design and implementation, and some essential general issues will be also considered in the framework. For simplicity, instead of a thorough design starting from zero, we utilized an existing project named Simulation Caching Framework [19] in our group which is designed for interactive remote supercomputing. Now for the new purpose of distributed collaborative simulation in this paper, we have extended the original framework to the new Silk Road Framework which considers the bi-directional collaboration of multiple users in a distributed environment. Before going deeper, it's necessary to introduce the original framework and first compare the differences in terms of background.

3.1 Background: Simulation Caching Framework

Simulation Caching Framework is a general-purpose framework for interactive remote supercomputing [19]. The strategy is to utilize a regional server to hide network delay by redundantly performing limited area and/or lower resolution simulation on regional server to give a timely response. In this former framework, the computing starts from central server which always keeps a better simulation result than regional server, so the communication for consistency control is purely a single-direction from central to regional server (refered to as downward quality control, DQC). We name the framework "Simulation Caching" since the role of the regional server is to cache the simulation program to perform simulation for timely response, and to receive downward simulation data in some frequency to restart the simulation for maintaining the data within acceptable accuracy. Although the former framework had considered to extend for multiple users' interaction [20], [21], the former endeavor is to share inputs to other regional users via central server, while current Silk Road Framework assumes users focus on different region without overlapping, and all information including inputs and simulation propagating are shared through boundary condition via the cen-

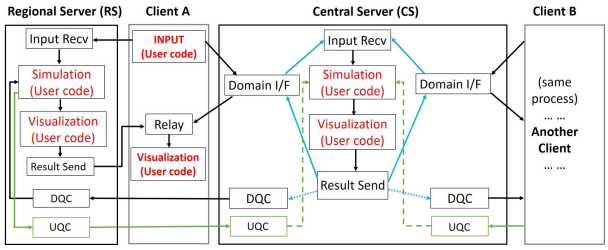

Fig. 2 The system design of Silk Road Framework.

tral server.

In Silk Road Framework serving multiple users, the computing starts from each regional server which is expected to maintain a better simulation result for a limited area, and the downward data is not for consistency updating any more, instead it is for boundary conditions which reflect the influences by other regional users. And since the regional servers know quite well about their domains of interest, an inverse upward communication from a regional server to the central server is necessary to help improve the global simulation, which we would like to refer to as upward quality control (UQC).

3.2 System Design of Silk Road Framework

Figure 2 illustrates the overview of the system design of our Silk Road Framework which includes the bi-directional quality control processing UQC and DQC. In this figure, the blocks in red color mean this process belongs to the user application, such as providing input data for interaction, and simulation and visualization program. Domain I/F denotes the interface between regional server and central server which is used to relay input data and occasional downward visualization data for observation and debugging.

The newly added upward quality control and revised downward quality control have achieved a bi-directional message passing route for regional servers and the central server in Silk Road Framework. Indeed, with the help of this bi-directional communication routes designed for general purpose, users can define any transferred data on their own, and develop their own synchronization or processing functions according to their applications. Here we are considering in general, and providing the data processing scheme for typical situations in our proposed distributed collaborative simulation.

Figure 3 shows the overview of flowcharts of DQC and UQC process. In DQC time step, central server side will extract part of its simulation data as the DQC data and send it to the regional server side for generating its boundary condition. Once the regional server identifies the received data is DQC data rather than dummy message in other time steps, the received data will be sent to the simulation program and participate into simulation in the next time step. In UQC, regional server side will extract its simulation data as the UQC data and send it to the central server for updating since we expect the high-resolution simulation on the regional server is more accurate then it could help refine the global simulation on central server.

Although Silk Road Framework is to build a bi-directional road for data exchanging, except for the message passing road, some

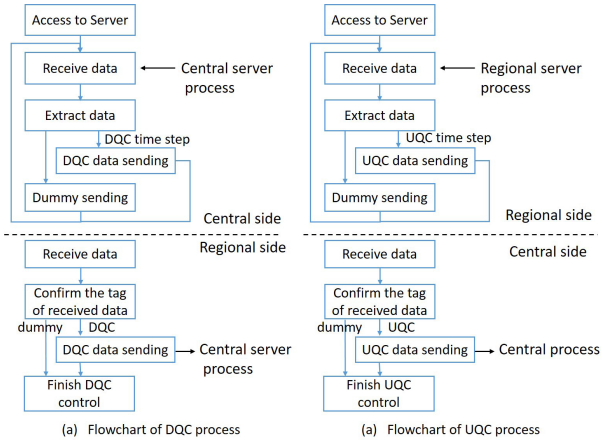

Fig. 3 The overview of flowcharts of DQC and UQC.

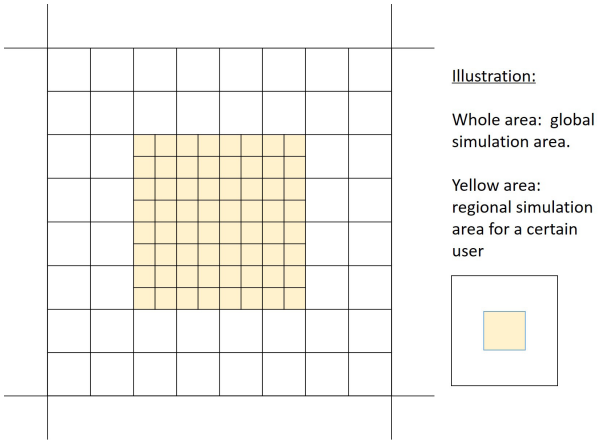

Fig. 4 A typical situation of multi-scale bi-directional refinement model.

common essential issues should also be considered for DQC and UQC process. The next two subsections tells the considerations of typical processing in DQC and UQC.

3.3 Essential Considerations in DQC

The typical situation is a regional server performs a high-resolution simulation on a limited area while central server performs a low-resolution but global area simulation which covers the regional simulation's area. **Figure 4** shows the situation when we assume the ratio of global and regional simulation's resolution is double on x-dimension and y-dimension separately in the two dimensional case. Due to limitations on space, this figure just shows part of the global area which covers the regional simulation area. Since the communication between regional server and central server are in a distributed environment where relatively large network delay exists, both DQC and UQC are scheduled in some frequency as a parameter rather than every time step.

In order to send the required boundary condition to a regional server from the central server at DQC time step, the first process is to extract the necessary data for sending. **Figure 5** shows this data extract process. On central server side, the yellow color area in the figure indicates the duplicate area with regional simulation, a little larger gray color area includes necessary information for regional simulation's boundary condition. To reduce the data size for transferring and corresponding overhead, the white color area's data does not need to be sent, and we use a parameter KK

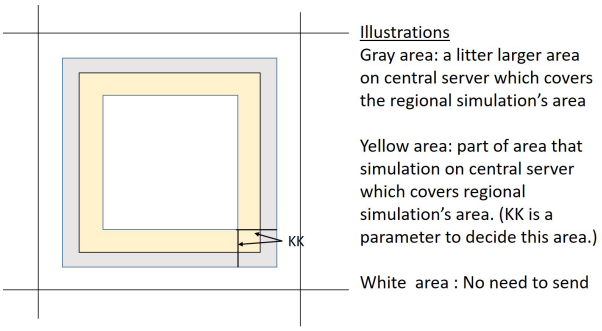

Fig. 5 Data extract in DQC.

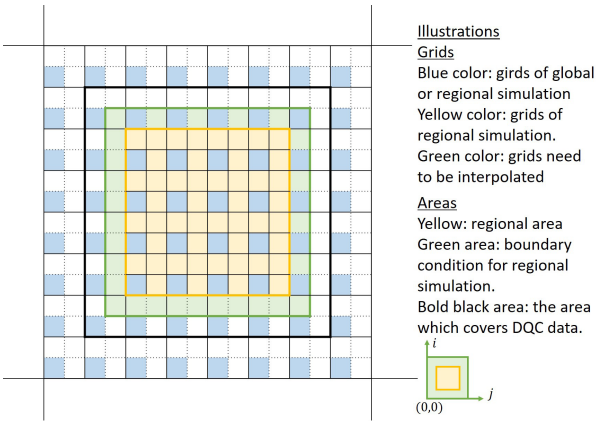

Fig. 6 Interpolation process in DQC.

to adjust the yellow area's size for application users. As a result, the gray and yellow area's simulation data will be sent from the central server to a regional server.

After regional server receives the DQC data, it still needs to perform two different kinds of interpolation using the lower resolution DQC data to generate the boundary condition for high-resolution regional simulation. **Figure 6** shows an example to illustrate this interpolation process. For simplicity of explaining, we use different colors to identify the grids, areas and situations. First of all, in the overall global simulation space, the innermost rectangular area which is surrounded by a bold yellow border refers to a regional server's domain of interest. Its adjacent area, which is mainly colored in green and surrounded by a bold green border, determines the boundary condition we need for regional simulation. The boundary consists of green grid and blue grid. For the green grid, we have to perform interpolation since there is no corresponding grid with the global simulation on central server, not like the blue grid. So, with the help of DQC data from central server, we have to perform some interpolation process on regional server side to generate the needed boundary condition in high resolution.

For simplicity of implementation, we mainly use averaging for interpolating the value of green grids. To make this easier to specify in the figure, we mark the coordinate $(0, 0)$ at the most left-down green grid of boundary condition, and use coordinate (i, j) to indicate each grid along the upward direction and rightward direction respectively.

Specifically, to interpolate the up-side grids of boundary condition, we directly use the existing value of DQC data at mapped position (when j is odd), or average the left and right neighbors of DQC data (when j is even). The same to the right-side grids of boundary condition, use the existing mapped DQC data (when i is odd), or average the up and down neighbors of DQC data (when i is even). For the down-side and left-side grids of boundary condition, since there is no existing mapped value in DQC data, it need a bit more attention. For the interpolation of down-side grids, do average using its up and down neighbors' value in DQC data (when j is odd), and find the average of the nearest four surrounding grid's value in DQC data (when j is even). The same to the left side grids, but use i instead of j to justify the interpolated grid's situation.

It is important to note that, the ratio of the resolution of fine-grid regional simulation and coarse-grid global simulation may be 3, 4, 5 or more times. If the ratio is odd like 3, the above discussed interpolation and mapping relationship between regional and global simulation will be different, for example, the number of ghost cells may decrease and interpolation may become a little simpler. And if the ratio is more like 4 times, the improvement from fine-grid simulation may be bigger whereas it may bring about more errors on boundary condition since the difference between global and regional simulation will become larger.

Furthermore, as far as we know, it's almost impossible to interpolate a perfect value without any error in this kind of various resolution simulations and develop a general interpolation process suits for any kind of simulations, since the interpolation inherently depends on the characteristics of the specific user applications. Currently we use the typical averaging method as the default interpolation processing in DQC, but application users of our framework may replace it with their own interpolation function, since the interpolation would affect the accuracy of simulation.

Since the Silk Road Framework itself is building a bi-directional message passing route for distributed collaborative simulation. It's a system to help realize multiple regional servers' collaboration via a central server, and we are providing the framework as a framework developer and considering essential general issues as an application user. So here we only discuss the essential interpolation process when the ratio of resolution on space equals 2 on x, y dimension separately (actually the ratio of resolution on time is also 2, which will be discussed later), and only consider relatively simple averaging for interpolating boundary conditions.

3.4 Essential Considerations in UQC

Since we expect the high-resolution simulation result on regional servers could help refine the global simulation on the central server, the UQC data is sent to update the simulation data on the central server at some frequency.

Figure 7 shows the process of UQC at each UQC time step. Firstly, extract part data from fine-grid simulation result on the regional server, it's recommended by us to directly pick the cells which have nature mapping relationship with the grids of coarse-grid simulation on the central server. Of course, it could be the average value of every four grids on regional server to update one grid on the central server if the user like. Secondly, after the central server received UQC data, it's time to update the simulation

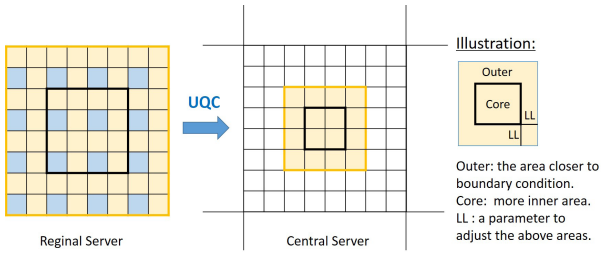

Fig. 7 Updating process in UQC.

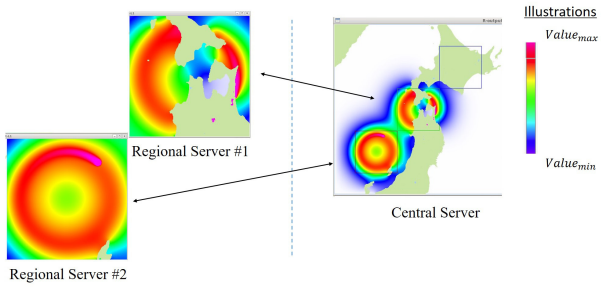

Fig. 8 An implemented example when two regional servers are in collaboration.

data on central server. Since regional server cannot receive DQC data every time step, in addition to some other factors we will discuss in Section 3.6, there exist errors on boundary condition areas. So we divide the regional area on the central server into two parts which is called *Core* and *Outer* area, and set a parameter LL for users to adjust the size of the two parts. In *Core* area, we can choose to totally trust the UQC data from regional server; but in *Outer* area, we can use the weight value to get a balanced result based on UQC data and central server's data.

Compared to DQC, the frequency of UQC could be much lower, since the central server has the ability to receive inputs and perform its simulation covering this regional area without UQC data, while the regional simulation have to rely on the DQC data to generate boundary condition. Another reason is that the data size of UQC data is much bigger, to reduce the data size to be transferred at UQC time step, we may just send sampled data instead of sending whole area's data.

3.5 Implementation

Based on the system design and the essential consideration of bi-directional quality control discussed in previous sections, we have implemented the framework that could support multiple users' collaboration.

Figure 8 shows an implemented example when two regional servers are working in collaboration. The simulation is a diffusion simulation which we will take as a case study in next section. The different colors in visualization represent a different level of simulation data. This snapshot shows that each regional server can correctly reflect the influences happening at their outer space which may be affected by other regional servers. To be specific, in the left-down corner of the regional server#1, the visualization result shows that the value of simulation data is higher around this corner, which reflects the inbound influences caused by the other regional server#2. Similarly, regional server#2 reflects the inbound influences at its up-right corner. In this figure, the main

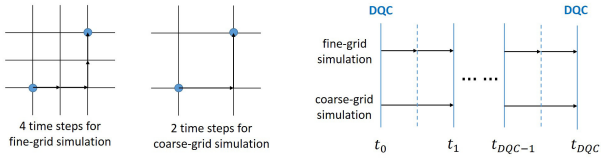

Fig. 9 Error source analysis.

parameters are that the frequency of DQC is every 30 time steps, and the geographical background cannot absorb the diffusion at all.

It optimistically shows that by collaboration, each regional server can get the boundary condition they need so that they could reflect the phenomena happened at outer space. However, visualization may deceive us since human eyes are not sensitive enough to recognize some differences in simulation data.

3.6 Error Analysis

Intuitively, it seems that regional simulation which is in higher resolution on space and time should be definitely more accurate than the data of same area in the low-resolution global simulation. However, there exist inevitable errors in our model and other multi-scale simulations. References [17], [18] call these errors "noises" and the attempt to reduce the errors "noises control". Since the inevitable errors are a general issue, it's necessary to analyze the sources in the context of our proposal.

First, since our research is for real-time interactive simulation under a distributed network environment, the quality control (DQC or UQC) of every time step is impossible. So, the regional server has no way to know the latest change on boundary conditions during the interval time steps between each DQC, that is the interval steps between t_0 and t_{DQC} in **Fig. 9**. And on the other hand, since the regional simulation is in higher resolution on space, for example if the resolution on a space is double on the x dimension and y dimension separately in the two dimensional case, the resolution of fine-grid simulation in time should also be double in order to propagate the simulation in the same physical area, which is shown in Fig. 9. Therefore at the interval between t_0 and t_1, the boundary condition for fine-grid simulation also exists error.

Secondly, the interpolation of boundary condition for fine-grid regional simulation is based on the coarse-grid global simulation's data. There are a lot of ghost cells as we have seen in Fig. 6. And although we did interpolation to generate these values, due to the characteristics of computing in simulation, it's inevitable to bring about errors in this interpolation process.

Thirdly, different simulation resolution (the mesh size) will make the phenomena of simulation itself different, for example, the speed of a diffusion phenomenon may be faster in a coarse-grid space than a fine-grid space. The error souring from different mesh size is also reported in Ref. [18].

Lastly, the updating process in UQC from fine-grid simulation to coarse-grid simulation may bring about few errors when refine the global simulation.

In order to reduce the errors, we have been developing some efforts to control the noises. For example, at the interval time steps between each DQC, we may use some predict models to

predict a boundary condition for next time step according to the current regional simulation data. Currently, in this paper, we are always using the last time interpolated boundary condition as a constant boundary condition for the interval times steps between each DQC.

4. A Case Study for Evaluation

4.1 The Diffusion Simulation as the Case Study

We develop a two dimensional diffusion simulation as a case study to evaluate the effectiveness of our framework and the model of distributed collaborative simulation. Since the computing is based on the partial differential equation which can describe a lot of phenomenon such as the diffusion of sound, heat or pollutant material. So we have no need to distinguish the practical meaning of the simulation itself only if it can correctly describe the physical phenomena of diffusion from the perspective of the Silk Road Framework.

Since regional user needs a more accurate simulation result on their domain of interest, for this region-based requirement, we introduce geographical information into the diffusion simulation which could interact and influence the simulation. The geographical information generated can be simply divided into the sea and the land. We assume that diffusion can propagate on the sea; and assume that the land can absorb the diffusion or cannot absorb it at all (be insulated to the diffusion), which are the two extreme ends of the effects to the simulation.

For visualization, we prepare a normal style visualization and a log-scale visualization mainly in gray-scale coloring for errors observation. The sea and the land are rendered in two colors to show the geographical structure. In normal style, we use different colors as the bar shows in Fig. 8 to linearly express the changement of simulation data in several segmented range. In log-scale visualization, we compute the base 2 logarithm of the simulation data for color mapping to magnify the visual observation of the simulation data which is close to zero, and except for the diffusion source rendered in red color, all data is expressed in linear gray-scale color.

4.2 Experiment Design (Single-machine Evaluation Program)

For the purpose of purely evaluating the effectiveness of the multi-scale bi-directional refinement model for distributed collaborative simulation in Silk Road Framework, we specially developed a single-machine evaluation program which combines diffusion simulation and all data processing functions such as data extraction, interpolation of boundary condition, updating and so on, but remove all data transferring. This is because we would like to concentrate on the evaluation of the simulation quality under various simulator configurations without suffering from the network delay for transferring large amount of data. As a result, this single-machine program is implemented to easily display the real-time visualization, compare the accuracy of simulation data, output measurement data for post analysis, and then evaluate the effectiveness of our model.

In this evaluation program, initially there are two simulations that regional simulation in high-resolution and global simulation in low-resolution collaborating with each other by utilizing DQC and UQC processing. In addition, we add another two simulations to be compared for evaluating the effectiveness of our model. Firstly, since both regional simulation and global simulation cannot be perfect due to the lack of correct boundary condition or details of regional geographical information, we introduce a simulation which covers a global area and be in high resolution (ultimate simulation), so that the result of this ultimate simulation can be thought as the ground truth for the quality comparison. It is important to note that such an ultimate simulation doesn't exist in real Silk Road Framework.

Table 1 Symbols denoting simulations in single-machine evaluation program.

Symbols	Denoting
R	high-resolution regional simulation w/ collaboration
C	low-resolution global simulation w/ collaboration
U	high-resolution global simulation
B	low-resolution global simulation w/o collaboration
$R_1/C_1/U_1/B_1$	the above simulations on a first region
$R_2/C_2/U_2/B_2$	the above simulations on a second region

Secondly we implemented a standalone low-resolution global area simulation, or base simulation, as our competitor. With this base simulation, concerning to the region of interest, we could investigate the effect of the tradeoff between the accuracy of the boundary condition and the simulation resolution, that is the difference between the low-resolution simulation with accurate boundary condition in every time step and high-resolution simulation with less accurate boundary condition. For easily referring, we use symbols to denote the above simulations described in **Table 1**.

4.3 A Snapshot and Observation

Figure 10 shows a snapshot of this single-machine evaluation program at one time step, in this figure, we use a log-scale visualization style in order to capture the very small simulation data close to zero, the minimum value in simulation data. The illustration shows the color ranges from black to white, except that the diffusion source is in red color. The layout shows the arrangement of four kinds of simulation in the evaluation [*2]. And we specially add a D_1 and D_2 window to show the differences between R_1 and B_1 (and R_2 and B_2), since the visual perception of humans is not sensitive enough to see tiny differences in visualization result.

To generate D_1 and D_2, in every time step, at first we compare each gird's value of regional simulation or base simulation against the ultimate simulation, then compare the absolute value of their differences in order to see which one is more closer to the ultimate simulation serving as truth. This comparison can be expressed in $D_1 = |R_1 - U_1| - |B_1 - U_1|$, and the according difference is rendered in different colors to have intuitional observation. Since B_1 is in lower resolution than R_1 and U_1, so in order to compare in same grid size, we simply use each grid's value in B_1 four times to map three nonexistent neighbouring grids in comparison($R_{1(i,j)}$

[*2] Since both C and B are low-resolution global simulation, the difference between them is whether using collaboration with regional servers or not, so in order to reduce the number of visualization windows for a clearer overall view, we use keyboard pressing event to let users choose showing C result or B result freely at the display window illustrated as "C or B."

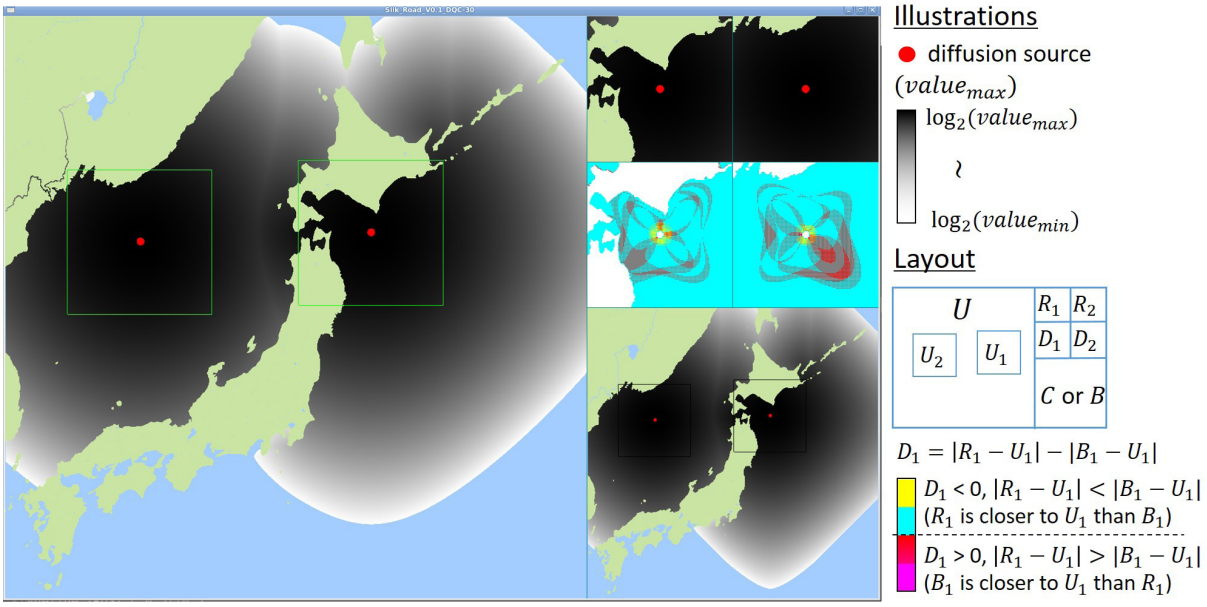

Fig. 10 A snapshot of single-machine evaluation program.

against $B_{1(i/2,j/2)}$ in some sense).

In Fig. 10, we can also observe some interesting phenomenon. At first, the different edge shape of diffusion in U and C shows that different resolution will make the diffusion speed different, which demonstrates the multi-scale resolution simulation will bring about some error that we have already discussed in section3.6. Secondly, if we look carefully at the up-left area where is the intersection place of the sea and the land, in high resolution U_1, the diffusion starts to diffuse on a small river upwards and will approach a small lake later in fact, but this diffusion will never approach this river in low resolution C or B, because the low-resolution geographical information ignored the entrance of the small river. This means high-resolution simulation sometimes means not only a more accurate value, but also the existence of phenomena or not. Thirdly, in D_1, the most area is rendered in light blue color or yellow color which means the data on R_1 is more accurate than B_1. Nevertheless, there are still some pixels in red showing that R_1 exists error, and actually as the time step goes, error may happen at the boundary area first then propagate toward inside. In the next section, we will give more details about measurement to evaluate the effectiveness of distribute collaborative simulation.

4.4 Measurement and Data Analysis

In order to evaluate the effectiveness of distribute collaborative simulation, we choose the result of ultimate simulation U as perfect truth, and anticipate that the regional simulation R and global simulation C could be closer to U than the competitor base simulation (B). To compare the simulation data of R, C, B to the according data of U, we adopt Root Mean Squared Error (RMSE) as the measurement.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - y_i')^2} \quad (1)$$

Since the effects on each regional user is the same, we focus on one region as the measurement area. Indeed, for each regional server, its own simulation result with interactions will try to effect outbound (flow out), and other regional server's simulation result will have an inbound effect (flow in). And essentially, inbound and outbound effects constitute the fundamental influence to the quality of collaborative simulation. So among diverse kinds of situations, we pick a typical situation as the test situation. **Figure 11** shows this situation, we fix a diffusion source locating at the center of a region for easier analysis, and fix an outside diffusion source, and the two sources have same distance to the left margin of the measured region. So in this region, there will be both inbound and outbound effects. And the region we choose is near to the Tsugaru Strait of Japan.

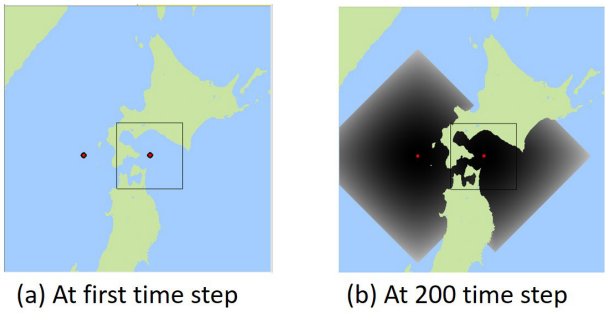

Fig. 11 The setted typical test situation.

We trace the data for 50 thousand time steps which is a reasonably long period, and set main parameters as follows: the physical size of global area is 4 times than regional area on x and y dimension separately; the grid size of C is 512 by 512, the grid size of R is 256 by 256, and the grid size of U is 1,024 by 1,024; the float precision is double type; the frequency of DQC is every 30 time steps, and the frequency of UQC is every 1,200 time steps; the geographical background can absorb the difussion in the simulation; during updating in UQC, on *Core* area of C_1, we totally believe all UQC data, and on *Outer* area of C_1 we believe

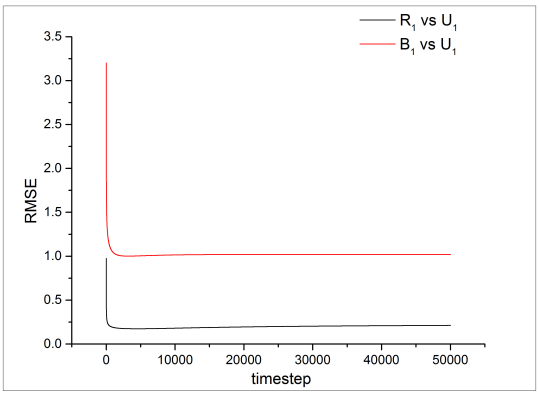

Fig. 12 The measurement result on regional server side.

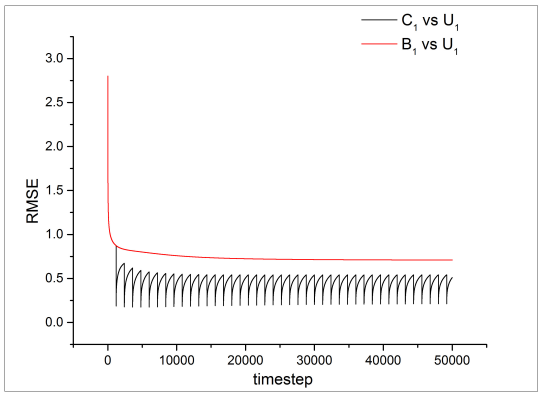

Fig. 13 The measurement result on central server side.

both UQC data and C_1 data half and a half. At the interval time steps between each DQC time step, we always use the last time interpolated boundary condition to compute.

Figure 12 shows the result of comparison on regional server side. By comparing the difference of simulation data with the result of ultimate simulation U, it shows that the result of regional simulation which utilizes collaboration is constantly more accurate than B_1. **Figure 13** shows that on central server side, the central global simulation can be refined by using UQC in the case of setting the frequency of UQC every 1,200 time steps.

As we could see in Fig. 13, there is the possibility that the simulation result user observed may contain periodic phenomenon incurred by the UQC process, this may mislead users' understanding on the simulation results. In order to avoid this misunderstanding, we could provide some additional information to help users justify the phenomenon, for example, using a progress bar to show the time from the previous UQC or DQC may be helpful.

5. Conclusion

In this paper, for high-resolution simulations and real-time interactions of users (men or machines) in wide geographical distributions, we propose a distributed collaborative simulation model. This model utilizes distributed regional servers to perform high-resolution simulations on limited areas with the help of achieving boundary conditions through collaborations among regional servers via a central server which performs a low-resolution but wider simulation to couple the regional simulations to propagate higher resolution simulation results for each other. This design enables a lot of advantages: high-resolution simulation result, real-time interactions, reliability, low cost because centralized high performance computer or supercomputer is not mandatory. Based on this model, we propose and implement a general purpose framework named Silk Road Framework to help realize such distributed collaborative simulations, which provides the bi-directional refinement of on-going simulation by exchanging adequate amount of simulation results occasionally without inducing unacceptable network delay. Through the multi-scale bi-directional refinements using DQC and UQC process, both simulations on regional servers and the central server can be refined. The availability and effectiveness of the framework and distributed collaborative simulation are demonstrated through a 2D diffusion simulation as a case study.

Currently the evaluation using the case study is relatively a very simple situation, the diffusion simulation itself is not so complicated, and the test scenario of case study is a simple configuration where the either regional servers are the same size area and not intersected. As future work, we would like to perform more evaluations and examinations in an actual remote network environment and in more complex test scenarios, and we also would like to investigate more complicated simulations.

Acknowledgments We would like to thank all the reviewers for their insightful comments for our paper. We are also grateful to the people who assisted with this work, Y. Matsuyama, Y. Yamamoto, Y. Nishimura, K. Hashimoto, and other members of HPC group at University of Fukui. This research was supported in part by JSPS KAKENHI Grants-in-Aid for Scientific Research (B) 25280042.

References

[1] Introduction to Parallel Computing (online), available from ⟨https://computing.llnl.gov/tutorials/parallel_comp/#WhyUse/⟩ (accessed 2017-06-01).
[2] Ma, K.-L.: In situ visualization at extreme scale: Challenges and opportunities, *IEEE Computer Graphics and Applications*, Vol.29, No.6, pp.14–19 (2009).
[3] Johnson, C. et al.: Interactive simulation and visualization, *IEEE Computer*, Vol.32, No.12, pp.59–65 (1999).
[4] Gubbi, J. et al.: Internet of Things (IoT): A vision, architectural elements, and future directions, *Future Generation Computer Systems*, Vol.29, No.7, pp.1645–1660 (2013).
[5] Hurlburt, G.: The internet of things of all things, *XRDS: Crossroads, The ACM Magazine for Students*, Vol.22, No.2, pp.22–26 (2015).
[6] Zhang, J. et al.: Consideration on the extension of Simulation Caching Framework to Distributed Co-simulation Environment, *Proc. Joint Conference of Hokuriku Chapters of Electrical Societies 2016* (*JHES2016, CD-ROM*), Hokuriku, Japan (2016).
[7] Zhang, J. et al.: A case study: A Distributed Collaborative Simulation using extended simulation caching framework, *Proc. High Performance Computing Symposium 2017* (*HPCS2017*), p.14 (2017).
[8] Wikipedia: Silk Road (online), available from ⟨https://en.wikipedia.org/wiki/Silk_Road⟩ (accessed 2017-06-10).
[9] Bonomi, F. et al.: Fog computing and its role in the internet of things, *Proc. 1st Edition of the MCC Workshop on Mobile Cloud Computing*, ACM (2012).
[10] Garcia Lopez, P. et al.: Edge-centric computing: Vision and challenges, *ACM SIGCOMM Computer Communication Review*, Vol.45, No.5, pp.37–42 (2015).
[11] Kitchin, R.: The real-time city? Big data and smart urbanism, *GeoJournal*, Vol.79, pp.1–14 (2014).
[12] Mone, G.: The new smart cities, *Comm. ACM*, Vol.58, No.7, pp.20–21 (2015).
[13] Distributed multi-scale weather forecase system using the Earth Simulator and edge computing for smart society (in japanese) (online), available from ⟨http://www.jamstec.go.jp/j/about/press_release/20160209/⟩ (accessed 2017-06-01).
[14] Onishi, R. et al.: About Developing of Smart Weather Forecast System

using the Earth Simulator and Edge Computing (in japanese), *Proc. High Performance Computing Symposium 2017* (*HPCS2017*), pp.45–46 (2017).
[15] Imato, Y. et al.: High-resolution, Quantitative Tsunami simulation on New Earth Simulator, *High Performance Computing Symposium 2016*, pp.1–8 (2016).
[16] Malakar, P. et al.: An adaptive framework for simulation and online remote visualization of critical climate applications in resource-constrained environments, *Proc. 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis* (*SC10*), IEEE Computer Society (2010).
[17] Debreu, L. and Blayo, E.: Two-way embedding algorithms: A review, *Ocean Dynamics*, Vol.58, No.5-6, pp.415–428 (2008).
[18] Koch, S.E. and McQueen, J.T.: A survey of nested grid techniques and their potential for use within the MASS weather prediction model, NASA Technical Memorandum 87808 (1987).
[19] Yamamoto, Y. et al.: Implementation of Simulation Caching Framework, *Journal of Information Processing*, Vol.57, No.3, pp.823–835 (2016).
[20] Yamamoto, Y. et al.: Simulation Caching Framework for Real-time Remote Sharing of Interactive Simulation *High Performance Computing Symposium 2016* (*HPCS2016*), pp.101–110 (2016).
[21] Zhang, J. et al.: Prototype Implementation of Simulation Caching framework for Multi-user Interaction, *The International Conference for High Performance Computing, Networking, Storage and Analysis* (*SC16*), Poster Session (2016).

Jiachao Zhang was born in 1987. He received his B.E. in Computer Science and Technology from Sichuan University at China in 2010 and M.E. in computer application technology from China Electronics Technology Group Corporation's Academy of Electronics and Information Technology at China in 2013. Now he is a Ph.D. student in information science at University of Fukui at Japan. His research interests include software project, parallel processing, simulation and visualization. He is a member of ACM and IPSJ.

Shinji Fukuma received his B.S. and M.S. degrees in electrical engineering from Nagaoka University of Technology in 1994 and 1996, respectively. He received the Dr. Eng. degree in Information and Control Engineering from the university in 1999. From 1999 to 2000, he was a Research Assistant at Nagaoka University of Technology as and also held a Fellow of Telecommunications Advancement Organization of Japan (TAO). From 2000 to 2004, he was an Assistant of Interdisciplinary Faculty of Science and Engineering at Shimane University. He is currently an associate professor in the Graduate School of Engineering, University of Fukui. His research themes are a data compression and a software/hardware co-design of embedded image processing systems using FPGA.

Shin-ichiro Mori was born in 1963. He received his B.E. in Electronics from Kumamoto University in 1987 and M.E. and Ph.D. in Computer Science from Kyushu University in 1989 and 1995, respectively. From 1992 to 1995, he was a research associate in the Faculty of Engineering, Kyoto University. From 1995 to 2006, he was an associate professor in the Department of Computer Science, Kyoto University. Since 2006, he has been a professor in the Department of Information Science, University of Fukui. His research interests include computer architecture, parallel processing and visualization. He is a member of IEEE, ACM, EUROGRAPHICS and IPSJ.