MGRID: A Modifiable-Grid Region Matching Approach for DDM in the HLA RTI

Shih-Hsiang Lo[1], Cheng-An Chiu[1], Fang-Ping Pai[2], Ding-Yong Hong[1] and Yeh-Ching Chung[1]
[1]Department of Computer Science, National Tsing Hua University, Taiwan
[2] Aeronautical Systems Research Division, CHUNG_SHAN Institute of Science and Technology, Taiwan
albert@sslab.cs.nthu.edu.tw, cachiu@sslab.cs.nthu.edu.tw, fppai118@gmail.com,
dyhong@sslab.cs.nthu.edu.tw and ychung@cs.nthu.edu.tw

Keywords: High Level Architecture, Data Distribution Management, Interest Management

Abstract
In a large-scale simulation Data Distribution Management (DDM) plays a prominent role in supporting simulation entities as many as possible. In order to provide a flexible solution of region matching to different simulation scenarios, we propose a region matching approach for DDM, MGRID. MGRID can gather run-time information about regions, then evaluate the information by our region matching cost model and make adjustments for present situation accordingly. MGRID has been implemented in our HLA RTI system using C++ language, which follows IEEE Standard 1516. In addition, we have implemented region-based approach, hybrid-based approach and sort-based approach for comparison. In our experimental evaluations MGRID performed well in all test cases.

1. INTRODUCTION

High Level Architecture (HLA) [9] is a general-purpose framework for distributed simulation and modeling. In this paper, we focus on Data Distribution Management (DDM), which aims to reduce unnecessary transmission between federates. As simulation objects increases significantly, DDM becomes a critical part in supporting the execution of a large-scale simulation. Through DDM services defined in the HLA Interface Specification, federates can clearly state data requirements, which are defined as regions in the HLA. Specifically, a federate can publish data in specified regions (i.e. publishing regions) which it has influences over. Likewise, a federate can subscribe data in specified regions (i.e. subscription regions) which it has an interest in. Afterwards, RTI will deliver data from the publishing federate to the subscription federate only if the publishing regions overlap with the subscription regions. The process of finding overlap between publishing and subscription regions is called region matching.

In general, region matching algorithms can be classified as region-based approach [15, 16], grid-based approach [2, 5, 6], hybrid-based approach [1, 3, 4, 14] and sort-based approach [8, 11, 12]. The proposed approaches, however, can not be used to deal with more complex simulation scenarios. For example, a large-scale military simulation, airplanes, warships, tanks, soldiers, etc. could be involved in simulation and interesting regions of these simulation entities may be changed or resized at day or at night for realistic simulation. Considering such simulations, we adopt a flexible approach that involves gathering profiling information about regions, processing profiling information and making adjustment for new condition at run-time easily. We therefore propose a cost model, along with profiling information, to evaluate the cost of region matching. Based on the cost model, we present a dynamic hybrid-based approach that can adjust the size of grid cells dynamically.

To evaluate the performance, we implemented our approach with comparisons to the region-based approach in [15], the hybrid-based approach in [14] and the sort-based approach in [11] in the HLA RTI that follows IEEE Standard 1516. The experimental results show that the proposed dynamic hybrid-based approach can successfully alter the size of grid cells to a proper one without much computation overhead and achieve better performances than those of the region-based approach, the hybrid-based approach and the sorted-based approach in most of test cases.

The organization of this paper is described as follows. In Section 2, we briefly discuss the approaches reported in the literature. The dynamic hybrid-based approach is presented in Section 3. Section 4 gives the experimental results. We conclude our work and present future works in Section 5.

2. RELATED WORK

2.1. Region-based Approach

In this approach, publishing regions need to be compared with all subscription regions for finding overlap between these regions [15, 16]. Its idea is very straightforward and the cost of the computation is quadratic in terms of number of regions. In a large-spatial simulation, this approach takes most of time to compare unrelated regions (i.e. the regions are not close in terms of distance). On the other hand, it can get better performance when all regions highly overlap with other regions [7, 12].

2.2. Grid-based Approach

In [2, 5, 6], A. Boukerche *et al*. proposed a method to reduce high computing operations of the region-based approach. This approach partitions *N*-dimension data space into grid cells and maps all regions on to these grid cells. If a publishing region and a subscription region are mapped on to the same grid cell, both regions are presumed to overlap with each other without carrying out exact region matching. This mechanism requires much less computation than region-based approach and hybrid-based approach. In fact, it will result in unnecessary network traffic [13]. Moreover, the additional computation is required to filter unnecessary messages. Gray Tan *et al*. in [14] suggested that if the overhead of computation to do region matching is less than the overhead of communication and filtering, the better way is to carry out exact region matching instead of directly delivering unnecessary messages to receivers. Additionally, with the advance of computing capability, to do exact region matching is a better policy.

2.3. Hybrid-based Approach

Hybrid-based approach [3, 4, 14] combines the concepts of region-based and grid-based approach, thereby removing the irrelevant messages generated and reducing computational overhead as well. For the case of small size of grid cells, a subscription region is compared with a publishing region several times if these two regions overlap with more than one grid cell at the same time. For the case of large size of grid cells, a subscription region and a publishing region are likely to locate in the same grid cell whereas it is possible that two regions are not overlapped. As a result, the chosen size of grid cells has substantial impact on the performance of the hybrid-based approach.

Rassul Ayani *et al*. in [1] proposed a detailed model to formulate the cost of grid-based filtering with regard to system factors, model factors and platform factors. By approximating the equation of the cost of grid-based filtering, the optimized size of grid cells can be solved. If a simulation scenario is unknown in advance or could be changed at run-time (i.e. those factors can not be on hand), the size of grid cells calculated by this approach is not adequate for such a case.

2.4. Sort-based Approach

C. Raczy *et al*. in [12] employ a novel approach to deal with region matching. The end points of each dimension of each region are recorded in sorted lists. For each dimension, it scans the sorted lists to get *N*-size arrays of bit vectors which record the overlap information between *N* regions. After that, it merges the overlap information of each dimension to attain the overall overlap information. It uses bit vector to manipulate insertion, removing and locating of data, thereby enhancing the speed of region matching. This approach is quite efficiently in region matching because the sorting process and overlap information can be accomplished before the execution of simulation. However, this approach is not applied to simulations where regions will be altered at run-time. For this reason, a dynamic sort-based algorithm for region matching in a large-spatial environment is presented in [11]. This approach reduces the data storage requirement in comparison with the static sort-based approach [12] and supports alteration of regions during simulation. Once the size or the location of regions is altered, this approach shifts the end points of regions from old positions to new positions and then scans the sorted end points within a dynamic range. This dynamic range is defined in accordance with the end points of current regions and the maximum size of publishing and subscription regions. Larger size of regions a simulation has, more time it spends in scanning end points. With our experiment results, it is not suitable for some case where a simulation comprises a large range of sizes of regions.

In [8], authors proposed an algorithm for DDM, P-Pruning. This approach builds a Region Projection Array to store information about regions. The information of a region will be stored at some certain elements according to the values of end points of the region, i.e. Bucket Sort. End points of all regions are being sorted until accomplishing inserting information of all regions. To find overlap information of a region, this approach will scan some elements to find whether any region overlaps with this region. The principle of this approach is similar to the works in [11, 12]. The sorting procedure of this approach is quite fast, albeit with scalability problems.

3. DYNAMIC HYBRID-BASED APPROACH

3.1. The Matching Cost Model

In the following, we first give several definitions used in this paper.

Definition 1: An *N*-dimensional space is defined as *SPACEN*=f, where D_i is the *i*th dimension. The size of $SPACE_N$ is defined as $DL_1 * DL_2 * \ldots * DL_N$, where DL_i is the length of D_i.

Definition 2: A $SPACE_N$ is partitioned by each dimension and forms a set of equivalent grid cells, denoted as $C=\{c_1,c_2,\ldots,c_N\}$. The size of a grid cell is $cS=cL_1 * cL_2 * \ldots * cL_N$, where cL_i is the length of a grid cell in the ith dimension.

Definition 3: Given a $SPACE_N$ and *cS*, the total number of grid cells can be defined as

$$TC(SPANCE_N, cS) = \prod_{i=1}^{N} \left(\frac{DL_i}{cL_i}\right). \quad (1)$$

Definition 4: In an *N*-dimensional space, a set of publishing regions is defined as $P=\{p_1,p_2,\ldots,p_m\}$ and the number of

publishing regions in a set is defined as $NumReg(P) = m$; a set of subscription regions is defined as $S=\{s_1,s_2,\ldots,s_n\}$ and the number of subscription regions is defined as $NumReg(S) = n$. We define p_kL_i and s_kL_i as the length of ith dimension of p_k and s_k, respectively. Thus, the size of p_k is $p_kS=p_kL_1 * p_kL_2 * \ldots * p_kL_N$ and the size of s_k is $s_kS=s_kL_1 * s_kL_2 * \ldots * s_kL_N$.

Definition 5: Given a set of publishing regions P, $AvgReg(P)$ gets an average region of P and the length of the ith dimension of the region p_{avg} is defined as follows:

$$AvgReg(P) = p_{avg}; p_{avg}L_i = \frac{\sum_{j=1}^{m} p_jL_i}{NumReg(P)}, i = 1 \sim N \quad (2)$$

Definition 6: Given a set of subscription regions S, $AvgReg(S)$ gets an average region of S and the length of the ith dimension of this region s_{avg}, is defined as follows:

$$AvgReg(S) = s_{avg}; s_{avg}L_i = \frac{\sum_{j=1}^{n} s_jL_i}{NumReg(S)}, i = 1 \sim N \quad (3)$$

In the following, we will give notations used in this paper.

Definition 7: Given a $SPACE_N$, cS and p_k, we can approximately estimate the number of the grid cells with which p_k overlaps by dividing the size of a publishing region by the size grid cells as follows:

$$ECP(SPACE_N, cS, p_k) = \prod_{i=1}^{N} (\frac{p_kL_i}{cL_i} + 1) \quad (4)$$

Definition 8: Given a $SPACE_N$, cS and s_k, we can approximately estimate the number of the grid cells with which s_k overlaps by dividing the size of a subscription region by the size grid cells as follows:

$$ECS(SPACE_N, cS, s_k) = \prod_{i=1}^{N} (\frac{s_kL_i}{cL_i} + 1) \quad (5)$$

Definition 9: Assume that all publishing regions, denoted as P_{all}, will be evenly distributed in the $SPACE_N$. Thus the average number of the publishing regions on a cell is calculated as follows:

$$ACP(SPACE_N, cS, P_{all}) = \frac{NumReg(P_{all}) * ECP(SPACE_N, cS, AvgReg(P_{all}))}{TC(SPACE_N, cS)} \quad (6)$$

Definition 10: Assume that all subscription regions, denoted as S_{all}, will be evenly distributed in the $SPACE_N$. Thus the average number of the subscription regions on a cell is calculated as follows:

$$ACS(SPACE_N, cS, S_{all}) = \frac{NumReg(S_{all}) * ECS(SPACE_N, cS, AvgReg(S_{all}))}{TC(SPACE_N, cS)} \quad (7)$$

Definition 11: The number of the subscription regions with which a publishing region p_k overlaps is estimated as follows:

$$unitPMatchS(SPACE_N, cS, S_{all}, p_k) = ECP(SPACE_N, cS, p_k) * ACS(SPACE_N, cS, S_{all}) \quad (8)$$

Definition 12: The number of the publishing regions with which a subscription region s_k overlaps is estimated as follows:

$$unitSMatchP(SPACE_N, cS, P_{all}, s_k) = ECS(SPACE_N, cS, s_k) * ACP(SPACE_N, cS, P_{all}) \quad (9)$$

Definition 13: Let P_{upd} be a set of updated publishing regions, i.e. these regions need to be recalculated overlapping. The number of the subscription regions with which P_{upd} overlap is calculated as follows:

$$PMatchS(SPACE_N, cS, S_{all}, P_{upd}) = NumReg(P_{upd}) * unitPMatchS(SPACE_N, cS, S_{all}, avgP(P_{upd})) \quad (10)$$

Definition 14: Let S_{upd} be a set of updated publishing regions. The number of the publishing regions with which S_{upd} overlap is calculated as follows:

$$SMatchP(SPACE_N, cS, P_{all}, S_{upd}) = NumReg(S_{upd}) * unitSMatchP(SPACE_N, cS, P_{all}, avgS(S_{upd})) \quad (11)$$

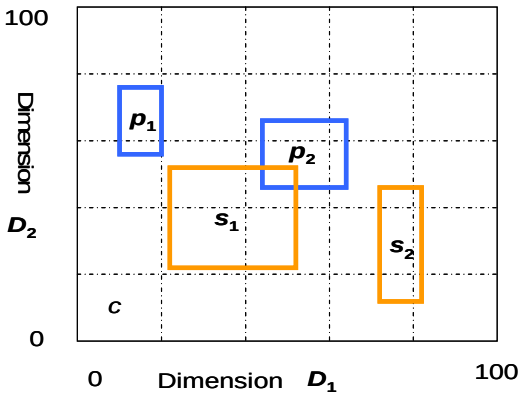

Figure 1. An example of 2-dimensional space with four regions

We now give an example to explain above definitions. Figure 1 depicts $SPACE_2=\{D_1,D_2\}$ with the size 100x100, all publishing regions, $P_{all}=\{p_1,p_2\}$, all subscription regions, $S_{all}=\{s_1,s_2\}$, and cS=20x20. In Figure 1, there exist four regions: p_1 with size 10x20, p_2 with size 20x20, s_1 with size 30x30 and s_2 with size 10x30. With equations from definition 3 to 10, we can get

$TC(SPACE_2,cS)$=25,

$AvgReg(P_{all})=p_{avg}$, $p_{avg}S$=15x20,

$AvgReg(S_{all})=s_{avg}$, $s_{avg}S$=20x30,
$ECP(SPACE_2,cS,avgS(P_1))$=3,
$ECP(SPACE_2,cS,avgS(P_{all}))$=3.5,
$ECS(SPACE_2,cS,avgS(S_{all}))$=5,
$ACP(SPACE_2,cS,P_{all})$=0.28,
$ACS(SPACE_2,cS,S_{all})$=0.4.

Once a set of updated (i.e. region updates) publishing regions, $P_{upd}=\{p_1\}$, is required to find the overlap information of P_{upd}, this operation averagely costs $PMatchS(SPACE_2,cS,S_{all},P_{upd})$=1.2 comparisons according to Definition 13. Similarly, once a set of updated subscription regions $S_{upd}=\{s_1,s_2\}$ is required to find the overlap information of S_{upd}, that averagely costs $SMatchP(SPACE_2,cS,P_{all},S_{upd})$=2.8 comparisons according to Definition 14.

3.2. Dynamic Hybrid-based Algorithm

Figure 2 illustrates the conceptual relationship of the dynamic hybrid-based approach. In the initial stage, DDMInit component creates two grids: one grid is as a *currentGrid*; the other grid is as a *previousGrid*. The GridCell component is responsible for partitioning *N*-dimensional data space into grid cells and dealing with region updates (i.e. mapping updated publishing or subscription regions on to grid cells). In the beginning, publishing and subscription regions are only mapped on to the *currentGrid*. The Match component scans the grid cells with which a publishing region (or a subscription region) overlaps. For each grid cell the Match component invokes matching procedure, i.e. region-based approach, to find overlap. The Matching Cost Model component collects information, including sizes of publishing region and subscription regions and the size of current grid cells, to calculate the cost of region matching. The Matching Cost Model component will execute at a fixed time interval. Given the result from the Matching Cost component, EvaluateAdjust component determines whether the size of current grid cells should be adjusted.

The EvaluateAdjust component first gets an average the EvaluateAdjust component first gets an average region of updated publishing regions and an average region of updated subscription regions at run-time. Next it evaluates the cost of region matching of *currentGrid* in accordance with the matching cost model. The new size of grid cells is estimated based on the ratio of matching cost of publishing regions and subscription regions in *currentGrid*. With the new size of grid cells, new matching cost of the new size of adjusted grid cells can be calculated . In the EvaluateAdjust component, a threshold, denoted as *thre*, is set as the minimum percentage of the improved cost of region matching. Once the percentage of improved matching cost is greater than *thre*, it performs the following steps: 1) swap the *currentGrid* with the *previousGrid*, 2) partition the *currentGrid* with new adjusted size and 3) re-map all regions in the *previousGrid* on to the *currentGrid*.

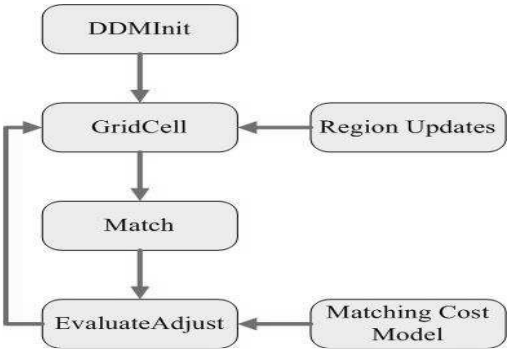

Figure 2. The flowchart of the MGRID approach

4. PERFORMANCE EVALUATION

4.1. Experimental Assumption and Platform

This section demonstrates the performance results of our proposed dynamic hybrid-based DDM approach (MGRID), compared with the region-based approach presented in [15] (REGION), the hybrid-based approach presented in [14] (HYBRID) and the sort-based approach presented in [11] (DSORT). In this paper we concerns processing cost in different region matching algorithms and therefore grid-based approaches are not considered.

We use 8 HP Blade Servers as our experimental platform. Each node of the HP Blade Server has two quardcore Xeon 2.66 GHz CPUs, 8 Gbytes DRAM and runs Linux operating system with kernel version 2.6.18. All machines are connected with Gigabit Ethernet network. We have implemented a RTI system following the specification of IEEE 1516 standard in C++ language. This RTI system is of client-server architecture and the matching algorithms runs at server nodes.

We design simulation scenarios to evaluate the performance of region matching algorithms. For all test cases, it simulates at most 20000 simulation objects distributed to 20 federates in a 10000-meters*10000-meter 2-dimensional battlefield. Each simulation object publishes its position attribute within a 2-dimension region and also subscribes object's position attribute within a 2-dimension region. In other words, each simulation object will obtain others' position attributes if regions are overlapped. In each test case, the size of both regions is set to the same value and the size of a region is denoted as *RS*. Initially, the simulation objects are placed at random locations in the battlefield. For each time step, the moving direction of each simulation object is randomly assigned to one in the five possibilities (Hold, North, South, East and West). During the simulation, the moving distance is set to half the region size of the simulation object. For all test cases, the time of region matching is averaged over a period of 30 time steps. Note that the hybrid-based approach

must initiate the size of grid cells and hence we evaluate three cases for hybrid-based approach: small *cS*=50m*50m (HYBRID(50)), medium *cS*=500m*500m (HYBRID(500)) and large *cS*=5000m*5000m (HYBRID(5000)).

4.2. The Performance of Single-Type Region

Figure 3 and 4 illustrate the comparison results of *RS*=50m*50m and *RS*=500m*500m, respectively. The x-axis represents number of regions in a simulation and the y-axis represents the time (in millisecond unit) which region matching algorithms take.

The best performances in Figure 3 and 4 are the HYBRID(50) and HYBRID(500), respectively. Since the *cS* of our approach can be adjusted to new one (49m*49m in Figure 3 or 493m*493m in Figure 4) at run-time, the results of our approach are very close to those best results with a slight overhead introduced by running cost model and adapting to new *cS*.

As *RS* becomes large, smaller *cS* will lead to repeat region matching in the hybrid-based approach such as the result of HYBRID(50) in Figure 4. Also, we can see that the performance of HYBRID(5000) is not as good as HYBRID(50) and HYBRID(500) in Figure 3. Evidently, the chosen size of grid cells is vital to the performance of hybrid-based approach.

In Figure 3, the performance of DSORT is better than those of HYBRID(5000) and REGION. However, in Figure 4 the result of DSORT is similar to those of HYBRID(5000) and REGION. For the case of *RS*=500m*500m, the scanning range is relatively large. The falling performance is attributed to the scanning range, as stated in Section 2. In addition, the larger regions are likely overlapped with each other. As a result, it costs more time in scanning.

4.3. The Performance of Compound-Type Regions

In this case, we test the performance of compound-type regions and measure the matching time of varied number of regions from 2000 to 40000, i.e. 1000 to 20000 objects. We set *RS* to {10m*10m, 20m*20m, 30m*30m, 40m*40m, 50m*50m} and evenly assign the *RS* to all regions. That is, in the case of total 40000 regions, 8000 regions are of size 10m*10m, 8000 regions are of size 20*20 and so on. Figure 5 illustrates the comparison results. The comparison results are almost the same as those in Figure 3 except that in two cases where the number of regions are 2000 and 4000 the performance of MGRID is not good as the performance of HYBRID(500). Owing to the smaller regions in existence, MGIRD will adjust the size of grid cells to 29m*29m. That results in repeat computation while doing region matching for larger regions (i.e. 40m*40m and 50*50m regions). However, in Figure 5 the difference between HYBRID(500) and MGRID is small: 5ms.

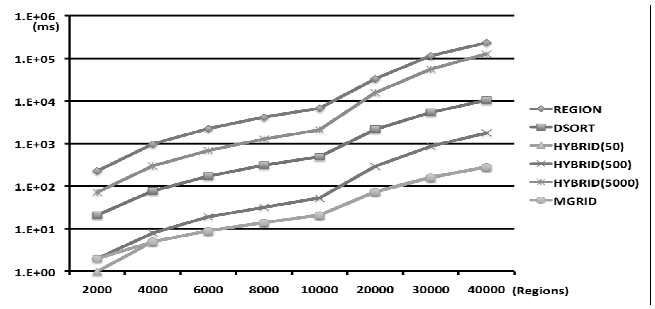

Figure 3. Single-Type Regions, *RS*=50*50

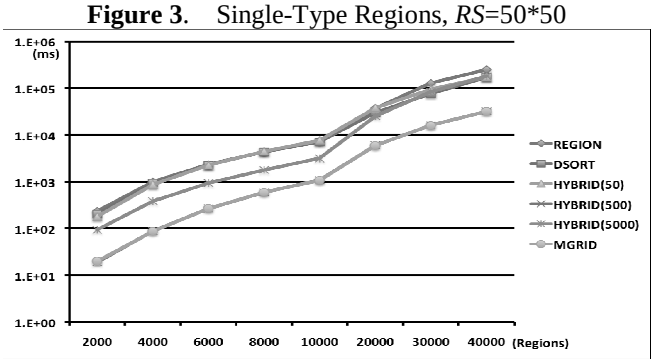

Figure 4. Single-Type Regions, *RS*=500*500

We also evaluate the performance of compound type of region for larger regions. Similarly, *RS* is set to {100m*100m, 200m*200m, 300m*300m, 400m*400m, 500m*500m} and there are the same number of regions for each size. The results are as shown in Figure 6. The result of HYBRID(500) is better than our approach, but in the cases where the number of regions is greater than 20000 the performance gap is not obvious. The size of grid cells of MGRID is set to 290m*290m.

4.4. The Performance of Flexible-Type Regions

In this test case, we consider that the size of all regions will be altered in federates as time steps passed. For every two time steps, the size of all regions will be resized according to the following sequence: {50m*50m, 150m*150m, 250m*250m, …, 750m*750m, 650m*650m, 550m*550m, …,150m*150m, 50m*50m}. We set the number of regions to 40000. The x-axis represents simulation time and the y-axis represents time in millisecond unit.

Figure 7 shows the comparison results. In Figure 7 the results of MGRID keep at quite good performance no matter how size of regions is altered. It shows that MGRID can adapt with new situations. As for hybrid-based approaches, HYBRID(50) presents good performance in the beginning but when the size of regions becomes larger, its performance is even worse than REGION. HYBRID(5000) and REGION have similar results because HYBRID(5000) partitions *2*-dimensional space into four grid cells, that is not much difference compared with REGION. As expected, REGION is insensitive to size of region. The result of DSORT shows

that its performance is connected with size of region as well. Larger region it has, more time it takes.

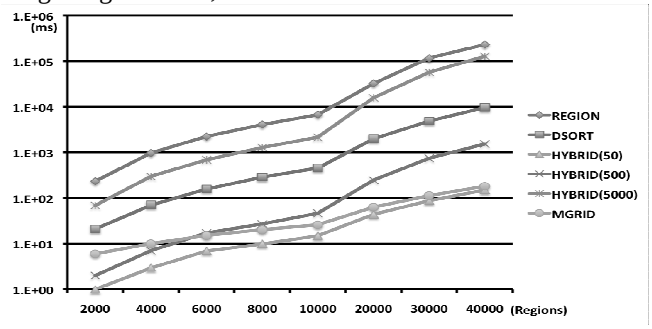

Figure 5. Compound-Type Regions, *RS*={10*10, 20*20, …, 50*50}

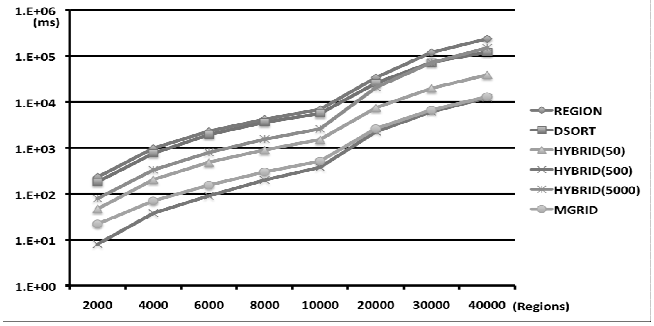

Figure 6. Compound-Type Regions, *RS*={100*100, 200*200, …, 500*500}

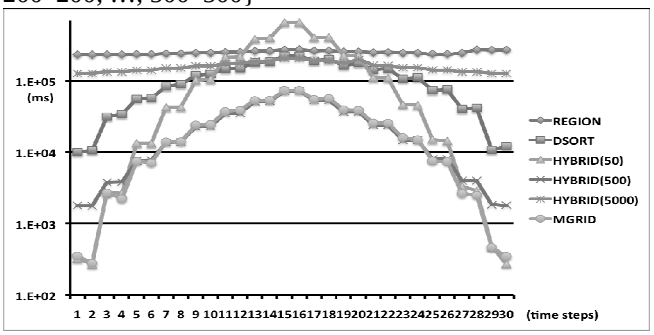

Figure 7. Flexible-Type Regions

5. CONCLUSION

Efficient data distribution is an important issue in large scale distributed simulations with hundreds of thousands of entities or more. DDM services in the HLA RTI provide a good mechanism to reduce unnecessary transmission and irrelevant reception over the network. To ensure only necessary data are transmitted, matching between publishing and subscription regions is required. In this paper we propose an approach, MGRID, to address the issue of choosing a proper size of grid cells. The MGRID uses the matching cost model with the profiling information about regions to evaluate the cost of region matching and dynamically adjusts the size of grid cells. The experimental results show that our proposed approach is more efficient than the region-based, the hybrid-based and the sort-based approach in most of cases.

Acknowledgments. The work of this paper is partially supported by CHUNG-SHAN Institute of Science & Technology under contract XW97184P. The authors would like to thank anonymous referees for their comments.

References

[1] R. Ayani, F. Moradi, and G. Tan, "Optimizing cell-size in grid-based DDM," in Proceedings of the fourteenth workshop on Parallel and distributed simulation Bologna, Italy: IEEE Computer Society, 2000.

[2] A. Boukerche, C. Dzermajko, and K. Lu, "Dynamic Grid-Based vs Region-Based Data Distribution Management in Multi-Resolution Large-Scale Distributed Systems," in Proceedings of the 18th International Parallel & Distributed Processing Symposium, 2004.

[3] A. Boukerche, N. McGraw, C. Dzermajko, and K. Lu, "Grid-Filtered Region-Based Data Distribution Management in Large-Scale Distributed Simulation Systems," in Proceedings of the 38th Annual Simulation Symposium, 2005, pp. 259-266.

[4] A. Boukerche, N. J. McGraw, and R. B. Araujo, "A grid-filtered region-based approach to support synchronization in large-scale distributed interactive virtual environments," in Parallel Processing, 2005. ICPP 2005 Workshops. International Conference Workshops on, 2005, pp. 525-530.

[5] A. Boukerche and A. J. Roy, "Dynamic Grid-Based Approach to Data Distribution Management," Journal of Parallel and Distributed Computing, pp. 366-392, 2002.

[6] A. Boukerche and A. J. Roy, "Dynamic Grid Based Multicast Group Assignment in Data Distribution Management," in Proceeding Fourth IEEE International Workshop on Distributed Simulation and Real-time Applications Workshop 2000, 2000, pp. 27-34.

[7] J. Y. C Raczy, G Tan, SC Tay, R Ayani, "Adaptive data distribution management for HLA RTI," in Proceedings of 2002 European Simulation Interoperability, 2002.

[8] P. Gupta and R. K. Guha, "A Comparative Study of Data Distribution Management Algorithms," The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology, vol. Volume 4 Number 2, 2007.

[9] IEEE, "IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) -- Framework and Rules." vol. 1516-2000, 2000.

[10] IEEE, "IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) -- Interface Specification." vol. 1516.3-2000, 2000.

[11] K. Pan, S. J. Turner, W. Cai, and Z. Li, "An Efficient Sort-Based DDM Matching Algorithm for HLA Applications with a Large Spatial Environment," in Proceedings of the 21st International Workshop on Principles of Advanced and Distributed Simulation: IEEE Computer Society, 2007.

[12] C. Raczy, G. Tan, and J. Yu, "A sort-based DDM matching algorithm for HLA," ACM Trans. Model. Comput. Simul., vol. 15, pp. 14-38, 2005.

[13] G. Tan, R. Ayani, Z. YuSong, and F. Moradi, "Grid-based data management in distributed simulation," in Simulation Symposium, 2000. (SS 2000) Proceedings. 33rd Annual, 2000, pp. 7-13.

[14] G. Tan, Z. Yusong, and R. Ayani, "A hybrid approach to data distribution management," in Distributed Simulation and Real-Time Applications, 2000. (DS-RT 2000). Proceedings. Fourth IEEE International Workshop on, 2000, pp. 55-61.

[15] D. J. Van Hook and J. O. Calvin, "Data Distribution Management in RTI 1.3," in Proceedings of the Spring Simulation Interoperability Workshop, Orlando, 1998.

[16] D. Wood, "Implementation of DDM in the MAK High Performance RTI," in Proceedings of the Simulation Interoperability Workshop.