Proceedings of the 2009 Winter Simulation Conference
M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin and R. G. Ingalls, eds.

PREDICTIVE ALGORITHMS FOR AGGREGATION AND DISAGGREGATION IN MIXED MODE SIMULATION

Benjamin Yuan Wei Chua
Malcolm Yoke Hean Low

School of Computer Engineering
Nanyang Technological University
Nanyang Avenue, Singapore 639798

ABSTRACT

One of the issues in mixed mode simulation is the need to achieve a believable and valid state of interoperability within the simulation itself. This is achieved in part through aggregation/disaggregation, Multi Resolution Entities or other methods, which deal with the interoperability and believability issues with different amounts of success. These approaches can be improved with the inclusion of predictive algorithms that can reduce the amount of aggregation/disaggregation in dense or thrashing scenarios. In this paper, we discuss the issues of consistency in mixed mode simulation in the context of the High Level Architecture and proposed a set of predictive algorithms to improve its efficiency. We carried out a set of experiments using these algorithms in a mixed mode simulation to assess their effects on consistency and efficiency. The experimental results show that the algorithms can improve the simulation performance by reducing the amount of aggregation/disaggregation in dense interaction scenarios.

1 INTRODUCTION

One of the issues with multi-resolution simulations is the need to achieve a believable and valid state of interoperability. Some simulations, such as the Joint Conflict And Tactical Simulation (JCATS) and the Joint Theatre Level Simulation (JTLS) achieve some degree of interoperability using the HLA RTI (Bowers and Prochnow 2003). This is achieved through the processes of aggregation and disaggregation in which represented objects convert between High Resolution Entities (HREs) and Low Resolution Entities (LRE) on a need basis. Issues arising from the aggregation and disaggregation processes, e.g. consistency and thrashing, and some of their possible solutions have been discussed in (Reynolds and Natrajan 1997, Natrajan 2000). We note, however, that there is still an unavoidable reliance on the process of aggregation/disaggregation in these solutions. In the case of Multi Resolution Entities (Natrajan et al. 1997), a similar mechanism for attribute regeneration is used when a multi-resolution simulation is carried out in space constrained situations.

In this paper, we reinvestigate the issue of consistency in a simulation experiment using HREs, LREs and Multi Resolution Entities as well as investigate a predictive algorithm approach to mitigate the aggregation/disaggregation problem in simulation efficiency in which the worst case scenario is thrashing.

The remainder of this paper is organized as follows. Section 2 briefly introduces aggregation and disaggregation, multi resolution modeling, multi resolution entities, data distribution management and predictive algorithms. Section 3 addresses the consistency and performance issues that were investigated in further detail. Section 4 details the algorithms used in testing the issues described in Section 3. Section 5 describes the experimental scenario in which the algorithms were tested. Section 6 presents the results of the experiment. And finally, Section 7 presents our conclusions and possible related work in the future.

2 RELATED WORK

Distributed simulation is an evolving field requiring the integration of many different technologies. In this section we cover some of the techniques and design ideas developed in previous work that were studied and used in our experiments.

2.1 High Level Architecture

The High Level Architecture (HLA) is an architecture for reuse and interoperation of simulations (DMSO 1998). It is an IEEE specification standard for simulation interoperability in a distributed environment. The HLA describes distributed simulations as a HLA 'federation' which consists of several functional components: individual simulations termed as 'federates'.

The HLA Runtime Infrastructure (HLA RTI) is a key component that enables simulations to communicate with each other. It is the system through which simulations pass messages to one another. The HLA RTI also provides a number of general purpose features for mixed mode simulations.

2.2 Multi Resolution Modeling

Multi Resolution Modeling (MRM) is defined as (1) building a single model with alternative user modes having different levels of resolution for the same phenomena; (2) building an integrated family of two or more mutually consistent models of the same phenomena at different levels of resolution; or (3) both (Davis and Tolk 2007).

Models are required to function at different levels of detail due to the strengths and weaknesses of the models at different levels of resolution (Davis and Bigelow 2002). Different levels of resolution provide different perspectives to the same problem.

A purpose of MRM is the efficiency of communication: to reduce overheads in communication when a high level of detail is unnecessary, and to reverse the process when a situation arises which requires the high level of detail. An example would be combining a number of tanks into a tank group for ease of communication of orders and the returning of the tanks to their individual control states when they enter combat with other individually controlled vehicles to facilitate combat calculations.

One of the ways MRM is achieved is through the processes of Aggregation and Disaggregation. This will be described in the following two subsections.

2.3 Aggregation and Disaggregation

The process of aggregation is one which a number of HREs are combined into LREs of a higher level of abstraction (Natrajan 2000). In this state, the HRE object models are either rendered inactive, slaved to the operations of the LRE or deleted.

Disaggregation would be the opposite process where the LRE is split into its component parts and the HREs are individually communicated with. Disaggregation forces the platoon to decompose into individual soldiers in situations that require greater precision (e.g. simulated combat at the individual level with units controlled by another simulation federate).

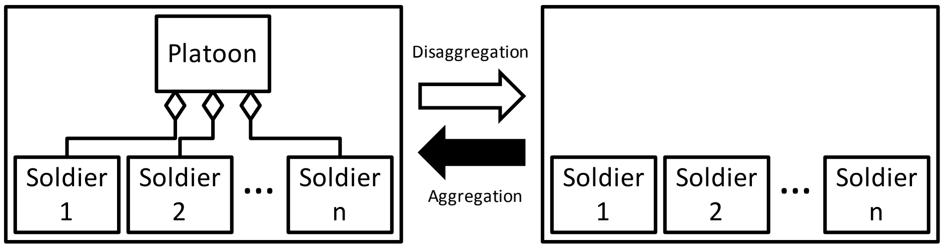


Figure 1: An example of the process of Aggregation and Disaggregation

Figure 1 shows conceptually that several soldier entities can be aggregated to form a platoon LRE. Likewise a platoon LRE can be disaggregated into its component soldiers.

In present day, simulating the basic HRE to LRE aggregation/disaggregation model is rare as data consistency could be lost through repetitive aggregation and disaggregation (Natrajan et al. 1997). Instead other methods such as composite model are used. One of the composite models which combine features of HREs and LREs is the Multi Resolution Entity.

2.4 Thrashing

Thrashing in the context of mixed mode simulation is the extreme situation where aggregation and disaggregation occurs so frequently, e.g. on every time iteration of the simulation, that the overheads for these two processes actually hinder the performance of the simulation (Reynolds and Natrajan 1997). Thrashing occurs in a spatial simulation when a relevant HRE enters the aggregation/disaggregation range of an LRE, or needs to interact with the LRE.

2.5 Multi Resolution Entities

Multi Resolution Entities or MREs are hybrid objects that combine features of both LREs and HREs (Natrajan et al. 1997). MREs maintain the information of both disaggregate and aggregate models at the same time, and therefore require more memory than LREs and HREs being modeled separately. But maintaining correctness and consistency of data is easier with MREs as a consistency checker function is called periodically to update relevant attributes even if that level of resolution of the model is not currently active.

For a fully implemented MRE, there is no need to generate the data of a different level of resolution. Instead, what would happen during aggregation and disaggregation intervals is replaced with a choice of which resolution to broadcast the data in; to select which attributes and control mechanisms should be active for the next point in the simulation.

While MREs can solve the consistency and simulation efficiency issues, they also require more memory and are more complex than HREs and LREs. A fully implemented MRE requires the space equivalent of an HRE and all its LREs at all times of the simulation. Regardless of which mode the MRE is operating in at the moment, more data have to be tracked and managed.

2.6 Data Distribution Management

Data Distribution Management (DDM) is a service provided by the HLA RTI to limit the amount of messages passed between its individual simulators (Van Hook and Calvin 1998). These individual simulators can use DDM to parse the entities that they subscribe to, filtering out messages about the entities outside their scope of interests. There are many other approaches to DDM (Kumova 2005, Tacic and Fujimoto 1998). The efficiency of the more common DDM methods, e.g. Region and Grid based DDM are investigated in (Boukerche and Dzermajko 2004).

Figures 2 and 3 show how a simulation in a distributed environment that subscribes to the same entity class might work, with or without DDM respectively. Sending simulators are the simulators that generate data about the one class of object that the receiving simulator is interested in.

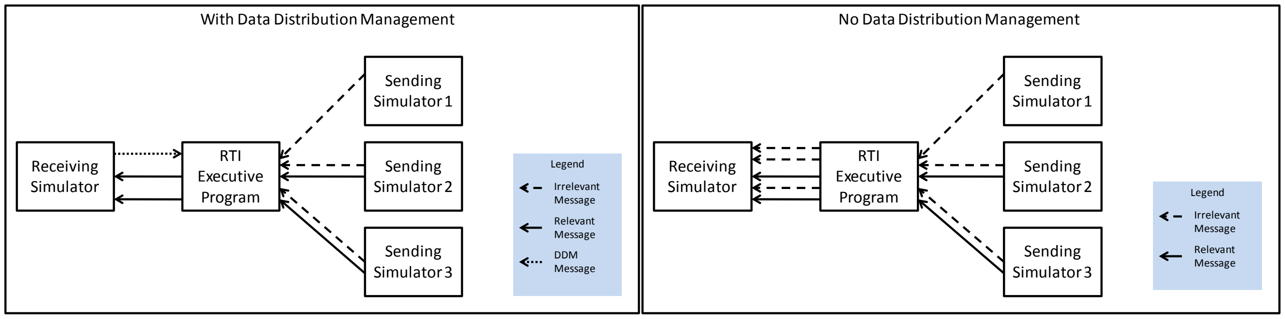


Figure 2: A distributed simulation with and without Data Distribution Management

If no DDM were used, all messages about this class of objects would be sent to the Receiving Simulator even if the Receiving simulator did not need the information. With DDM implemented, the receiving simulator would be able to send a DDM message to the RTI Execution Program, specifying what ranges of data is relevant to it. The RTI Executive Program would then filter irrelevant updates from reaching the receiving simulator. In this case, the receiving simulator is only interested in 1 message each from sending simulators 2 and 3. The RTI Executive Program filters out all the other messages.

2.7 Predictive algorithms

To the best of our knowledge, predictive algorithms have not been used to control aggregation/disaggregation. However, predictive algorithms have been used in distributed simulations to reduce network message load. One of these predictive algorithms is dead reckoning, a technique in which future positions of objects in a spatial simulation are estimated by interpolating from their current velocity and position (Zhang et al. 2006, Cai et al. 1999). The error of the interpolated value from the measured value from the next update is used to determine the frequency of future updates.

While dead reckoning is not used in this project, the predictive algorithms investigated use the same methodology: they collect historical data to decide future actions taken by the simulation. The motivation of predicting aggregation and disaggregation in this manner is to produce a simulation that can decide if the computational tradeoff between an LRE reaggregating immediately (after it is no longer necessary to stay disaggregated) and simply waiting in its disaggregate state to its next disaggregation is worth it.

3 ISSUES OF MIXED MODE FEDERATION

Regardless of what scenario is to be modeled, simulations operating on different resolutions have to deal with a set of common issues, including data consistency and simulation performance.

3.1 Consistency

Consistency is an issue of MRM simulations that involves the accuracy and believability of information produced by aggregation/disaggregation (Reynolds and Natrajan 1997, Natrajan 2000). Problems with consistency could occur when an LRE meets an HRE and has to disaggregate to interact with it fruitfully.

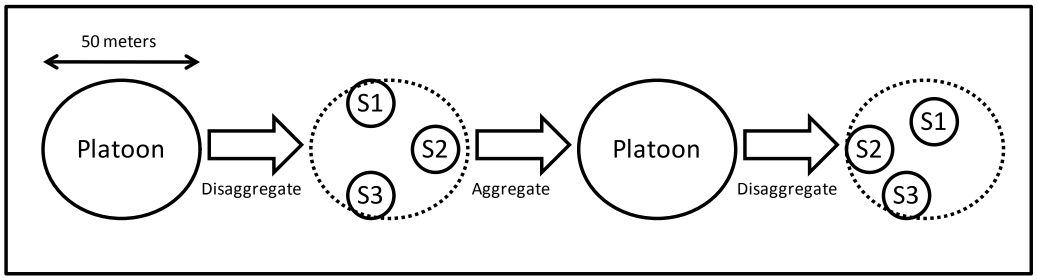


Figure 3: An example of mapping inconsistency in a simple MRM simulation

Figure 3 illustrates spatial mapping inconsistency where the physically impossible occurs due to rapid aggregation and disaggregation. Soldiers S1, S2 and S3 are deleted and recreated with random positions each time the platoon LRE aggregates and disaggregates. Assume in the above scenario each arrow represents a 1 second time frame. That would mean that the soldier S2 had 'jumped' an impossible distance in a short time.

For this research, we were primarily concerned with observing spatial mapping inconsistency as it is the most obvious inconsistency phenomenon. We observed that the inconsistency problems were most evident during periods of rapid aggregation and disaggregation. The main problems consisted of the center of gravity of LREs shifting sporadically over a short period of time, causing HREs 'jumping' around impossibly (as shown in Figure 3).

3.2 Performance

The performance of a distributed simulation is a measurement of the effectiveness of its resource utilization. The performance, or simulation efficiency, can be measured by the passage of logical time, the amount of memory space utilized, the number of RTI messages received and sent by the simulation, and the number of aggregation/disaggregation in the simulation.

For this research, we define interaction density as the amount of interaction (or aggregation/disaggregation occurring) in a simulation with respect to the amount of real world entities (LREs, HREs) being modeled in the simulation. For example, suppose a single HLA federate sends and receives an average of 10 messages per iteration cycle. In an HLA federation which involves 1000 entities, the interaction density of that federate would be considered relatively sparse. However if it was part of an HLA federation which involves 50 entities, its interaction density would be relatively dense.

To test the viability of different predictive algorithms, we measured the simulation efficiency of aggregation/disaggregation under various forms of control and under differing simulation population densities from sparse interaction density to dense interaction density. The performance of these methods were benchmarked against the naïve method (no control) of approaching aggregation/disaggregation. For our experiments, we measured the efficiency of an algorithm by counting the number of aggregation and disaggregation that occurs in these scenarios and the number of messages passed in and out of the simulation.

The motivation of the experiment is to reduce the total number of messages produced and to eliminate the problem of thrashing without using a full MRE model. The primary reasoning being a full MRE model takes up more space than similarly structured HREs permanently in full disaggregate state.

4 APPROACHES

We detail here the various approaches to controlling aggregation/disaggregation which were tested in our experiments using the HLA RTI.

4.1 Naïve

This is the basic message passing approach in HLA RTI simulations, no message filtering beyond publishing and subscribing checks is done. The naïve approach processes a lot of unnecessary information but is simple, performing well for scantly populated spatial based simulations. It was used in the research as a benchmark against which the other approaches were compared.

4.2 Region-based Data Distribution Management (DDM)

Region-based DDM is provided by the HLA RTI as a service, reducing unnecessary information sent to receiving simulators by utilizing the RTI Executive to filter messages which are not in the receiving simulators region of interest. Data distribution management by itself does not prevent thrashing or reduce load efficiently in dense interaction situations but it has a great effect on simulation efficiency in sparse to moderation interaction conditions.

4.3 Region-based DDM with Constant Wait Interval

An addition of a static value set by the user, delays aggregation by a number of logical cycles. The main purpose of this constant wait interval is to reduce the effects of aggregation/disaggregation due to thrashing at certain frequencies. This approach has the disadvantage of preventing thrashing only at the selected delay intervals and could actually make thrashing worse if thrashing occurred at a slightly slower frequency. It also has the added disadvantage of slowing the simulation down in sparse conditions, where the constants wait interval is not required.

4.4 Region-based DDM with Predicted Wait Interval

The main limitation of the wait interval described above is its static nature. If the wait interval could change according to the simulations current interaction density, the issues it introduces could be resolved. To estimate the wait interval for the current situation requires the simulation to collect data from past iterations. There are several means to achieve this measurement, but due to time limitations only two methods of predicting this value are investigated in this paper.

4.4.1 Quantity of Entity Prediction

The first method for determining the wait interval t_{wait} is the Quantity of Entity Prediction (QoE) method. QoE is particularly useful for simulations that revolve around crowds and based on simple observation: if a LRE encounters one HRE, it is likely the LRE will continue to encounter more HREs. The more HREs and LRE is simultaneously encountering, the higher the likelihood that the LRE will interact with more HREs within a short amount of time. This prediction method checks for the number of HREs interacting with a LRE and uses this value to estimate how long it should wait before re-aggregating should the HREs stop interacting with any LRE in the next logical time step.

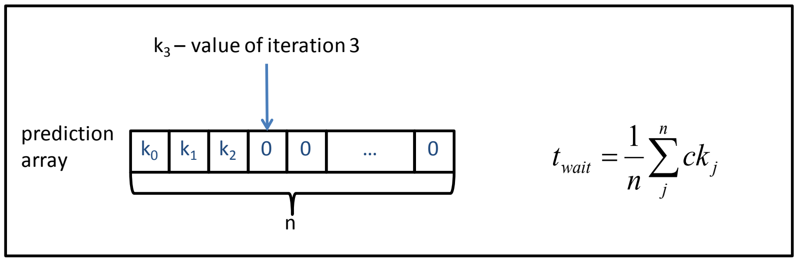

k_3 – value of iteration 3
prediction array
k_0
k_1
k_2
0
0
...
0
n
$t_{wait} = \frac{1}{n}\sum_{j}^{n} ck_j$

Figure 4: Quantity of Entity Prediction

Figure 4 shows how t_{wait} could be measured using QoE. The number of HREs within a LRE's region of interest, k_j , is determined by DDM and recorded every iteration into a location in the array determined by iteration number modulus array length. During an iteration the values are summed, averaged and multiplied by a constant c to determine the length of the next wait interval. The algorithm is adaptive and the interval changes according to the number of HREs measured in the region over a specified number of n previous iterations.

4.4.2 Frequency-based Prediction

Frequency-based Prediction measures HREs on an individual basis. It keeps track of the time between the beginning of each HRE's interaction and its end, discarding the information if the HRE appears relatively infrequently with other HREs within the region of interest. Using the information collected, the algorithm checks if staying in the disaggregate state till the reappearance of the entity will save more effort than aggregating now and disaggregating again later. A decision is then made based on the information calculated.

Each HRE is measured on an individual entity basis. Frequency based prediction is less prone to sudden fluctuations in interaction density, but requires a higher 'setup' time for data collection of each entity's activities. This setup time could be very long if the HREs only interact sparsely. Frequency-based prediction also requires the designer to formulate a balance between space and time efficiency. This requires some technical expertise but means that frequency-based prediction can be flexible for different simulations.

4.5 Summary of Algorithms

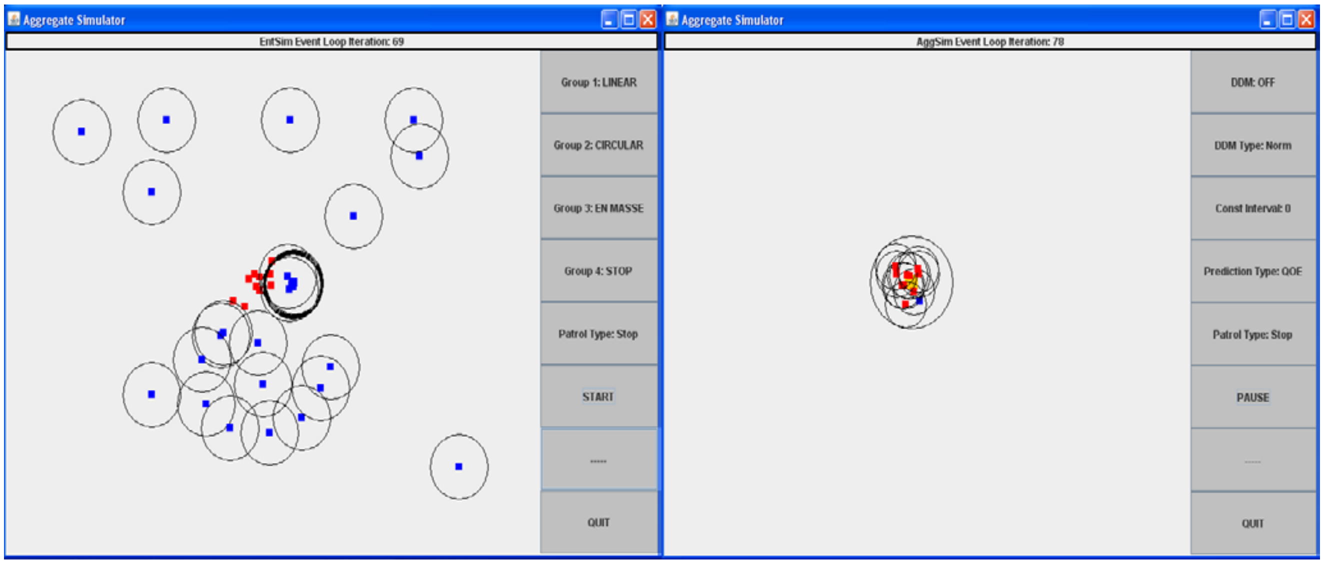
Table 1 summarizes and compares the algorithms discussed in this section.

Table 1: Comparing the four different types of aggregation/disaggregation algorithms

	Naïve	**DDM**	**Constant Wait Value DDM**	**QOE Prediction DDM**	**Frequency-based Prediction DDM**
Complexity	Simple	Simple	Simple	+O(n) complexity	Simple
Aggregation Disaggregation Management	Not efficient	Not efficient	Good in some invariant cases	Good in dense interaction but not in sparse conditions	Good in dense interaction scenarios.
Adaptability	None, fails at thrashing	None, fails at thrashing	None	Adapts quickly. Solves thrashing problem in most cases	Adaptation rate is dependent on frequency of encounters with entities.
User Knowledge Requirement of Simulation	Low	Low	Low	Low	Less prone to sudden fluctuations. Requires formulation of space/time trade off.

5 EXPERIMENTAL MODEL

The algorithms were tested in a 2D spatial world populated by HREs and 1 LRE with a region of interest that is a subset of the spatial world. In the simulation, the LRE disaggregates into a number of entities when HREs enter its region of interest to operate at the HRE's level of resolution. The simulation federation consists of 1 LRE simulation, which disaggregates when necessary, and up to 3 HRE simulations controlling 1 to 80 HRE entities depending on the interaction density load requirement of the tests. In total the experiments were run on a total of 4 computers with similar specifications.



EntSim Event Loop Iteration: 69

Group 1: LINEAR

Group 2: CIRCULAR

Group 3: EN MASSE

Group 4: STOP

Patrol Type: Stop

AggSim Event Loop Iteration: 78

DDM: OFF

DDM Type: Norm

Const Interval: 0

Prediction Type: QOE

Patrol Type: Stop

Figure 5: HRE simulation and LRE simulation view of the simulation world respectively

Figure 5 shows a stationary LRE (e.g. a base or town entity) controlled by the aggregate federate is positioned in the middle of the simulation world, staying in its default LRE state but disaggregating into mobile HREs (e.g. person, car, tank or troop entities) when a HRE (e.g. spy plane, UAV, helicopter, vehicle entity) controlled by other simulations comes within range. HREs controlled by the HRE simulations moved in a number of different paths to produce different interaction density situations. The left image shows the view of the simulation world from the perspective of the HRE simulation and its entities. The right image shows the view of the simulation world from the perspective of the LRE simulation and its region of interest (represented by the largest circle). The experiment measured the number of messages passed between the LRE federate and the RTI Executive and the number of aggregation and disaggregation work being done by the LRE federate.

A total of 5 scenarios were tested with no changes to the single LRE federate in any of the scenarios. The HRE simulation in the first scenario produces a sparse interaction density situation, meaning little aggregation and disaggregation occurs, by controlling one HRE that intersects with the LRE's region of interest at long intervals. The second scenario simulates a situation where there are 20 HREs, but none actually interact with the LRE. The third scenario simulates regular interaction between the HREs and the LRE. The fourth scenario simulates thrashing at regular intervals to test the performance of the algorithms in the worst case scenario of distributed multi-resolution simulations. The last scenario simulates a high load, very dense interaction density environment where the LRE has little or no chance to re-aggregate at all.

Table 2: Configuration of the different scenarios tested

Scenario No.	Number of HRE Simulations	Number of HRE Entities	Expected Interaction Density
1	1	1	Sparse
2	1	20	Zero
3	1	20	Moderate
4	1	20	Thrashing
5	3	80	Very Dense

Table 2 shows a summary of the 5 scenarios which the algorithms were tested on. The scenarios differed in the density and frequency of interaction between the HREs and the LRE entity. Algorithm performance of each algorithm was determined in each case as shown in the next section of this paper.

6 RESULTS

From our experiment results in Table 3, it can generally be seen from rows 3 and 4 that simulations utilizing predictive algorithms perform better than those that do not in cases with high aggregation/disaggregation. Only in cases with little or no aggregation/disaggregation, e.g. scenarios 1, 2 and 5, do predictive algorithms not make a notable difference.

Table 3: Number of messages per iteration.*

Scenario	Naïve	DDM	DDM CW	DDM QOE	DDM FREQ
1: Sparse Interaction	5.51/10.724	4.35/12.271	5.45/12.377 (Wait 5)	4.37/12.268	4.63/12.312
2: No Interaction	41.04/0.894	1.02/0.537	1.02/0.537	1.02/0.537	1.02/0.537
3: Moderately Dense Interaction	52.76/18.708	16.01/18.992	13.54/9.592 (Wait 3) 16.04/17.712 (Wait 1)	14.92/15.697	12.42/3.772
4: Thrashing	63.36/27.936	36.06/39.056	21.36/12.617 (Wait 3) 38.78/44.253 (Wait 1)	21.24/11.615	21.28/11.406
5: Dense Interaction	168.82/17.358	17.52/8.660	17.52/8.660	17.52/8.660	17.52/8.660

*The first number shows the mean value and the second number shows the standard deviation.

The predictive algorithms tested saved the most effort in dense thrashing situations, e.g. scenario 4, but they also increase the efficiency of the simulation in scenario 3. It can be seen from our results that implementing a constant wait interval could also improve the efficiency of the simulation for certain situations and could be detrimental in others.

Figure 6 shows the problem with constant wait intervals. The sharp spikes in the number of messages in the graph indicate that disaggregation is occurring. In this situation, aggregation/disaggregation thrashing occurs at an interval of 3 logical cycles. In this case, having a constant wait interval of 3 prevents thrashing altogether. But if the constant wait interval was 1, the constant wait interval does not stop thrashing from occurring at all. While the designer may tune the wait interval to fix one issue, the wait interval could likely cause another problem sometime in the future.

The predictive algorithms are also not flawless. Figure 7 shows an issue of the QoE prediction which relies on the selection of a good value for the constant multiplier of the QoE equation for the weight of entities. In the experiment, the value of the constant multiplier used was not large enough and fails when only one entity is thrashing. However the constant multiplier selected works when 3 entities are thrashing. As can be seen in the graph, the QoE predictive algorithm adapts the wait interval to prevent thrashing from occurring after 3 disaggregation processes have happened.

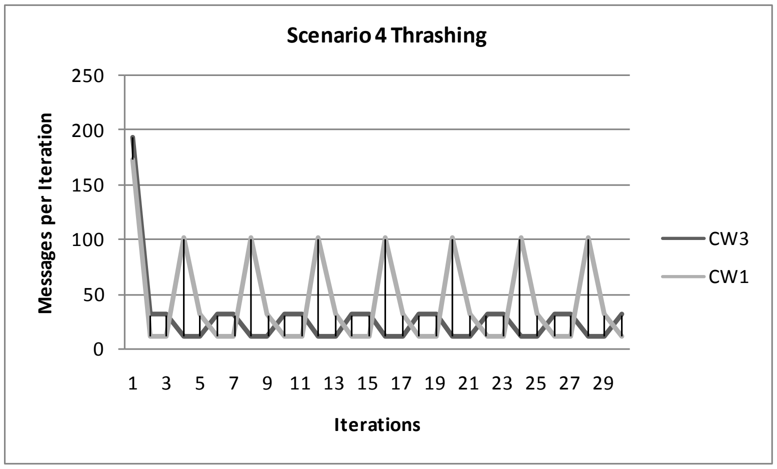


Figure 6: Comparing wait intervals 1 and 3

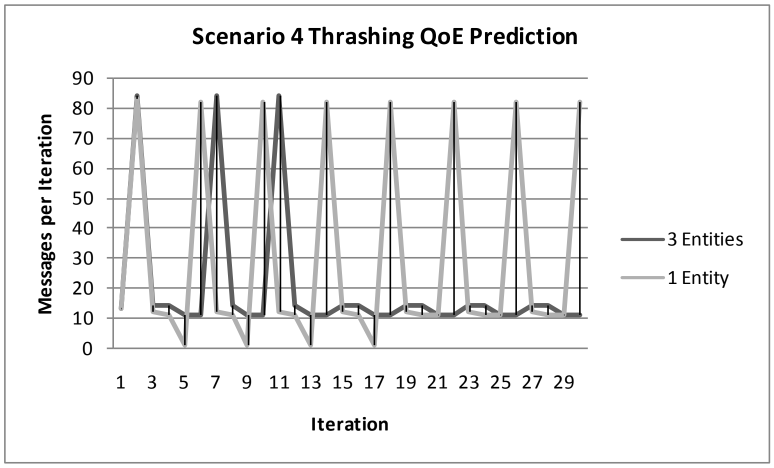


Figure 7: An issue of QoE prediction

7 CONCLUSION

It is our conclusion that predictive algorithms can improve the efficiency of mixed mode simulations especially in dense interaction or thrashing scenarios. Predictive algorithms can likely eliminate thrashing, providing an alternative in space constrained simulations where the use of full MREs is not feasible.

While the algorithms tested have their weaknesses or scenarios in which they do not work, it is our opinion that these are weaknesses that can be overcome with further research and study, or with additional heuristic weights that can balance the predictive algorithm's equations.

REFERENCES

Boukerche A., and C. Dzermajko. 2004. Scalability and Performance Evaluation of An Aggregation/Disaggregation Scheme for Data Distribution Management in Large-Scale Distributed Interactive Systems. In *Proceedings of the 37th Annual Simulation Symposium*, 238-245.

Bowers F. A., and D. L. Prochnow. 2003. JTLS-JCATS Federation support of emergency response training. In *Proceedings of 2003 Winter Simulation Conference*, eds. S. E. Chick, P. J. Sanchez, D. M. Ferrin, D. J. Morrice, 1052-1060. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Cai, W., F. B. S. Lee and L. Chen. 1999. An auto-adaptive dead reckoning algorithm for distributed interactive simulation. In *Proceedings of the 13th Workshop on Parallel and Distributed Simulation*, 82-89.

Davis P. K., and H. J Bigelow. 2002. Motivated metamodels: Synthesis of cause-effect reasoning and statistical modeling. RAND.

Davis P. K., and A. Tolk. 2007. Observations on new developments in composability and multi-resolution modeling. In *Proceedings of the 2007 Winter Simulation Conference*, eds. S. G. Henderson, B. Biller, M.-H Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, 859-870. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Defense Modeling and Simulation Office (DMSO). 1998. High Level Architecture Interface Specification, Version 1.3. Washington D.C.

Kumova, B. I. 2005. Dynamically Adaptive Partition-Based Data Distribution Management. In *Proceedings of the 19th Workshop on Principles of Advanced and Distributed Simulation*, 292-300.

Natrajan, A. 2000. *Consistency Maintenance in Concurrent Representations*, Ph.D. Thesis, University of Virginia.

Natrajan, A., P. F. Jr. Reynold and S. Srinivasan. 1997. MRE: a flexible approach to multi-resolution modeling. *ACM SIGSIM Simulation Digest*, 27(1):156-163.

Reynolds, P. F. Jr., and A. Natrajan. 1997. Consistency maintenance in multiresolution simulations. *ACM Transactions on Modeling and Computer Simulation*, 7(3):368-392.

Tacic, I., and R. M. Fujimoto. 1998. Synchronized data distribution management in distributed simulations. In *Proceedings of the 12th Workshop on Parallel and Distributed Simulation*, 108-115.

Van, H. D. J., and J. O. Calvin. 1998. Data Distribution Management in RTI 1.3, 98S-SIW-206. In *Spring Simulation Interoperability Workshop*.

Zhang, Y., L. Chen and G. C. Chen, 2006. Globally synchronized dead-reckoning with local lag for continuous distributed multiplayer games. In *Proceedings of 5th ACM SIGCOMM Workshop on Network and System Support for Games.*

AUTHOR BIOGRAPHIES

BENJAMIN YUAN WEI CHUA is currently a final year undergraduate student in the School of Computer Engineering at the Nanyang Technological University of Singapore. His research interest is in the area of mixed mode distributed simulation. His e-mail address is <chua0308@ntu.edu.sg>.

MALCOLM YOKE HEAN LOW is currently an Assistant Professor in the School of Computer Engineering at the Nanyang Technological University (NTU), Singapore. Prior to this, he was with the Singapore Institute of Manufacturing Technology, Singapore (SIMTech). He received his Bachelor and Master of Applied Science in Computer Engineering from NTU in 1997 and 1999 respectively. He was awarded a Gintic (now SIMTech) Postgraduate Scholarship in 1999. In 2002, he received his D.Phil. degree in Computer Science from Oxford University. His current research interest is in the application of parallel and distributed computing for the modeling, simulation, analysis and optimization of complex systems. His email address is <yhlow@ntu.edu.sg>.