Guidelines for the Design of Multi-resolution Simulations

A Research Proposal

Anand Natrajan, Paul F. Reynolds, Jr., Sudhir Srinivasan
anand@virginia.edu, reynolds@virginia.edu, sudhirs@mystech.com

1 Introduction

This document proposes Multiple Resolution Entities (MREs) and Attribute Dependency Graphs (ADGs) as part of a framework for maintaining consistency in multi-resolution simulations. We briefly present our past work in this area and then outline our new directions. In particular, we demonstrate how our approach applies to existing simulations, and analyze the cost benefits achieved by our scheme over traditional multi-resolution schemes. We offer guidelines for future simulations from these analyses, and draw milestones for future research. The primary objective of the direction of our research is to provide guidelines for designers and developers of new and legacy battlefield simulations for resolving representation issues in multi-resolution simulations.

2 Background

We have identified critical issues that must be solved in order to make simulations at multiple levels of resolution feasible. Existing schemes do not address some or all of these issues in a coherent manner, causing entities to become inconsistent. Since inconsistency may compromise the usefulness of the simulation, many simulations may require consistency maintenance. We proposed Multiple Resolution Entities as a mechanism for maintaining consistency. We explored means by which MREs could be created and kept consistent. In pursuit of this goal, we made some Fundamental Observations, which we believe should guide the design of all multi-resolution simulations.

2.1 Definitions

- ***Object***: A fundamental element of a conceptual representation that reflects the real world at levels of abstraction and resolution appropriate for a planned simulation.
- ***Entity***: A unit of organization at some level of abstraction, such as a tank, human, platoon, battalion, cloud or radar.
- ***Model***: A mathematical abstraction of the behavior of an object at a level appropriate for the planned simulation. Models are usually instantiated in simulation source code.
- ***Resolution***: The conceptual level at which an entity is simulated.
- ***Disaggregated Entity (DE)****:* A high-resolution entity, such as a CCTT tank simulator.
- ***Aggregated Entity (AE)***: A low-resolution entity that simulates several aggregated objects, such as a battalion.
- ***Multiple Resolution Entity (MRE)***: An entity that can be perceived at multiple levels of resolution concurrently.
- ***Simulation***: A dynamic representation of one or more objects, involving some combination of executing code, control/display interface hardware and interfaces to real-world equipment.
- ***Multi-level Simulation***: A simulation or exercise that involves entities at different levels of resolution.

Note: Levels of resolution and levels of aggregation are inversely related: high-resolution means low level of aggregation, and low-resolution means high level of aggregation.

2.2 Problems with Current Multi-Resolution Approaches

Current approaches subject entities to one of Full Aggregation (FA), Full Disaggregation (FD), Partial Disaggregation (PD) or Pseudo-Disaggregation (SD). The first two schemes represent extremes in the way they address multi-resolution issues. We briefly re-introduce these issues. A fuller discussion can be found in [Reyn96].

Temporal Inconsistency: Temporal inconsistency is said to occur when two (or more) entities have mutually inconsistent views at the same simulation time. This may arise when simulations at different resolution levels proceed at time steps that differ by orders of magnitude. In particular, inconsistency may occur during attrition computation, while perceiving the state of another entity, during line-of-sight computations or during dead-reckoning. Temporal inconsistency will become very significant as large multi-level simulations are planned and executed.

Chain Disaggregation: Chain disaggregation (also called spreading disaggregation) is a phenomenon wherein many AEs are forced to disaggregate in a short period of time. Chain disaggregation usually causes unnecessary disaggregation, putting a burden on computing and network resources. Many approaches introduce temporal inconsistency in attempting to solve the chain disaggregation problem.

Network Flooding: Network resources may be strained by the acts of aggregation and disaggregation, depending on the scheme used. Even if only the entity state messages generated by all the entities are taken into account, disaggregation increases network traffic by virtue of creating more entities. Aggregation and disaggregation protocols typically also require a number of messages such as “Request to disaggregate”, “Refuse to disaggregate” and “Request to aggregate”. With the network projected as the biggest bottleneck for simulations, these messages may represent an unacceptable overhead.

Transition Latency: The time taken to effect an aggregation or disaggregation, the *transition period*, can be significantly long depending on the complexity of the protocol. Long transition periods are incompatible with real-time constraints in human-in-the-loop simulations because they may cause visual or conceptual inconsistencies. Long latencies may also cause entities to thrash, wherein they spend most of their time just changing levels.

Mapping Inconsistency: Mapping inconsistency arises when the attributes at one level of resolution are not consistently mapped to the attributes at other levels. The problem is observed when an entity performs actions in an interval of time in a simulation that it could not have performed in reality.

2.3 Fundamental Observations

We present some fundamental observations regarding multi-resolution simulations. These are *observations* rather than theorems because the truth of their statements is argued informally rather than proven rigorously.They are fundamental because any general solution to the multi-resolution problem *must* take them into account. These observations are a result of a thorough analysis of the issues concerning multi-resolution modeling. The fundamental observations provide the foundation for the theory of multi-resolution modeling and guide the development of long-term solutions to the various issues in multi-resolution modeling. The arguments supporting the fundamental observations are in [Reyn96]. In this discussion we merely present the observations with a brief explanation.

FO-1: In general, effective linkage requires entities to be modeled at appropriate levels of resolution.

FO-2: The effects of concurrent interactions at multiple levels of resolution must be combined consistently.

FO-3: Overlapping interactions may often not be independent.

FO-4: Time-step differentials can amplify ineffectiveness due to dependence violations.

Consider a linkage between models E_A and E_B at levels of resolution L_A and L_B respectively (Figure 1). For most applications, either E_A must be represented at L_B or E_B must be represented at L_A for E_A and E_B to interact with each other. FO-1 states that only linkages following a combination of a vertical and a horizontal link can be effective — a diagonal linkage cannot.

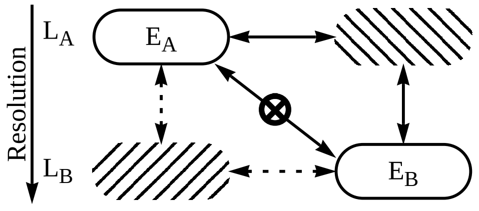

FIGURE 1: Fundamental Observation 1

In order to satisfy FO-1, entities must dynamically transition to the appropriate level as required, as is done with aggregation-disaggregation. The costs associated with the overheads of dynamic transitions can be reduced by reducing the number of transitions. Significant reductions in overhead can be achieved by limiting the propagation of transitions (for example, by controlling chain disaggregation). Ideally, a transition should be restricted to a single entity instead of propagating. This leads to the following two requirements: (i) entities must be able to handle interactions at multiple levels *concurrently*, and (ii) the effects of these concurrent interactions must be combined without compromising effectiveness (realism, validity, consistency, etc.). These requirements are captured in FO-2.

The second requirement is difficult to satisfy. Serialization fails in the context of real-time interactions because it detracts from the appearance of concurrency of interactions overlapping in real-time. Alternatively, interactions could be processed in parallel and their results combined. However, when interactions with smaller time-steps (S_i) are allowed to occur concurrently with those with larger time-steps (L_i), the assumptions made by the L_i will be invalidated due to the S_i during a time-step, leading to ineffective linkages.

The problem of combining the effects of concurrent interactions consistently arises primarily due to a fundamental underlying problem, namely, *independence*. As FO-3 states, two interactions that overlap in (i) simulation time, and (ii) the entities involved in the interaction, may not in general be independent simply because they can affect the outcome of each other. If two dependent interactions are executed independently, the results of the combination of these interactions may be invalid. FO-4 says that time-step differentials tend to aggravate the inconsistencies created due to dependency issues.

The fundamental observations present the basic issues that must be addressed by any general, scalable approach to multi-resolution modeling and thus provide the beginnings of a theoretical foundation for the same. The key to multi-resolution modeling is a holistic approach that internalizes issues of consistency and is designed to solve them. In the next section, we present one such approach based on the fundamental observations.

3 A General Framework

We describe a general framework we have developed aimed at facilitating the design of multi-resolution simulations. The framework builds on our Fundamental Observations and consists of two components: the Multiple Resolution Entity and the Attribute Dependency Graph.

3.1 Multiple Resolution Entity

Traditional approaches towards aggregation/disaggregation maintain, at any given time, the attributes at only *one* level of resolution — the level at which the entity is being simulated. In contrast, we believe each entity should possess attributes at multiple levels of resolution. These Multiple Resolution Entities (MREs) can be perceived at multiple resolutions because they either maintain state information at all desired levels of resolution or furnish state information at a requested level in a timely manner. Simulation of the MRE entails handling incoming interactions at all desired levels. Each MRE is responsible for enforcing logical consistency across resolution levels: the effect of any incoming interaction should be reflected consistently in the attributes of all levels of the MRE.

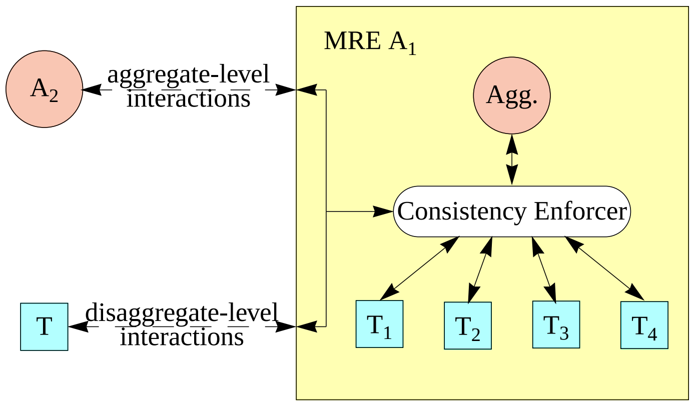

FIGURE 2. Multiple levels of resolution

The MRE is temporally consistent because every interaction is consistently reflected across all levels of resolution. Each MRE determines the level of resolution at which it perceives another MRE, and the perceived MRE is able to present consistent views of itself to its perceivers. Since the concept of disaggregation doesn't exist, chain disaggregation is eliminated. This, in turn, reduces network load and transition latency. We proposed a core set of attributes for each MRE, which comprised attributes that will always be maintained for that entity. We identified some guidelines for choosing attributes that went into the core set. As the following sections will show, we have had insights into how the MRE can be designed in order to stay consistent.

3.2 Attribute Dependency Graph

The key feature of the MRE is its ability to stay consistent in the face of concurrent interactions at multiple levels of resolution. In order to gain insights into its implementation, it is useful to visualize a model of the MRE as a graph.

This graph, known as an Attribute Dependency Graph (ADG), depicts the various attributes and sub-entities of the MRE, and portrays the relationships between them. ADGs are an encoding of the concurrent multi-resolution interactions problem, and are also an encoding of solutions to this problem. For the rest of this discussion we assume away FO-4, i.e., we assume that the time-step differential between simulations at different levels of resolution does not exist. We believe that making this assumption does not adversely affect our theory.

In order to model how an entity behaves at multiple levels of resolution, it is important to be able to express the relationships between attributes. These relationships can be modelled by a directed, weighted graph wherein the nodes represent attributes and the edges between the nodes represent relationships. An example serves to illustrate the concept better. Assume an entity E at two levels of resolution: a low-resolution aggregate level and a high-resolution disaggregate level. The low-resolution level is the aggregate level and the high-resolution level is the disaggregate level. Let E_1 and E_2 be the sub-entities of E. Let the attributes of interest be *S* (strength) and *F* (firepower). Therefore, the attributes of E (and hence the nodes in the Attribute Dependency Graph) are:

S_A = aggregate strength	S_1 = strength of E_1	S_2 = strength of E_2
F_A = aggregate firepower	F_1 = firepower of E_1	F_2 = firepower of E_2

A phantom node, *Outside*, is used to represent interactions from the environment or other entities. This node may be omitted altogether without loss of generality. For every dependency between a pair of attributes, a directed edge must be drawn from the dependee to the depender (e.g., if $y = g(x)$, then draw the edge $x \rightarrow y$). The edges fall into various categories depending on their associated nodes, as is explained in the following section. A pictorial representation of the MRE E is shown in Figure 3.

The selection of nodes clearly reveals how the ADG satisfies FO-1. Attributes at all levels are present in the ADG. Therefore, the MRE is represented at all levels of resolution.

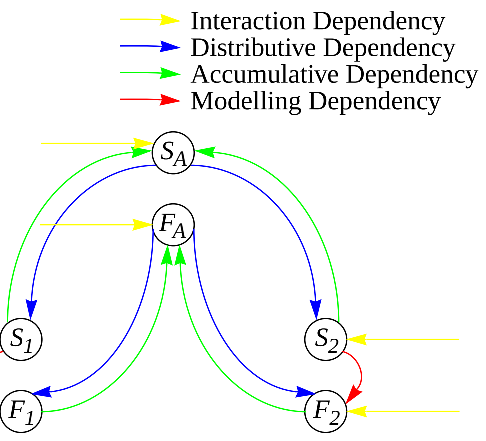

FIGURE 3: Attribute Dependency Graph

3.3 Attribute Dependencies

The dependencies between attributes fall into four classes. The semantics of these dependencies (and hence the edges in the graph) are as below:

- **Interaction Dependencies**: These are edges from *Outside* to some other node in the graph. Interaction dependencies capture interactions that may cause attributes to change values. Typically each attribute that can be changed as a direct result of an interaction would have an interaction dependency.
- **Distributive Dependencies**: These are edges from a node representing an aggregate-level attribute to a node representing the corresponding disaggregate attribute for a particular sub-entity. Obviously, distributive dependencies exist from each aggregate attribute to many disaggregate attributes.
- **Accumulative Dependencies**: These are edges from a node representing a disaggregate-level attribute for a particular sub-entity to a node representing the corresponding aggregate attribute. Each disaggregate-level attribute has an accumulative dependency with one aggregate-level attribute.
- **Modelling Dependencies**: These are all edges that are not one of the above. Typically, these edges represent relationships between attributes that exist due to the nature of the entity being modelled.

To construct the graph for E, we assign each attribute to a node. The distributive dependencies are: $S_A \rightarrow S_1$, $S_A \rightarrow S_2$, $F_A \rightarrow F_1$ and $F_A \rightarrow F_2$. The accumulative dependencies are: $S_1 \rightarrow S_A$, $S_2 \rightarrow S_A$, $F_1 \rightarrow F_A$ and $F_2 \rightarrow F_A$. Aggregate interactions can cause changes in the aggregate S_A or F_A, and disaggregate interactions can cause changes in S_1, F_1, S_2 or F_2 depending on which of E_1 and E_2 or both are involved. Assume E_2 is involved in a disaggregate engagement and concurrently, there is an aggregate engagement in progress. Therefore we have edges *Outside* $\rightarrow S_A$, *Outside* $\rightarrow F_A$, *Outside* $\rightarrow S_2$ and *Outside* $\rightarrow F_2$. In addition, assume that the modelling of the sub-entities stipulates that as *S* reduces, so does *F*. In this case, we have the edges $S_1 \rightarrow F_1$ and $S_2 \rightarrow F_2$.

After constructing the graph (Figure 3), given an interaction, it is possible to trace a path in the graph to account for changes to nodes. For example, suppose an aggregate-level enemy attrits the MRE. The paths through which the attrition must be reflected are: *Outside* $\rightarrow S_A \rightarrow S_1 \rightarrow F_1 \rightarrow F_A$ and *Outside* $\rightarrow S_A \rightarrow S_2 \rightarrow F_2 \rightarrow F_A$. This is equivalent to saying that the aggregate interactions caused a change in the individual disaggregate strengths, causing a change in the disaggregate firepowers, which affected the aggregate firepower. Recall that E_2 was involved in a disaggregate battle elsewhere. We may translate this to removing (or weighting with zero) the edge $F_2 \rightarrow F_A$, in essence saying that $\mathtt{E}_2$ does not contribute to the firepower of the aggregate. $F_A \rightarrow F_2$ should be weighted to zero for the same reason. Note that this does not preclude $\mathtt{E}_2$'s firepower being decreased due to the effects of an aggregate attrition.

The ability to weight graphs (with binary or fractional weights) provides a means to solve the issue raised by FO-3. For our particular example, a judicious choice of weights ensures that $\mathtt{E}_2$'s strength can be affected by both aggregate and disaggregate interactions, but its firepower is targeted either towards the aggregate battle or the disaggregate one, not both. This is because firepower expenditures are independent interactions — an aggregate-level expenditure precludes a disaggregate-level expenditure in the same time-step.

3.4 Graph Traversal

For any interaction, a path can be traced in the graph to account for changes to attributes represented by nodes. A traversal of the ADG is initiated every time an interaction occurs. In other words, graph traversal begins with an interaction dependency. Further traversals occur along the other dependencies. In the course of one traversal, if a node has been visited it is colored to indicate that for that traversal the node should not be evaluated again. After a node has been colored (and the effects of the changes in the corresponding attribute computed), the edges going outward from that node are addressed. The changes in the nodes at the other end of the edges are computed, and the process is continued till no more nodes can be visited either because they have all been colored or there does not exist an outgoing path from any one of the colored nodes to the uncolored ones. In simulation terms this translates to the state where the effects of every attribute to every other possible attribute have been propagated.

The order in which the nodes are traversed is somewhat crucial. Assume an interaction modifying S_A arrives. A depth-first traversal of the graph might yield the following paths: *Outside* $\rightarrow S_A \rightarrow S_2 \rightarrow F_2 \rightarrow F_A \rightarrow F_1$ and $S_A \rightarrow S_1$. Notice that in the second path F_1 is not taken because it has been visited already in the first path. This is clearly incorrect because it violates the design requisites of the sub-entity. A breadth-first traversal works slightly better in this case because it yields the paths: *Outside* $\rightarrow S_A \rightarrow S_2 \rightarrow F_2 \rightarrow F_A$ and $S_A \rightarrow S_1 \rightarrow F_1$. Note that this is also unsatisfactory because the new value of F_A has been computed without taking into account the change in F_1. General rules for traversal of the Attribute Dependency Graph need to be researched.

The distributive and accumulative dependencies put together form simple cycles. If a particular traversal is in progress and a distributive dependency is taken, then the corresponding accumulative dependencies must not be taken. Likewise, if an accumulative dependency is taken, then the corresponding distributive dependency must not be taken. Essentially, this means that once the attribute at one level is changed it does not have to be re-computed after its corresponding attribute at the other level has changed. If visited nodes are colored, then this will be enforced. If the modelling dependencies form a cycle, then during a traversal the nodes in the cycle must be visited only once. If the attributes at some level cyclically depend on each other, then it is reasonable to assume that the designer does not wish that the attributes be modified infinite number of times per interaction.

3.5 Multiple Incident Edges

An important issue is managing multiple dependencies on an attribute, characterized by multiple edges incident on the corresponding node in the graph. When changes to an attribute's value arrive on more than one incident edge concurrently, a decision must be made as to how the effects of those interactions will be reflected on the attribute. Note that this issue directly corresponds to FO-2 and FO-3. Choosing a policy for resolving multiple incident edges on nodes in the ADG addresses the problem of combining concurrent interactions on the MRE. We attempt to identify classes of nodes for which we can provide solutions to the problem of concurrent interactions. The application may choose different solutions for different nodes.

One solution addresses attributes for which interactions with independent effects are incident. Such interactions have effects on the corresponding nodes that are independent of other interactions. Note that this does *not* imply that the interactions are independent with respect to the nodes they affect. If they do affect different nodes altogether the problem becomes trivial. Rather, we account for concurrent interactions whose intersection set of affected attributes is non-empty, but whose effects are such that one does not preclude or affect the outcome of the other. For such interactions, we can show that their effects can be propagated to the corresponding nodes in any order. Therefore, we can instantaneously reflect the effects of such interactions on the corresponding nodes.

Another solution is to delay the effects of interactions for an average of half-a-timestep. In effect, each time-step becomes a window during which interactions are incident on a node. At the end of that time-step, the node possesses a set of interactions that arrived during that time-step, but have not been incorporated yet. The node may then reflect the effects of these interactions by choosing one of the following (incomplete) list of options[*]:

- The node could reflect the effects of the interactions in the order they arrived on the node. This is akin to serialization, but with some delay because the interactions are made to wait for an average of half-a-timestep before their effects are reflected. If such a policy is adopted, it may be more beneficial to not make the interactions wait and instead just reflect their effects as and when the interactions occur. Concurrent interactions may be arbitrarily settled by choosing one over the other as occurring earlier.
- The node could arbitrarily re-order the interactions and then reflect their effects. This may devolve to the independent interactions solution described earlier, except that the effects are reflected after some delay.
- The node could have static policies for each combination of concurrent interactions since the types of interactions that are incident on a node are known *a priori*. If n types of interactions could be incident on a node, then the total number of combinations of types of interactions that could be incident on that node is $S = O\left(\sum_{i=0}^{n} \binom{n}{i}\right) = O(2^n)$. Thus, choosing a policy for each combination is an exponential-growth solution. However, we expect many of the cases could be collapsed into one another. Another approach is to design for exceptions alone and catch all other cases in a default. Also, n is expected to be small.

4 Applicability to Existing Simulations

In order to demonstrate the applicability of Attribute Dependency Graphs, we applied them to the Federation Object Models (FOMs) of existing battlefield simulations. We considered `JPSD` and `Eagle` as candidate simulations, and attempted to create ADGs for entities simulated therein. We were able to draw some conclusions regarding these simulations after creating the ADGs. Since the `JPSD` FOM does not focus on aggregate interactions, we attempted to introduce these by constructing a “mock” linkage between the `Eagle` and `JPSD` FOMs. We believe this linkage helps bring out important issues to be considered in actual implementations.

4.1 JPSD

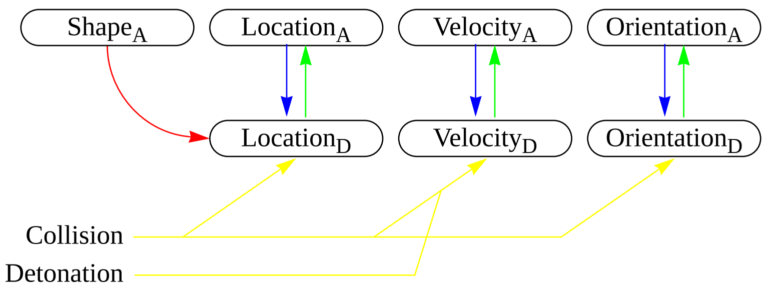

FIGURE 4. `JPSD` **Aggregate MRE**

The `JPSD` FOM includes the Class Structure Table, the Attribute Table and the Interaction Table, The ADG in Figure 4 was constructed from these tables. Some of the Aggregate and Entity attributes (subscripted $_A$ and $_D$ respectively) have been omitted because they are not immediately relevant to our theory. The omitted attributes are mostly enumerations, and their correlation with other attributes is minimal, if any. The blue arrows stand for Distributive dependencies from that attribute to all other entities/attributes at a lower level of resolution. The green arrows stand for Accumulative dependencies. The red arrow stands for a Modelling dependency. Clearly, the disaggregate-level locations depend not just on the aggregate-level location, but also on the shape of the aggregate. The yellow arrows stand for Interaction Dependencies.

An interesting observation we made was regarding acceleration and angular velocity attributes. The `JPSD` FOM indicates that a Helicopter Company may be an aggregate entity. Assuming that such an aggregate would comprise Entity.Platform.Air.AttackHelicopter, it is somewhat strange that the aggregate entity has no provision for acceleration or angular velocity attributes even though the disaggregate entities have such attributes. Also, `JPSD` seems to ignore the strength of the disaggregate entity (save for displaying a damage state) and its firepower. The

[*] The time required to actually reflect the effects of these interactions must be small compared to the size of the time-step.

aggregate equivalent of such a policy is to just count the number of disaggregate entities that are in the aggregate and submit this as some notion of strength.

The FOM Interaction Table lists the possible `JPSD` interactions and shows the attributes that would be affected in case such an interaction occurs. Most of these interactions are "read" interactions, and are of lesser importance. The Collision interaction is an example of interactions that affect some entity attributes. Figure 4 shows how the Collision interaction affects the MRE. The Detonation interaction is also depicted. Aside from these two, the only other interaction that affects attributes in an MRE is the DisaggregateRequest interaction. We foresee that this interaction will be unnecessary in the context of an entity that is able to consistently represent itself across multiple levels of resolution. The contents of the interaction messages, as shown in the FOM, make it clear as to how the affected attributes are changed.

`JPSD` does not possess aggregate interactions, apart from DisaggregateRequest. The simulation proceeds in the aggregate when nothing of interest is going on, but when something interesting happens, the simulation switches to the disaggregate. The problem of concurrent interactions along the edges incident on a node is thus solved trivially. However, `JPSD` could be augmented by assuming concurrent multiple levels. In such case, we would have to add Interaction dependencies to the nodes representing aggregate attributes too.

4.2 Eagle-JPSD

Figure 5 shows the `Eagle`-`JPSD` linkage in the form of Attribute Dependency Graphs. The upper half of the

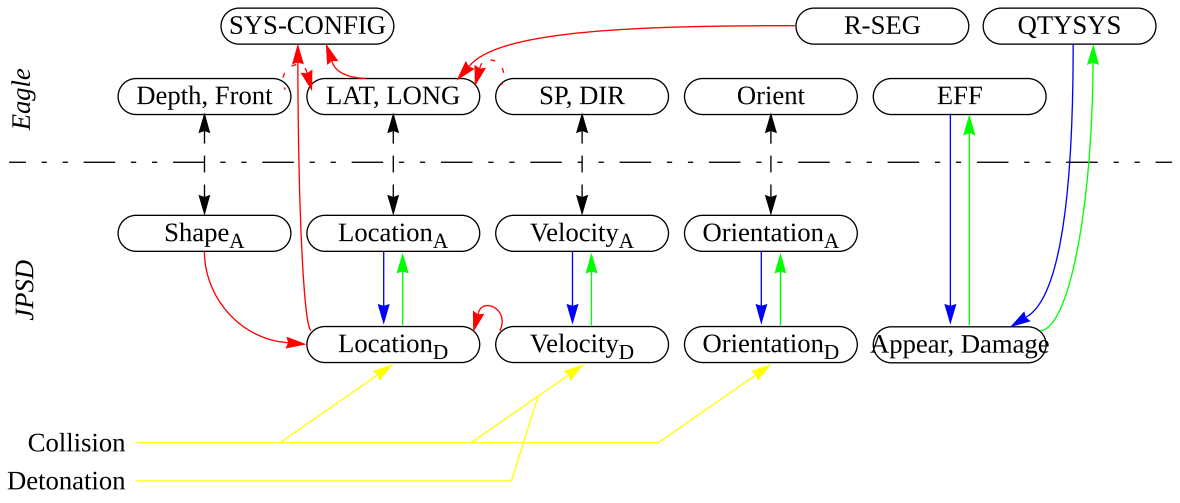

FIGURE 5. `Eagle-JPSD` **Attribute Dependency Graph**

diagram shows `Eagle` nodes for their "Military-Units". The lower half of the diagram details `JPSD` attributes. Again, only attributes immediately relevant to our theory are shown. The black two-headed arrows stand for `Eagle` attributes and their immediate counterparts in `JPSD`.

The meanings of some of the arrows are obvious. However, some others merit explanation. SYS-CONFIG is an attribute maintained by `Eagle` Military-Units to record the percentage of the Unit that lies in each of four cells: front, rear, right and left. This attribute may be calculated from the position of the aggregate and the positions of the disaggregates. R-SEG is an `Eagle` attribute that is a set of points representing the future path of the unit. QTYSYS represents the quantity of each type of system in the composition of the aggregate unit. EFF is the percentage effectiveness of the `Eagle` unit. `JPSD` entities have no notion of effectiveness, therefore the mapping is not obvious. The best recourse may be to link the Appearance and Damage-State of the entity to its effectiveness. Likewise, one way to keep QTYSYS consistent with entity-level information is to link it to the Appearance and Damage-State of the entities. Both these mappings are not satisfactory, but it is the best we have thus far.

As of writing, we have been unable to procure a complete FOM for `Eagle`. The FOM we possess does not list interactions well. The few interactions that are shown are shown as affecting *internal* attributes, not the attributes that are in the FOM. The mapping between internal attributes to FOM attributes is not obvious and missing in many cases. We believe we can do more justice to the `Eagle` ADG once we are in possession of a more complete FOM.

The most interesting node in the above graph is the one labelled LAT,LONG. This represents the position of the `Eagle` unit. Clearly, this node is affected by the disaggregate-level locations, which in turn are affected by the shape of the aggregate and the disaggregate-level velocities. LAT,LONG is also affected by the pre-chosen path of the unit, namely, R-SEG. The dashed arrow from the unit's velocity (SP,DIR) to LAT,LONG indicates that modelling dependency is captured by the path SP,DIR $\rightarrow$ Velocity$_D$ $\rightarrow$ Location$_D$ $\rightarrow$ LAT,LONG. R-SEG represents where the unit wishes to be, while the Location$_D$s represent where the unit actually is. The policies to resolve any conflict between these two could be, depending on the situation, one of: ignore R-SEG, force Location$_D$s to conform to R-SEG, and average out Location$_D$s and R-SEG.

5 Cost of MRE

It is important to compare the cost of maintaining consistency with the cost of simulation of various techniques of managing multi-resolution simulations. We consider Full Aggregation (FA), Full Disaggregation (FD) and the MRE approach (MRE). FA and FD represent two ends of a spectrum of solutions, whereas the MRE approach represents a middle ground. For analysis purposes we use a simplified notion of a multi-resolution simulation. The simplifications merely make it easier to effect a comparison between the various techniques. Figure 6 shows an entity in such a simulation. The assumptions of this simplified multi-resolution simulation are:

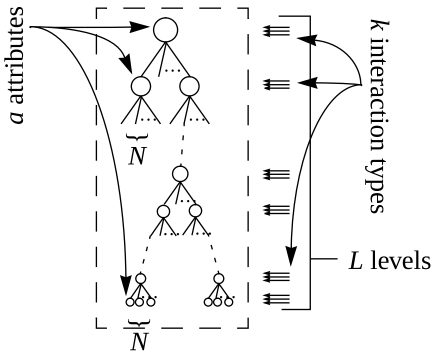

FIGURE 6. Simple MRE

- There are *L* levels of resolution, level *0* being the lowest (most aggregate) and level *L–1* being the highest (most disaggregate).
- There are *N* higher-resolution sub-entities per lower-resolution entity, i.e., an entity at a resolution level of *i* comprises of exactly *N* entities that are at resolution level *i+1*. This is true for all *i* = *0* to *L–2*.
- All entities at a particular resolution level are exactly identical in composition, i.e., they have the same number of sub-entities (as stated earlier), and also have similar attributes. Note that these entities may perform different tasks in the simulation, but for analysis purposes, they are similar in composition.
- All entities at all levels have exactly *a* attributes. All the attributes of an entity at a particular level are modified by every interaction at that level.
- There are exactly *k* types of interactions at each level of resolution.

Therefore,

Total number of entities possible, given a low-resolution entity, = $\Psi(N, L) = \sum_{i=0}^{L-1} N^i = \frac{N^L - 1}{N - 1}$.

Total number of interaction types = *kL*

5.1 Consistency Cost

Consistency Cost is comprised of a Static Consistency Cost (SCC) and a Dynamic Consistency Cost (DCC). SCC is incurred during the design phase and is a one-time cost reflecting the amount of effort required to design a consistent entity. DCC is incurred for every interaction at run-time, and reflects the number of operations required to maintain consistency in the face of interactions.

5.1.1 Full Aggregation

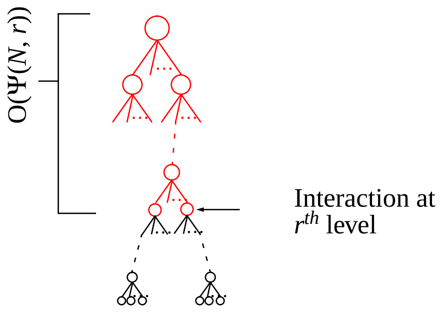

FIGURE 7. DCC for FA

In FA, an entity is simulated at the *0*th level of resolution until a higher-resolution interaction occurs. At that point, the entity is disaggregated to the appropriate level and simulated. In order to maintain consistency, the designer of the simulation has to roll back the effects of the interaction to attributes at all lower levels of resolution. Therefore, each interaction affects $O(La)$ attribute types (not attributes, but types of attributes). Assuming all interactions have independent effects (i.e., the combination of the effects of any set of interactions is the same as the effect of the combination of the same interactions),

$SCC_{FA} = O(kL \times La) = O(kL^2a)$

This is the cost of designing a function for reflecting effects of each interaction type on each attribute type. However, if we assume that pairs of concurrent interactions could be dependent,

$SCC_{FA} = O(k^2L^2 \times La) = O(k^2L^3a)$

In general, if sets of *n* concurrent interactions could be dependent,

$SCC_{FA} = O(k^nL^n \times La) = O(k^nL^{n+1}a)$

Assume an interaction at the *r*th level ($0 \le r < L$) arrives at an entity. The entity must disaggregate to level *r*, reflect the effects of this interaction at this level and aggregate back to level *0*. In order to disaggregate to level *r* from the current level *0*, the costs incurred are $O(\Psi(N, r))$. The cost of aggregation is presumably of the same order as the cost of disaggregation. Thus,

DCC_{FA} (shown in red in Figure 7) = $O(\Psi(N, r))$

If we assume that the entity does not step through every level between *0* and *r* during disaggregation/aggregation, then DCC is vastly reduced. However, SCC is then vastly increased because mapping functions must be found for each level so that the entity can “jump” the hierarchy. This optimization not only violates the strict hierarchical nature of the simulation entity, but also may lead to increased inconsistency. This is because the various levels are reachable from one another only through level *0*. Thus, they may be inconsistent with each other. In general, FA could cause inconsistency because of this tendency to revert back to level *0*, wherein there is a loss of information with respect to higher levels.

5.1.2 Full Disaggregation

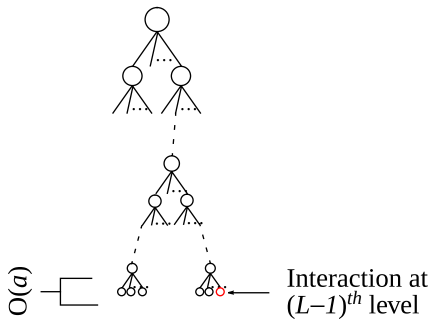

FIGURE 8. DCC for FD

In FD, all entities are always simulated at the $(L-1)$th level of resolution. Thus, there exists only one level of resolution, namely the highest. Clearly, consistency has to be maintained only within one level, a task far easier than maintaining consistency across many levels of resolution. Therefore, making $L = 1$, each interaction affects $O(a)$ attribute types. Assuming all interactions are mutually independent,

$SCC_{FD} = O(k \times a) = O(ka)$

However, if we assume that pairs of concurrent interactions could be dependent,

$SCC_{FD} = O(k^2 \times a) = O(k^2a)$

In general, if sets of *n* concurrent interactions could be dependent,

$SCC_{FD} = O(k^n \times La) = O(k^na)$

The run-time consistency costs for FD are also low. All interactions occur at the $(L-1)$th level, where $L = 1$. Therefore,

DCC_{FD} (shown in red in Figure 8) = $O(a)$

5.1.3 The MRE Approach

In MRE, an entity is simulated consistently at all levels of resolution. The relationships between low-resolution attributes and their corresponding high-resolution attributes are well-defined. These relationships — accumulative and distributive dependencies — are essentially the same across levels. In other words, the dependencies between level *k* and level *k+1* attributes are much the same as the dependencies between level *k–1* and level *k* attributes. The relationships between attributes at various levels are determined without knowledge of the expected interactions on these attributes. Therefore, during the design, each interaction affects O(*a*) attribute types. The effects are reflected to other attributes by virtue of the pre-set dependencies. Assuming all interactions are mutually independent,

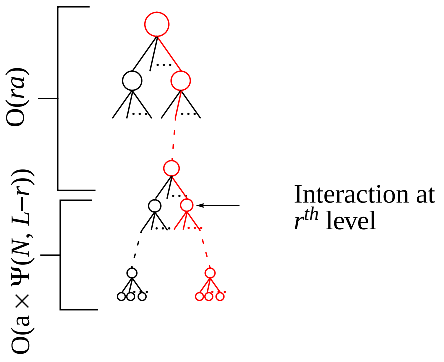

FIGURE 9. DCC for MRE

$SCC_{MRE} = O(kL \times a) = O(kLa)$

If pairs of concurrent interactions could be dependent,

$SCC_{MRE} = O(k^2L^2 \times a) = O(k^2L^2a)$

In general, if sets of *n* concurrent interactions could be dependent,

$SCC_{MRE} = O(k^nL^n \times a) = O(k^nL^na)$

The run-time costs for MRE are computed as follows. Assume an interaction at the r^{th} level ($0 \le r < L$). The entity must reflect the effects of this interaction at level = *r*, all levels > *r* and all levels < *r*. In order to reflect the interactions to higher resolution levels, the cost incurred is $O(a \times \Psi(N, L-r))$. The cost incurred in reflecting the effects to lower resolution levels is merely O(*ra*). Thus,

DCC_{MRE} (shown in red in Figure 9) $= O(ra + a \times \Psi(N, L-r))$

5.2 Simulation Cost

Simulation Cost (SC) is the cost of simulating the entities during a run of the simulation. SC may include costs of processing, memory and communication. For the purposes of this discussion we will not distinguish between these. For FA, the entity is simulated in the aggregate unless there is a need to be disaggregated. After disaggregation FA effectively becomes FD. Therefore, before disaggregation,

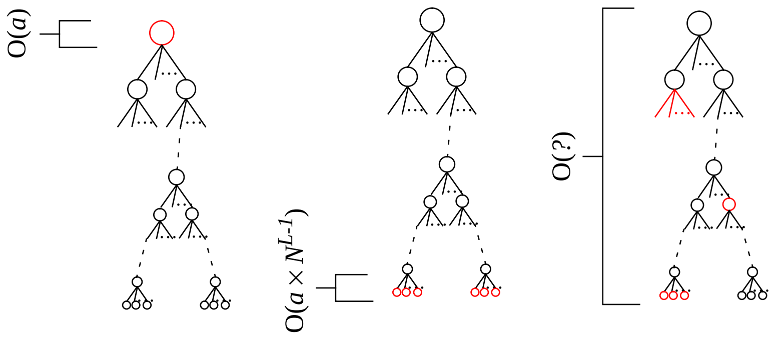

FIGURE 10. (Left to Right) SC for FA, FD and MRE

$SC_{FA} = O(a)$

In the case of FD, the entity is always simulated at the $(L-1)^{th}$ level. Therefore,

$SC_{FD} = O(a \times N^{L-1})$

Lastly, SC for MRE lies between SC for FA and FD, because in the worst case, the entity may have to be simulated entirely at the disaggregate level, but in the best case, simulation at the aggregate level may be enough. If there are concurrent interactions at all levels of resolution for an entity, and all the sub-entities at all levels need to be instantiated,

$SC_{MRE} = O(a \times \Psi(N, L))$

However, if there are interactions only at level *0*,

$SC_{MRE} = O(a)$

Figure 10 shows SC for FA, FD and MRE (from left to right) in red.

5.3 Discussion

TABLE 1: Cost Comparison

Scheme\Cost	SCC	DCC	SC
FA	O($k^nL^{n+1}a$)	O($\Psi(N, r)$)	O(a)
MRE	O(k^nL^na)	O($ra + a\Psi(N, L–r)$)	between
FD	O(k^na)	O(a)	O($aN^{L–1}$)

Table 1 compares the various costs for the different schemes considered. Based on this table, Figure 11 shows a rough diagram of expected simulation and consistency costs for FA, FD and MRE.

Consistency costs decrease with schemes that run more in the disaggregate. However, simulation costs increase. A scheme running mostly in the aggregate has low simulation costs, but high consistency costs because aggregation tends to cause information loss. The MRE scheme lies between these extremes of multi-resolution schemes. In other words, SC for MRE is expected to be lower than FD, but higher than FA, whereas DCC for MRE is expected to be higher than FD, but lower than FA.

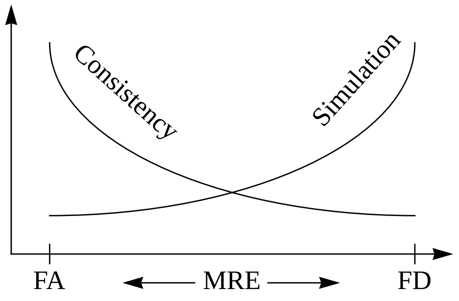

FIGURE 11. Cost Diagram

It is worthwhile to note that in a pathological case DCC_{MRE} may be slightly more than DCC_{FA}, though they will be of the same order. This is because the consistency gained by the MRE scheme is actually better than the consistency gained by the best FA scheme. FA causes information to be lost when it reverts back to level *0*. This is avoided in the MRE scheme. Likewise, in another worst case, SC_{MRE} may be slightly more than SC_{FD}, though of the same order. However, this is also justifiable in the light of MRE being able to process interactions at all levels, whereas FD is unable to accept interactions at any level except the most disaggregate.

Note the nature of $\Psi(N, L)$. Ψ is polynomial in N, but exponential in L. Since the exponential function grows faster than the polynomial one, it is recommended that for simulations with a flexible object hierarchy, effort should be directed towards making the resolution tree as broad and shallow as possible.

The effects of relaxing the introductory assumptions follow. Clearly, changing L has the most dramatic effect in changes to simulation and consistency costs. Changing N has less dramatic effects. If N is different for entities at different levels of resolution, the function Ψ becomes somewhat involved, but its basic nature does not change. If k changes with the level of resolution, then the total number of types of interactions becomes $k_0+k_1+\ldots+k_{L-1}$ instead of kL. Likewise, we could further complicate the calculations involving the number of attributes, a, by asserting that the attribute count at different levels of resolution is different. None of these modifications to the initial set of assumptions change the order of the costs; they merely make the equations more intricate.

6 Guideline Issues

We identify a number of issues to be explored using our framework towards establishing guidelines for multi-resolution simulations. These issues fall into two broad categories: consistency maintenance issues and cost reduction issues. The former are directly linked to the use of Attribute Dependency Graphs for designing MREs. The latter are more general because they are not linked to any specific methodology. Rather, any scheme with characteristics similar to ours is likely to exhibit the cost characteristics we outline. The issues outlined in the next two sections are pertinent to the formulation of multi-resolution design guidelines.

6.1 Consistency Maintenance Issues

Multi-resolution simulations must pay heed to the Fundamental Observations if they are to ensure consistency. The Attribute Dependency Graphs outlined earlier encode the consistency maintenance problem and solutions to it. The important contributions of the graph theory are:

- Imposing semantics on the nodes and edges identifies the components that could be modified concurrently.
- Indicates the order in which the attributes must be modified per interaction respecting the dependencies between the attributes.
- Semantics of the arrows ensure that the reversible mapping functions problem is addressed.
- Accommodates a very heterogeneous set of nodes, wherein some nodes may be entities, others sets of

entities, some others attributes and yet others attributes of attributes.
- By assigning fractional weights to the edges, a solution obtained can be shown to subsume traditional notions of disaggregation.
- Reduces the original problem of concurrent dissimilar interactions at multiple levels on the MRE to the easier problem of multiple similar concurrent interactions at the same node.

Issues that must be addressed with the graph theory are:
- Edge-weighting algorithms: We expect to leverage off existing algorithms used for partial disaggregation, because of the similarity between the two concepts.
- Multiple concurrent interactions at a node: There does not exist one general solution to this problem because any solution choice satisfying the requirements of one node would fail for another. Since a multiplicity of choices exists, designers must choose from a set of potential solutions or general solution characteristics. The choice of solutions for a node may be made statically or dynamically.

For a given application, the designer of the MRE must choose appropriate functions for each node to account for concurrent interactions on that attribute, where appropriateness is defined by the semantics of that attribute's role in the MRE. The MRE can then be made to behave in a consistent manner in the face of concurrent interactions at multiple levels of resolution. The graph emphasizes the relationships between the attributes and also depicts the manner in which the effects of an interaction ripple through the MRE. The quality of the consistency achieved by a solution derived from the graph theory does not depend on the application-specific functions at each node. Rather, those functions affect merely the semantics of the MRE, not its consistency. Consistency is captured in the graph itself, and if fractional weights are introduced, then a few simple rules (e.g., sum of the fractional weights on the distributive edges of an attribute must sum to 1) are enough to ensure that the MRE remains consistent.

6.2 Cost Reduction Issues

Simulation-specific features can help reduce costs. As illustrated earlier, broad and shallow resolution hierarchies are better than deep hierarchies. Interactions with independent effects help reduce static design costs. One of the initial assumptions is that every interaction affects all *a* attributes of the entity (and thence to other levels). This need not be true for a real simulation. Typically, interactions in real simulations would affect only a subset of *a*, thus bringing down costs further. The MRE approach may be thought of as an adaptive approach that swings between the extremes of FA and FD depending on the state of the simulation.

7 Proposed Work

Over the last ten months, we have developed a general framework for designing multi-resolution simulations, consisting of two important components: the Multiple Resolution Entity and the Attribute Dependency Graph. This framework helps capture the essential issues in consistency maintenance in a multi-resolution simulation. In doing so, it helps identify potential solutions as well. The primary objective of the continuing work proposed here is to formulate guidelines for resolving multi-resolution representation issues. These guidelines will benefit designers and developers of new and legacy systems to be used in HLA federations that span multiple levels of resolution. To achieve this objective, we will further develop the framework described here and apply it to real-world simulations/ federations. The ADGs represent the foundation of our framework, and we believe that solutions to the representation issue will emanate from ADGs. Issues with the ADG that represent our next challenges are: (i) semantics of nodes and edges, (ii) multiple edges incident on nodes, (iii) graph traversal, and (iv) weighting edges. We believe we have a good understanding of (i) and, to some extent (ii); we intend to explore algorithms for (iii) and (iv).

The costs involved in designing a multi-resolution simulation are also of crucial interest. On the one hand, a simulation that achieves consistency at a prohibitive cost may not be desirable. On the other hand, a low-cost simulation that compromises consistency is of little value. We expect the costs of implementing MREs based on the Attribute Dependency Graphs will be overshadowed by the benefits of the resulting improved consistency. In addition, we believe our approach may be able to achieve the right balance between simulation and consistency costs.

Environmental factors present unique challenges to the simulation community. While the bulk of simulations address issues involved with simulating entities such as tanks, platoons and companies, most simulations treat the environment unsatisfactorily. The MRE model is not restricted to just tanks, platoons or companies, but also environmental factors such as terrain and atmosphere. We have had preliminary thoughts on the issue of designing terrain as an MRE. We would like to compare this design with existing schemes and measure the costs involved.

The following are the major milestones for the proposed work:

- *Apply our framework to existing simulations*: This would be done by studying existing FOMs and extracting information from them leading to the design of ADG-based MREs for the entities involved.
- *Complete cost analysis*: Our current work represents a rough cut at characterizing the costs of consistency maintenance in multi-resolution simulations. We would like to perform this study on all important existing schemes and compare them with ours. Our analysis could be supported by simulation.
- *Explore optimizations to MRE design*: We have a notion of a core attribute set maintained at all times that might be sufficient for consistency. This and other optimizations would be geared towards reducing simulation cost without compromising consistency.
- *Study effect of relaxing Fundamental Observations*, particularly FO-4: Compatible time-steps are an important issue in our theory. However, we realize that for the short term it may be unrealistic to expect simulation designers to be able to coerce legacy simulations into running at compatible time-steps. Thus, we would like to study the effect of relaxing this assumption (and maybe others) in order to educate designers about the expectations they may entertain about simulations that do not conform to FO-4.
- *Educate simulation community* about multi-resolution problems and offer guidelines towards solutions: Multi-resolution issues are manifest in many existing simulations because the community is largely unaware of the implication of the Fundamental Observations. The long-term goal of our work is to educate designers about the FOs, depict the costs of various approaches towards multi-resolution simulations and suggest guidelines on where costs might be cut without compromising consistency.

8 Cost Estimate

Item	Cost
Labor	42,500
Travel	4,000
Materials & Supplies	2,000
Technical Services	4,000
In-state tuition	5,047
University of Virginia overhead	27,047
Total	**84,595**