AFRL-RI-RS-TR-2010-188

MODELING AND SIMULATION WITH OPERATIONAL DATABASES TO ENABLE DYNAMIC SITUATION ASSESSMENT & PREDICTION

WarpIV Technologies, Inc.

November 2010

FINAL REPORT

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the 88th ABW, Wright-Patterson AFB Public Affairs Office and is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2010-188 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/s/

MICHAEL T. GACEK
Work Unit Manager

/s/

JULIE BRICHACEK, Chief
Information Systems Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
NOVEMBER 2010	Final Technical Report	February 2008 – September 2010

4. TITLE AND SUBTITLE

MODELING AND SIMULATION WITH OPERATIONAL DATABASES TO ENABLE DYNAMIC SITUATION ASSESSMENT & PREDICTION

5a. CONTRACT NUMBER
FA8750-08-C-0070

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
62702F

6. AUTHOR(S)
Craig Lammers
Jeffrey S. Steinman

5d. PROJECT NUMBER
558S

5e. TASK NUMBER
08

5f. WORK UNIT NUMBER
01

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
WarpIV Technologies, Inc.
Ste 306
5230 Carroll Canyon Road
San Diego CA 92121

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory/RISB
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RISB

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-RI-RS-TR-2010-188

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited. PA# 88ABW-2010-5672
Date Cleared: 26 October 2010.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The Air Force Research Laboratory is investigating a next-generation planning capability that combines advanced optimistic discrete-event modeling and simulation technologies with operational databases to enable dynamic situational assessment and prediction in a live effects-based dynamic Intelligence, Surveillance and Reconnaissance (ISR) Joint Command and Control (C2) environment. Combined capabilities were designed to facilitate dynamic re-planning in an effects-based C2 environment. The software developed for this effort provides a semi-automated planning system that suggests alternative courses of action when, based on intelligence data, re-planning is required. The plan is in the form of an Air Tasking Order (ATO) that declares mission objectives for aircraft assets. Measuring the effectiveness of a plan and determining when a plan has gone off course requires knowledge and representation of both the intended plan and real world execution of the intended plan over time. This document captures the software design for a next-generation dynamic situational assessment and prediction capability that combines the latest advances in modeling & simulation technologies with control theory algorithms, optimization techniques, and operational databases.

15. SUBJECT TERMS
Multi-Core, Cluster, Dual Core, Quad Core, HPC, Modeling, Simulation, WarpIV, High Performance Computing, Shared Memory, Open UTF, PDMS-SSG

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			MICHAEL T. GACEK
U	U	U	UU	52	19b. TELEPHONE NUMBER (*Include area code*) N/A

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 Introduction

The Air Force Research Laboratory is investigating a next-generation planning capability that combines advanced optimistic discrete-event modeling and simulation technologies with operational databases to enable dynamic situational assessment and prediction in a live effects-based dynamic Intelligence, Surveillance and Reconnaissance (ISR) Joint Command and Control (C2) environment. Dynamic situational assessment refers to the ability to determine the current operational state of the real world battlespace based on the arrival of potentially incomplete, inaccurate, and/or noisy data from multiple sources in real-time. Dynamic situational prediction refers to the ability to rapidly predict potential outcomes of alternative courses of action. These combined capabilities were designed to facilitate dynamic re-planning in an effects-based C2 environment.

The software developed for this effort provides a semi-automated planning system that suggests alternative courses of action when, based on intelligence data, re-planning is required. The plan is in the form of an Air Tasking Order (ATO) that declares mission objectives for aircraft assets. Measuring the effectiveness of a plan and determining when a plan has gone off course requires knowledge and representation of both the intended plan and real world execution of the intended plan over time. A simulation was developed using the WarpIV Kernel to address this by providing a (1) real-time scripted execution of the ATO for aircraft bombing missions, (2) improved state-estimate of the Common Operational Picture (COP), and (3) mechanism to detect deviation between the intended plan and reality. Interfaces were developed to enable live operational databases to inject real-time insight into the state of neutral, friendly, and hostile forces. Position data injected into the simulation can be taken at face value or assumed to be noisy, inaccurate, and/or delayed. Kalman Filters are used to fuse position data from multiple sources to form entity tracks. The WarpIV simulation can output Distributed Interactive Simulation (DIS) Protocol Data Unit (PDU) messages to visualize the state in a DIS-compliant simulator.

Scenarios that require dynamic re-planning could include the (1) failure of an asset to complete a mission, (2) unexpected destruction of a pivotal asset, (3) introduction of a newly detected hostile threat into the battlespace, and (4) change in behavior of a neutral, friendly, or hostile entity. Currently, the re-planning process is automatically triggered due to the (1) unexpected introduction of a red-force entity into the battlespace or (2) unexpected destruction of a blue-force aircraft. The general framework can be extended to support other re-planning triggers. Alternative courses of action are generated by an external optimization tool (STOMP), and rapidly evaluated and ranked using the WarpIV HyperWarpSpeed simulation technology.

This document captures the software design for a next-generation dynamic situational assessment and prediction capability that combines the latest advances in modeling & simulation technologies with control theory algorithms, optimization techniques, and operational databases. WarpIV Technologies, Inc. leveraged its WarpIV Kernel to provide these capabilities to the DoD community. In addition, performance benchmarks of the enabling M&S technology conducted on a wide variety of computing architectures are presented. Finally, lessons learned and opportunities for future work are discussed.

1.1 Objective

The objective of this effort is to provide a next-generation planning capability that combines advanced optimistic discrete-event Modeling & Simulation (M&S) technologies with operational databases to enable dynamic situational assessment and prediction in a live effects-based dynamic Intelligence, Surveillance and Reconnaissance (ISR) Joint Command and Control (C2) environment. Dynamic situation assessment and prediction helps facilitate dynamic re-planning in an effects-based C2 environment.

1.2 Motivation

There is a need for technology to facilitate dynamic re-planning in real-time. The battlespace is a highly dynamic and changing environment. The optimal plan based on current information may no longer be optimal as additional information is received. Unforeseen red-force entities can enter the battlespace and critical blue-force assets can be destroyed, thereby invalidating the current plan. It is possible that the execution of a plan may not proceed as expected. In addition, plans can become ineffective over time.

It is difficult to determine when, and by how much, a plan has gone off course and therefore requires re-planning. There is a need for a real-time dynamic re-planning capability to automatically suggest alternative courses of action. A real-time capability requires highly efficient algorithms and parallel processing resources. These fundamental requirements form the cornerstone of our approach to the problem. In particular, independent performance benchmarks conducted by AFRL demonstrated scalability on tests using up to *4-million entities* and *700 processors*.

1.3 Summary

The scope of this effort is to combine advanced optimistic discrete-event M&S technologies with operational databases to enable dynamic situational assessment and prediction in a live effects-based dynamic ISR Joint C2 environment. The solution will utilize advanced optimistic (i.e. rollback-based) simulation technologies in conjunction with control theory techniques and operational databases to provide an improved state-estimate of the battlespace in real-time. In addition, a new five-dimensional simulation capability will evaluate alternative courses of action when re-planning is required. The five-dimensional simulation technology enables a wider exploration of potential outcomes by supporting multiple decision branches within a single predictive execution of a simulation, while simultaneously taking into account real-time data to ensure useful predictions.

Inputs into the system include Air Tasking Orders (ATO) declaring mission objectives for aircraft assets, and data from live operational databases such as the Theater Battle Management Core Systems (TBMCS) providing potentially incomplete, inaccurate, and noisy ground-truth estimates on the state of the real world. Outputs produced by the system include alternative courses of action that have been automatically generated, evaluated, and ranked when re-planning is deemed necessary.

Various elements are required to solve the problem. First is the need for a real-time simulation technology capable of receiving, interpreting, and utilizing live real world intelligence data during execution to provide an improved state-estimate of the real world. Second is the need for a capability to determine when the real world execution of the commander's plan has gone awry; this will trigger the re-planning process. Third is the need for a capability to automatically generate viable alternative courses of actions (this capability is provided by Charles River Analytics). Fourth is the need for an advanced predictive simulation environment to rapidly evaluate these alternative courses of action. Fifth is the need for a plan ranking capability that automatically determines the best overall course of action based on predictive simulation results. Sixth is the need for an underlying architecture that ties all of these components together. Since a completely automated re-planning capability is unrealistic and potentially dangerous, the commander must be able to override the system and manually generate or select alternative courses of action.

The scope of this problem is immense, but the benefits gained by developing a next generation system that performs these integrated capabilities in an efficient manner would be of tremendous value with a wide range of application domains beyond DoD Command and Control. Examples might include Air Traffic Management (ATM) for the Federal Avionics Administration (FAA), congestion management to monitor, predict, and direct freeway traffic for the Department of Transportation (DOT), electrical grid power management for the Department of Energy (DOE) to manage energy consumption during periods of stress on the system, nuclear power plant operation, the world economy, world health and infectious disease control, and a wide variety of business applications.

2 Supporting Technology

The WarpIV Kernel and HyperWarpSpeed provided the core enabling technology for this effort. These technologies are discussed in the following subsections.

2.1 WarpIV Kernel

The WarpIV Kernel provides the core software infrastructure for WarpIV Technologies, Inc. high-performance computing technologies. It is comprised of a suite of high-speed communications services, software utilities, and advanced compute engines. At the heart of the WarpIV Kernel is an advanced parallel discrete-event simulation engine that enables scalable performance of models executing on multicore computing architectures.

The WarpIV Simulation Kernel hosts discrete-event simulation in parallel and distributed environments. The WarpIV Simulation Kernel provides a next-generation object-oriented computational framework that combines state-of-the-art distributed event processing with powerful modeling constructs and support utilities. This delivers scalable performance across different computer and network architectures, while simplifying the effort of constructing models for simulation developers. The WarpIV Kernel supports heterogeneous networked applications and executes over Linux/Unix/Windows (PC, Mac, SGI, Sun, HP, etc.) platforms.

The WarpIV Simulation Kernel offers several adaptive time management algorithms that coordinate discrete-event processing in a parallel and distributed environment. Sequential, conservative, optimistic, and five-dimensional time management capabilities are provided in an integrated framework to maximize performance while maintaining the overall system stability through dynamic flow control techniques. A full-featured rollback framework provides automatic rollback support when running optimistically.

The WarpIV Kernel promotes an open-source component-based plug-and-play modeling paradigm to integrate reusable models. The WarpIV Kernel provides an open-source reference implementation of the Open Unified Technical Framework (OpenUTF) that is under investigation for possible standardization within the SISO Parallel and Distributed Modeling and Simulation Standing Study Group (PDMS-SSG).

The WarpIV Kernel is the culmination of 20 years research and development. The technology began in 1990 at JPL/CalTech with the development of the Synchronous Parallel Environment for Emulation and Discrete Event Simulation (SPEEDES). Development of the WarpIV Kernel began in 2001 as the next-generation replacement for SPEEDES in a backward compatible manner. The WarpIV Kernel has continued steady development through the present under several programs. The WarpIV Kernel contains over 340,000 lines of C++ and Java code.

2.2 HyperWarpSpeed

Computer simulation plays an important role in the decision making process. Factors such as financial constraints, safety concerns, physical impracticalities, and time constraints often prohibit conducting real-world studies, making simulation a viable alternative. Achieving a valid statistical representation from a simulation often requires executing many independent Monte Carlo runs with different parameter values, scenario excursions, and/or random number seeds.

Time constraints prohibit the majority of experiments from exploring the full set of experimental permutations, although the ability to do so in an efficient manner would be of tremendous value.

A new five-dimensional approach to modeling and simulation, known as HyperWarpSpeed, was developed to address this very problem. HyperWarpSpeed is unique in its ability to explore multiple decision branches within a single simulation execution while eliminating redundant computations between overlapping timelines. In effect, HyperWarpSpeed supports the mutual interaction of multiple event timelines continually being created and merging as they converge within a five-dimensional simulated universe.

HyperWarpSpeed transparently runs on multicore computing architectures using discrete-event optimistic processing. Rollbackable event processing allows this technology to support real-time predictions with live data feeds that continually calibrate the models. In this manner, five-dimensional predictions are always based on best estimates of the dynamically changing current world state. [1]

WarpIV Technologies, Inc. developed HyperWarpSpeed in 2006 under funding by the Air Force Research Laboratory in Rome, NY [2]. This effort involved the application and extension of HyperWarpSpeed.

2.2.1 Introduction

Traditional computer simulations represent systems in four dimensions, i.e. space plus time. HyperWarpSpeed [3] is unique in that it can internally spawn multiple behavior timelines at key decision points within a single simulation execution. These behavior timelines interact as necessary within a five-dimensional parallel universe.

One way to understand how this technology works is to consider a chess game, where instead of two players taking turns exploring possible moves, there are an unlimited number of players exploring possible moves whenever necessary. This capability provides an invaluable mechanism for efficiently conducting "what if" type analysis. Depending on the scenario, literally billions of decision permutations can be explored within a single simulation execution that perhaps takes just a few times longer to complete than the execution of a single permutation. Whereas Monte Carlo simulation experiments may re-compute up to 90% of the same calculations within each replication, HyperWarpSpeed eliminates these redundancies by sharing computations between identical timelines.

2.2.2 Technical Background

As opposed to cloning an entire simulation or cloning simulated objects at key decision points, HyperWarpSpeed offers a more flexible, robust, scalable, and efficient solution to exploring multiple decision paths. Coordinating how the alternative timelines interact is completely automated and transparent through unique multivariable state management structures, event splitting, and event merging techniques. These techniques are briefly discussed in the following subsections.

2.2.2.1 Replication Sets

Fundamental to HyperWarpSpeed is the notion of replication numbers and replication sets. Monte Carlo simulations uniquely identify individual runs with a replication number. A replication number can be used to (1) set different starting seeds for pseudo random number generation, (2) specify a set of model performance parameters, (3) identify a particular set of decisions in a battle plan, and/or (4) identify enemy responses.

A *replication set* is simply a collection of replication numbers. HyperWarpSpeed represents replication sets as bit fields, where each bit set to one identifies a replication in the set. For example, a replication set with 32 potential replications containing replications {0, 1, 4, 9, 15, 29} would be stored as binary 0010 0000 0010 0000 1000 0010 0001 0011, where the rightmost bit represents replication 0, the next bit to the left represents replication 1, etc.

To achieve scalable performance, high-speed bit-masking instructions automate most of the critical bookkeeping operations for replication sets. Both events and state variables are associated with replication sets. State variables manage an array of values where the array index is associated with the replication number. Events keep track of which replication numbers are represented by the event. At the start of the simulation before any branching occurs, (1) state variables store the same value for all possible replications and (2) all initially scheduled events represent the full set of replications. Branching, event splitting, event merging, and multireplication state variable assignments occur as events are processed.

2.2.2.2 Branching

Branching enables a HyperWarpSpeed model to simultaneously explore multiple decision points within a special event type known as a process. An event is a method on a C++ object; a process is an event that is able to pass time without exiting the method. Special modeling constructs are provided by the WarpIV Kernel to allow processes to (1) wait specified periods of time before waking up, (2) wait for interrupts to occur, and/or (3) wait for necessary resources to become available before continuing processing. Processes can have multiple wait statements within a single method.

HyperWarpSpeed allows processes to branch using syntax similar to the standard switch-case statement. Each decision point is currently limited to twenty branches. At the decision point, model developers specify the probability of taking each branch. HyperWarpSpeed schedules new processes that continue execution with subdivided replication sets based on the specified probabilities for each branch label. The maximum number of branches within an individual timeline is dependent on the replication set size and branch probabilities. For example, a replication set of 128 could evenly subdivide 7 times, resulting in 128 unique permutations for a particular behavior operating within a larger simulation context modeling a vast number of behaviors. Any additional branch statements encountered would flip a coin to determine which branch to evaluate, as in a Monte Carlo run.

2.2.2.3 Multireplication State Variables

Multireplication state variables manage an array of values, where each element in the array corresponds to a particular replication number. State variables are potentially accessed and/or modified by events with arbitrary replication sets. Multireplication data types are implemented as

C++ objects representing integers, doubles, and Boolean values. With operator overloading, these data types behave as normal variables. Under the hood, they are managing multiple values for different replication subsets.

2.2.2.4 Event Splitting

Event splitting occurs when an event or process with a particular replication set accesses multireplication variables with different values. For example, a Boolean multireplication state variable data type might store the value ***true*** for replications {0, 1, 4, 9, 15, 29}, and ***false*** for all other replications. An event or process with replications {5, 9, 12, 15, 20} will likely produce different results depending on the stored value. So, HyperWarpSpeed automatically processes the event twice, once for replications {5, 12, 20} and then again for replications {9, 15}. During event splitting operations, HyperWarpSpeed automatically tracks replication set dependencies. Any event or process may access several multireplication state variables, causing arbitrarily complex event splitting.

2.2.2.5 Event Merging

Event merging enables identical timelines to recombine whenever possible. It is entirely possible for (1) identical events to be scheduled during event splitting, and for (2) pending unprocessed events to be identical. HyperWarpSpeed merges identical events before they are processed to consolidate event scheduling and processing overheads. Without this merging capability, HyperWarpSpeed would branch and split events in an exponential out-of-control manner. If decision points are independent of one another, the simulation will continually diverge and merge between decision points.

2.2.2.6 A Simple Example

Imagine within a simulated world that two aircraft, EAGLE and HAWK, are en-route to a target of opportunity. During engagement, there is a probability that EAGLE's communication system is impacted, and as a result, becomes inoperable. In HyperWarpSpeed, EAGLE branches into two overlapping parallel universes. In one universe, EAGLE's communication system is functional. In the other universe, EAGLE's communication system is inoperable. Meanwhile, other missions continue unaffected by the state of EAGLE's communication system. Both branches (universes) of EAGLE share (overlap) the modeling of these independent missions.

During battle, HAWK noticed shots were fired at EAGLE and radios to him. At this point, HAWK automatically splits into two. In one universe, HAWK successfully communicates with EAGLE. In the other universe, HAWK is unable to communicate with EAGLE. After successfully completing the mission, EAGLE and HAWK return to base. Assuming they do not speak to one another, their universes that had previously branched and split now merge because their flight back to base is independent of the state of EAGLE's communication system.

While en-route to base, EAGLE attempts communication with air traffic control (ATC). This automatically causes EAGLE to split. The EAGLE that had previously exited battle unscathed establishes communication, but the EAGLE in the alternate universe that was impacted was unable to establish communication. What caused the split is the fact that the state of EAGLE's communication system depended on the outcome of events earlier in the day. The Futures Graph of this scenario is shown in Figure 1.

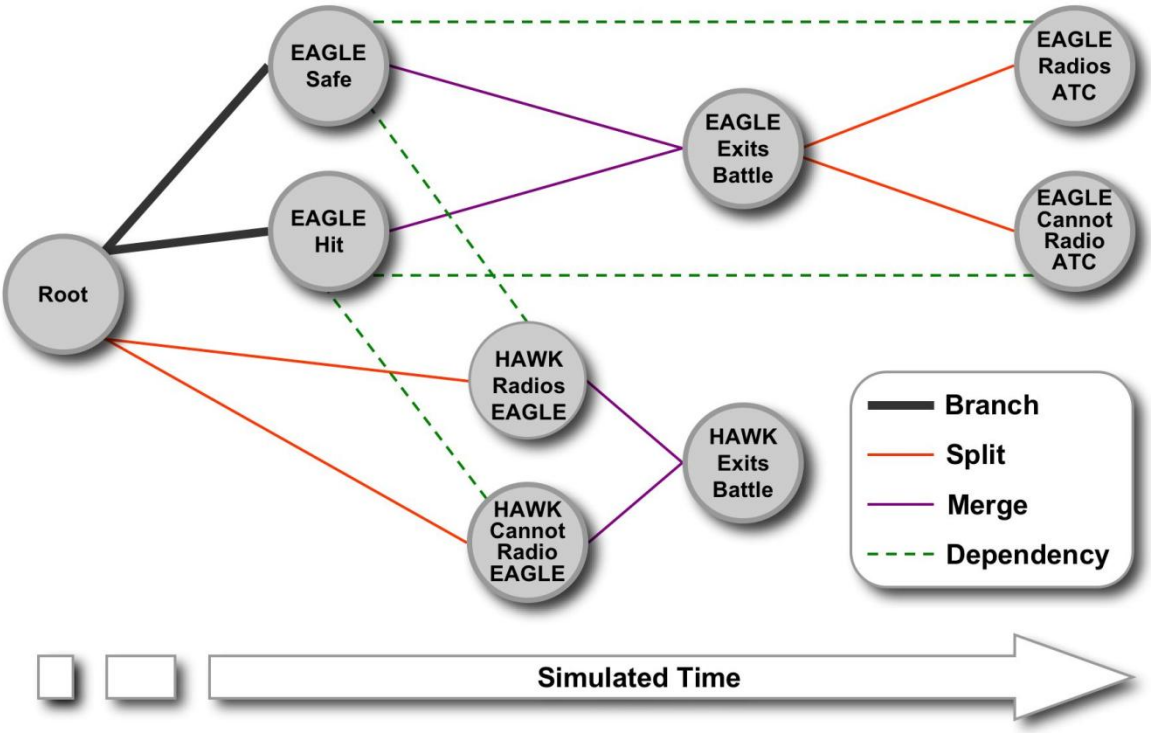

Figure 1: Futures Graph representation of EAGLE and HAWK.

3 Methods, Assumptions and Procedures

A high-level description of the Dynamic Situation Assessment and Prediction (DSAP) framework developed to support this effort is illustrated in Figure 2. The framework consists of five specific pieces: (1) battle plans via Air Tasking Orders (ATO) and real-time operational battlefield data via Theater Battle Management Core Systems (TBMCS), (2) WarpIV real-time simulation, (3) alternative plan generation, (4) HyperWarpSpeed five-dimension predictive simulation, and (5) plan ranking.

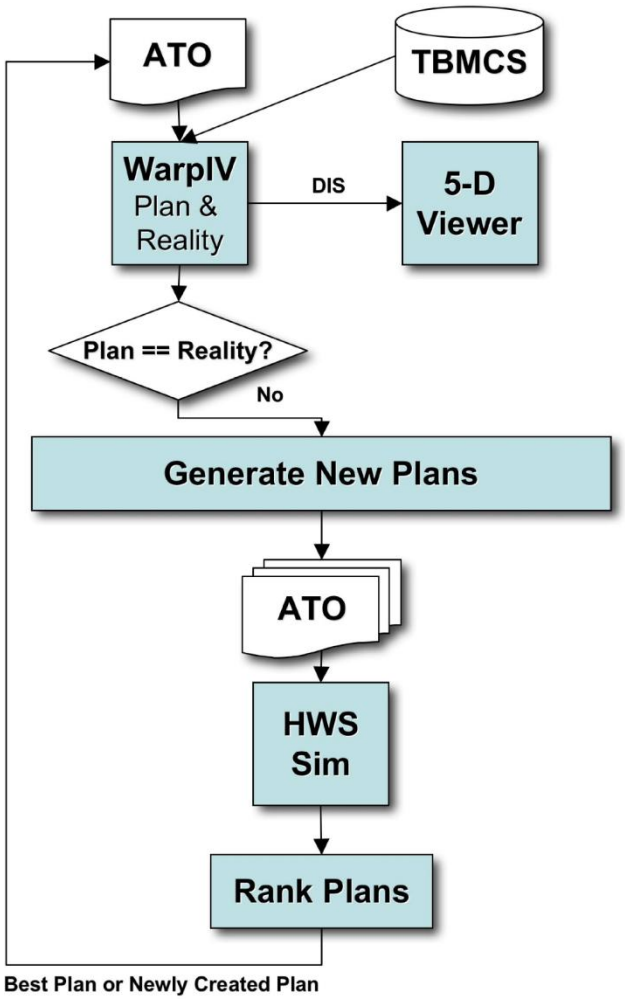

Figure 2: The prototype Dynamic Situation Assessment and Prediction framework.

An ATO represents the commander's intended plan for blue-force assets and is used to initialize the system. TBMCS is a source of battlefield ground-truth data. WarpIV provides a real-time simulation of the commander's intended plan and provides a continually updated state-estimate of the real world battlespace within a single real-time execution. WarpIV is capable of responding to state updates from outside sources, such as those originating from operational databases. The WarpIV simulation forms the basis of comparison that determines when the commander's plan has gone awry and re-planning is required. At the re-planning stage, an optimization component generates alternative courses of action based on the current state of the

battlespace and air tasking orders, and each plan is simulated and ranked based on its effectiveness using HyperWarpSpeed predictive simulation. At this point in time, the commander can accept one of the alternate plans or manually generate a new plan, and the system repeats the cycle using the new plan. The initial implementation was designed to allow WarpIV to indirectly receive TBMCS inputs through JSAF via DIS. Development is underway on another effort to provide a 5-dimensional display to help commanders visualize uncertain outcomes over time. In a sense, this provides a 5-D Common Operation Picture (COP).

The specific software components developed to provide the DSAP capability are described in the following subsections.

3.1 TBMCS Client

The TBMCS Client program extracts an Air Battle Plan from the TBMCS database and saves it in the form of a (1) scenario composition file that can be simulated within the WarpIV Kernel, and (2) STOMP Scenario and AODB file that can execute within STOMP. In addition, the TBMCS Client is capable of sending detection, track, and detonation information to the real-time simulated representation of the commander's intended plan and COP (refer to Section 3.3). A graphical interface to the TBMCS Client program was developed using Java to ensure cross-platform support.

A screen capture of the Graphical User Interface (GUI) is shown in Figure 3. In this screen capture, the program is extracting the Pacifica scenario from TBMCS and, upon completion, will convert it to a composition file that defines and dynamically composes a WarpIV simulation at run-time. A drop-down menu enables the user to select the scenario from TBMCS, and a progress bar displays the progress of the query. Clicking on the *Save Directory* text field brings up a file chooser for changing the default save location of the WarpIV composition file. Data extracted from the database is displayed in the terminal window in the background.

A TBMCS account is required to use the TBMCS Client program. After logging in with a valid user name and password, a comprehensive list of Air Battle Plans within TBMCS is displayed in a drop-down menu. After selecting an Air Battle Plan, the WarpIV composition file and STOMP scenario and AODB files will be saved in the user-specified directory.

The TBMCS Client has a few limitations. First, the program only extracts objectives for aircraft bombing missions; pure surveillance missions are currently ignored. Second, the WarpIV models do not model in-flight aircraft refueling, so any refueling orders within the battle plan are currently ignored. Lastly, the program only extracts the first target if multiple target locations are specified. This limitation is attributed to the fact that the STOMP mission optimization tool only handles one-to-one mappings between aircraft and targets.

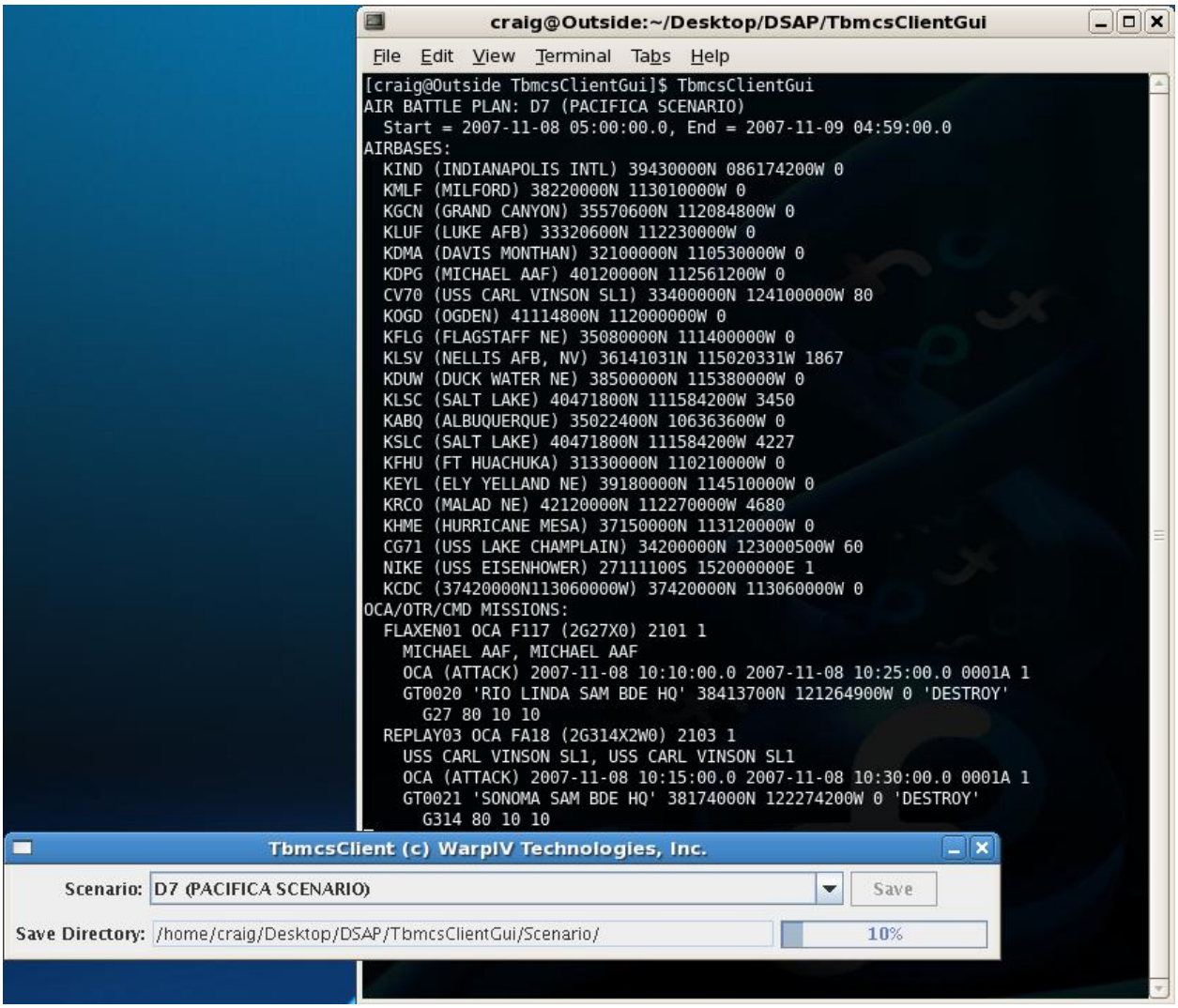

Figure 3: Screen capture of TBMCS Client GUI.

3.2 TBMCS Modifier

As a mission unfolds, changes in the operational state of the battlespace are likely to occur. In an operational environment, those changes are relayed to the TBMCS database. Depending on the severity of those changes, mission re-planning may be required. A TBMCS Modifier program was developed to mimic dynamic operational changes in state within TBMCS. Screen shots of the GUI are shown in Figure 4 and Figure 5.

Specifically, this program allows for the creation and deletion of newly defined red-force threats within pre-defined TBMCS scenarios. Newly defined entities are created or deleted within a standalone TBMCS sandbox as not to affect other users of the system. The TBMCS Client, discussed in Section 3.1, automatically and regularly monitors the current scenario within TBMCS to check for new red-force threats. When new red-force threats are detected, this information is relayed to the simulated representation of the commander's scripted plan and estimated state of the operational world.

Figure 4: Screen capture of TBMCS Modifier GUI at login.

Figure 4 illustrates the TBMCS Modifier GUI at login. For security reasons, the Password field displays the characters typed in the text field with an asterisk.

Figure 5 illustrates the TBMCS Modifier program in action. Adding a new target entity to the TBMCS database requires the specification of a unique Air Battle Plan (ABP) Request ID and target name. The ABP Request ID is automatically generated based on the number of entities in the current scenario, and guaranteed to be unique. The program verifies the target name is unique. Tooltips describe each button, menu, and text field in the GUI. To protect against accidentally obliterating a scenario, the only entities that can be removed from a scenario are entities that have been created using the GUI. In this case, no new entities have been created yet, so the *Remove* button is disabled.

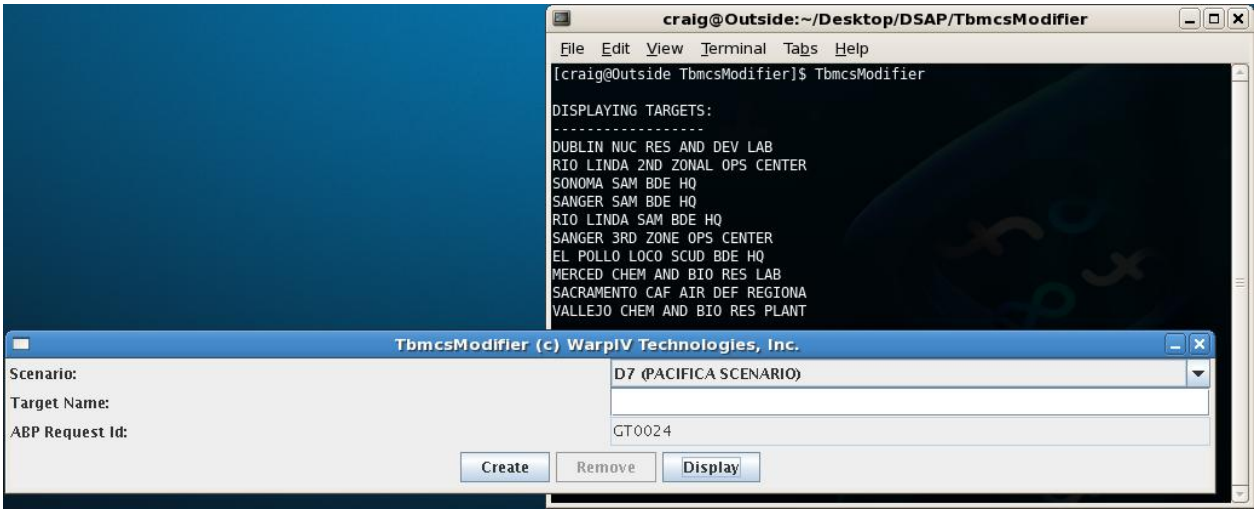

Figure 5: Screen capture of GUI displaying the list of targets in the Pacifica scenario.

3.3 Simulation

A simulation was developed to represent aircraft bombing missions based on an Air Tasking Order. The simulation is capable of executing in (1) real-time, (2) predictive, and (3) simulated TBMCS mode. The real-time simulation mode provides a representation of both the commander's intended plan and Common Operational Picture (COP) over time, as well as a mechanism to automatically determine when re-planning is necessary. Re-planning is required when the real world execution of the mission has deviated from the intended plan. To allow visualization in an external viewer or simulation (such as JSAF), the real-time simulation outputs DIS EntityState, Fire, and Detonation Protocol Data Units (PDU) from both the scripted plan and COP. The predictive simulation mode offers faster-than-real-time execution of alternative courses of action using HyperWarpSpeed decision-branching technology. The TBMCS

simulation mode provides a simulated representation of the true state of the battlespace over time. This mode is useful for demonstration purposes when direct access to TBMCS is unavailable. By varying the command line arguments, one of these three modes can be executed at runtime. Slightly different models are used for the predictive HyperWarpSpeed simulation.

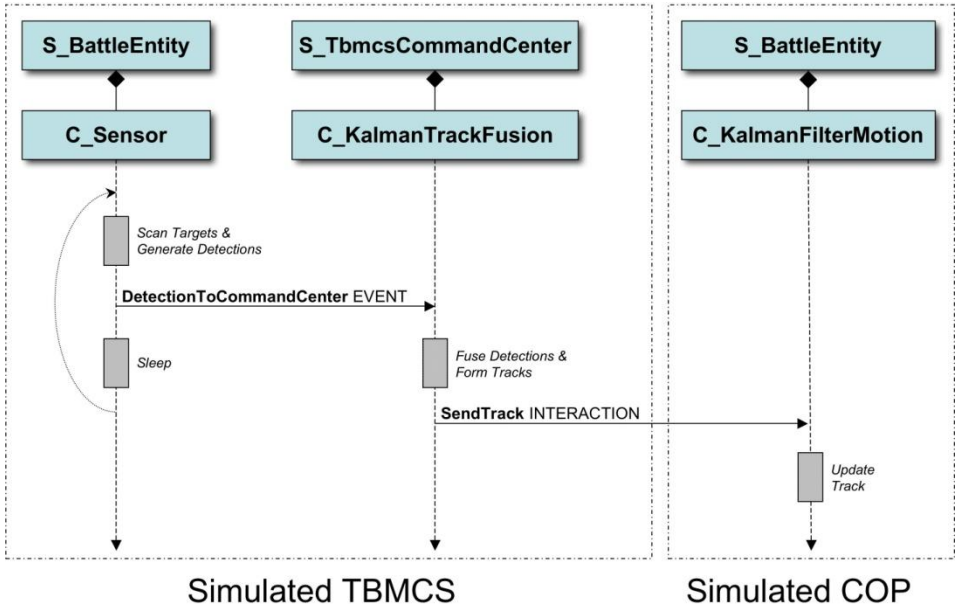

Figure 6: Interaction between the simulated TBMCS & COP.

The interaction between the simulated TBMCS and COP is shown in the sequence diagram in Figure 6. Each entity in the simulated TBMCS contains a sensor component that periodically scans for targets within range and generates raw detections. Afterwards, the sensor schedules an event that passes the raw detection to a central command center. The command center contains a KalmanTrackFusion component that fuses the raw detections and forms tracks for each entity. As tracks are formed, the command center schedules an interaction for the appropriate component in the COP simulation with the new track. Each entity in the COP simulation contains a KalmanFilterMotion component that guides its motion by responding to the interaction.

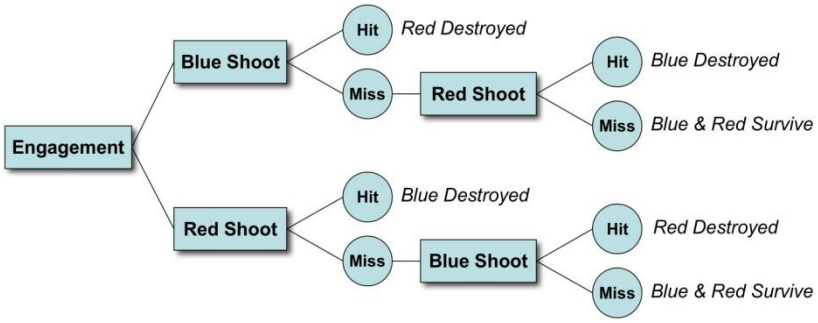

Figure 7: Decision branch points within the HyperWarpSpeed predictive simulation.

The decision branch points within the HyperWarpSpeed predictive simulation are shown in Figure 7. In the example, there are six permutations. Extending this example to *N* aircraft, there

are 6^N permutations. Even though there are six permutations for each aircraft, there are only three unique outcomes. The unique outcomes are (1) Red is destroyed, (2) Blue is destroyed, or (3) Red and Blue survive. The overlap enables computation sharing within HyperWarpSpeed.

The predictive simulation currently branches based on the attack sequence and detonation result. The probability of a blue or red-force entity taking the first shot is specified in the Scenario.rtc parameter file. The probability of a red-force entity detonating is based on the effectiveness of the munitions used by the blue-force aircraft. The probability of a blue-force entity detonating is based on a probability specified in the Scenario.rtc parameter file.

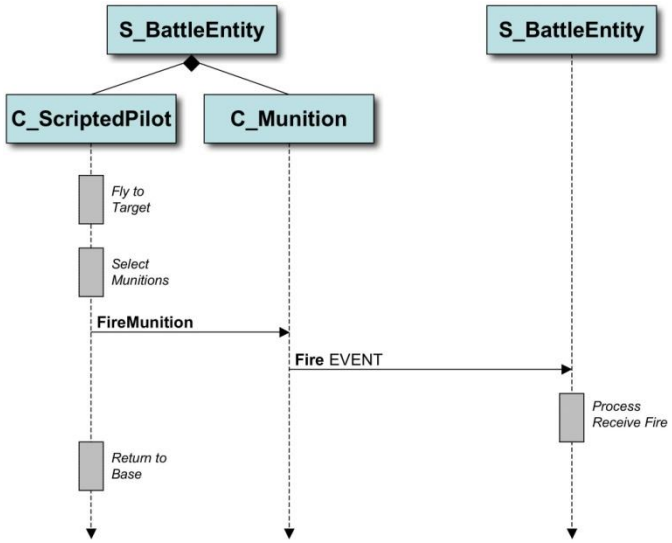

Figure 8: Sequence diagram of a scripted aircraft pilot conducting a mission.

The sequence diagram of a scripted aircraft pilot conducting a mission is illustrated in Figure 8. The pilot flies the aircraft to the location of its assigned target, selects the best munitions from those available, and fires at the target when within range. The target then determines the impact of the munitions. If the aircraft is still withstanding after engagement, the pilot returns the aircraft to base.

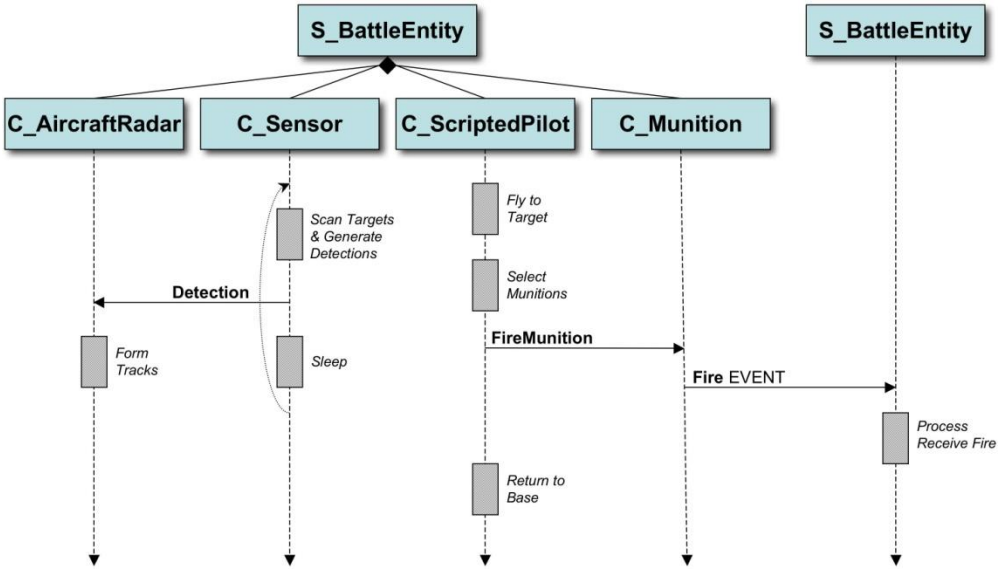

Figure 9: Sequence diagram of an aircraft entity.

The sequence diagram of an aircraft entity is illustrated in Figure 9. The aircraft sensor periodically scans for targets within range, generates raw detections for targets within range, and forwards the detections to its radar system to form tracks. Meanwhile, a scripted pilot conducts a mission.

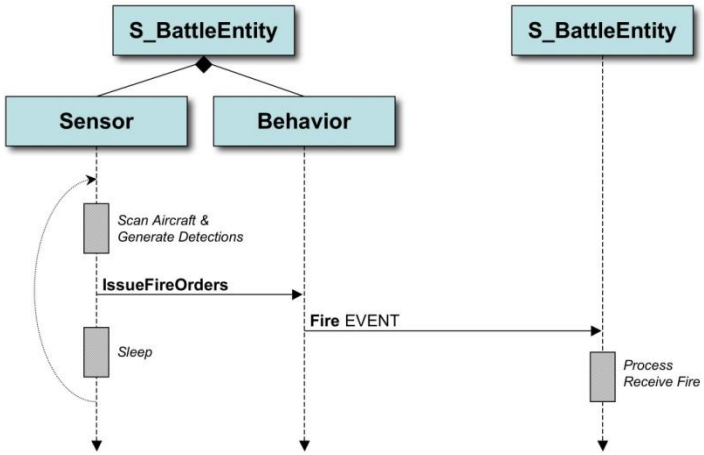

Figure 10: Generic sequence diagram of a red-force entity threat.

The sequence diagram for a generic red-force entity threat is illustrated in Figure 10. The threat contains a sensor component that periodically scans for blue-force aircraft and generates raw detections. If the aircraft is within range, the behavior component schedules an event to fire at the aircraft. The aircraft then determines the impact, if any, of the attack.

3.4 Sim Controller

The WarpIV DSAP Sim Controller provides a command-line and GUI interface into the WarpIV DSAP real-time simulation. This utility enables the user to remotely alter the state of the WarpIV COP in real-time and, in the event of re-planning, reschedule missions for aircraft in the scripted

plan simulation without having to terminate the program. Specifically, capabilities are provided to create, detonate, and set the track of an entity in the COP. In addition, capabilities are provided to create an entity, force a squadron to return to base, and schedule a new mission in the scripted simulation. Finally, options are provided to toggle between outputting DIS messages from the commander's scripted plan and the COP.

The WarpIV Kernel enables an external client to communicate with a simulation client in real-time. An external client communicates with a simulation client by sending an interaction, which is an instantaneous call of a particular method on a simulation object. Figure 11 illustrates the Sim Controller sending an interaction to a particular object within the external simulation. An external client sends an interaction to an entity in the simulation client through the WpServer.

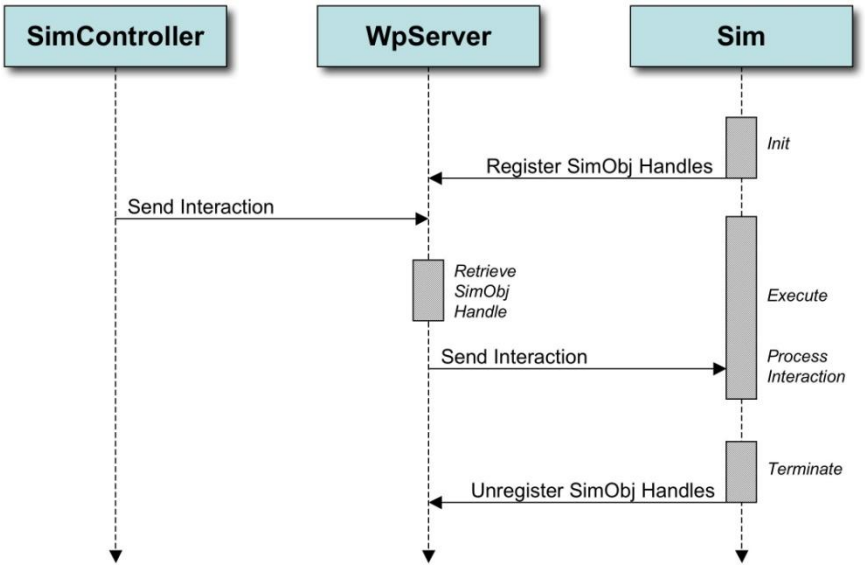

Figure 11: Sim Controller sequence diagram.

When the real-time simulation initializes, each simulation object registers its unique name and object handle with the WpServer that provides network communication services between the WarpIV simulation and the external world. Given the name of a simulation object, the Sim Controller can schedule specific interactions for specific objects in the real-time simulation. Each of the interactions listed in Figure 11 are implemented as a method within specific entities or components in the real-time simulation. When the real-time simulation terminates, each simulation object un-registers its object handle with the WpServer.

The Sim Controller command line options are listed in Table 1. The Sim Controller command-line interface provides error checking to ensure the correct number of parameters is specified for each command.

Table 1: Sim Controller command line options.

COMMAND	DESCRIPTION
DETONATE	Detonate an entity in the COP simulation.
SET_TRACK	Set the track of an entity in the COP simulation.
CREATE	Dynamically create an entity in the COP simulation.
CREATE_PLAN	Dynamically creates an entity in the scripted simulation.
RTB	Force a squadron in the scripted simulation to return to base.
SCHEDULE_MISSION	Schedule a mission for a squadron in the scripted simulation.
ENABLE_DIS_PLAN	Enable the scripted simulation to output DIS messages.
DISABLE_DIS_PLAN	Stop the scripted simulation from outputting DIS messages.
ENABLE_DIS_REALITY	Enable the COP simulation to output DIS messages.
DISABLE_DIS_REALITY	Stop the COP simulation from outputting DIS messages.
SIMOBJS	List the names of all registered SimObjs from the scripted/COP simulation.
SHUTDOWN	Shuts down the scripted/COP simulation.
?	Help command.

The main view of the Sim Controller GUI is shown in Figure 12. The GUI provides an interface for all the command line options listed in Table 1. For simplicity, the GUI provides a dynamically populated drop down list of all active simulated entities. Various buttons enable the user to perform a set of actions on a specific simulated entity. Clicking on each *Action* button brings up a new interface with additional parameters.

Figure 12: Main view of Sim Controller GUI.

3.5 Sim Analyzer

The Sim Analyzer provides a basic capability to evaluate the results of a HyperWarpSpeed predictive simulation execution of an alternative COA. Currently, the Sim Analyzer measures the

effectiveness of a particular COA by analyzing the health of each entity in the predictive simulation. Specifically, the Sim Analyzer computes the raw effectiveness, average blue-force health, and average red-force health for each replication within the HyperWarpSpeed simulation. Each metric is then summarized in a histogram to represent the final results of a HyperWarpSpeed simulation. The Sim Analyzer displays each histogram in the terminal window and posts the histograms to the Blackboard where they can be visualized in a GUI. The Blackboard is described in Section 3.9.

3.6 Sequence Analyzer

The Sequence Analyzer provides a basic capability to evaluate the results of a specific decision branch of a HyperWarpSpeed predictive simulation execution. As opposed to the Sim Analyzer, discussed in Section 3.5, which computes the overall effectiveness of a simulation, the Sequence Analyzer computes the effectiveness of a specific decision branch.

Currently, the Sequence Analyzer measures the effectiveness of a specific decision branch by analyzing the health of each entity in the predictive simulation. Specifically, the Sequence Analyzer computes the raw effectiveness, average blue-force health, and average red-force health for each replication within the HyperWarpSpeed simulation. Each metric is then displayed in a histogram to summarize the results of a HyperWarpSpeed simulation. The Sequence Analyzer displays each histogram in the terminal window and posts the histograms to the Blackboard where they can be visualized in a GUI.

3.7 PDU Interceptor

The WarpIV PDU Interceptor allows JSAF or any other DIS-compliant simulator to send state information into the WarpIV real-time COP in the form of DIS PDUs. This capability, along with the Sim Controller and WarpIV TBMCS simulation, offers a third alternative for feeding state updates into the WarpIV COP. Currently, the WarpIV COP responds to updates in the form of either entity tracks or entity health status. Entity tracks are in the form of EntityState PDUs, which contain position, velocity, and acceleration information for a particular entity. Health status is in the form of a Detonation PDU.

In order to provide a clean interface, the WarpIV COP does not directly receive DIS PDUs. The WarpIV COP has a well-defined interface for updating the track or health status of an entity. The PDU Interceptor extracts, interprets, and converts data from incoming PDUs and forwards updates to the WarpIV COP. The WarpIV COP processes the updates as externally generated events.

Entities in the WarpIV COP are mapped to entities in the DIS-compliant simulator by the EntityState PDU Marking field. Thus, for a pre-existing entity in the WarpIV COP to receive a state update from an external simulator, the 11-character Marking field in the EntityState PDU must match the name of the Simulation Object (SimObj) in the WarpIV COP. When initializing the WarpIV real-time simulation, every entity identified in the ATO is represented in both the commander's scripted plan and WarpIV COP. Markings not matching the name of any pre-existing entity within the WarpIV COP are regarded as new battlespace entities. These entities are dynamically created within the WarpIV COP, and registered with the WpServer using the

Marking as the SimObj name. The PDU Interceptor determines the type of entity based on the EntityType record within the EntityState PDU. Predefined text files define the composition for each type of entity that is dynamically created within the simulation. For consistency with STOMP, the composition files define the structure of Airbase, Aircraft, Mobile, Person, Radar, SAM, and Structure entities.

3.8 STOMP Translator

The STOMP Translator is required to provide interoperability between STOMP and WarpIV Kernel simulations. The STOMP Translator performs three basic functions: (1) convert from STOMP Scenario and AODB XML input files to a WarpIV real-time composition file, (2) convert from STOMP Scenario, AODB, and Retask XML input files to a WarpIV predictive composition file, and (3) convert the results of a WarpIV real-time simulation to STOMP Scenario and AODB XML input files. Both a command line and graphical user interface to the STOMP Translator was developed.

Rather than integrate a cumbersome third-party XML parser, such as XERCES, a non-validating XML parser was developed in C++ from scratch (in two days) for this task. This XML parser is efficient, easy to use, and has shown to be extremely robust.

Figure 13 illustrates one of several views of the STOMP Translator GUI. Tooltips provide an explanation of each parameter and button. The locations of input files are specified using a file chooser interface.

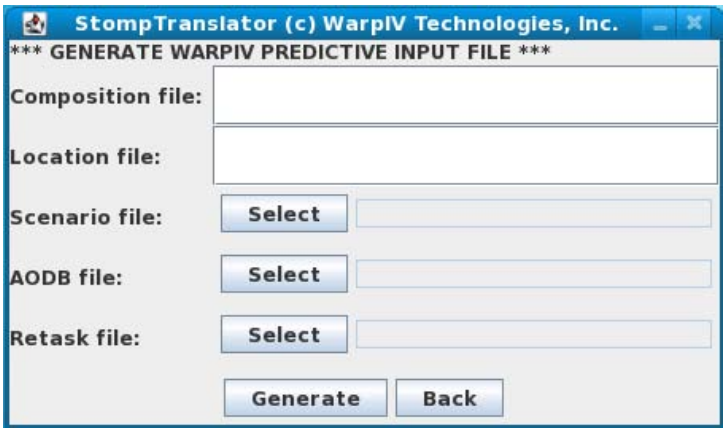

Figure 13: One of several views of the STOMP Translator GUI.

STOMP executes on Windows platforms, whereas WarpIV executes on UNIX and Windows platforms. For performance reasons, UNIX is the preferred platform for running the WarpIV Kernel [4]. The format of Windows and UNIX text files differ slightly. Line returns in Windows text files contain a line feed and carriage return, whereas UNIX text files only contain a line feed. To ensure cross-platform interoperability, a C++ utility was developed to convert between Windows/UNIX output files.

Integration of the STOMP optimization tool is discussed in Section 3.10.

3.9 Blackboard

The Blackboard displays and ranks the effectiveness of HyperWarpSpeed simulations. The Sim Analyzer and Sequence Analyzer programs, discussed in Section 3.5 and Section 3.6 respectively, compute and automatically post the effectiveness of a HyperWarpSpeed simulation to the Blackboard. Specifically, these programs compute the raw effectiveness, average blue-force health, and average red-force health for each replication within a HyperWarpSpeed simulation. Each of these metrics are collapsed into a histogram to summarize the results, and automatically posted to the Blackboard where they can be visualized.

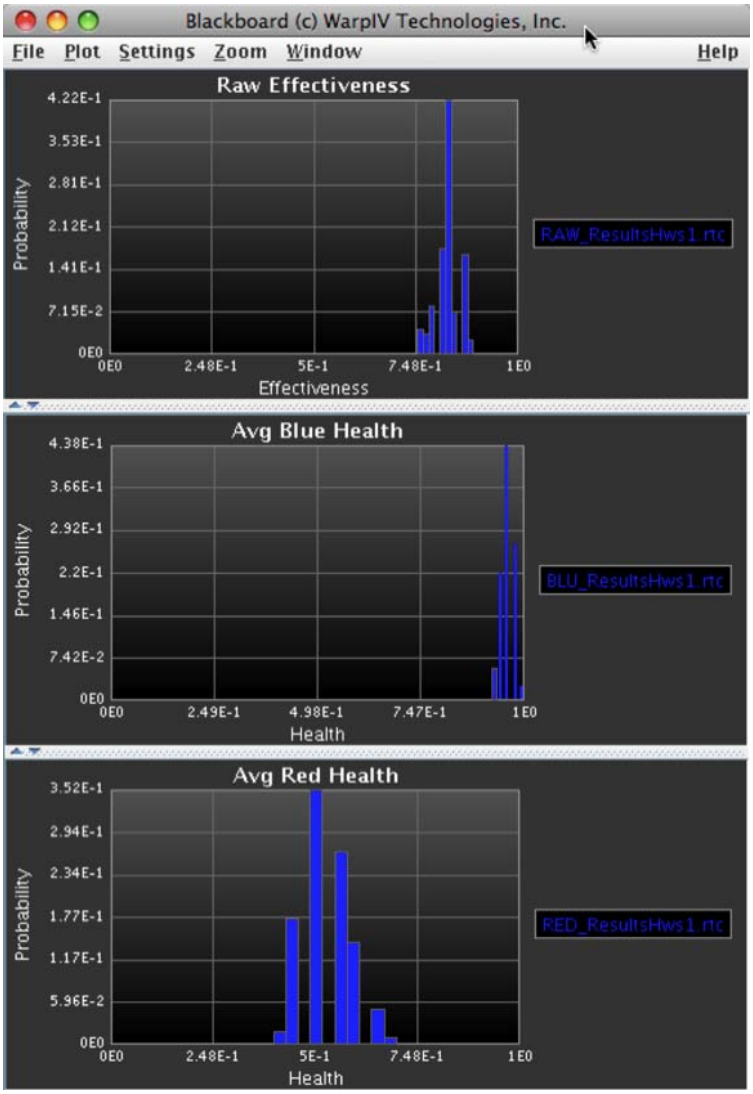

Figure 14: Blackboard screenshot.

A screen capture of the Blackboard is shown in Figure 14. Three unique graphs are incorporated to display the (1) raw effectiveness, (2) average blue-force health, and (3) average red-force health of each HyperWarpSpeed execution. Although not shown in the screenshot, the Blackboard is able to plot and rank the effectiveness of multiple HyperWarpSpeed executions. Multiple plots overlaid on top of one another can be difficult to visualize and interpret. However,

each execution is assigned a different color to distinguish one from another. To filter the data, users can specify which executions to display in the graphs. Users also have control over the order in which histograms are painted to the screen. Finally, users have control over transparency levels in order to uncover any histograms buried in the background.

3.10 DSAP Framework

The DSAP framework was designed to automate the execution of all simulation, translation, and analysis tasks. The framework is composed of various tasker and worker components that request and perform jobs, in addition to a manger that coordinates and relays all communication between components.

All DSAP framework components are illustrated in Figure 15. Tasker components are shown on the left, and worker components are shown on the right. Separate workers exist to conduct the real-time and predictive simulation tasks, thereby allowing these unique jobs to be executed on different machines. Multiple predictive simulation workers can be instantiated to conduct the predictive simulation tasks in parallel. A translation worker conducts all STOMP/WarpIV file translation tasks, and an analysis worker conducts analysis of the HyperWarpSpeed predictive simulation results.

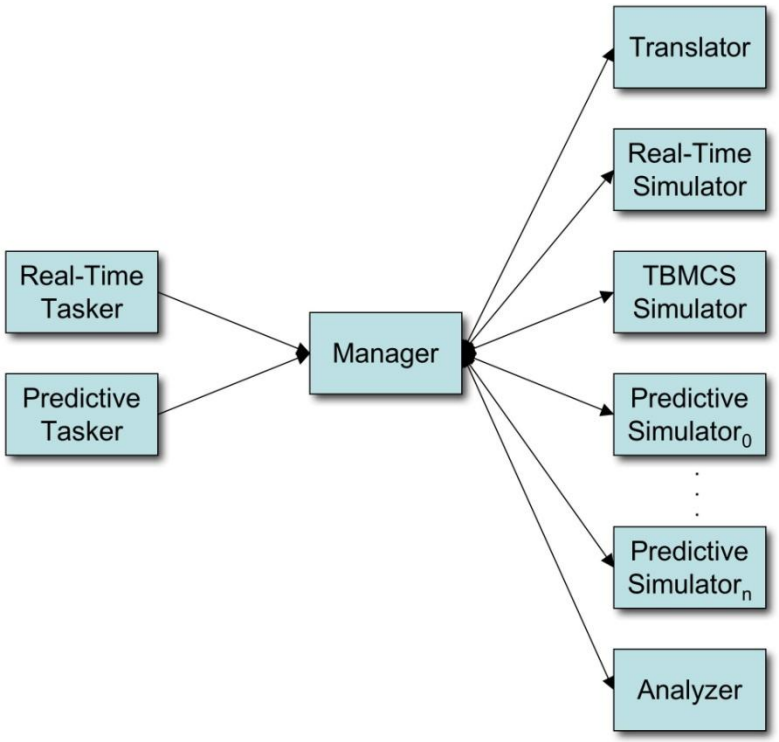

Figure 15: DSAP framework components.

Missing from the DSAP framework is a worker component to automatically launch, run, and return STOMP-generated alternative courses of action. STOMP introduces a discontinuity in the framework in that it cannot be driven from the command line and requires interaction with a GUI. Charles River Analytics has considered modifying STOMP such that it could be driven from the command line, but that capability does not currently exist. Currently, STOMP is

manually launched and operated. However, the goal is to automatically launch, run (based on preset or default parameters), and return the STOMP-generated alternative courses of action to the Manager component when this task completes. The WarpIV Kernel executes on both UNIX and Windows platforms in a distributed manner, so communication with STOMP on a networked Windows machine is straightforward. WarpIV Technologies, Inc. has demonstrated scalable execution on heterogeneous clusters and distributed heterogeneous machines.

Figure 16 illustrates the real-time processing sequence for the DSAP framework. The *Tasker* component submits an ATO from TBMCS to the *Manager* component. The ATO will be translated into an input file that defines a scenario that can be executed within a simulation. The *Translation Worker* connects to the *Manager* and requests an ATO for translation. The *Manager* submits an ATO to the *Translation Worker*, which then translates the ATO. The translated ATO is issued and executed within the *ATO Simulation Worker* and *TBMCS Simulation Worker* components that have requested work from the *Manager*.

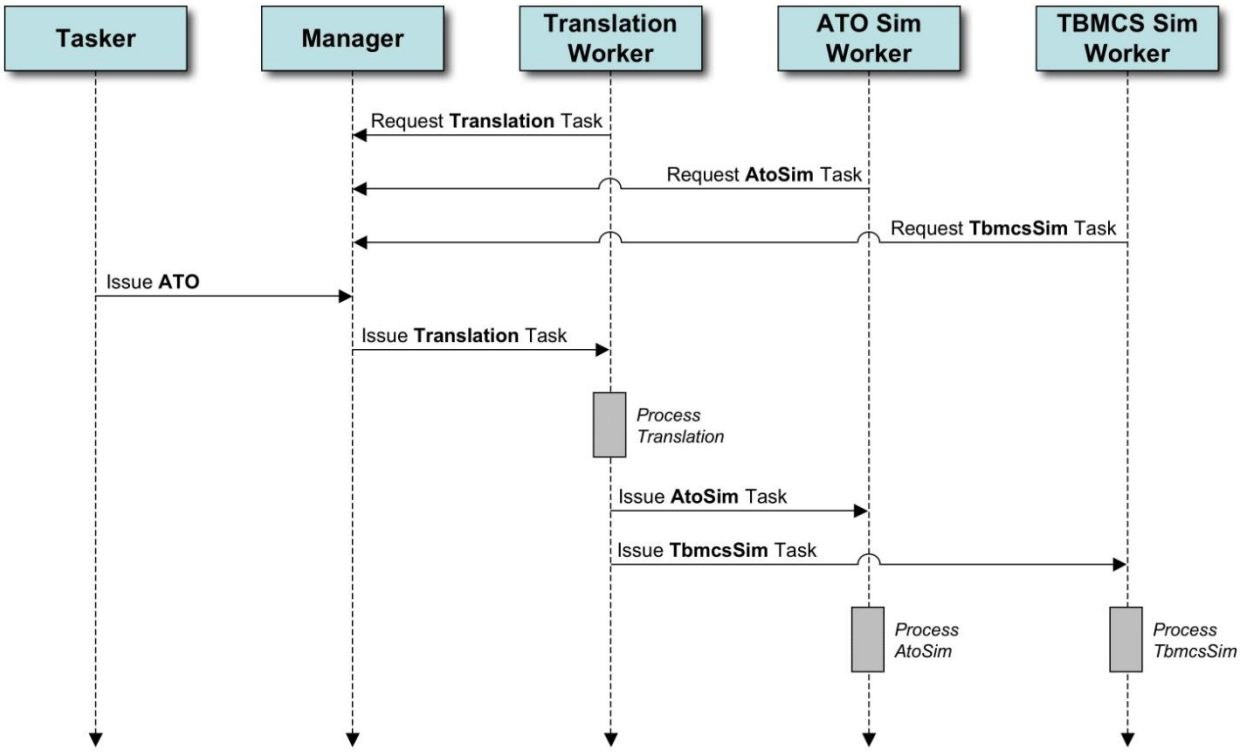

Figure 16: DSAP framework real-time processing sequence.

Figure 17 illustrates the predictive simulation processing sequence for the DSAP framework. The *Tasker* component submits alternative Courses of Action (COAs) from STOMP to the *Manager* component. The COAs will be translated into an input file that defines a scenario that can be executed within a simulation. The *Translation Worker* connects to the *Manager* and requests a COA for translation. The *Manager* submits a COA to the *Translation Worker*, which then translates the COA. The translated COA is issued and executed within the *Predictive Simulation Worker* that has requested work from the *Manager*. The results of the predictive simulation are passed to an *Analysis Worker* that analyzes the results of the predictive simulation.

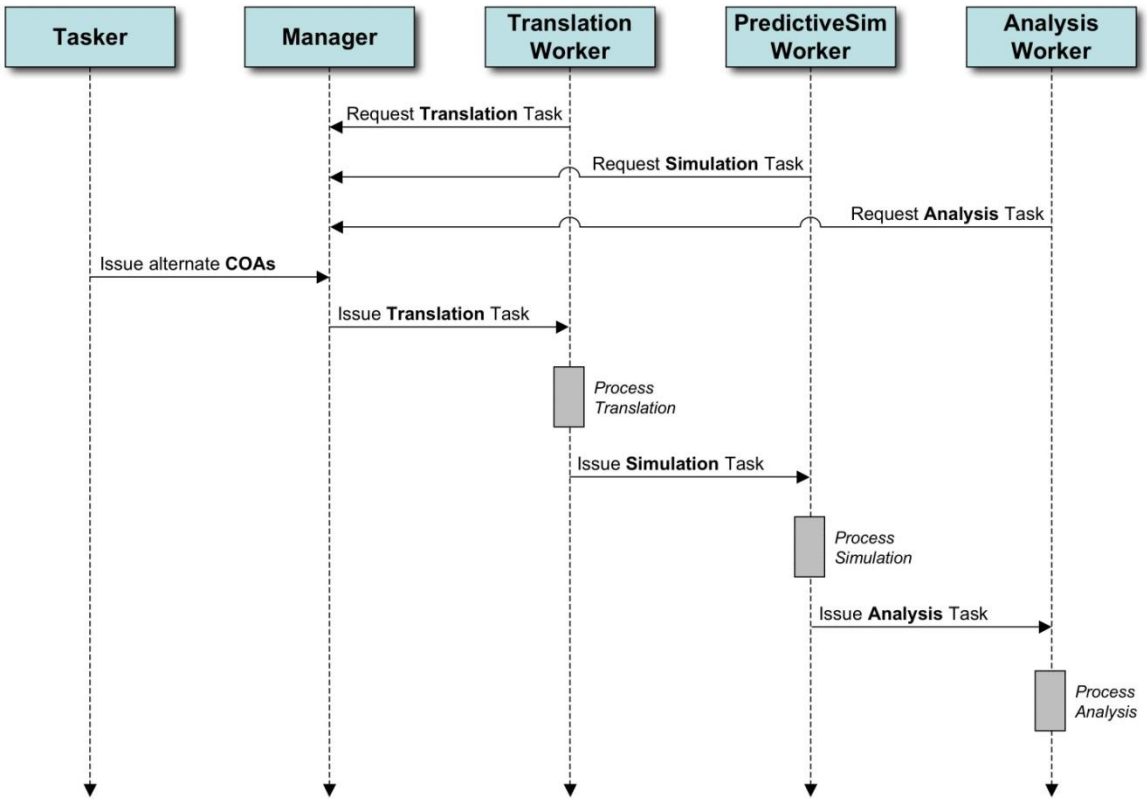

Figure 17: DSAP framework predictive simulation processing sequence.

3.11 Visualization

JSAF was initially used to provide a visualization capability. The WarpIV Viewer was leveraged to provide an alternative visualization capability. The visualization capability, developed from scratch using OpenGL and C++, provides an efficient, cross-platform, flexible, and interactive 3D Earth environment for viewing the movement and engagement of objects. This capability was developed on another effort and will eventually be extended to provide a 5D capability for viewing multiple timelines within a HyperWarpSpeed execution.

The WarpIV Viewer contains a trackball feature that enables the user to orient the rotation of the Earth using a mouse. Additional features enable the user to adjust the zoom and change the viewing perspective. Multi-resolution satellite imagery was integrated to provide more detail as the zoom level increases. Bill boarding was implemented to ensure 2D objects always face the viewer, regardless of the viewing perspective. Fading trajectories can be displayed for moving objects, and detections from sensors can be drawn. A star field is also displayed in the distance around the Earth. Currently, the prototype 3D viewer provides basic support for DIS and responds to EntityState, Fire, and Detonation PDUs.

The prototype OpenGL/C++ based viewer is shown in Figure 18. The image on the left zooms in on Iraq and displays the expected and actual location of aircraft entities. Lines, varying in width based on range, show the distance between the expected and actual locations. The image on the right zooms in on the Caribbean and displays the trajectory of an aircraft in the background.

Figure 18: WarpIV Viewer.

3.12 Results and Discussion

Two unique simulations were developed using the WarpIV Kernel to conduct performance benchmarks on a wide variety of machines. Whereas the smaller fixed-size ATO scenarios were used to demonstrate proof-of-concept of the enabling technology, these two unique simulations can easily be reconfigured to conduct performance benchmarks with extremely large entity counts. The following subsections discuss the design of the simulations.

3.12.1 Lanchester5D Simulation

A Lanchester simulation was developed to conduct performance benchmarks using the WarpIV Kernel and HyperWarpSpeed. The Lanchester simulation contains a user-definable number of grid cells in which blue and red forces engage in battle using Lanchester equations. Having a user-definable number of grid cells enables the simulation to be stressed with high entity counts. The simulation is configurable at run-time by modifying a parameterized input file. The input file specifies the number of grid cells, maximum number of blue and red-force troops per grid cell, and blue and red-force attrition level. The actual number of troops varies by grid cell and is randomly determined at initialization.

The simulation randomly disperses blue and red forces among grid cells. In each grid cell, blue and red forces battle to the end. When the battle finishes, the simulation branches by allowing the victor to move up, down, left, and right (if possible) to a neighboring grid cell. Each grid cell is represented as a SimObj, so parallelism is achieved by distributing grid cells to processors.

3.12.2 NASM Simulation

A low-fidelity representation of the National Air Space Model (NASM) was developed using the WarpIV Kernel to conduct performance benchmarks on a wide variety of machines. The model contains *2,036* airports, *2,664* air sectors, and *14,875* domestic flights represented over a 24-hour simulated period. Data used for the simulation is freely available for download from the Bureau of Transportation Statistics website (www.bts.gov).

The scenario is defined in a Composition.rtc input file. On the average, sectors are 60 nautical miles in diameter, and allow a maximum occupancy of 15 aircraft at a time. Flight state information is contained within an event message. The event message contains the (1) code and location of the origin and destination airport, (2) planned departure and arrival time, (3) flight path including sector enter/exit times, and (4) flight delays encountered along the way. Sector enter/exit times are updated as delays are encountered due to sector backlogs.

A sequence diagram is illustrated in Figure 19. The simulation consists of Airport, Aircraft, and Sector SimObjs. Airport SimObjs contain flight details regarding their domestic departures and arrivals. Aircraft SimObjs determine the flight path by computing which sectors to fly through en-route to their destination. Sector SimObjs manage a region of airspace by determining when aircraft are allowed to enter and occupy it. Flight state information is contained within an event message that persists throughout the duration of a flight.

Airports schedule an *AircraftDeparture* event for each domestic aircraft flight. This event initializes the aircraft event message and pre-determines the flight path required by the aircraft to reach its destination. The flight path is determined based on Great Circle motion. Afterwards, the aircraft schedules an event for the first sector it requests to enter. The sector contains a process loop that (1) waits until the capacity permits the aircraft to safely enter the airspace, (2) updates flight state information if delays have occurred due to waiting, (3) waits for the aircraft to traverse the sector, and (4) schedules an event for the aircraft to enter the next sector.

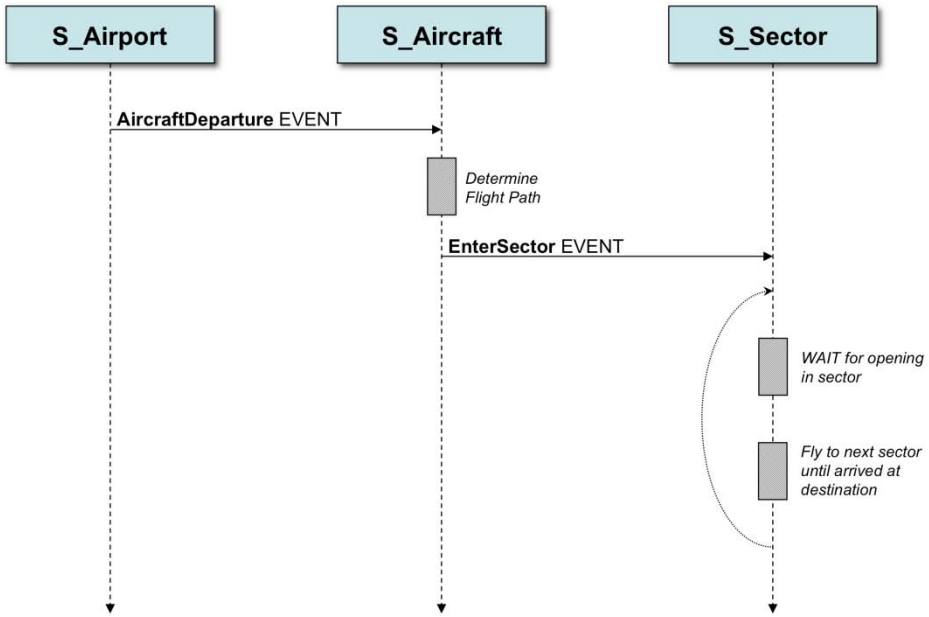

Figure 19: Sequence diagram for NASM simulation.

A visualization utility, shown in Figure 20 and Figure 21, was developed to verify and validate the simulation. The simple time-stepped visualization utility plots the 24-hour aircraft traffic represented in the WarpIV NASM simulation using the Google Maps API V3 and JavaScript. A program was developed that automatically generates an HTML page from a Composition.rtc file that can be viewed in any web browser with Internet connectivity. The webpage displays the

movement of all flights, approximate flight path of all flights, and highlights sectors that become congested over time.

Users have complete access to Google Map controls including zoom, pan, and map type. Start, stop, pause, and resume buttons are provided to control the simulation. Drop-down menus enable the user to dynamically modify the simulation rate and frame rate (number of frames per second). Check boxes enable the user to dynamically toggle between animating aircraft movement, flight paths, or congested sectors. Text areas display the number of active flights and time with respect to Eastern Standard Time (EST). Disabling the animation drastically improves the performance of the simulation. Flight paths have a transparency control that can be increased or decreased to improve clarity during times of congestion. Eastbound flights are shown as red circles, and westbound flights are shown as blue circles. The visualization helps understand (1) flight patterns, and (2) areas of congestion that introduce bottlenecks. In addition, the visualization gives insight into how the (1) routing and (2) parallelization of the simulation could be improved.

Due to the massive number of Google Maps markers that are rendered in the simulation, the visualization utility is best viewed using a web browser with an efficient JavaScript implementation, such as Safari, Firefox, or Google Chrome. Internet Explorer is not recommended.

Figure 20 contains a screen capture of the NASM simulation visualization utility. Eastbound flights are shown as red circles, and westbound flights as blue circles. A transparent line displays the approximate flight path for each active flight. Yellow rectangles, visible in the background, illustrate sectors that are congested. As of 8:14 AM EST, congestion is observed in the sectors occupied by Atlanta, New York City, and Charlotte. For more clarity observing sectors experiencing congestion, the flight path and aircraft animation can be disabled.

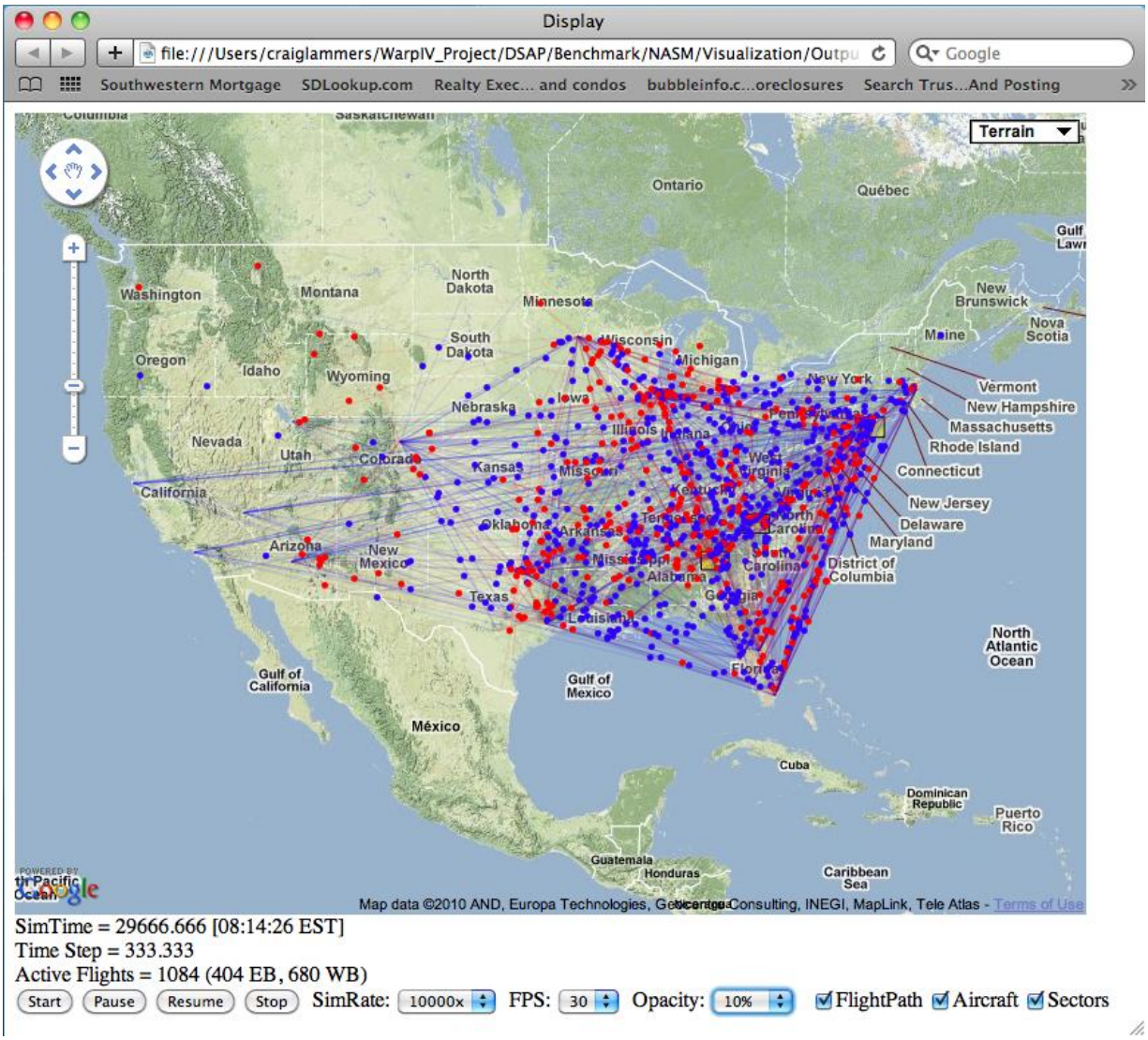

Figure 20: Screen capture of NASM simulation visualization utility.

Figure 21 better illustrates the high-traffic areas of the simulation by disabling the flight path and aircraft animation. The flight path and aircraft animation were disabled to focus on and highlight high-traffic sectors. The simulation time was 9:57 AM EST.

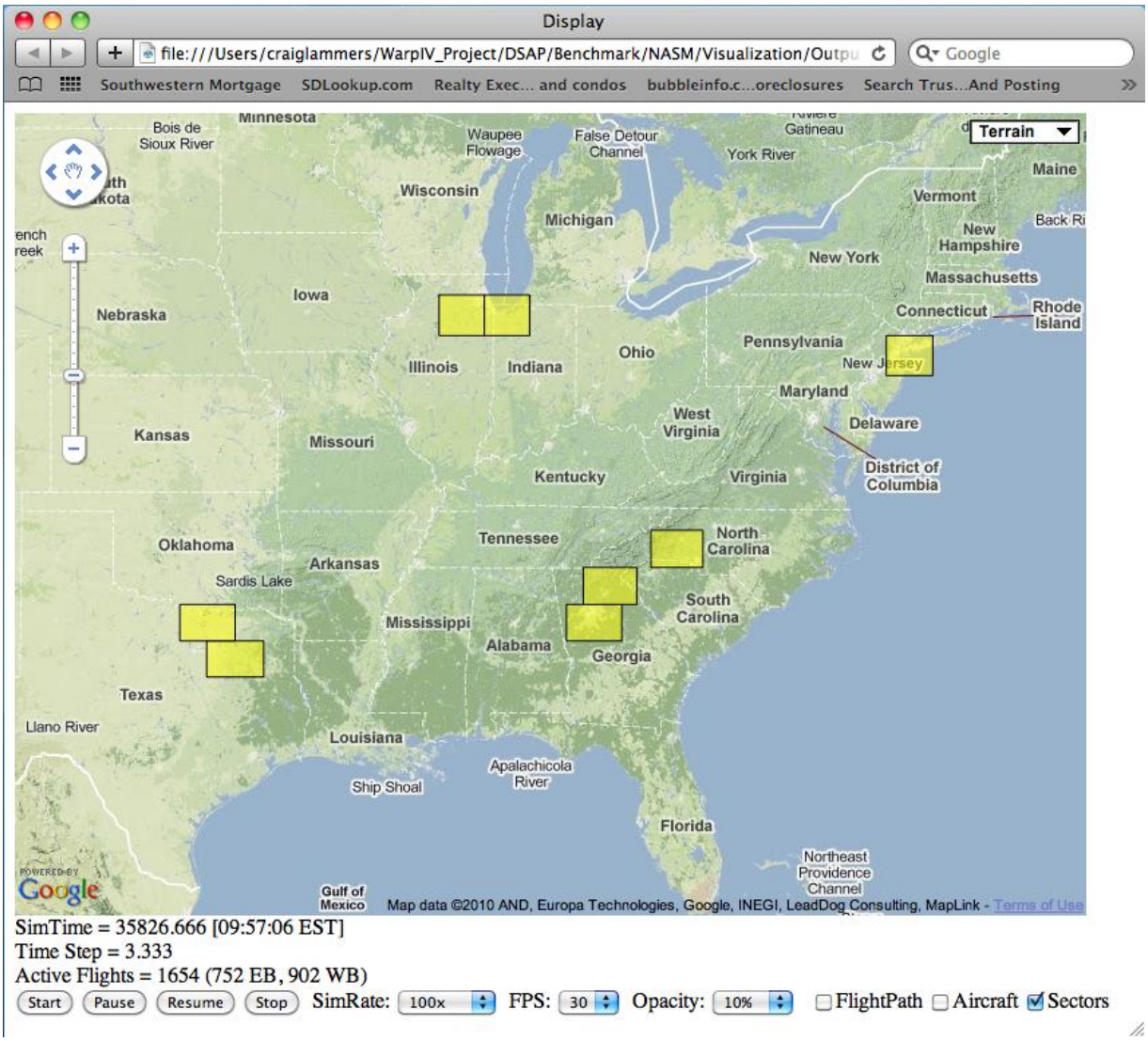

Figure 21: Congested sectors within the NASM simulation.

4 HyperWarpSpeed Validation

This effort conducted a validation of HyperWarpSpeed. Two simulations, *BranchAirMission* and *Lanchester5D*, were independently validated to ensure the results of 128 Monte Carlo replications were equivalent to that of a single HyperWarpSpeed execution. When toggling between executing Monte Carlo and HyperWarpSpeed runs, the source code was recompiled to use the correct type of state variables. The Monte Carlo runs used rollbackable state variables, and the HyperWarpSpeed run used rollbackable multi-replication state variables.

4.1 Branch Air Mission

The *BranchAirMission* simulation contains 1,000 aircraft that bomb 1,000 ground targets in sequence. Each aircraft flies in a single-file formation over the same sequence of targets and drops a bomb on the target if it had not already been destroyed. If the dropped bomb does not destroy the target, a surface to air missile is fired back at the aircraft. A 128-replication set was used in the HyperWarpSpeed algorithm to support this test. In this scenario, branching occurs at two places: (1) to determine if the bomb destroyed the target, and (2) to determine if the surface-to-air missile destroyed the aircraft. In both branching cases, a probability of 0.5 was used to determine the effect of (1) the bomb on the target, and (2) the missile on the aircraft.

Each aircraft and target within the simulation contains a Boolean state variable representing health status. 128 Monte Carlo replications were executed using a unique random number seed, and the results were combined into a histogram. A single HyperWarpSpeed simulation was executed, and the result was fit into a histogram. The histogram contained 10 bins, where each bin represented 100 SimObjs. These histograms are shown in Figure 22. Visually, there does not appear to be a significant difference between the results of 128 Monte Carlo replications and a single HyperWarpSpeed execution.

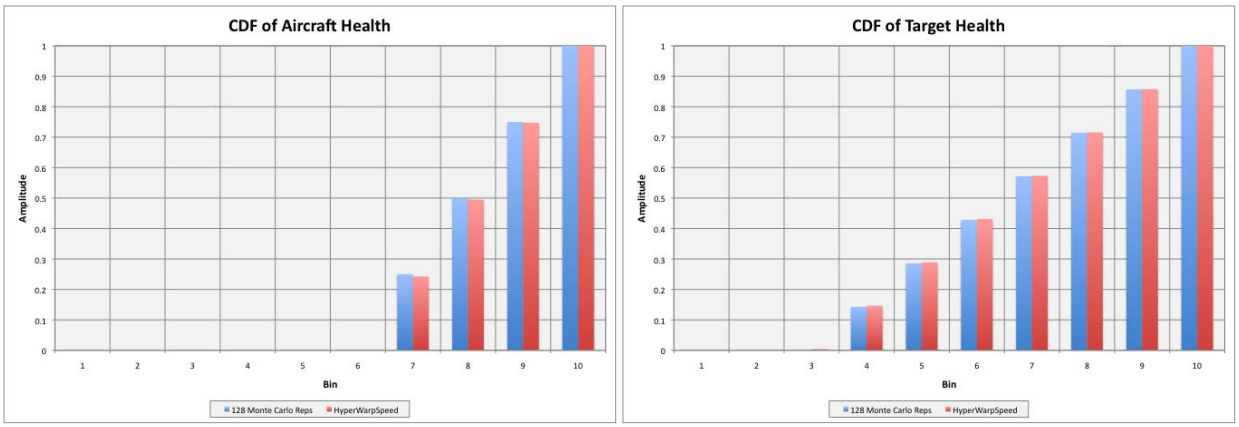

Figure 22: Cumulative Distribution Function of Aircraft and Target Health.

The mean and standard deviation of the data in Figure 22 is given in Table 2. Blue bars represent the normalized health for 128 Monte Carlo replications, and red bars represent the normalized health for a single HyperWarpSpeed execution. In each case, the difference between the mean and standard deviation of the state variables between 128 Monte Carlo replications and single HyperWarpSpeed execution is less than one percent.

Table 2: Mean and Standard Deviation of BranchAirMission simulation state variables.

Metric	128 Monte Carlo Reps	HyperWarpSpeed	Difference
Aircraft Health Mean	799.2	800.678	0.2%
Aircraft Health StdDev	111.692	111.24	0.4%
Target Health Mean	649.35	647.574	0.3%
Target Health StdDev	199.8	201.12	0.7%

The independent two-sample *t*-test was performed to statistically determine whether the means of the two distributions were equivalent. This test assumes the two sample sizes are equal, and the two distributions have the same variance. The *t*-test computed a 99.3% probability that the distribution representing aircraft health for 128 Monte Carlo replications and one HyperWarpSpeed execution have the same mean. In addition, the *t*-test computed a 99.2% probability that the distribution representing target health for 128 Monte Carlo replications and one HyperWarpSpeed execution have the same mean. Statistically, there does not appear to be a significant difference between the results of 128 Monte Carlo replications and a single HyperWarpSpeed execution.

4.2 Lanchester5D

The *Lanchester5D* simulation was described in Section 3.12.1.

Each grid cell within the simulation contains an integer state variable representing the number of healthy blue and red-force entities. The simulation contained a total of 25 grid cells, from which the average number of blue and red forces per cell was computed. 128 Monte Carlo replications were executed using a unique random number seed, and the results were combined into a histogram. A single HyperWarpSpeed simulation was executed, and the result was fit into a histogram. The histogram contained 4 bins, where each bin represented the average number of healthy entities per cell. These histograms are shown in Figure 23. Visually, there does not appear to be a significant difference between the results of 128 Monte Carlo replications and a single HyperWarpSpeed execution.

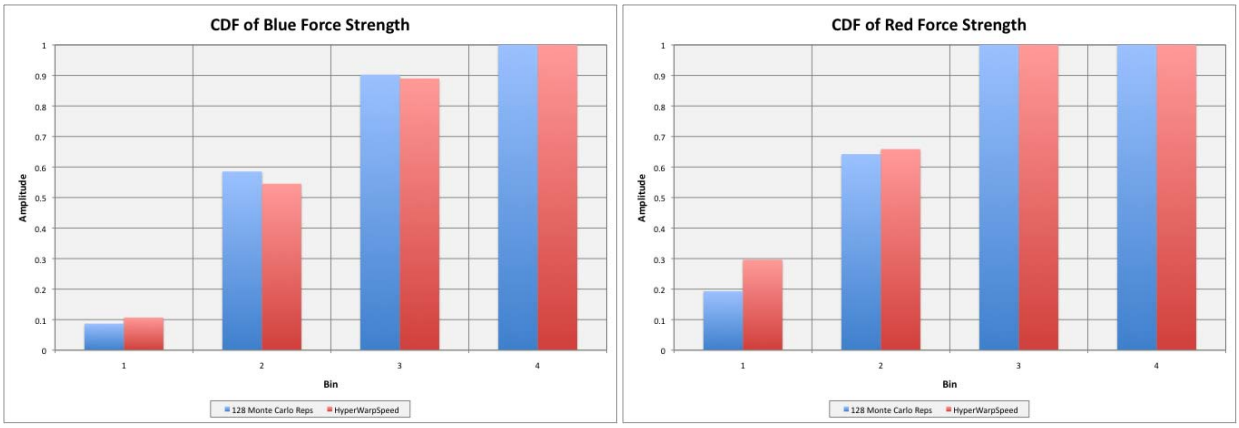

Figure 23: Cumulative Distribution Function of Blue and Red-Force Strength.

The mean and standard deviation of the data in Figure 23 is given in Table 3. Blue bars represent the normalized strength for 128 Monte Carlo replications, and red bars represent the normalized strength for a single HyperWarpSpeed execution. In each case, the difference between the mean

and standard deviation of the state variables between 128 Monte Carlo replications and single HyperWarpSpeed execution is less than ten percent.

Table 3: Mean and Standard Deviation of Lanchester5D simulation state variables.

Metric	128 Monte Carlo Reps	HyperWarpSpeed	Difference
Blue Strength Mean	1.925	1.958	1.7%
Blue Strength StdDev	0.783	0.826	5.5%
Red Strength Mean	1.665	1.545	7.2%
Red Strength StdDev	0.724	0.797	10.1%

The independent two-sample *t*-test was performed to statistically determine whether the means of the two distributions were equivalent. This test assumes the two sample sizes are equal, and the two distributions have the same variance. The *t*-test computed a 97.8% probability that the distribution representing blue strength for 128 Monte Carlo replications and one HyperWarpSpeed execution have the same mean. In addition, the *t*-test computed a 91.0% probability that the distribution representing red strength for 128 Monte Carlo replications and one HyperWarpSpeed execution have the same mean. Statistically, there does not appear to be a significant difference between the results of 128 Monte Carlo replications and a single HyperWarpSpeed execution.

5 Conclusions

This effort conducted large-scale performance benchmarks of the enabling technology provided by the WarpIV Kernel and HyperWarpSpeed on a wide variety of machines ranging from common desktops to supercomputers and compute clusters. The Air Force Research Laboratory in Rome, NY independently conducted benchmarks on several supercomputers and compute clusters. In most cases, at least eight independent runs were averaged for each test configuration.

In summary, the WarpIV Kernel and HyperWarpSpeed demonstrated scalability on tests using up to ***4-million entities*** and ***700 processors***. These specific results have not been tabulated as of the time of this report [5]. In particular, the SGI Altix 4700 *(Hawk)* at Wright Patterson Air Force Base demonstrated a speedup factor of 60X on 100 processors.

5.1 Test System Configurations

Hardware configurations for the systems utilized are described in Table 4.

Table 4: Test System Configurations.

MACHINE	CORES	PROCESSOR	RAM	OPERATING SYSTEM
IBM p690	32	1.3 GHz Power4	32 GB	SUSE Linux Enterprise Server 10
HP Superdome	48	550 MHz PA8600	48 GB	HP-UX 11i
SGI Altix 4700	9,216	1.6 GHz Itanium 2	2 or 4 GB/core	SUSE Linux Enterprise Server 10
Dell PowerEdge M610	9,216	2.8 GHz Intel Nehalem	3 GB/core	Linux
Phantom Desktop	8	3.2 GHz Intel Xeon	32 GB	Linux
Dell Inspiron 530	4	2.4 GHz Intel Core 2	4 GB	Fedora Core 9

The IBM p690 supercomputer, codenamed *Bengal*, contains 32 1.3-GHz Power4 cores. The HP Superdome supercomputer, codenamed *Hercules*, contains 48 550-MHz PA8600 cores. Bengal and Hercules are both located at SPAWAR Systems Center Pacific.

The SGI Altix 4700 modular blade design, codenamed *Hawk*, contains 9,216 cores. The system contains 18 blade systems, where each system contains 256 Itanium2 dual-core processors. The SGI Altix 4700 implements a global shared memory space. This provides each core with access to the entire shared-memory space, thereby improving read/write time and scalability over traditional blade architectures. Jobs are submitted to the machine using the PBS batch scheduling system. Hawk is located at Wright Patterson Air Force Base. [6]

The Dell PowerEdge M610 cluster, codenamed *Mana*, contains 9,216 cores. The system contains 1,152 M610 blades, where each blade contains two Intel Nehalem quad-core processors. The system is interconnected using Dual Data Rate Infiniband. Mana is located at the Maui High Performance Computing Center. [7]

The Phantom and Coyote clusters are located at the Air Force Research Laboratory in Rome, NY. Specifications for these clusters were not provided.

The Phantom and Dell Inspiron 530 are commonplace and affordable multicore desktop computers. The Phantom contains 8 nodes, whereas the Dell Inspiron 530 contains 4 nodes. Phantom is located at the Air Force Research Laboratory, Rome NY, and the Dell Inspiron 530 is located in the offices of WarpIV Technologies, Inc.

5.2 Decomposition Strategies

Parallelism is achieved in the WarpIV Kernel by distributing SimObjs to processors in order to distribute the workload. Parallelism is achieved at the SimObj level. There are several methods for distributing SimObjs to processors. SCATTER, BLOCK, and 2D_GRID decomposition strategies were utilized during benchmarking to assess the relative strength of each approach.

SCATTER decomposition card deals one object at a time to each processor, looping through processors if necessary. BLOCK decomposition deals an equivalent fraction of objects to processors in a consecutive manner in a single pass. 2D_GRID decomposition divides objects into rectangular grid cells and disperses them to processors. These decomposition strategies are illustrated in Figure 24. This example maps 16 SimObjs to four processors using each decomposition strategy. The processors are color coded for clarity.

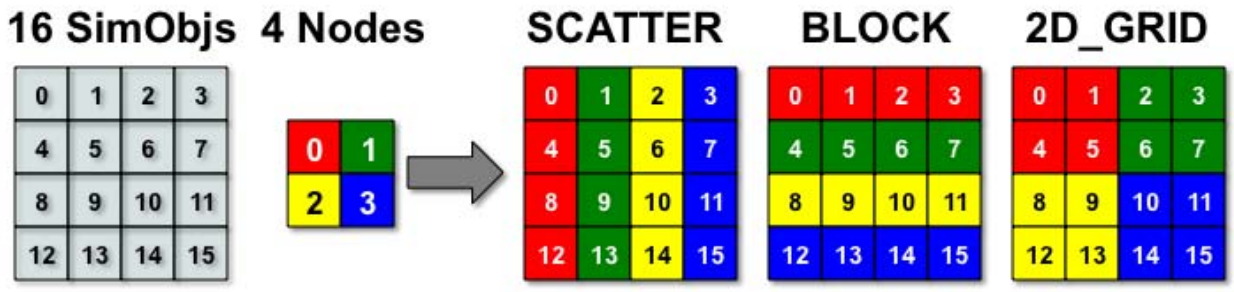

Figure 24: Object decomposition strategies.

The goal is not to eliminate rollbacks in an optimistic simulation, as this can effectively serialize a parallel simulation. Rather, the goal is to balance the workload between processors while minimizing communications. In general, there is no "best" decomposition strategy. The best decomposition strategy is entirely dependent on the simulation at hand.

While parallelism was achieved at the SimObj level for these benchmarks, it is important to note that the WarpIV Kernel was recently extended to provide a capability that can parallelize and distribute the workload of a single simulation object. This parallelism is in addition to the parallelism achieved by distributing objects to processors. This capability can help reduce bottlenecks introduced by CPU-intensive SimObjs, such as command centers that process and fuse detections from sensors.

5.3 NASM Benchmarks

The NASM simulation was previously described in Section 3.12.2. Figure 25 illustrates NASM speedup on the common and affordable Phantom and Dell Inspiron 530 multicore desktop machines using SCATTER decomposition. The Dell Inspiron 530 is a dual-boot system containing both Linux and Windows XP operating systems. Scalability with smooth linear

speedup was shown on all desktop configurations. An approximate speedup factor of 3X was achieved on all systems when using four processors. Identical speedup was achieved on Linux and Windows, although the execution time on Linux was faster (not shown here). Improper load balancing had an effect on the scalability beyond four nodes.

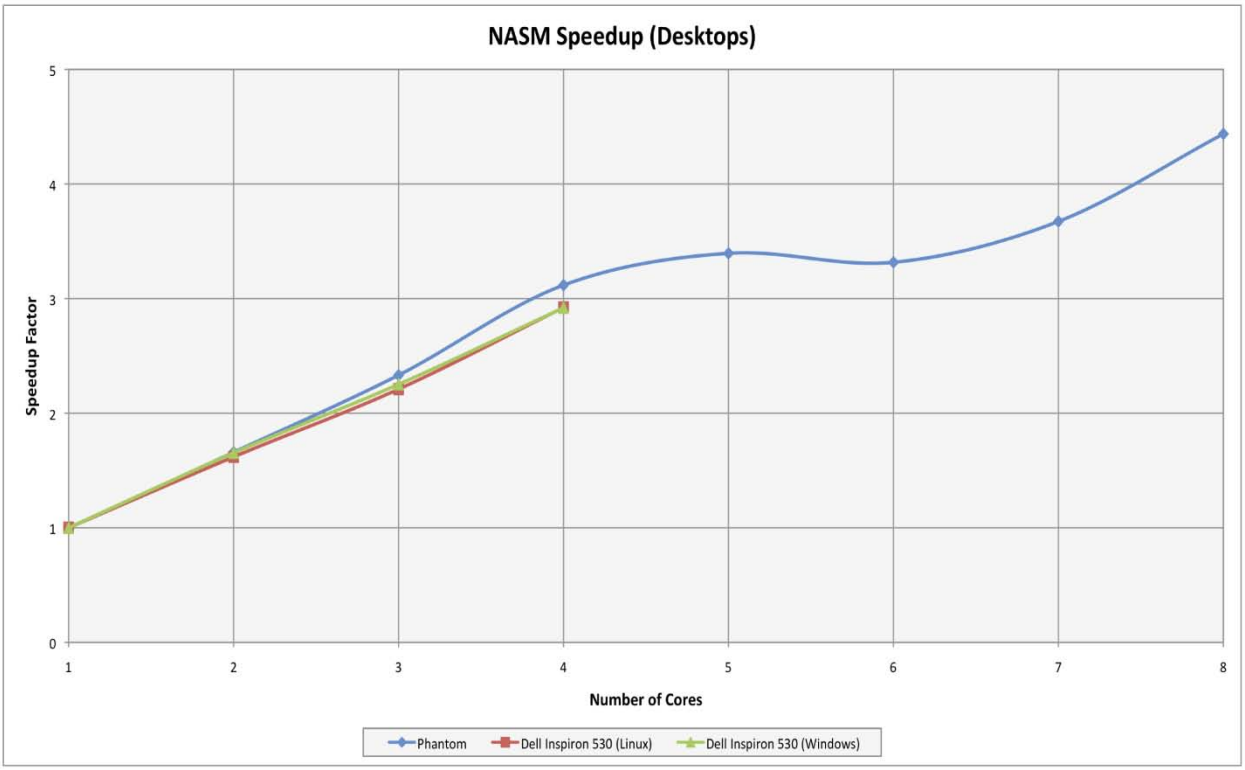

Figure 25: NASM speedup on common multicore desktops.

Figure 26 illustrates NASM speedup on the HP Superdome using SCATTER, BLOCK, and OPTIMIZED decomposition strategies. The OPTIMIZED decomposition strategy is similar to the 2D_GRID strategy, and was developed prior to its existence. The goal of the OPTIMIZED decomposition strategy was to optimize the placement of SimObjs to processing nodes specifically for the NASM benchmark simulation. In the NASM simulation, some airports receive significantly more traffic than others. This creates uneven load balancing when objects are distributed to multiple processors. As a result, benchmarks using SCATTER decomposition have shown fluctuating sin-wave like behavior instead of a smooth increase in the simulation speedup. This algorithm attempts to balance the expected number of events across processors by breaking up the continental United Status into a set of rectangular regions equal to the number of processing nodes. Rectangular regions help to keep computations local as neighboring sectors are bundled on the same processor. The boundaries of the regions (longitude bands) are determined using a genetic algorithm that attempts to balance the expected number of events in each region. A mapping within the simulation provides an efficient lookup to the Simulation Object ID when scheduling events for specific objects.

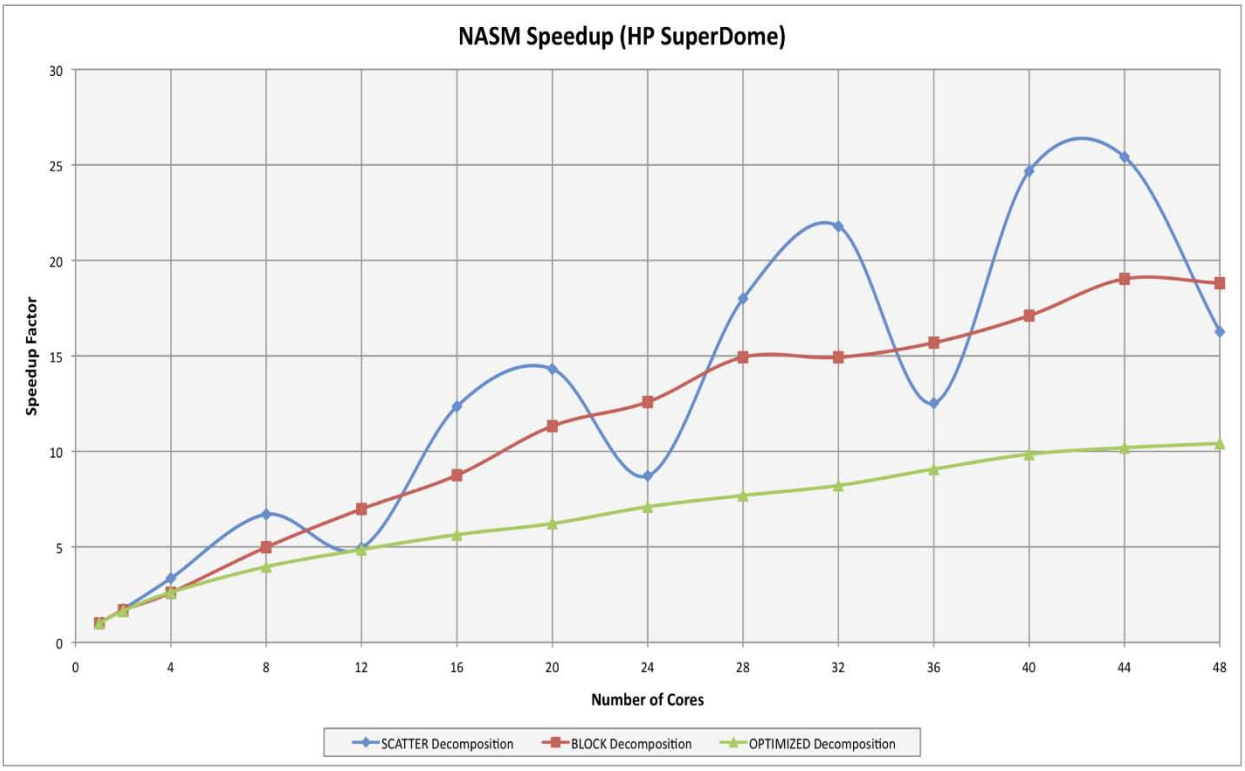

Figure 26: NASM speedup using a variety of decomposition strategies on the HP.

SCATTER decomposition exhibited erratic sine-wave behavior with performance hiccups at multiples of 12 processors. This was due to improper load balancing in which several congested airports were placed on the same processor. BLOCK and OPTIMIZED decomposition strategies produced more consistent, although not as impressive, speedup. The object placement strategy offered by OPTIMIZED decomposition was anything but optimized and proved to be suboptimal. The algorithm did not consider the variability in sector and airport congestion as a function of time. In addition, the attempt at maximizing local computation by minimizing event messages and rollbacks ended up serializing large portions of the problem.

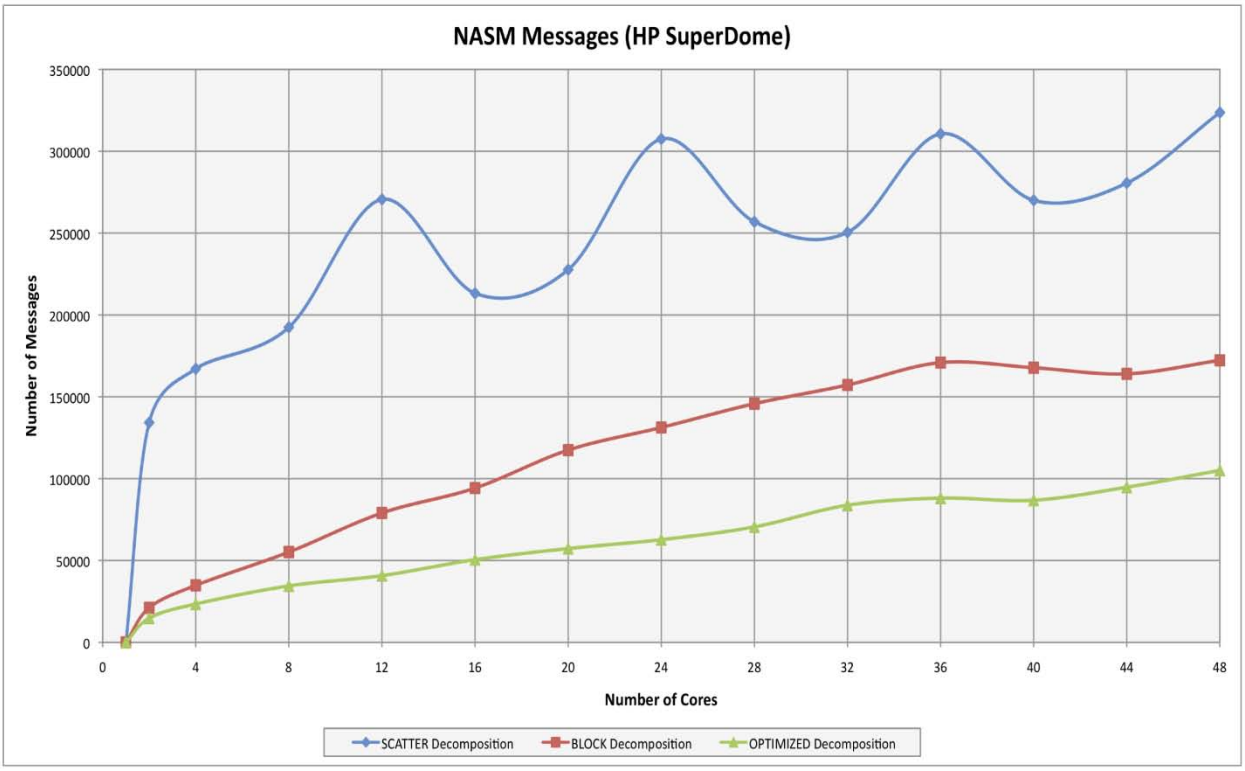

Figure 27: NASM event messages using a variety of decomposition strategies.

Figure 27 illustrates the number of event messages sent between objects located on different processors. There are zero messages when executing on one processor because every object is located on the same processor. BLOCK and OPTIMIZED decomposition helped minimize event messages by keeping neighboring sectors on the same processor. SCATTER decomposition introduced noticeable spikes occurring at multiples of 12 processors. This is consistent with the results of the previous figure. The next figure illustrates that these spikes in event messages were attributed to excessive rollbacks.

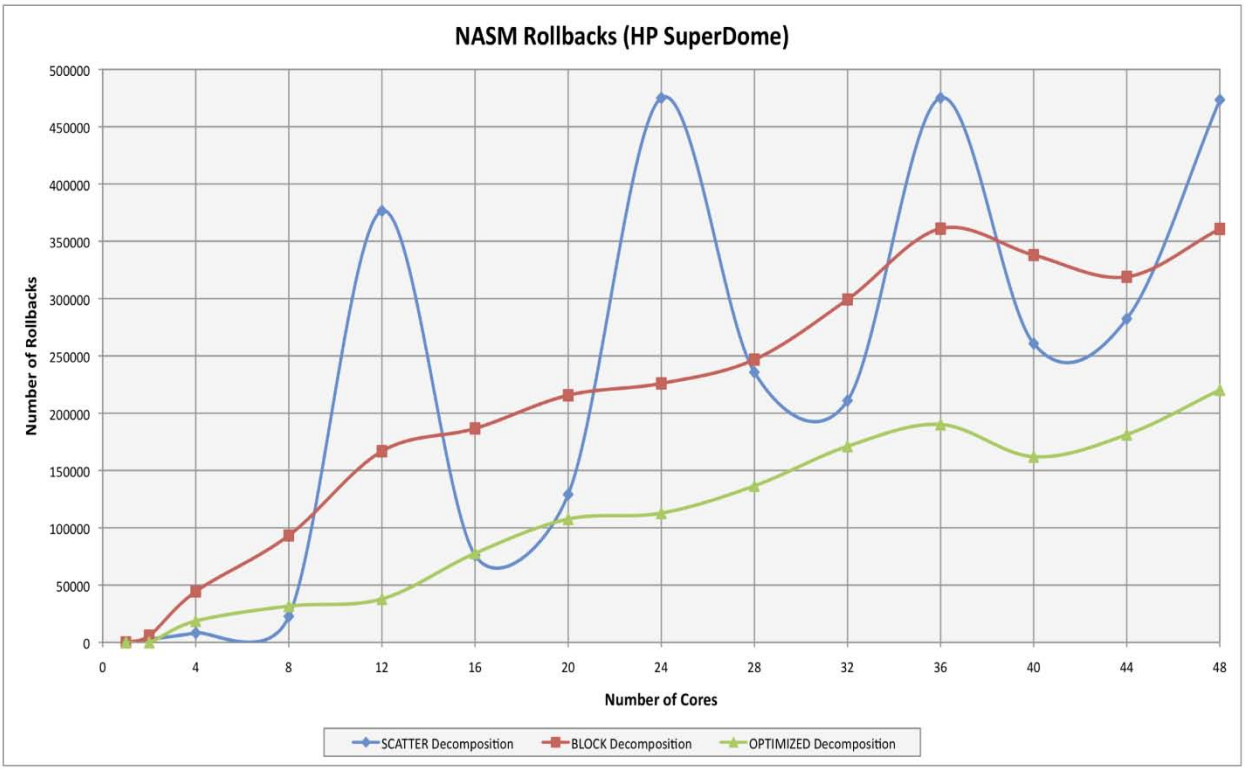

Figure 28: NASM rollbacks under a variety of decomposition strategies on the HP.

Figure 28 illustrates the number of rollbacks using SCATTER, BLOCK, and OPTIMIZED decomposition strategies on the HP Superdome. The spike in the number of rollbacks at multiples of 12 processors using SCATTER decomposition is consistent with the previous figures and results.

5.4 Lanchester5D Benchmarks

The Lanchester5D simulation was described in Section 3.12.1. Figure 29 illustrates speedup of the Lanchester5D simulation on the common and affordable Phantom and Dell Inspiron 530 multicore desktop computers.

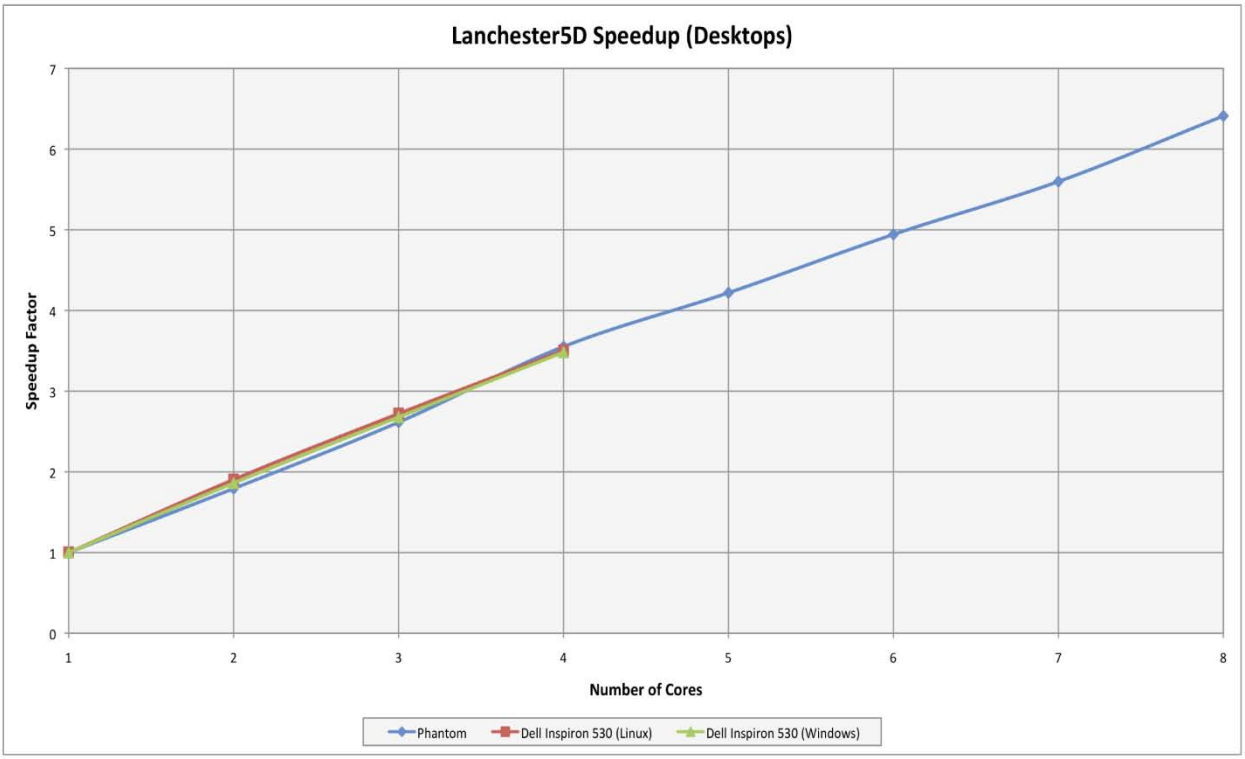

Figure 29: Lanchester5D speedup on common multicore desktops.

Scalability with smooth linear speedup was shown on all desktop configurations. An approximate speedup factor of 3.5X was achieved on all systems when using four processors. Identical speedup was achieved on Linux and Windows, although the overall execution time on Linux was faster (not shown here). Beyond four nodes, the Phantom continued to demonstrate linear speedup. At eight processors, the Phantom showed a speedup factor of 6.5X.

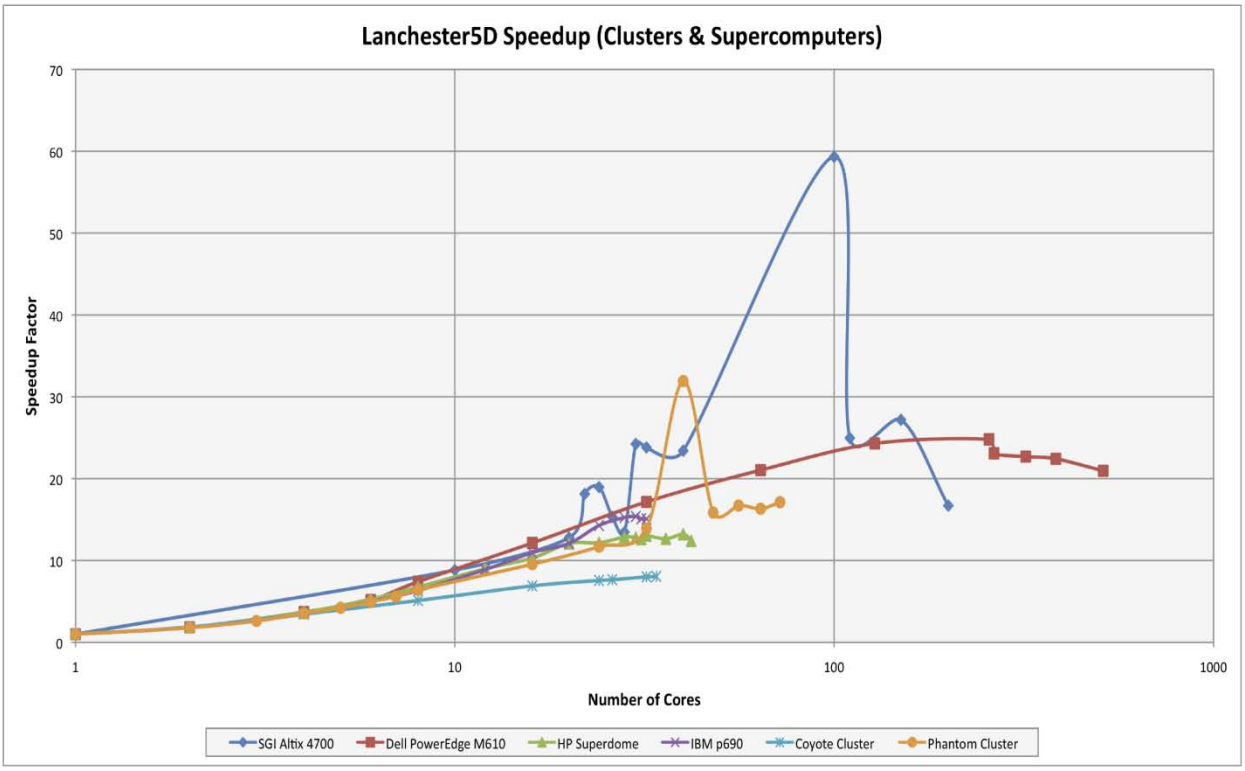

Figure 30: Lanchester5D speedup on clusters and supercomputers using WarpSpeed.

Figure 30 illustrates Lanchester5D speedup using WarpSpeed time management on a variety of clusters (Coyote and Phantom) and supercomputers (SGI Altix 4700, Dell PowerEdge M610, HP Superdome, and IBM p690). These benchmarks tested the Lanchester5D simulation using up to one million entities.

Since the WarpIV Kernel is not currently optimized for running in distributed environments, it was expected that the Coyote and Phantom clusters would not perform as well as the shared memory machines. However, the results for the compute clusters were promising and demonstrated scalability. There was an interesting performance spike on the Phantom cluster at 40 processors. This jump in performance was due to a significant drop in the number of event messages sent across the wire. A better dispersion of objects resulted in more local computation and less network traffic.

The HP Superdome and IBM p690 produced relatively comparable results. However, the SGI Altix 4700 demonstrated significant performance gain over the Dell PowerEdge M610. This is attributed to the fact that the SGI Altix 4700 is a shared memory machine, whereas the Dell PowerEdge M610 sends messages between blades over a wire.

Multiple runs were executed and averaged for each test configuration. Simulation results were repeatable in each and every case. However there was relatively wide variability in execution time between some of the runs. The HP Superdome and IBM p690 had relatively zero variability between runs, since WarpIV Technologies, Inc. had sole use of the machines and could guarantee there were no other users or CPU-intensive update or backup processes running in the background. Variability is expected in the cluster configurations due to the unpredictability and

inconsistency of network communication. Memory caching may have played a role in the variability of the SGI Altix 4700. Neglecting outliers and excessively variable data points, the SGI Altix 4700 demonstrates increasing linear speedup beyond 100 processors.

It is important to note that subsequent Lanchester5D benchmarks performed by Mike Gacek demonstrated scalability with four million entities.

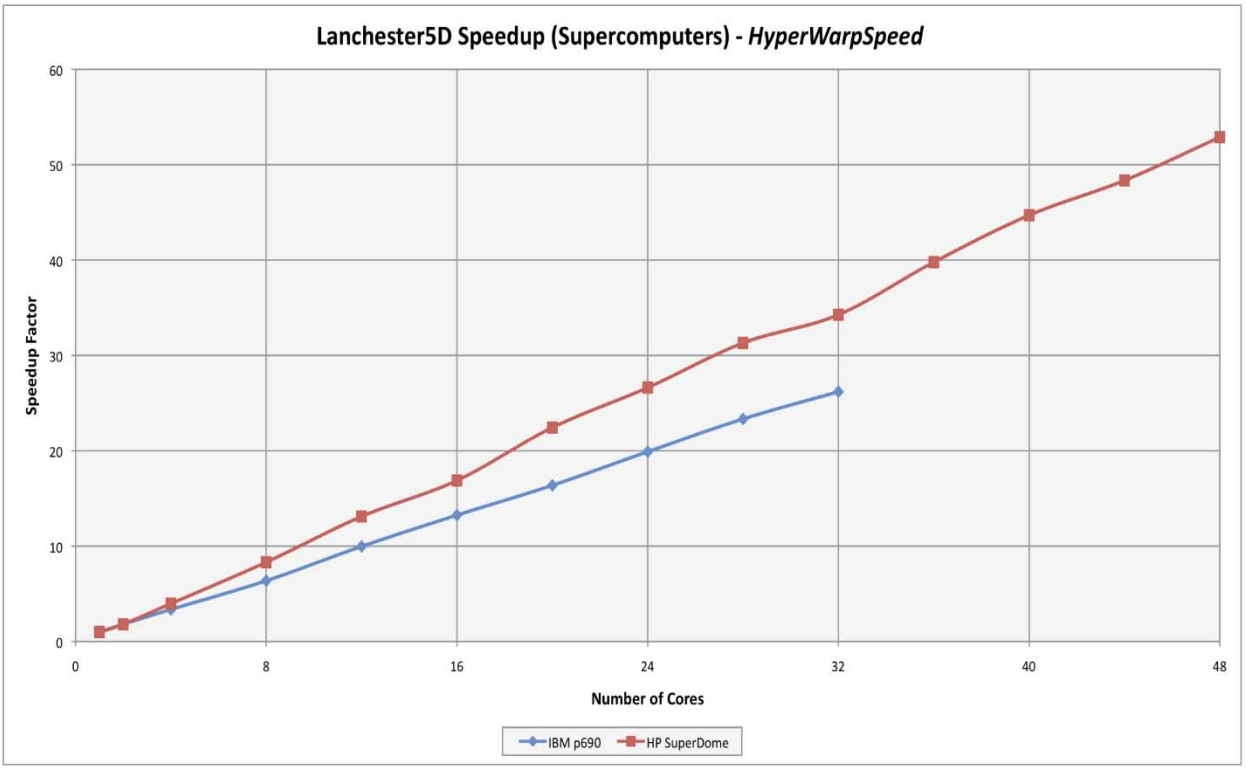

Figure 31: Lanchester5D speedup on supercomputers using HyperWarpSpeed.

Figure 31 illustrates Lanchester5D speedup using HyperWarpSpeed time management on the HP Superdome and IBM p690 supercomputers. Both supercomputers demonstrated scalable performance with smooth linear speedup. A speedup factor of 26X was achieved on the IBM p690 at 32 processors. The HP Superdome exhibited super-linear speedup with a speedup factor of 52X on 48 processors. The super-linear speedup is attributed to the overhead produced by the massive Lanchester5D event queue. Distributing the event queue reduced overhead while improving performance.

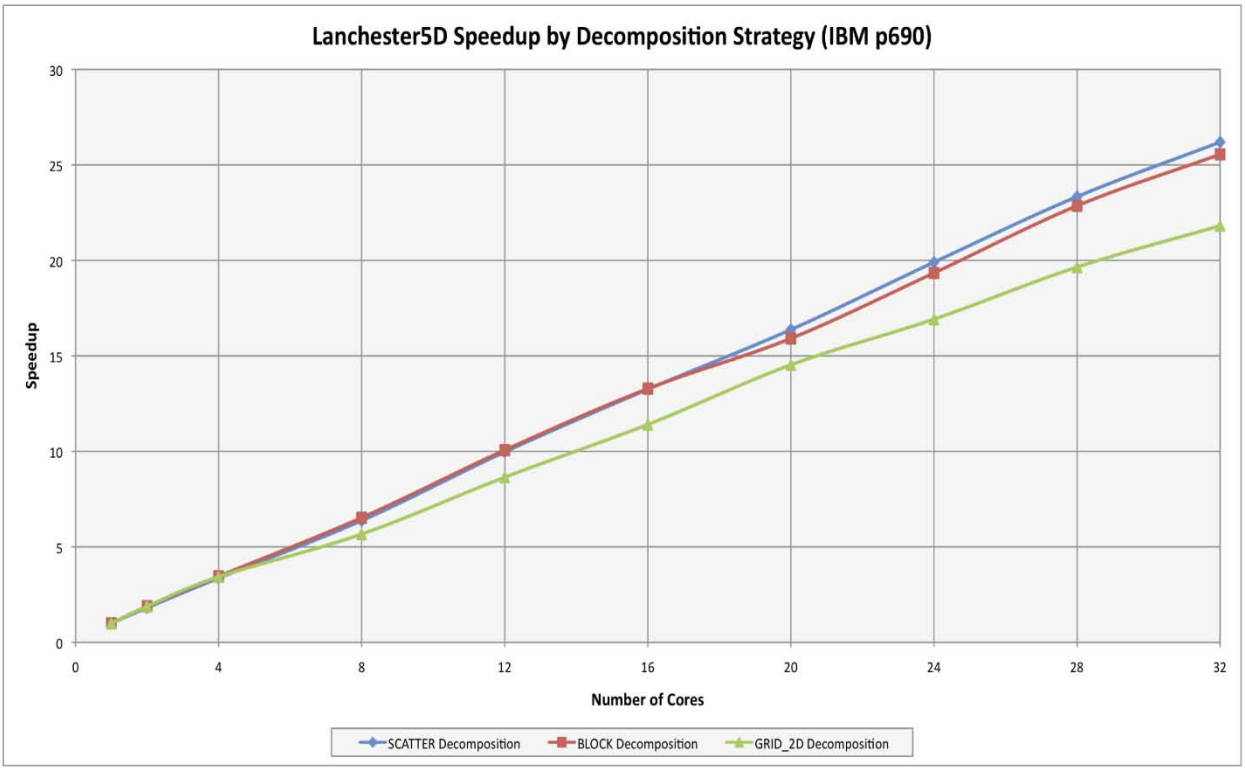

Figure 32: Lanchester5D speedup on the IBM p690.

Figure 32 illustrates Lanchester5D speedup using SCATTER, BLOCK, and 2D_GRID decomposition strategies with HyperWarpSpeed on the IBM p690 supercomputer. Decomposition strategies had a minimal impact on scalability. The following figures illustrate the potential impact of decomposition strategies on communication overheads.

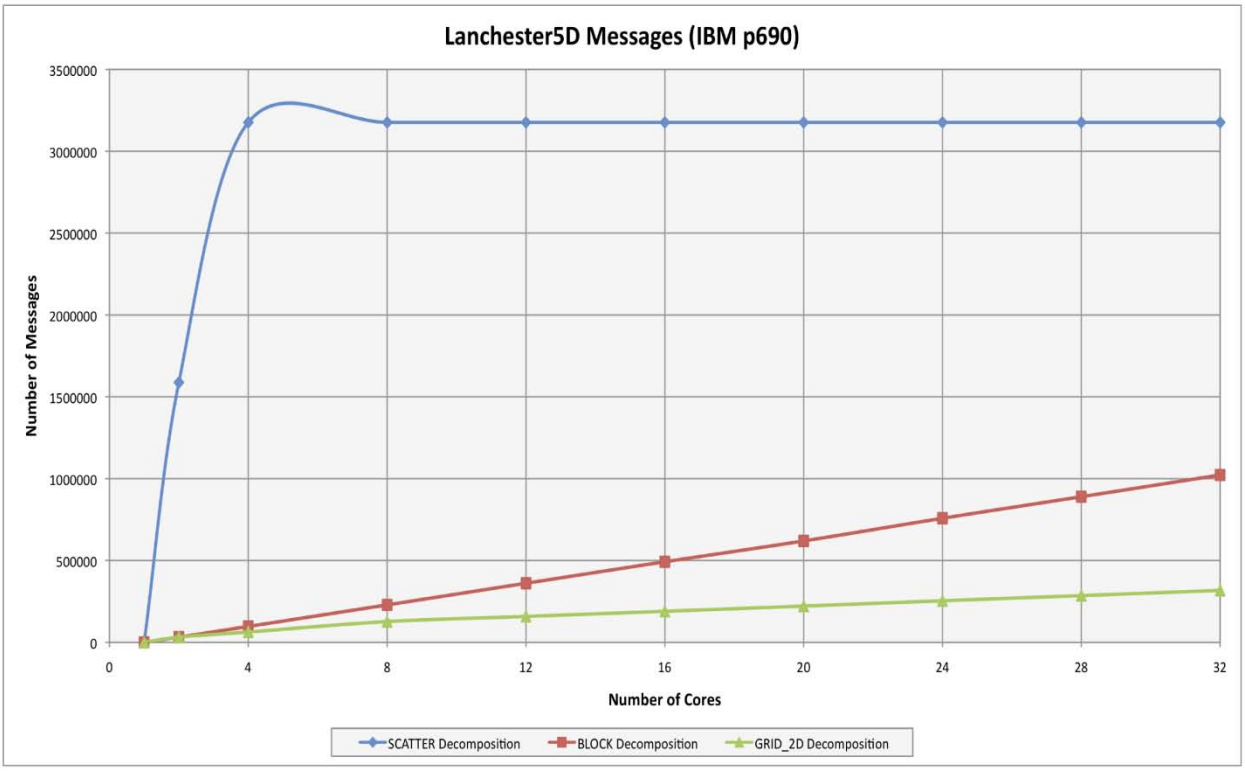

Figure 33: Lanchester5D event messages on the IBM p690.

Figure 33 illustrates the number of event messages sent using SCATTER, BLOCK, and 2D_GRID decomposition strategies with HyperWarpSpeed for the Lanchester5D simulation. There are zero messages for one processor since every object is located on the same processor. Using SCATTER decomposition, the number of event messages is fixed at 3.2 million beyond four processors. This is due to the fact that neighboring grid cells are almost always located on a different processor.

The number of event messages sent using BLOCK and GRID_2D decomposition increases linearly as a function of the number of processors. Neighboring grid cells tend to exist on the same processor with these strategies, but more so with GRID_2D decomposition since it bundles neighboring cells in both the X and Y directions on the same processor. The GRID_2D decomposition strategy generated the least number of event messages and kept computations local. Minimizing the number of event messages is critical when running on compute clusters and event-processing time is negligible.

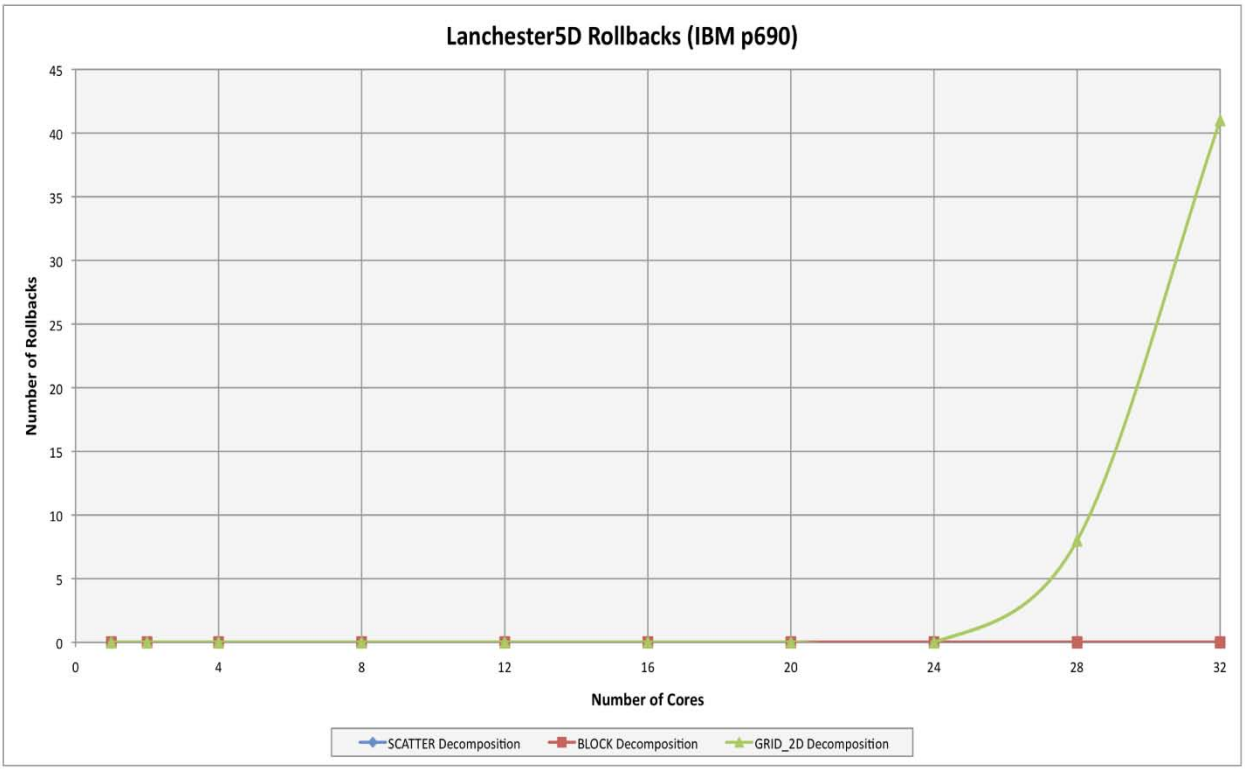

Figure 34: Lanchester5D rollbacks on the IBM p690.

Figure 34 illustrates the number of rollbacks that occurred using SCATTER, BLOCK, and GRID_2D decomposition strategies with HyperWarpSpeed time management on the IBM p690 supercomputer. Due to near-synchronous event processing and a near-perfectly balanced workload, rollbacks were essentially nonexistent under each and every decomposition strategy. Note that the worst-case measurement was only 40 rollbacks out of millions of committed events.

6 Recommendations

Future work could investigate several technical aspects. *First*, additional computational acceleration techniques such as Graphics Processing Units (GPUs) and Field Programmable Gate Arrays (FPGAs) should be explored to improve the performance of HyperWarpSpeed when necessary. *Second*, new modeling techniques must be explored to develop simulations having estimated and predicted state values based on inputs from external sources while having statistical properties without generating random numbers. *Third*, other estimation and prediction techniques such as extended Kalman Filters, Bayesian estimation theory, and Particle Filters should be explored to address non-linear systems, stability/convergence issues, and situations where critical measurements are missing or unavailable. *Fourth*, analysis techniques and tools must be developed to characterize the different measures of effectiveness and performance. *Fifth*, optimization techniques can be developed to automatically analyze outcomes of branches and prune those that do not produce desired outcomes while generating new ones.

A final step is to develop validated models and to then integrate this capability into real-world systems. Live intelligence data feeds coming from the Global Information Grid (GIG) will drive the estimation and prediction processes. Analysis tools will give rapid feedback to the users to describe effectiveness and performance of the system. They will help users understand the current estimated state of the system as real-time data is received. The analysis tools will also provide feedback to the users to help them understand the predicted outcomes of the system.

The benefits of developing these concepts are far-reaching and can be used to estimate, predict, and optimize many real-world systems. Examples where this technology would have a significant impact include: urban traffic management for the Department of Transportation, air traffic control for the Federal Aviation Administration, energy grid and individual power plants for the Department of Energy, space exploration missions for the National Aeronautics and Space Administration, and military battlefield operations for the Department of Defense.

Several recent factors are converging that demand a broader vision for developing complex systems of systems. These factors include the net-centric transformation of military armed forces, emerging multicore computing revolution, plug-and-play component/composite interoperability methodologies, Service Oriented Architectures, web technologies, Live, Virtual and Constructive (LVC) interoperability standards, cognitive modeling, open source software, standardization of data models, the Department of Defense Architecture Framework, and now technologies such as HyperWarpSpeed that introduce a new simulation paradigm for supporting advanced decision making.

To accomplish this vision, WarpIV Technologies, Inc. is leading the *Parallel and Distributed Modeling & Simulation Standing Study Group* (PDMS-SSG) within the *Simulation Interoperability Standards Organization* (SISO). The intent of this study group is to report and raise awareness within the M&S community concerning the need for a standard multicore-ready architecture for Force Modeling and Simulation.

7 References

[1]. Lammers Craig, Steinman Jeff, Valinski Maria, and Roth Karen, 2009. "Five-Dimensional Simulation for Advanced Decision Making". In proceedings of the *SPIE – Enabling Technologies for Simulation Science XIII*, Orlando, FL. 2009.

[2]. Estimation and Prediction Using Optimistic Simulation and Control Theory Techniques. Contract #FA8750-06-C-0218. POC: Dawn Trevisani, Program Manager, AFRL/RISB, Rome, NY. Email: Dawn.Trevisani@rl.af.mil, Phone: (315) 330-7311.

[3]. Steinman Jeff, Lammers Craig, Valinski Maria, and Roth Karen, 2008. "Simulating Parallel Overlapping Universes in the Fifth Dimension with HyperWarpSpeed Implemented in the WarpIV Kernel". In proceedings of the *Spring Simulation Interoperability Workshop*, Providence, RI. 2008. **Nominated for Best Paper**.

[4]. Lammers Craig, Valinski Maria, and Steinman Jeff, 2009. "Multiplatform Support for the OpenMSA/OSAMS Reference Implementation". In proceedings of the *Spring Simulation Interoperability Workshop*, San Diego, CA. 2009. **Nominated for Best Paper**.

[5]. Mr. Michael Gacek, Computer Scientist, AFRL/RISB, Rome, NY. Email: Michael.Gacek@rl.af.mil, Phone: (315) 330-3675.

[6]. http://www.afrl.hpc.mil

[7]. http://www.mhpcc.hpc.mil/doc/mana.html

8 List of Acronyms

ISR: Intelligence, Surveillance, Reconnaissance.
ATO: Air Tasking Order.
COP: Common Operational Picture
DIS: Distributed Interactive Simulation.
PDU: Protocol Data Unit.
STOMP: Software Toolkit for Optimizing Mission Plans
ATM: Air Traffic Management.
FAA: Federal Avionics Administration.
DOT: Department of Transportation.
DOE: Department of Energy.
SISO: Simulation Interoperability Standards Organization.
PDMS-SSG: Parallel and Distributed Modeling and Simulation Study Group.
SPEEDES: Synchronous Parallel Environment for Emulation and Discrete Event Simulation.
ATC: Air Traffic Control.
DSAP: Dynamic Situation Assessment and Prediction.
GUI: Graphical User Interface.
AODB: Air Operations Data Base.
ABP: Air Battle Plan.
SIM: Simulation.
COA: Course of Action
XML: Extensible Markup Language.
NASM: National Air Space Model.
HTML: HyperText Markup Language.
PBS: Portable Batch System.
SPAWAR: Space and Naval Warfare Systems Command.
GPU: Graphics Processing Units.
FPGA: Field Programmable Gate Array
GIG: Global Information Grid.
LVC: Live, Virtual and Constructive