Bringing Control Theory to C2
An Update on the DARPA JFACC Program

Col. Daniel C. McCorry, USAF, JFACC Program Manager
Dr. H. Stephen Morse, SM&A, Chief Technologist
JFACC Program Office
DARPA
3701 N. Fairfax Drive
Arlington, VA 22203-1714
703 – 526 – 6607
dmccorry@darpa.mil
hmorse@snap.org

Abstract

The current emphasis of the JFACC program is on the applicability of control theory, broadly conceived, to selected problems in military command and control. The program is organized around the conduct of a special type of experiment, in which a control technology is matched against an externally given plant. We discuss the approach and initial experimental results of a number of the research teams, as well as the work of the system architect. A final section reviews current challenges as well as anticipated future results and developments.

1. Introduction

Control Theory and its supporting technologies have progressed to the point where they can be applied to selected problems in military command and control and achieve quantum improvements in the effectiveness and efficiency of military operations. The Defense Advanced Research Projects Agency (DARPA) Joint Force Air Component Commander (JFACC) Program aims to verify (or disprove) this simple yet elegant hypothesis.

The JFACC Program is fundamentally different from what it was just a short while ago. As originally conceived, the program set out to apply planning and scheduling technologies, developed within the Artificial Intelligence community, to problems in Air Operations – in particular, to activities of the Air Operations Center (AOC) and its chief daily product, the Air Tasking Order (ATO). It quickly became clear, however, that other service programs were pursuing similar efforts. In line with DARPA's charter, the program required a new vision – a truly revolutionary approach to military Command and Control. Therefore, in the Winter of 1998, the old effort was terminated, and a new effort begun.

This paper will describe how the current program is structured, our progress to date, and possible future directions. Let us begin, however, with a few foundational ideas which motivated the selection and understanding of our central hypothesis.

Command and control (C2) is the function which dictates all action in any military operation. And since every action is the result of a decision, we conclude that *decision making* is the essence of command and control. The JFACC Program, therefore, focuses on the how to achieve

more effective and efficient decisions. We recognize that while information is clearly necessary to support military C2, it is not sufficient. In fact, it is the required decision which generates the necessity for appropriate information. Just as raw materials alone are not sufficient to produce a final product in the world of manufacturing, so also information alone is not enough to produce good decisions. The decision *is* the product of C2.

A guiding metaphor for the JFACC Program is a *prosthesis* – a device which enhances and extends human capability. The application of control theory should enable humans to operate routinely with enhanced capability not previously available due to human limitations. As an example, the flight control system of the F-16 fighter permits a radically different aircraft design, beyond the capability of normal human reflexes to control. The F-16 flight control system enables a more maneuverable (or *agile*) weapon system without sacrificing *stability* (in fact stability is increased throughout the entire flight envelope). Human limitations no longer are an obstacle to such a design. The pilot can now focus on employing the weapon system rather than controlling the aircraft. In the same way, we expect that new advances in control theory, applied to *enterprises*, will enable military operations to routinely function with both agility and stability heretofore not achievable. Military decision makers should focus on commanding rather than controlling an operation.

In a similar sense, the product of our research should enable human beings to routinely and confidently operate in the face of unprecedented levels of complexity and reduced latency. Central to this is the ability to reduce a complex, high-dimensional space down to a few essential and comprehensible parameters that are of interest at the moment. The system must compute and provide feedback, not only about recommended courses of action, but about the shape and sensitivity of the decision "landscape." How critical is the upcoming decision? Which piece of additional information is pivotal? Are many good options available, or is the decision space highly constrained? We believe that control theory, and the rigorous modeling and mathematical analysis that are at its core, will enable us to create C2 systems that act as a powerful and intuitive decision support prosthesis. In this sense, we believe that the military C2 infrastructure should be *human centric* – not human *intensive*– enabling the commander to make effective decisions in an increasingly complex and time-critical environment.

Some final comments about the program before turning to more substantive considerations. First, despite its name, the JFACC program is *not* exclusively, or even primarily, concerned with Air Operations. The technologies we are exploring will have application at all levels of military command and control – strategic, operational, and tactical. And while an Air Operations scenario is the basis for our initial investigation, our researchers have selected problem definitions with widely varying temporal and spatial characteristics. Second, we are doing research; we are *not* building working prototypes. The results of our work will take the form of research papers presented, and published, both at a series of symposia we are sponsoring as well as in open literature. Our goal is to understand thoroughly both the power and limitations of the technology *before* we build.

In the remainder of the paper, we will describe our experimental procedure (Section 2), present some of the approaches our researchers are taking (Section 3), briefly discuss architectural

implications (Section 4), and provide an initial glimpse of future directions and challenges (Section 5).

2. **Experimental Approach**

As we noted above, the goal of the program is to verify (or disprove) a hypothesis concerning the applicability of control theory to selected problems in military command and control. To achieve this goal, we have designed the program around a specially constructed type of experiment – one that is particularly well-suited to the verification of controllers. Briefly, the experiment consists of operating a *controller* against an independently instantiated *plant*. For those without a control theoretic background, the term *plant*, when used in this context, has its roots in the chemical industry, where automatic controllers were used to monitor and adjust the input rates of reactants in industrial *plants*. Thus, the word *plant* refers generically both to the process being controlled and to all relevant aspects of the environment in which it resides. For our purposes, this might include weapons systems, targets, sensors, command and control entities, logistics, weather, terrain, etc., for both friendly and enemy forces.

Since we do not have access to a real plant, we must settle for second best – that is, a simulated plant. This is a piece of software capable of accepting control commands from the controller, driving the dynamics of the plant forward in time (including modeling the effects of engagements between friendly and enemy assets), and returning appropriate information about the state to the controller (see Diagram 2.1). In particular, there must be a clean separation between the plant being controlled, on the one hand, and the controller and its internal models and algorithms, on the other. In the real world, that separation is inescapable. In a computer simulation, it means that each model must be developed and implemented twice: once for the computer simulation that is the plant; and once within the controller (to do its own internal bookkeeping, tracking, state estimation, and prediction).

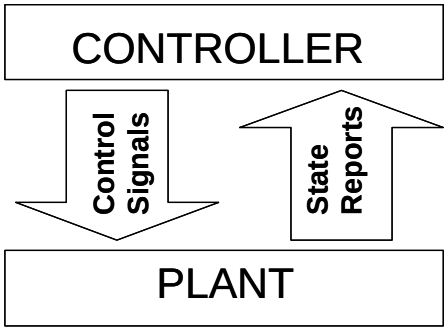

Diagram 2.1: The Basic Experiment

A key issue, then, is the means by which the models used by the controller, on the one hand, and the models used within the plant simulation, on the other hand, are made to agree. In the real world, the process of tuning or adjusting controller models to best match plant behavior is called *system identification* (or *plant identification*). In our experimental framework, however, we have

the luxury of deferring this difficult task until the algorithms, and their theoretical characteristics, have been thoroughly wrung-out and understood. This can be done by the simple expedient of specifying the desired behavior of the plant (that is, of the models used to implement the plant in the simulation). Eventually, of course, we will face up to the more difficult problem of interfacing to a real plant – that is, the real world. At this stage of research, however, we have permitted our researchers the luxury of specifying the plant behavior (that is, the implementing models) they need to validate the basic algorithmic characteristics of their proposed controllers.

Even so, we have not permitted the controller research teams to proceed completely independently. While they may develop plant models internally for their own research and experimentation, we are also requiring that each team interface to an externally provided version of the plant. These versions (and there is potentially a different one for each researcher) are produced by an *Enterprise Modeler*. The job of an Enterprise Modeler is to provide a plant model against which the experimental claims of a reasearcher can be validated, and to ensure the experimental integrity of the interface (that is, to ensure that no information flows between the controller and the plant other than agreed-to state reports and control signals).

As the project continues, the very simple initial models proposed by the researchers will, it is hoped, increase in complexity and fidelity. We are currently about half way through the 18-month investigation and design phase of the program, and results are still preliminary (but encouraging). Another task of the Enterprise Modeler is to assist the researchers by suggesting and helping to implement greater modeling fidelity, in this way promoting the process of transition from the laboratory to the real world. It remains to be seen how much progress each of the teams will be able to make.

In designing their experiments, the controller research teams have been given two types of guidance. First, they should design and perform experiments that clearly demonstrate the *value* of their proposed technology. That is, the scenarios (implemented within the plant) against which the controller must execute should be tailored to play to the strengths of the technology. Second, and just as important, they should also conduct experiments that explore the *value landscape* within which the controller operates. By this, we mean that plant characteristics should be selectively varied about nominal values to show where and under what conditions the controller breaks (or performance degrades). Examples include: noise or missing data in the state observation signal; force imbalance; unexpected enemy actions; changes in expected rates of attrition; mis-matches between plant and controller models; etc. The result should be a complete, balanced, and objective assessment of both the strengths and limitations of the proposed technologies. In the next section we will present some of our initial results.

In addition to verifying the performance and limitations of the control technology, these experiments will be instrumented to measure data rates, loading, and latencies. This will be important information for use by the *System Architect* (whose role is discussed at greater length in Section 4). For the moment, we only mention that this experimental process will also produce useful performance metrics contributing to the design and understanding of an appropriate supporting communications and reporting infrastructure.

3. **Examples**

In this section, we'll look at four approaches, or ways of thinking about the problem, that span the set of researchers, and illustrate well the breadth and diversity among team members. They include: model predictive control; modeling an active adversary; hierarchical problem decomposition; and emergent behavior.

3.1 ***Model Predictive Control***

The key notion in Model Predictive Control is that the predictive function explicitly models and takes into account *its own activity* as part of the decision-making process. It is widely recognized that the replan cycles for planning and scheduling systems must match, or exceed, the typical rates at which dynamic events occur. There is all the difference in the world between ConOps where the replan period is (say) 24 hours, and where the replan period is (say) 3 hours. Further, they must be capable of responding to unexpected, ad hoc events – either repairing the existing plan (small to moderate modifications), or throwing it out and starting from scratch (total replan). To do this, a typical planning and scheduling tool will look out some distance into the future (the planning horizon), and predict as best it can what the future may hold. Using this prediction, it will then optimize (as best it can) over the planning horizon, and issue the revised plan for execution.

By contrast, *model predictive control* explicitly models the fact that replanning will take place as an integral part of the process – *not* as an "exception." That is, in its model of the future, it explicitly recognizes that certain types of information will be available at known future points, and (therefore) that it may be better to defer decisions and allocation of resources until that point in time. In effect, model predictive control attempts to trade off the penalty for delay (missed opportunity now) vs the penalty for acting too soon (missed opportunity later).

Another advantage of model predictive control is that it becomes possible to explicitly take plan-to-plan variability into account. That is, successive plans issued by the controller overlap, and it becomes possible to compare these plans and to compute the amount of change (measured, say, by logistical impact). It even is possible to explicitly factor this into the optimization formula – that is, refusing to accept large changes in the plan that yield only modest expected performance improvement. This provides the best approach, to date, for trading off *agility* (the ability to react to new information with low latency) against *stability* (the ability to adhere, as much as possible, to published plans).

Finally, we note that this approach is quite flexible, and is capable of accommodating a number of different modeling techniques. As such, it provides an architectural framework for incorporating the modeling and control approaches from other researchers.

The biggest challenges facing this approach include: building adequate models; investigating sensitivity (that is, under what conditions does this approach significantly improve performance); design of an adequate objective function; and computational complexity (the stochastic search tree can become unmanageable, requiring suitable pruning heuristics). For example, one of our researchers has significantly extended the traditional Lanchester attrition model to permit rapid

closed-form calculation of future probability distributions. Another researcher has proposed to accelerate the tree search using *Neuro-Dynamic Programming* – a technique that can learn how to estimate value-to-go in the Bellman equations.

3.2 ***Active Adversary***

Rather than model adversary actions as noise or disturbance, several of our researchers are attempting to anticipate, and optimize, against an active and intelligent adversary. The basic approach is to model the situation as a *game* in which the adversary's goal is assumed to be opposite to one's own. Under certain assumptions about mathematical characteristics of the objective function (usually, a utility balancing loss of assets on each side), saddle-point solutions can be found which solve the mini-max game theoretic formulation. This yields conservative, "worst case" strategies which can subsequently be tempered as additional information about actual enemy actions becomes known.

These games can be formulated is a number of ways, and in each case a different collection of mathematical tools can be brought to bear. In one approach, the researcher has formulated the problem as a deterministic *differential game*. Optimal trajectories of interacting objects are computed over a time horizon of interest (typically 30 minutes or so in low-level tactical encounters), which are then used to build the optimized controller. One advantage of this approach is that, as a by-product of controller formulation, a nominal future state is generated. This enables the controller to track actual behavior against its prediction (that is, to compute the *residuals*), and to alert the commander when expected and observed actions are straying outside of acceptable performance bounds. It might even be possible for the system to recompute its optimized solution, becoming in this way *adaptive* to the changing circumstances of the observed encounter.

Another researcher is explicitly modeling the uncertainties of encounters – both in terms of knowledge of enemy locations and in terms of the outcome of local engagements. In this formulation – an optimized stochastic game -- the goal is to optimize the expected value of some key parameter: attrition, time, probability of reaching a desired end state, etc. As with the previous example, an optimized trajectory can be formulated over a time horizon, and residuals (expected minus observed) can be computed.

Two researchers have chosen to model the problem as an *abstract board game*, but in each case with very different optimization approaches. By board game, we mean that the geographic extent of the battle space is gridded (hexagonally or rectangularly), and objects move fixed numbers of cells at each turn. Controls include options for moving, firing and hiding. In one approach *Linguistic Geometry* has been applied to "solve" the game. Linguistic geomoetry is a computational approach for enumerating and pruning the search space of an abstract board game. The geometrical and dynamical characteristics of the pieces (= objects in the battlespace) are exploited to rapidly prune and eliminate unpromising options. This scalability enables the approach to extend to very complex situations, involving many pieces and a large geographical extent.

The other researcher has chosen to compute the solution to the abstract board game using Stackelberg "leader-follower" ideas. The key notion here is that a player can influence friendly or enemy behaviors by estimating their local payoff matrices, and then adjusting his own actions accordingly. This approach has already shown significant promise, based on a simulated encounter in which Red and Blue forces are evenly matched. Using the traditional Nash solution, Red emerges victorious. But, when Blue uses a strategy based on leader-follower ideas (a Stackelberg solution), it is enough to turn the tables. Further, because the enemy intent is explicitly modeled and estimated, it becomes possible to use state observation data obtained during the encounter to automatically detect and adjust to changes in intent. That is, it is possible to infer the hidden (not directly observable) entries in the enemy's payoff matrix by fitting an estimate to the observed data. It is even possible to select Blue behaviors deliberately to provoke a Red response, thereby obtaining data to support improved estimates.

To address the issue of target valuation, one of our research teams has proposed a new theoretical approach called *Ordinal Games*. The advantage is that, rather than assigning numerical values and costs to objects in the battlespace, a commander need only express a *preference* among possible outcomes. Ordinal Games are also applicable to problems in economics and the social sciences that are intractable when modeled using numerical utility. This is an exciting, and unanticipated, outcome from the ongoing research.

The biggest challenges facing this approach include: building adequate models; investigating sensitivity (that is, under what conditions does this approach significantly improve performance); design of an adequate objective function; computational complexity; and brittleness in the face of inadequate support for state reports.

3.3 ***Hierarchical Decomposition***

As we noted in Section 1, despite its name the JFACC program is interested in potential solutions at all levels of the C2 hierarchy – from strategic, through operational, to tactical. A few of our researchers have focused on schemes that permit state estimation data to flow upward, and control data to flow downward, across these boundaries. The approach begins by recognizing that the models needed at each of these levels are different, and represent different levels of aggregation – both of time and of objects. Thus, mappings are needed between models at the different levels. Ideally, these mappings, or even portions of the models themselves, could be generated automatically.

Next, it is recognized that state observation data may enter at any of the levels, and flow upwards or downwards. In one important case, low-level tactical state data (say, position and velocity of aircraft) enters at the bottom. The hope is that only very rarely – say, when the reports indicate significant deviation from expectation – would this data need to percolate to higher levels. This shields the upper levels from unnecessary information, and reduces the total amount of traffic across organization boundaries. However, when a significant event does occur, it is important that higher levels be notified promptly, and in a form that matches their own understanding and semantics. This is illustrated in Diagram 3.1, based on a design approach offered by one of our researchers, which shows one node in a control hierarchy. Depending on the type of event and/or state information, four different data flows through the node are possible (as indicated by

the numbering scheme). Three of the four possible data flows shield the upper levels from lower level detail.

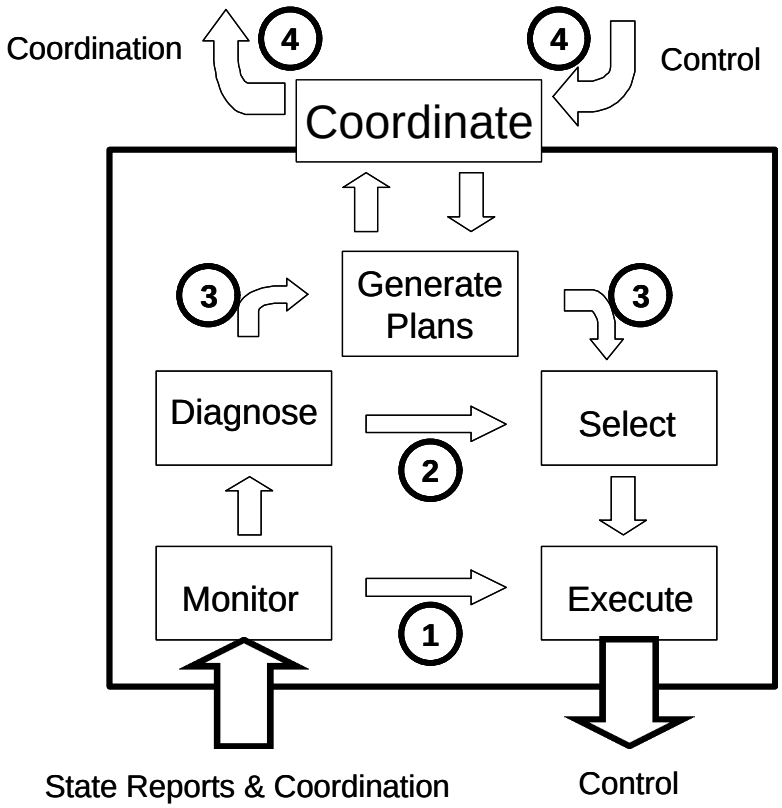

Diagram 3.1: Nominal controller node in hierarchy

Third, it is recognized that command information must flow down the hierarchy, terminating ultimately in low-level tactical entities. Again, a mapping is required, but this time in the opposite direction: *from* higher levels of aggregation *to* lower levels, with less coarse temporal, spatial, and functional granularity. It is also important to preserve tracability, and to derive the value or priority of lower level tasks from higher level statements of intent.

Finally, optimization algorithms can benefit from problem decomposition. Complexity is often super-linear (quadratic, or even exponential) in the number of variables, so it is often of benefit to decompose a single large problem into a number of smaller ones. This permits parallelism (the small problems can be solved independently and concurrently), and can also considerably reduce overall run time. This addresses the notion of the *scalability* of the proposed algorithms -- from small problems in the laboratory, up to large problems on the scale of real world operations. The need for problem decomposition can benefit from the natural partitioning given by the different levels in the tree, but there remains the difficulty of actually decomposing the problem and examining the extent to which the decomposition results (if at all) in significantly sub-optimal results.

To briefly summarize, the biggest challenges facing this approach include: efficient and consistent mapping (that is, aggregation and dis-aggregation) between levels; shielding higher levels from unnecessary detail while, at the same time, responding rapidly to critical events; consistency and tracability as command decisions move down the hierarchy; and problem decomposition and scalability.

All of these issues are being addressed by one or another of our researchers. For example, one research team has a formal, automatic decomposition of an LP optimization into multiple independent sub-problems. This makes a formerly intractable problem solvable with reasonable latency. And, since lower levels give sensitivity feedback to upper levels, the algorithm rapidly converges to a global optimum. Another research team is investigating the possibility of automating the process of aggregation (upward flows) and dis-aggregation (downward flows). This is an ambitious project, but initial results are encouraging. For problems like those being modeled here, reductions in data of 1 to 2 orders of magnitude have been achieved. As a final example, one research team is applying principles from parallel compilers (load balancing, consistency) to formally decompose problems stated in top-level semantics into smaller problems and tasks at lower levels.

3.4 *Emergent Behavior*

As a final example of a control approach, one of our research teams is using the metaphor of insect swarms and pheromones to model large-scale military engagements. This is an offshoot of complexity theory, which has received considerable attention recently for its ability to model non-linear effects *without* the need to resort to differential equations or stochastic optimization.

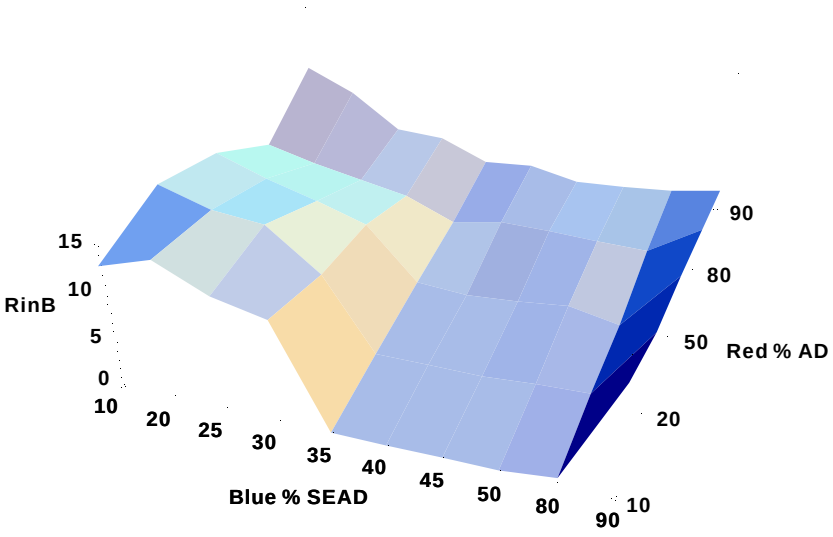

Diagram 3.2: Performance Landscape

Briefly, a pheromone is a chemically specific substance that can be deposited by an insect at a location, disperses at some rate (like perfume) through the environment, and eventually dissipates entirely. Insects sense the presence of pheromones, and respond using simple rules (e.g., follow the gradient). Out of this simple underlying model, very complex and highly optimized behaviors emerge in ways that are both non-intuitive and robust.

The major thrust of the research is to investigate whether an appropriate pheromone vocabulary, coupled with context-sensitive rules (in the form of simple state machines), is capable of statistically predicting the outcome of complex engagements, and of suggesting appropriate course of action to optimize the expected results. Because of the counter-intuitive and non-linear nature of the mechanism, there is a strong experimental component to the investigation which would have been impossible (or prohibitively expensive) on previous generations of computer hardware. A key aspect of the investigative technique is visualization of the dynamic interactions over time. This permits rapid detection of suggestive phenomena, followed by more detailed sampling in the nearby portions of the large search space.

An example of preliminary results is shown in Diagram 3.2 , which plots expected outcome of an engagement (Red forces in Blue territory) against two critical input controls – levels of Blue SEAD, and levels of Red Air Defense. The key feature that emerges is the steep gradient that occurs when Blue SEAD exceeds 30%, with only modest improvements beyond that point. The message to the commander is clear: there is a floor below which you should not venture; but above that threshold, there is plenty of room for flexibility. In other words, the modeling can present to the commander, in an easily understood way, an important characteristic of the "problem space" within which he finds himself. This type of capability is representative of what we mean by a prosthesis – the ability to reduce complex, higher dimensional spaces down to a few key terms which can be understood, and which are actionable.

4. **Architecture**

The JFACC program rests on a three-legged stool: controllers, models, and architecture. In the previous section, we dealt at some length with models and controllers, and the various technical approaches being taken by the researchers. However, each of these approaches makes assumptions about the supporting infrastructure and organization – the rates, latency, and accuracy of state reports (that is, ISR); the bandwidths and reliability of communications; operator proficiency; etc. The task of the *System Architect* within the JFACC program is to deal with these ancillary but critical issues.

Of special interest is the potential for various kinds of pathological behaviors within the enterprise in which these controllers are embedded. Feedback loops are very powerful mechanisms, but they also can give rise to non-intuitive pathologies, particularly when they are part of a complex infrastructure with dependencies, and when driven by stochastic processes with large variability. An early task of the system architect was to identify a number of candidate pathological behaviors the system might exhibit, and develop analytical and modeling tools to evaluate the ability of candidate system architectures built around the proposed control technologies to suppress these pathologies.

A short list of some of the most significant and interesting pathologies includes:

Race Conditions: This arises when a check against some condition is required by more than one agent, during a time window. If the condition changes between the time when the first agent checks and when the second agent checks, they may receive different answers and, hence, lose coordinated behavior. The SA must check when correct controller behavior requires logical simultaneity, and ensure that the infrastructure can provide it (or, at least, detect when it may have been violated).

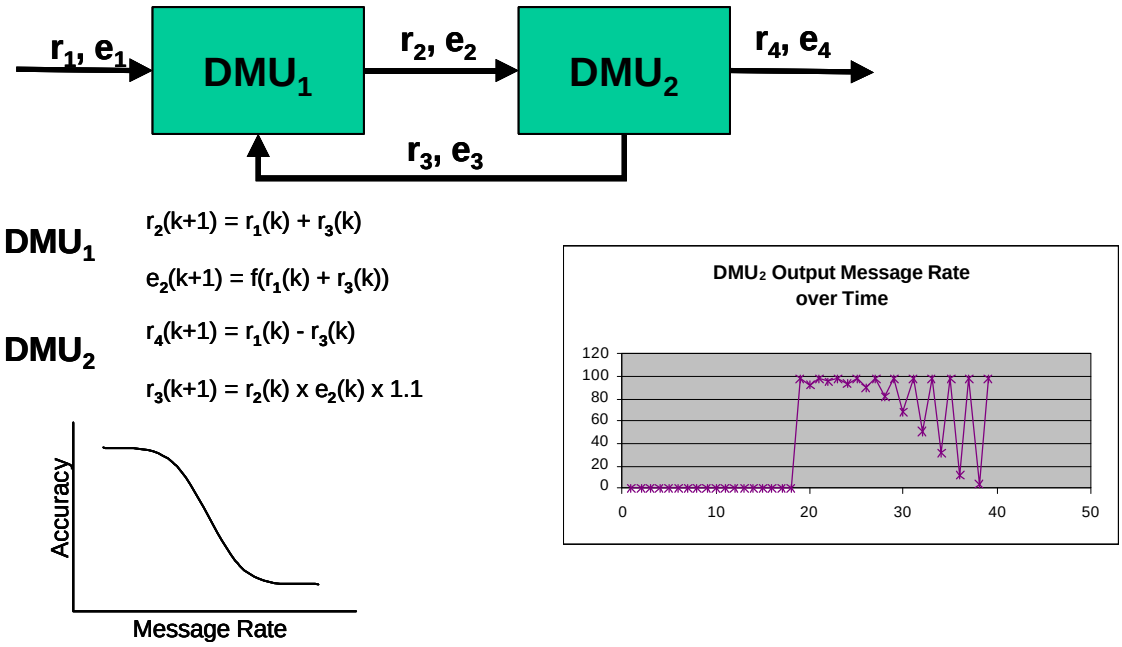

Diagram 4.1: Non-intuitive Thrashing in Feed-back Loop

Thrashing/Livelock: Closed-loop system work best when the decision dependency tree contains no cycles. If cycles exist, then a process, ***A***, may find itself waiting for input from process ***B***, when the producer of that input is itself waiting for input from process ***A***. A similar situation can arise when input conditions cause the system to oscillate rapidly back and forth between states, without making any forward progress toward the goal. The system architect must identify topological, connectivity relationships among the components that could permit such behaviors, and ensure that mechanisms are in place to detect and correct them, should they arise. Diagram 4.1 shows an example of the kind of non-intuitive behaviors that can occur, and the analysis used by the system architect to detect and assess them.

Positive Feedback: In some circumstances, the decision made by a controller may cause (via the feedback loop) an input which intensifies the signal which triggered the decision in the first place. While well-designed controllers attempt to eliminate (or inhibit) such behaviors, they may have made infrastructure assumptions as part of their mitigation strategy. The system architect must identify the paths that are susceptible to this phenomenon, and validate that derived requirements on the architecture can be satisfied.

Hierarchical Inconsistency: In Section 3.3, we saw how several of the researchers are attempting to aggregate (as information flows up the hierarchy) and decompose (as control signals flow down the hierarchy). At any such point, the possibility for inconsistency arises. That is, because information is lost (upward) or inferred (downward), the pictures of the plant being used by different levels of the hierarchy may be inconsistent. The system architect should identify the opportunity for such inconsistency to arise, and suggest tests or rules for detection and correction.

Hunting: Ideally, a control system should settle rapidly into a new equilibrium in response to external stimuli. Due to quantization (and other effects), however, it may successively overshoot, and then undershoot, its intended target. The system architect should devise experiments intended to stress the system in these ways, and validate that the system response is within acceptable bounds.

The architectures being examined are driven, to a large extent, by the proposed controller technologies. A number of provisional, candidate architectures have been proposed, spanning from fully centralized to distributed self-organizing concepts. However, detailed analysis is just beginning. The eventual product of this effort will be an operational architecture (or more than one) which effectively incorporates and integrates the various successful controller technologies.

5. **Observations and Conclusions**

Before getting to the good news (the program is on track, with significant results already in-hand, and more soon to come), we consider three obstacles that must be overcome to achieve our long-term goal of transition to an operational environment.

The first of these we have alluded to in passing – the definition of an *acceptable objective function* for the optimization algorithms. Many of the approaches rely on the existence of such an objective function, and for many it should take the form of a utility: {value of target destroyed} – {cost of assets lost}. The problem is that assigning such values and costs is notoriously difficult and counter-intuitive. Our researchers assume the existence of these costs/values (in some cases with sophisticated variants reflecting temporal and aggregation effects), but in a deployed system, we must face up to the problem: where are these values to come from? Who assigns them, and how are they to make the transition from a heuristic (but comprehensible) expression of commander's intent to the more arcane language of numerical valuation?

Some possible approaches are being pursued. First, we have funded a white paper on the topic, suggesting possible computational procedures, and suggesting promising avenues of further research. For example, the issues that arise in effects based targeting are clearly relevant to our problem also. In addition, some of the researchers have cast the problem in such a way that target value (and asset cost) is not required. For example, in a Markov approach, the commander need only specify the desired end state, and the algorithm can evaluate alternative courses of action in terms of their probability of reaching that state. We also mentioned another novel approach – the theory of Ordinal Games (see Section 3.2 above) which requires preferences as input, but not numerical ratings.

A second important obstacle concerns *model building, initialization, and adaptation*. A well-known Achilles Heel of decision support systems is the need to develop models tailored to the specifics of the situation at hand. The ability to automatically generate models (from historical data, or detailed simulations) would be very useful, and we are considering adding this to our "to do" list. We should also admit that, in candor, the level of fidelity for many of our researchers' models is low. The effort, at this stage, has been to understand the algorithms, and their strengths and weaknesses. As the program progresses, we must add additional complexity and fidelity to more faithfully emulate real world characteristics. And, eventually, we must step up to a "live" scenario, with all that that implies in terms of plant identification, noisy data, deception, and an active and hostile adversary.

A third obstacle, and opportunity, is *state estimation*. Even if the controllers being developed by our researchers are never used as "automatic pilots" for conducting an opertion, they do provide the ability to predict the evolution of the battlespace (by modeling the decision making process as well as battle dynamics), and hence permit tracking of predicted actions against what is observed. The ability to calculate the "residuals" is a powerful computational tool whose implications are yet to be fully appreciated. As the program proceeds, we expect an increased emphasis on using the control technologies in this way – verifying that the battlespace is evolving in accordance with predictions, or (alternatively) alerting the commander when observations begin to deviate significantly, and in a biased way, from nominal projections.

In conclusion, we are at the halfway point of an 18 month effort, and are just now preparing for the follow-on. Initial experimental results are very encouraging. The core ideas for several different controllers have been formulated and verified in a "laboratory" environment. Ahead lies the task of adding increased modeling fidelity, logistical constraints, and robustness in the face of noisy and incomplete state data. Efforts in these directions are already underway.

Historically, tools for command and control have been developed using approaches from the broader theory of Artificial Intelligence: rule-based systems, Bayesian nets, reasoning in the face of uncertainty, knowledge bases, etc. What the JFACC Program has done is to give experts in control theory a shot at this problem. Through a series of symposia, as well as through the funded research, the JFACC program is building a community with a shared vision and a rapidly evolving set of tools and techniques. We believe this investment will pay big dividends, eventually leading to a revolution – not only in the way military command and control is conceived and implemented, but in the way large enterprises in general are modeled and controlled. We are very excited about this work. It has scientific and mathematical rigor, it has already produced some major surprises and innovations, and all indications are that the best is yet to come. Stay tuned!