Model Predictive Risk Control of Military Operations

Jan Jelinek

Honeywell Laboratories,
3660 Technology Drive
Minneapolis, MN 55418
U.S.A.
jelinek_jan@htc.honeywell.com

Abstract. Work reported in this paper was done as part of the DARPA Joint Force Air Component Commander (JFACC) project. Its objective was to investigate the possibility of improving the stability and agility of military operations using the concepts of modern control and game theories within the framework of state-of-the-art computer science and operations research.

Military operations are always defined and executed within the context of a command and control (C2) hierarchy. The questions we have studied at the Task level were twofold. What is the minimal effective task force size and composition for a given task, and how to guarantee successful task execution in the presence of uncertainties in combat, both due to random effects of weapons and an intelligent adversary? At the Task Group level, we asked how to optimally allocate and schedule available resources to satisfy the force size requirements for as many concurrent tasks as possible. We have developed probabilistic Markov models of combat dynamics, and then used them to build the Model Predictive Task Commander and Model Predictive Resource Allocator systems, which are briefly described in the paper along with experimental results showing their performance in simulated battles.

1 Introduction

War or, on a smaller scale, any battle can be viewed as a trade in which opponents mutually "trade" their resources until one side is forced to declare bankruptcy due to its bad trading decisions. The traditional mathematical approach to optimize this process is to view it as a resource allocation problem, whose objective is to match resources to targets in the most "profitable" way as defined by a suitable criterion. Obviously, the fundamental question is how to properly value resources to be traded. Dollars spent to procure them do not make much sense in war since their military values derive largely from expected opportunity costs and not from the numbers listed in accountants' ledgers. The military value of a bomber is surely different at the outbreak of war than on the V day and on any day in between, and the difference depends strongly on the war strategy and many other factors.

In any model of war based on the notion of trade the issue of resource valuation cannot be eventually avoided. However, the responsibility for resource valuation should be assigned to those who can be expected to possess knowledge and information relevant for such decisions, which, in practice, means the higher rungs of the command and control (C2) hierarchy. If the resource value derives mostly from the opportunity costs, then it is neither fair nor reasonable to ask a field commander to decide, say, how many of his bombers are worth a given heavily defended enemy air base he was ordered to destroy. Yet he somehow has to compose his strike packages, devise their offensive and defensive capabilities, schedule their employment and then manage the battle to success once it gets underway.

In the project we are reporting on here, our objective was to investigate the potential of improving the stability and agility of military operations. We have focused mostly on the lower rungs of the C2 hierarchy. For the reasons outlined above, we have rejected the more traditional design concepts based on straightforward resource allocation. Instead, we proceed in two steps. For every task we first try to establish what it takes to get it done regardless of whether the resources are actually available. We call the answer the *minimal effective force* (MEF). Only then we attempt to allocate available resources to the set of given tasks in amounts sufficient for their successful completion. As it turns out, most tasks can be successfully completed in more than one way so that their MEF is actually a set of alternative solutions. This extra degree of freedom provides the planner with greater flexibility to reconcile competing demands. Moreover, the breakup of planning into the two phases improves the transparency of the planning and battle management algorithms, significantly simplifies their computer implementation and speeds up their execution. Below we explain the gist of our approach, show some experimental results and outline issues for further work.

2 Concept of operations

The technology underpinning our approach cannot be judged properly without some notion of concept of operations. The concept emerging from our current work is as follows.

Superficially, the proposed C2 structure remains hierarchical. In its functionality, however, there are substantial differences in the way information flows about the hierarchy. The traditionally strong top-down information flow is complemented by a comparably strong bottom-up flow, producing a structure, in which decisions are reached through a negotiation process, whose objective is to optimally match the war objectives at the top with the resources and operational constraints existing in the field at any given moment. Conceptually, we are trying to move away from directive- to more consensus-based command and control, in which different levels of the hierarchy are seen more like peers united by a shared interest in winning the war rather than disinterested subordinates asked to fulfill orders passed down to them.

The hierarchy, its levels and the kinds of information passed between them are shown in Figure 1. The Mission Execution Level has a place in our hierarchy, even though we have not addressed it in our work. We have focused on investigating the Task, Task Group and, to some degree, Operations Levels. We have not studied levels above the Operations Level.

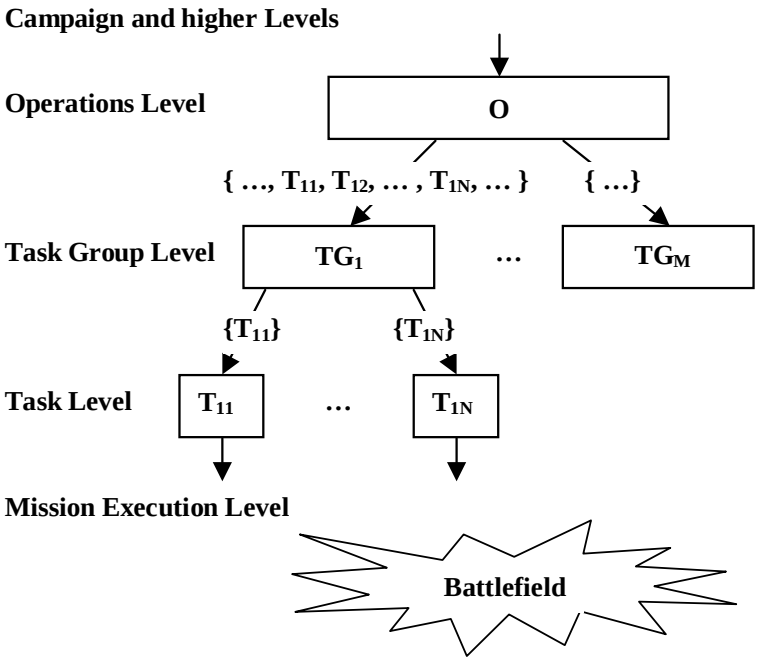

Figure 1. The Command and Control hierarchy

The basic notion in our approach is that of the *task*. Task is a relatively simple activity, whose objective is already stated in military terms (i.e., not strategic or political). It has a deadline, by which it has to be completed, and the issuer specifies the urgency of its successful completion by providing the desired probability of success. He also specifies what is the acceptable own loss to fulfill the task. This formulation reflects a realistic view of combat as always involving a significant random component, which is much better to be dealt with in the open than to pretend that it does not exist. The *battle* is then the process of executing a task. In this sense, task and battle are more or less synonymous words, one stressing more the objectives, the other the process of achieving them. Battles are generally viewed as sequences of simpler combat activities. The reader can interpret them as missions, sorties, etc.

The Task Level commander's role is in designing appropriate minimal effective forces (e.g., strike packages) for each mission of the given task and then making the necessary corrections to them prior to the next mission

depending on the *battle damage assessment* (BDA) feedback he receives. Executing the missions is the mission commander's responsibility and as we have pointed out above, is not the subject of our interest.

Any operation will likely consist of a large number of tasks running concurrently, with overlapping demands on resources. The Task Group Level commander, who is generally seen as the "owner" of resources, is responsible for apportioning his resources to the individual tasks that were assigned to his group for execution.

The Operations Level job is to translate the broader, strategic objectives (e.g., destroy enemy air base Alpha) into an interconnected web of individual tasks, which are then passed down to the Task Group Level commander for execution. The interface between the two levels is a queue of tasks $\{ \ldots, T_{11}, T_{12}, \ldots \}$, which is continuously filled by the former on one end and emptied by the latter on the other.

The reader may have noticed that in our concept of operations there is no notion of target and hence of target value. To be sure, targets are discussed, but only internally at the Operations Level, when tasks are being defined. In our C2 philosophy, the notions of targets and their values are not used in communication between levels below the Operations Level, because commanders working there lack the knowledge context to properly understand them.

A task exists since the moment the Operations Level pushes its statement into the task queue, well before any resources have been allocated to it. Indeed, once the Task Group commander retrieves it from the queue, his first job is not to allocate his resources to it, but to find out what is the minimal effective force to successfully prosecute it. To do so, he assigns the task a Task commander, effectively passing it down to the Task Level. This officer calculates the Minimal Effective Force needed to fulfill it. Since tasks can generally be fulfilled in more than one way, he compiles a list of alternative solutions and sends it back to the Task Group commander, who then attempts to choose the alternative that best utilizes his resources. If none of the alternatives can be reconciled with other already scheduled tasks, he can

1. either report back to the issuer, i.e., the Operations Level, or
2. can attempt to exchange resources with other Task Group commanders, or
3. can possibly define down or even drop some of the existing tasks, if the issuer has given him a specific permission to do so on his behalf.

Infeasibility to simultaneously provide for all the tasks currently in the task queue can trigger an extensive negotiating process running up and down the hierarchy, when the issuer can ask for sensitivity analysis of other tasks already in progress, their status and prospects. He can modify or drop some of them in order to make room for new ones. All those activities are conceptually supported in our system and, in fact, can happen automatically.

3 The nature of a task

As mentioned above, battle is the process of achieving a task's objective. Actually, we should use the plural, because the enemy has his objectives as well. This competitive process has its dynamics, which arise from the dynamic interaction of several components as depicted in Figure 2, and which each combatant tries to control in his favor. Prior to the first mission, each commander composes his package and sends it off to the battlefield. When the combat is over and survivors return to their bases, the commanders evaluate their battle damage assessment information and based on this feedback put together the second mission. Such iterations proceed until either one side succeeds in attaining its objectives or misses the deadline stated in its task specification and subsequently terminates the battle.

A control theorist readily recognizes in the description something that looks like a batch control problem. The Blue commander, who is the good guy whom we want to help, resembles a discrete controller which responds to the observed plant output by calculating a new control value to be applied next in order to bring the plant closer to meeting his objective. Admittedly, the control "value" is unusual and rather abstract, being represented by the composition, weapons, munitions and other attributes of the package sent into combat to drive the battle state in the desired direction. The BDA block represents sensors that map the state into the observable plant outputs, most likely with a lot of distortion, incompleteness, false readings and latency.

In addition to similarities, there are some important differences, too. As Figure 2 shows, Blue's links to the system go through the battlefield, which we view as a giant trading floor, where combatants trade their assets. In our approach, all deal-making is governed by probabilistic laws to reflect the random effects of weapons and other uncertainties of combat. Mathematically, we describe the trading which goes on on the battlefield as a Markov decision process, whose control variables are the package attributes. Such models addressing different

kinds of combat have been developed and studied [Clark 1969], [Ancker 1982], [Ancker & Gafarian 1992], [Jelinek 2001a], [Jelinek 2001b]. Interpreting the battlefield as the trading floor also readily offers an important generalization of the notion of the package, namely viewing it as a set of all the assets the combatants bring for trade. While some of them will undoubtedly be weapons systems, others may be passive assets like bridges, airfields, power stations, etc.

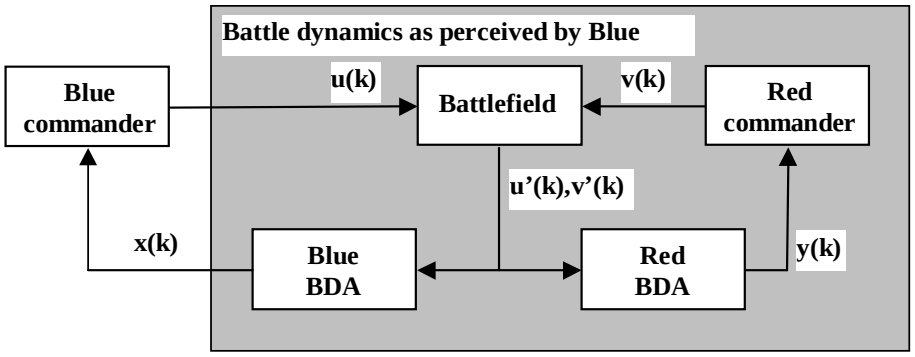

Figure 2. The internal structure of battle when viewed as a dynamic system

Unlike the mother Nature, which is an indifferent player in industrial applications, the Red commander has vested interest in the outcome of trading. Blue does not generally know what Red's interests exactly are, nor what strategy Red is going to pursue in advancing them. This brings in a different kind of uncertainty than the randomness of combat, one that must be addressed by the game-theoretic means. These problems have been intensively studied since the 1950's [Dresher 1961], [Basar & Olsder 1982].

Battle damage assessment poses yet another obstacle. In the world of smokescreens and deceit, even Blue's own BDA cannot be completely trusted. It is very difficult to find out, how much one actually does not know and somehow quantify this ignorance. For those reasons, we include the Blue's own BDA block into the battle dynamics model to stress the fact that the ground truth of the battlefield is not available to him for battle planning and management and that he can only see it through the lenses of his own BDA.

In order to rationally calculate the minimal effective force for a task, the Task Level commander has to model all the components enclosed in the gray box in Figure 2 along with their dynamic interaction. Compared to a control engineer facing a batch controller design problem, his plant-battle has not yet been "built" so his model cannot be, say, a neural network or some other regressor to be fitted to the plant using experimental data. Furthermore, in a typical operations center, hundreds of tasks are handled every day, so the ease and speed of model construction are paramount. Once built, the model will be run only once and then discarded.

The Task Group commander, whose job is to provide requested resources for tasks, deals only with their respective minimal effective forces. He is not interested in the details of how their they were calculated nor how the tasks' execution will be managed, and thus has no need to know the battle models. As it will become clear shortly, the minimal effective force specification is, in fact, the specification of a set of controlled random processes. It includes not only the immediate resource allocation request for the next mission, but, more importantly, provides a forecast of the expected battle evolution, including expected losses and resource demands for all future missions all the way to the task completion deadline. The challenge we are now facing is how to build a planner and scheduler that would take full advantage of this information to maximize the resource utilization and minimize disruptive plan and schedule modifications. Put in more mathematical terms, we are interested in the planning and scheduling of activities, which are characterized by sets of partially observable Markov decision processes operating over finite horizons [Sondik 1971], [Monahan 1982] or, in the most general form, partially observable competitive Markov processes [Filar & Vrieze 1997].

4 Minimal effective force

We have seen that each of the four components of the battle model in Figure 2 introduces uncertainties so fundamental that they cannot be ignored in the minimal effective force calculations. Uncertainty entails risk, which can be reduced by appropriate design, but never completely eliminated for very practical reasons: The cost of risk reduction tends to progressively escalate the faster, the closer we are approaching certainty, until at some point a low risk solution becomes unaffordable. We are thus caught between two conflicting interests. On one side, the task issuer wants to minimize the risk of task failure, because he will have to live with its consequences. On the other side, the task executor wants to lower the demand on his resources, which always seem in short supply, and is thus interested in relaxing the risk threshold. Furthermore, uncertainty in combat is not a constant, but varies in time as new information becomes available and effects of earlier decisions make an impact on the battlefield. This time variability requires continuous reevaluation of risk and making adjustments to the deployed forces, if the acceptable risk level is to be maintained with the minimal amount of resources. Having adopted this perspective as the central theme of our approach, we do not consider the notion of risk management to be just a metaphor. On the contrary, we view the C2 hierarchy as a hierarchical control system, which is to be explicitly designed to manage (control) risks arising from the uncertainties present at its different levels.

We define the risk ρ as the probability of task failure. Alternatively, we may be using the probability of success P_s, which is related to risk as $P_s = (1 - \rho)$. Let $u(k)$, $v(k)$ be vectors, whose components are the attributes that characterize the packages employed in the k-th mission by the Blue and Red commanders, respectively. The attributes may be, for example, the numbers of bombers and escort fighters in a strike package, their weapon lethalities against the opponent's assets, combat tactics to be used, etc. Recall that not all assets brought for trade by the combatants are necessarily weapon systems. In the course of combat, some of the attributes are traded for others so that when the fighting is over, the status of the surviving packages will be $u'(k)$, $v'(k)$. Due to the randomness inherent in combat, the particular values of $u'(k)$, $v'(k)$ are impossible to predict. The best we can hope for is to find their probability distribution $P\{u'(k), v'(k)\}$. It can be computed providing that we know the conditional probability distribution $P\{u'(k), v'(k) \mid u(k), v(k)\}$ which we call the *combat model*. This model, in contrast to the full model of battle dynamics, describes only the block named Battlefield in Figure 2. Assuming that the vectors $u(k)$, $v(k)$ take on only a finite number of discrete values, their sets $\mathcal{U}(k)$, $\mathcal{V}(k)$ are also finite and the combat model $P\{u'(k), v'(k) \mid u(k), v(k)\}$ can be viewed as the set of transition probabilities of a Markov chain.

The vectors $u'(k)$, $v'(k)$ generally are not directly available to the commanders for observation in their entirety. While the commanders usually get a good grasp of own losses, their information concerning their opponent's losses often is at best incomplete and at worst wrong. In our approach, we capture their less-than-perfect BDA capabilities by a pair of conditional distributions $P\{x(k) \mid u'(k), v'(k)\}$ and $P\{y(k) \mid u'(k), v'(k)\}$, which relate, although only in the probabilistic sense, the battle state $\{u'(k), v'(k)\}$ to the vectors $x(k)$ and $y(k)$, which represent the observables available to the Blue and Red commanders after the k-th mission, respectively.

Let $\mathcal{X}(k)$ be the sets of all possible vectors $x(k)$. Using Blue's task objective, we identify the subsets $\mathcal{X}_S(k) \subset \mathcal{X}(k)$, $\mathcal{X}_F(k) \subset \mathcal{X}(k)$ that Blue considers task success for himself and Red (i.e., Red's success is Blue's failure), respectively. If the battle reaches any one of those states, he terminates it. All other vectors are clearly indecisive situations, for which the battle continues with the $(k + 1)$-st mission unless Blue has reached his task deadline, in which case he declares all such indecisive vectors $x(k)$ a success for Red (i.e., failure for Blue). Now it is easy to calculate the probability of Blue's success in the k-th mission

$$P\{x(k) \in \mathcal{X}_S(k) \mid u(k), v(k)) = \sum_{\mathcal{X}_S} \sum_{\mathcal{U}\times\mathcal{V}} P\{x(k) \mid u'(k), v'(k)\}\, P\{u'(k), v'(k) \mid u(k), v(k)\} \quad \textbf{(1)}$$

where we assume that the set $\mathcal{U} \times \mathcal{V}$ of all admissible values of the pairs $\{u'(k), v'(k)\}$ is finite. Let $J(n \cdots m)$ denote the probability of success in any of the missions between n and m

$$J(n \cdots m) = \sum_{i=n}^{m} P\{x(i) \in \mathcal{X}_S(i) \mid u(i), v(i)) \quad \textbf{(2)}$$

Then the minimum effective force $\{u^*(k), \ldots, u^*(K)\}$ calculated prior to the k-th mission is

$$\{u^*(k),\ \ldots,\ u^*(K),\ v^*(k),\ \ldots,\ v^*(K)\} = \arg \min_{\mathcal{U}(k)} \max_{\mathcal{V}(k)} J(k \cdots K)$$

so that

$$J(1 \cdots (k-1)) + J(k \cdots K) \geq P_S \qquad (3)$$

where P_S is the desired probability of task success. For the sake of notational clarity, the obvious assumptions regarding the nonnegativity of attributes, etc. are omitted.

Note that the optimization is always done over the entire remaining task horizon. As a result, the minimum effective force specifies not only the package to be immediately employed in the upcoming *k*-th mission, but provides the estimates of all future packages all the way up to the task deadline. As will be shown in the next section, the estimates are continuously updated after each mission. Numerous experiments suggest that if Blue's battle model is reasonably accurate, total upsets of his expectations are rare. In most cases, the updates will be only modest corrections of the previous estimates, which enables the stable operation of forward-looking resource allocation planning and scheduling algorithms supporting the tasks at the Task Group Level. Second, as a side benefit of this game-theoretic optimization, Blue also develops his best estimate, $\{v^*(k),\ \ldots,\ v^*(K)\}$, of how Red is going to conduct the battle in the future given his current knowledge of Red's constraints. Although the problem statement (3) implies an assumption that the Red's sole objective is to prevent Blue from reaching his objective, which may be unrealistic in individual cases (and which turns our problem into a neat zero-sum game), other, more general game-theoretic formulations preserve this useful feature.

Since the problem statement (3) does not restrict what are the acceptable attribute values in the minimum effective force $\{u^*(k),\ \ldots,\ u^*(K)\}$, we call it the *Victory-at-Any-Cost* formulation. Most tasks, however, limit the amount of resources the Task Level commander is allowed to sacrifice. This extra constraint appears in the following *Victory-With-Acceptable-Loss* formulation [Jelinek & Godbole 2000].

$$\{u^*(k),\ \ldots,\ u^*(K),\ v^*(k),\ \ldots,\ v^*(K)\} = \arg \min_{\mathcal{U}(k)} \max_{\mathcal{V}(k)} J(k \cdots K)$$

so that

$$J(1 \cdots (k-1)) + J(k \cdots K) \geq P_S \qquad (4)$$

$$u_i^*(1) - u_i^*(K) \leq l_i, \qquad i \in \mathcal{I}$$

where $\mathcal{I}$ is the set of attributes whose change between the first and last mission we want to limit so as not to exceed the given threshold l_i.

5 Model predictive risk control

Unless the battle terminated after the (*k-1)*-th mission, the commanders have to determine the package composition for the *k*-th mission. How the Red commander actually makes his decisions is rarely known to the Blue commander. Blue may know Red's doctrine and rules of engagement, and often has intelligence assessing Red's resources. If Blue is proactive and holds the initiative, he may reduce most of the Red's objectives to attempts at stopping him. Calling the games affords Blue an additional foresight into what to expect of his opponent. Furthermore, it is reasonable to expect the Red commander to be rational. All this constrains Red's decision space and enables Blue to produce a qualified estimate of Red's likely decisions when calculating the minimal effective force.

The above constraints are known a priori and are not tied to any particular task. In addition to them, however, Blue also receives real time BDA feedback from the battlefield in the form of the observation vector $x(k)$. This additional information allows him to either strengthen some of them or add new ones that reflect the customized, up-to-date knowledge about the task being prosecuted. When translated into the mathematics of the minimum effective force calculations (3) or (4), the feedback loop is closed by making the sets of admissible attribute values $\mathcal{U}(k), \mathcal{V}(k)$ dependent on $x(k)$. Since combat results in asset attrition, these sets generally shrink with each mission, reducing the combatants' options in the process.

Using the model predictive control paradigm as a framework we have integrated the minimum effective force and real time feedback concepts into a system which we call the Model Predictive Task Commander (MPTC) and

whose purpose is to assist the Task Level Commander in the planning and execution of individual tasks. The next section offers a simple illustration of how the MPTC works.

6 Example

The task specification may look as follows.

At 0600 the commander of the Blue Task Group TG_1 is given the order to destroy Red's surface-to-air missile (SAM) assets made up of L_R real sites and D_R decoy by 1800 tomorrow. Because the objective is needed to clear the way for an already planned subsequent offensive, the Operations Level requests the order be executed with a very high degree of certainty, say less than 1 in 20 chances that it will not be met in full. The Red's SAMs are known to have the lethality λ_R against the attacking aircraft that Blue is intending to use. They also have a good radar tracking capability to know the accurate numbers and positions of attackers in real time.

The MPTC helps answer the following questions:

- How many airplanes, L_B, does Blue need in his strike package, if his kill rate on the Red SAM's is known to be λ_B ?
- How many missions (sorties), K, he should divide his objective into, one, two, or perhaps ten?
- If he decides to fly more missions, how should he define their individual objectives, against which he could measure the task's progress once it gets underway? Without them, he would not be able to identify looming problems until it may be too late for any correction.
- If he decides to fly more missions, how should he optimally assemble the strike packages for each one? On one side, gradual enemy attrition will lower the threat, but he will have his losses as well. How big? What is the total number of aircraft he should ask to be allocated for the task?
- If, for whatever reason, the task execution does not proceed as planned, what corrective action to take?

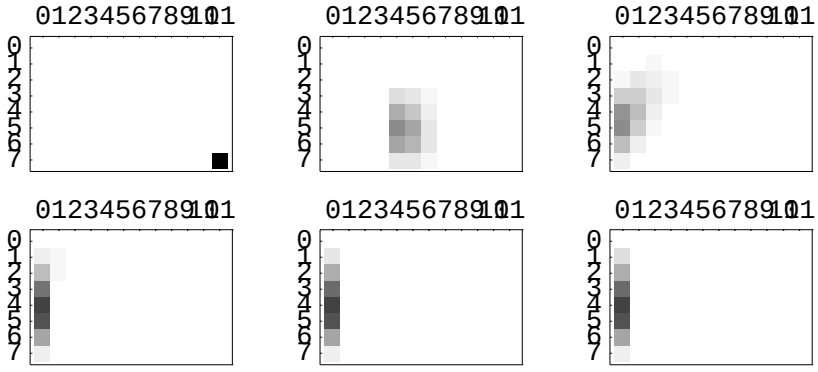

Figure 3. Probability distributions of the battle state $\{u'(k), v'(k)\}$ forecast how battle is going to evolve over the next 5 missions if Blue applies the minimum effective force calculated prior to the first mission. The first column are the win states for Blue.

To be specific, let us say that Blue's intel tells him that Red has $L_R = 11$ sites and $D_R = 1$ decoys. His SAMs are known to have the lethality against Blue aircraft $\lambda = 0.2$. On the other hand, Blue's weapons system data tell him that he can expect to kill this particular type of SAMs with the lethality $\lambda_B = 0.9$. Blue assumes that, when attacked, the Red commander will always employ all his surviving SAM sites to defend himself (which is actually the best policy for him in the game-theoretic sense). Because the mission turnover time is 6 hours due to the target distance, Blue can fly at most 6 missions before hitting the deadline. When Blue calculates his minimum effective force for $P_S = 0.95$ using the data, he is advised to employ 7 strikers for the first mission and then fight the remaining ones with survivors only (which is also the best policy for him). The MPTC also tells him that he can expect to lose slightly less than 3 aircraft on average in this job. The expected course of battle is shown in

Figure 3. The rows and columns in the frames correspond to the numbers of Blue and Red units that are alive, and plot densities are proportional to the state probabilities.

The top left frame is the initial state of battle, when Blue knows with probability 1 the initial numbers ($L_B = 7$, $L_R = 11$). The outcome of the first mission shown in the next frame is not that unequivocal anymore. The most likely number of survivors will be ($L_B = 5$, $L_R = 4$), but other outcomes still rather close to those numbers are possible. As the battle progresses (read Figure 3 row-wise), the cluster spreads more and more until it eventually splits into two, i.e., the distribution becomes bimodal. (This is not well visible in the figure.) The last frame in the bottom right corner says that Blue is most likely to win with 3 to 5 survivors, but battles with 2 or 6 survivors are a fairly likely outcome as well.

The MPTC repeats the same minimum force calculations after each mission upon receiving BDA feedback. The plots of their distributions would be similar except that the initial distribution, that is, the first frame in Figure 3, would be replaced by the latest battle state estimate and the planning horizon gets shorter with each new mission. The closed loop behavior is also a random process, whose distributions - of which we do not have their analytic forms to readily obtain their density plots - would not be much different from the open loop distributions shown in Figure 3.

If we performed multiple Monte Carlo simulations of the task execution under MPTC control, then each run would produce a new realization of the closed loop random process. A couple of runs shown in Figure 4 invokes the feeling for their variability. What matters, though, is that in spite of their vastly different appearance, Blue's MPTC will drive over 95% of all runs to success as requested by the task statement regardless of good or bad luck, which is always a factor in combat. As can be seen in the plot on the left, here the MPTC decided after the first and second missions that this battle was progressing better than expected and withdrew at first 3 and then 1 more survivors from action. In the battle on the right, Blue was down on luck and lost a lot more than expected in the first mission. Subsequently, the MPTC called for 2 additional airplanes to be added to the survivors to fly the second mission. Both battles shown happen to terminate after the third mission, but this is a coincidence. Battles can and do terminate anytime between 1 and 6 missions.

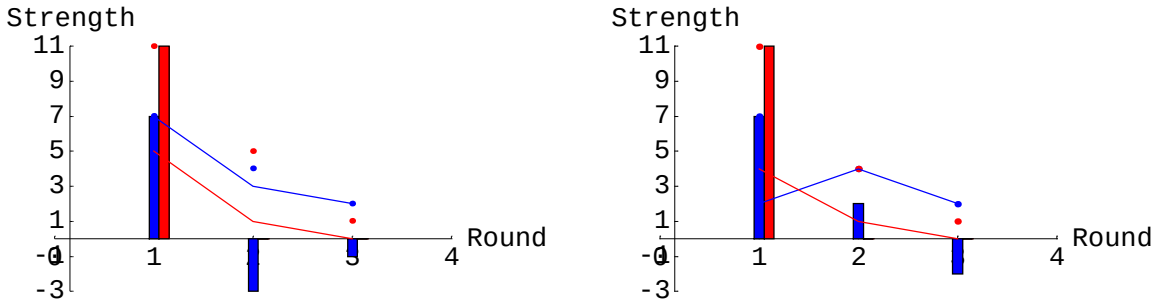

Figure 4. Monte Carlo simulation of two battles. Bars, dots and solid lines represent deployment increments/decrements, actually deployed and surviving units in each mission, respectively. Lacking color, the plots lose some of their explanatory power. The left and right bars in the first round (= mission) are Blue and Red, all bars in rounds ≥ 2 are Blue. The segmented line, which ends at the x axis is Red, because Red gets wiped out.

The true value of robust feedback control can be best appreciated when Blue does not get his battle model quite right. With so many unknowns to consider, such situations are very likely in the real world. Figure 5 shows on the left the average of 100 randomly generated battles managed by the MPTC, when Blue's model was right. On the right, the user underestimated the Red SAM's lethality by 50%, i.e., instead of entering $\lambda_R = 0.2$, he entered $\lambda_R = 0.1$. We see that the battles are generally longer and average withdrawals per mission are smaller. The simulation statistics for the perfect model example show that 99 battles were won, none lost in fight and 1 was lost by not being completed by the deadline. For the mismatched model, 93 battles were won, 1 lost on the battlefield and 6 were lost by not being completed in time. Although these numbers pertain only to the random samples of 100 battles, they are indicative of general results. The potential improvement the MPTC offers

becomes clear once we calculate that in the model mismatched case 17 battles out of 100 would have been lost on average without it.

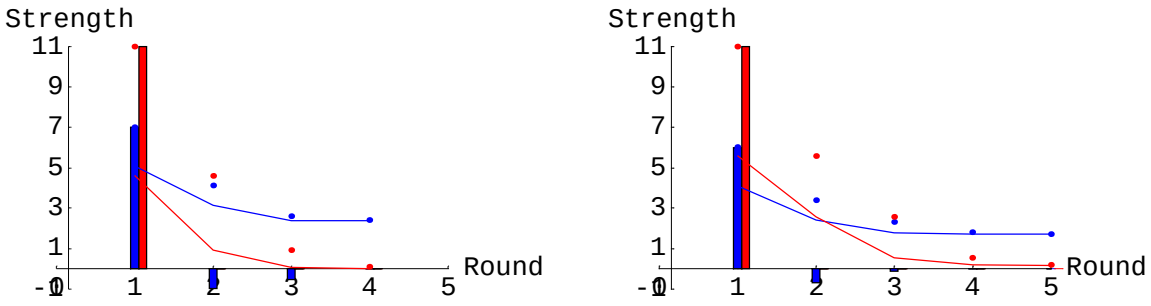

Figure 5. The expected evolution of battle computed as the average of randomly generated 100 battles for the perfect (left) and mismatched (right) models

7 Designing Package Defenses

A package can be viewed as an abstract weapon system that the Task Level commander custom designs and, with help from the Task Group Level commander, then "manufactures" for each mission to effect the desired change in the battlefield. As any other weapon, this one also has its offensive and defensive capabilities, which are characterized by the package attributes. While designing the offensive capabilities is, at least conceptually, straightforward, the design of defensive capabilities often is more difficult. As long as defensive components are an explicit part of a package, we use the formulation (4). However, many factors contributing to the package defense are intangibles that are very hard to model and quantify. The following concept offers a possible solution [Jelinek 2000].

Improving defense against enemy weapons, in effect, means lowering his weapons' lethality against own weapons. In other words, in this view Red's lethality λ_R should not be treated as a constant, but as a variable that Blue can actively manipulate in his favor. For example, flying a mission at night against the enemy whose fighters cannot fly by instruments would greatly lower his fighters' lethality against intruding Blue bombers. Likewise, adding a jammer aircraft to a package will lower the SAM's lethality by disabling their radar tracking. Just the threat of Wild Weasels leading a group of bombers might suffice to convince Red not to turn on his SAM radars at all, thus effectively lowering their lethality as well.

Let us assume that Blue can indeed manipulate the lethality of Red weapons and concern ourselves with the following question: How much down has Blue to drive the Red's lethality λ_R in order to keep his losses below a given threshold l_i? Once the MPTC suggests the needed lethality reduction, the Task Commander uses his military expertise to interpret it in terms of particular defensive options and suggest package defenses which are best suited for the task at hand. Mathematically, the solution is obtained by modifying the optimization problem (4) so that the minimization is carried out not only over the original set $\mathcal{U}(k)$, but also over the lethalities of Red weapons, which are now added to the Blue's decision variables.

Consider an example, in which Blue is tasked to destroy a bridge deep in the Red territory. Blue knows that on their way to the target, his bombers are likely to be intercepted by L_R Red fighters whose lethality $\lambda_R = 0.5$. The task specifies that his loss of bombers must not exceed 5 airplanes. How much must he lower λ_R to conform with the order? Figure 6 shows the results of the minimum effective force calculations for the acceptable maximum loss values ranging from 0 (the curve on the left closely following the y axis) to 9 (the last curve on the right, which is the lower envelope of the family). The thick curve provides the desired answer for maxLoss = 5. The example nicely demonstrates that our optimization problem does not have a unique solution as the needed number, nLB, of Blue bombers depends on their protection level that Blue is willing to provide. Any point of the thick curve meets the task objective, but offers a different offense-to-defense ratio for the package. Another fact worth pointing out is that trading off offense for defense has its limits. Even an unlimited number of bombers in the package cannot guarantee that their loss will not exceed 5 airplanes unless Blue defends them enough to drive the Red lethality (= pLR) down from 0.5 to about 0.38.

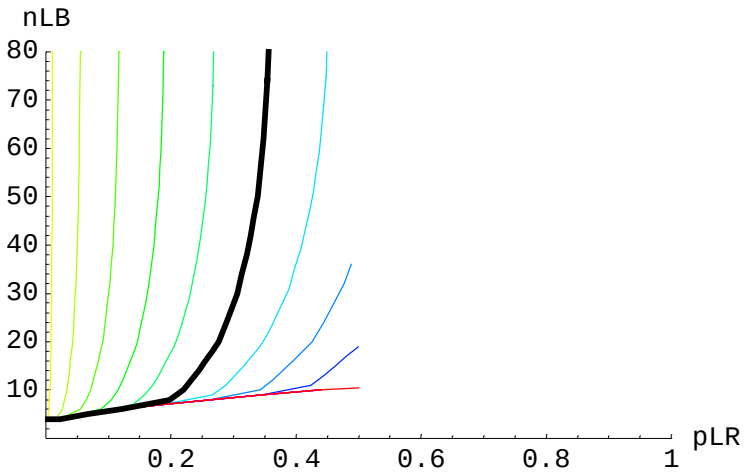

Figure 6. Many minimum effective force calculations offer multiple solutions. Any point of the thick curve meets the task objective, but offers a different offense-to-defense ratio.

8 Resource Allocation and Task Scheduling

Managing a set of tasks, which compete for shared resources, brings up new problems. The Model Predictive Resource Allocator (MPRA) proposed in [Tierno 2000] aims to achieve, whenever possible, the desired probability of success for all given tasks, while minimizing combat exposure of assets. It uses the market oriented programming to reconcile the competing demands. This is trivial as long as there is enough resources to satisfy all of the tasks' demands. It becomes a very difficult problem at the moment when the demands are irreconcilable and the MPRA is expected to provide some meaningful advice to the Task Group commander as to which tasks would be hurt the least if deprived of some of their resources, what effect such step would have on their probability of success, losses, etc. The MPRA may also be asked to advise which tasks can be redefined down without causing much harm or dropped altogether.

An appealing feature of this approach is the "currency" that bidders use in their attempts to acquire resources, which is based on the probability of success computed for each task by its MPTC. This injects some degree of objectivity into the way the algorithm works and makes its behavior easier to control and understand.

Mathematically, resource allocation is a packing problem. Such problems are notoriously hard to solve numerically. In real world applications, the Task Group commander has to handle tens or even hundreds of tasks simultaneously. This eliminates many potential algorithms, which simply cannot scale up to this problem size. For those reasons, work reported on in [Deshpande et al. 2001] investigated the use of greedy search and genetic algorithms to find only approximate solutions but in times more likely to be considered "real".

The above work concerned resource allocation done in a fixed time instant. There is no notion of the future in the resource allocation algorithms that were studied. Although the minimum effective force of each task offers an glimpse into its likely future, this information was ignored.

So far our team has not addressed the task scheduling problem at all. In their breakdown into individual, sequentially executed missions tasks do involve the notion of time, but this is only the "mission", not physical time. The job of scheduling is to map this mission time onto the physical time axis, and stack up the missions there so that they can be actually executed in the real world. We are addressing those issues in our current project.

Acknowledgements

This document is based upon work supported by the DARPA JFACC program through SPAWAR Systems Center San Diego under contract no. N66001-99-C-8510. Any opinions, findings, conclusions or recommendations expressed in this document are those of the authors and do not necessarily reflect the views of DARPA and SPAWAR Systems Center.

References

Ancker, C.J., Jr.: "One-on-One Stochastic Duels", Operations Research Society of America, 1982

Ancker, C.J., Jr. and Gafarian, A.V.: "Modern Combat Models", Operations Research Society of America, 1992

Basar, T. and Olsder, G.J.: *Dynamic Noncooperative Game Theory*, 2nd edition, SIAM 1999

Clark, G.M.: The Combat Analysis Model, Ph.D. Dissertation, Ohio State University 1969

Dresher, M.: *The Mathematics of Games of Strategy*, Dover Publications 1981. Originally published in 1961 by Prentice Hall, Inc.

Deshpande, R., Tierno, J., Parthasarathy, S.: Resource Assignment Across Multiple Tasks, DARPA JFACC Project Report, Honeywell Laboratories, September 2001

Filar, J. and Vrieze, K.: *Competitive Markov Decision Processes*, Springer 1997

Jelinek, J. and Godbole, D.: Model Predictive Control of Military Operations, Proc. 39th IEEE CDC Conf., Sydney, December 2000

Jelinek, J.: Winning Battles Without Losses, DARPA JFACC Project Report, Honeywell Laboratories, June 2000

Jelinek, J.: Predictive Models of Battle Dynamics, Proc. SPIE AeroSense Conf., Orlando, FL, April 2001

Jelinek, J.: Modeling Battles With Asynchronous Firing, DARPA JFACC Project Report, Honeywell Laboratories, September 2001 (a shorter version will appear in 2002 SPIE AeroSense Conf. proceedings)

Monahan, G.E.: A Survey of Partially Observable Markov Decision Processes: Theory, Models, and Algorithms, Management Science, Vol. 28, No.1, 1982

Sondik, E.J.: The Optimal Control of Partially Observable Markov Processes, Ph.D. Dissertation, Dept. of Electrical Engineering, Stanford University, May 1971

Tierno, J.: Distributed Multi-Model Predictive Control, DARPA JFACC Project Report, Honeywell Laboratories, June 2000