Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

Optimization-based Decision Support Algorithms for a Team-in-the-Loop Planning Experiment*

Huy Bui, Xu Han, Suvasri Mandal, and
Krishna R. Pattipati, Fellow IEEE
Dept. of Electrical and Computer Engineering
University of Connecticut
Storrs, CT 06269, USA
E-mail: krishna@engr.uconn.edu

David L. Kleinman,Fellow IEEE
Dept. of Information Sciences
Naval Postgraduate School
Monterey, CA 93943, USA
E-mail: dlkleinm@nps.edu

***Abstract*—Asset assignment and scheduling algorithms were developed and implemented to support a team-in-the-loop planning experiment conducted at the Naval Postgraduate School (NPS) in March 2009. The experiment examined planning and information flows among three cells in an abstracted and simplified Maritime Operations Center (MOC). This paper describes two optimization-based modules that focused on the Future Operations (FOPS) cell's planning activities. Module 1, a *FOPS Planning Module*, was a decision aid that presented the planners with *N*-best asset packages that would meet individual task requirements, while maximizing task execution accuracy. Module 2, *a Scheduling Module*, was an optimization-based scheduling algorithm that was used by experiment designers to set the conditions for the mission planning activity (e.g., asset types and numbers, task requirements and asset capabilities), and to assure that the tasks presented to the human planners would be achievable to a specified level of accuracy. A third module, termed Current Operations (COPS) Risk Analysis module, not discussed in detail here, was also implemented to assist COPS players on the consequences of redirecting assets from an ongoing task.**

***Keywords*— mission planning, scheduling, decision aids, *N*-best assignments, mixed integer programming, asset packages**

I. INTRODUCTION

A. Motivation

For over 15 years, the University of Connecticut (UConn) and the Naval Postgraduate School (NPS) have collaborated in an Office of Naval Research (ONR) supported program that has integrated analytical modeling and empirical research with human teams to design and evaluate alternative organizational structures and processes for Naval Command and Control (C2). The ways in which the integration is instantiated within this Adaptive Architectures for C2 (A2C2) program are three-fold: (1) Models are used prior to conducting an experiment to formulate hypotheses and guide selection of (independent and dependent) variables for the subsequent empirical study, (2) Models are used within information and decision aids that are presented to human players in the actual experiments, and (3) Models are used post-experiment to compare actual data with model predictions, and also provide a structure with which to interpret the empirical results.

Previous A2C2 research has had a strong tactical focus, that is "who does what", "who owns what (assets)", "who sees what", "who coordinates with whom", etc. In these experiments, human decision-makers (DMs) were involved in mission execution – moving assets, gathering information, attacking targets, etc. – using a real-time team-in-the-loop networked, distributed, software system called the Distributed Dynamic Decision-making (DDD) simulator [10]. Current A2C2 research has begun to shift its focus to the Maritime Headquarters with Maritime Operations Center (MHQ/MOC), as the Navy attempts to standardize operational processes and structures across Fleets and operation centers, and to be consistent with Joint operations. A MOC consists of 100's of people organized across a number of boards, centers and cells. At the operational (vs. tactical) level, MOC emphasizes standardized processes and methods, centralized assessment and guidance, networked distributed planning capabilities, and decentralized execution for assessing, planning and executing missions across a range of military operations. The assessment is a continuous process, combining the monitored outcomes of mission execution and the analyzed effects of operations (diplomatic, information, military or economic), with the situational awareness to inform future development of plans, to prioritize intelligence, surveillance and reconnaissance (ISR) activities and to allocate forces. The planning process is informed by guidance from higher headquarters and the assessment process, and is highly collaborative vertically with higher headquarters at the strategic level and with subordinates at the tactical level, and horizontally with other MOCs and Joint components. The maritime planning processes focus on the desired objectives and operational effects required by higher headquarters' guidance. Time scales, for planning, are generally 24 hours and beyond. A MOC does not "own" any forces, but rather gives directives to subordinate commands and forces. Important to effective execution is operational environment awareness, horizontal and vertical integration with other commands, and continuous assessment. A major challenge now facing the A2C2 research team is how to distill relevant MOC – oriented issues in such a manner that they can be studied empirically within a controlled research laboratory environment.

* This work was supported by the Office of Naval Research under contract # N00014-09-1-0062

This paper describes the model-based components of the first MOC–oriented human team experiment (MOC-1) that was conducted at NPS in March 2009. The experiment was designed to examine alternative structures and processes for the coordination among three key cells: Future Operations (FOPS), Current Operations (COPS), and Intelligence Surveillance and Reconnaissance (ISR) necessary to operationalize a plan generated by a Future Plans Cell. The FOPS cell drove this experiment – its goal was to develop the "plan" for allocating assets across a number of interdependent future tasks. The ISR and COPS cells supported this activity by obtaining and providing relevant task information needed by FOPS to construct the plan. For details of the experiment, see [11].

B. Related Research

The problem of allocating assets to tasks, in its simplified form, is a well-known assignment problem introduced in 1955 by Kuhn [8]. When there are a large number of assets available, it is related to the *Cutting Stock problem* [3]. There exist many optimization tools that provide a solution to such allocation problems (e.g. Lpsolver, ILOG CPLEX, LINGO, etc.). However, in some cases involving mixed initiative decision support systems, it may be useful to present the second, third and fourth best assignments (i.e., the N^{th}-best assignment after excluding the first *(N-1)* minimal cost assignments) to human decision makers. This concept has been popularized by Murty [1] for determining a ranked set of solutions to the assignment problem, and was enhanced by Cox et al [2] and Popp et al. [6] in the context of multi-target tracking. It is also applied to inference problems arising in multiple fault diagnosis with unreliable tests [4].

Scheduling problems are generally NP-hard. These problems can be approached via a set of well-known algorithms: branch-and-bound, dynamic programming (DP), dynamic list scheduling (DLS), and pair-wise exchange (PWE) [5], [7]. The benefit of using DLS and PWE is that their computational complexity is polynomial in the number of assets, tasks and related constraints (such as the number of precedence arcs in the task precedence graph and dimension of resource requirement/capability vector). The DLS method obtains fast near-optimal schedules, which can be further improved by the PWE algorithm. Another advantage of DLS is that it provides a dynamic scheduling procedure and accounts for on-line changes in the mission and/or asset capabilities. There are two main steps in the DLS algorithm: (1) Select the task to be processed, and (2) Select the group of assets to be allocated to the selected task. The process of task selection is based on task priority coefficients, which can be determined with one of the following three algorithms: critical path, level assignment, and weighted length. Based on the simulation results in [7], DLS provides a fast near-optimal schedule. If assets are shared among multiple geographic areas, schedules for combined areas can be constructed by using Benders' decomposition [3], [9] and Dantzig-Wolfe [3] decomposition.

C. Organization of the Paper

The paper is organized as follows. Section II describes the overall mission scenario that was used for the study, and defines the basic mission parameters (i.e., tasks, assets, task-resource requirement and asset-resource capability vectors, etc.). The *FOPS Planning Module,* and its embedded algorithm that generates *N*-best options for task accomplishment from a selected subset of available assets, are discussed in section III. Section IV discusses the *Scheduling Module* and algorithm that was developed for the pre-experiment activity to help select task and asset parameter values, to adjust numbers and type of assets, and to assure that the planning task could be accomplished such that no task would have accuracy less than a specified threshold (e.g., 75%). Algorithm performance and simulation results are given in section V. The paper concludes with a summary in section VI.

II. Overall Mission Scenario

In the MOC-1 Experiment at NPS, there were two FOPS planners, one for geographical area A and one for area B. Their goal was to build an effective plan by allocating a given set of 43 assets to a given set of 23 tasks distributed over those two areas. For each area, the planner was given a mission task graph, a worksheet with Gantt chart**,** and two tables similar to Tables 1 and 2 (below). Table 1 shows names of asset types, how many available assets of each type (e.g. 1 CVN, 2 CCG, 4 P3, etc.), and their resource capability vectors. Table 2 contains task information, such as task names and their resource requirement vectors. A task is deemed successfully accomplished if all its resource requirements are satisfied by the resource capabilities of allocated assets. For example, to successfully attack MSL bases with 100% accuracy requires 2 units of C2, 7 units of STRK, 3 units of ISR(G), and 7 units of BDA. All tasks must be completed in 30 hours. In addition, commander's guidance was provided to FOPS players to help them prioritize in their overall mission planning activity.

Similar to real-world situations, there may not be adequate assets to complete all tasks with 100% accuracy. One useful objective is to maximize the mission's average accuracy. Most assets were preallocated to either area A or B; but some assets were shareable between the two areas. The following are some asset constraints. An asset can only be assigned to one task at any given time. Furthermore, assets once committed to a task remain with that task until the task is finished. Since the number of asset combinations and task sequences is very large, FOPS players were provided a decision aid to rapidly provide *N*-best asset package options for each task and evaluate the achievable accuracy. The parameter *N* was set to 4.

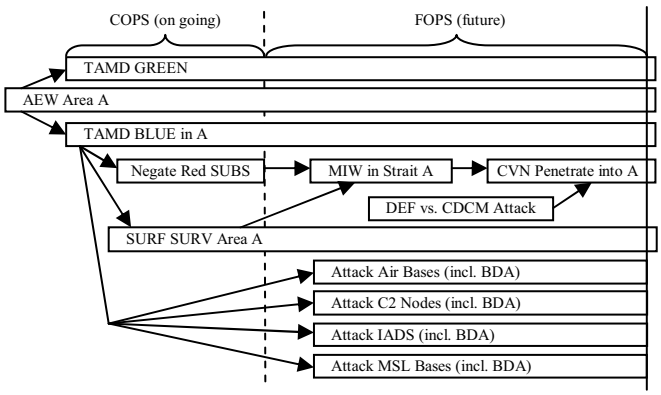

Figure 1: Mission task graph for area A

Table I. Gantt chart for Area A

Area A		FOPS Area A Plan for 30 hours														
Task Description	**Time Req'd for Task (hours)**	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30
Sample Task	14 H	CG_1, DDG_1, UAV_2														
AEW Area A	30 H															
TAMD GREEN	30 H															
TAMD BLUE in A	30 H															
SURF SURV Area A	30 H															
MIW in Strait A	10 H															
CSG Penetrate into A	4 H															
DEF vs. CDCM Attack	24 H															
Attack Air Bases	24 H															
Attack C2 Nodes	30 H															
Attack IADS	18 H															
Attack MSL Bases	30 H															

Each bold box indicates the duration of each task. Write the asset ID numbers assigned to each task within the bold box.

Table II. An example of asset resource capabilities (Area A)

ASSET	num	C2	STRK	AW	BMD	CMD	SUW	USW	MIW	ISR(A)	ISR(S)	ISR(G)	BDA
CVN	1	5	6	5		2	5	2	1	5	5	2	5
CG	2	3	5	8	7	6	4	3	3	7	5		
DDG	3	2	5	8	7	6	4	3	3	6	4		
SSN	3		3				5	4	2	1	3	1	
P3	4	1					6	2			6		3
MH53	1								4				
AWACS	2	5		5						8	3	1	
JSTAR	1	3									1	6	3
U2	1									3	2	4	3
RJ	1	2								4	3	3	2
UAV	3										1	5	5
CAP	1			7									
AEF	2	5	7	5			1			4	1	3	3

Table III. An example of task resource requirements (Area A)

ASSET	C2	STRK	AW	BMD	CMD	SUW	USW	MIW	ISR(A)	ISR(S)	ISR(G)	BDA
AEW Area A	5		5						5			
TAMD GREEN	5		12	14	10				12		4	
TAMD BLUE in A	3		8	8	7				6		4	
SURF SURV Area A	2									7		
MIW in Strait A	2							5		4		
CSG Penetrate into A	5		10		8	10	6		5	5		
DEF vs. CDCM Attack	2				8					5		
Attack Air Bases	2	5	5								2	3
Attack C2 Nodes	2	6									3	5
Attack IADS	2	8	5								2	5
Attack MSL Bases	2	7									5	7

Table IV. An example of risks associated with releasing ISR assets

Tasks	Primary Task Asset(s)	ISR Support Asset(s)	Risk of Losing Borrowed ISR Asset
AEW Area A	CAP	AWACS and Rivet Joint	AWACS: Lose wide area search
TAMD GREEN	CG	U-2 and UAV	U2: I&W of potential launch
TAMD BLUE in A	CG and DDG		
SURF SURV Area A	P3 and JSTAR	P3 and JSTAR	P3: Lose periscope detection and ESM capability JSTARS: Lose wide area search
Negate Red Subs	2 SSN and 3 P3	P3 and AWACS	P3: Lose fast revisit and response time AWACS; Lose potential periscope detection and ESM

In order to determine the best allocation of assets to tasks, the FOPS players send Requests for Information (RFIs) to the ISR cell for an Intelligence Preparation of the Environment (IPE). These provide FOPS with accurate and up-to-date data on resource requirements on a task-by-task basis. In response to the FOPS requests, ISR players *recommend* to COPS the ISR assets that would yield the best data as per the RFI. The COPS players direct the ISR platforms (i.e., Rivet Joint, U-2, UAVs, AWACS, etc.) that are "owned" by subordinate forces. Thus, ISR acts as a bridge between FOPS and COPS. But, there is a trade-off here. Releasing ISR assets from COPS would negatively impact the performance of *current* task execution; however, the useful data obtained via this action would positively impact FOPS in building its *future* plan.

A. Tasks

A task, which is derived from a mission decomposition, is an activity that requires relevant resources (provided by organization's assets) so that it could be carried out by a DM or a group of DMs under certain mission objectives, i.e. maximizing the overall mission accuracy, minimizing the mission's completion time, or balancing the workload. In our model, we characterize a task i (i = 1, …, H, where H is the number of tasks) by specifying the following attributes: (i) $t_{s,i}$ = required start time of task i , (ii) $t_{p,i}$ = processing time of task i, (iii) $[R_{i,1}, \ldots, R_{i,L}]$ = resource requirement vector, where $R_{i,l}$ is the number of units of resource type l (l = 1, …, L) required for successful processing of task i

B. Assets

An asset is a physical entity with given resource capabilities and is used to process tasks. For each asset type j (j = 1, ..., K), we define their attributes as follows: (i) a resource capability vector $[r_{i,1}, \ldots, r_{i,L}]$, where $r_{i,l}$ is the number of units of resource l possessed by asset type j, and (ii) number of available assets for asset type j, D_j,

III. THE FOPS PLANNING MODULE

A. Problem description

For each area, there is a list of available assets and their resource capabilities, as well as a list of tasks and their resource requirements. The objective is to provide FOPS players with a list of N-best asset packages suitable for executing a user-selected task to maximize the task's execution accuracy. Moreover, there should be no more than M assets assigned to any task. Each package shows: a) the % accuracy attainable, and b) the resource categories in which more capability is being planned than what is needed, expressed as a % of "overkill". Values of M (≤ 4) and N (= 4) are determined by experiment designers, not the planners.

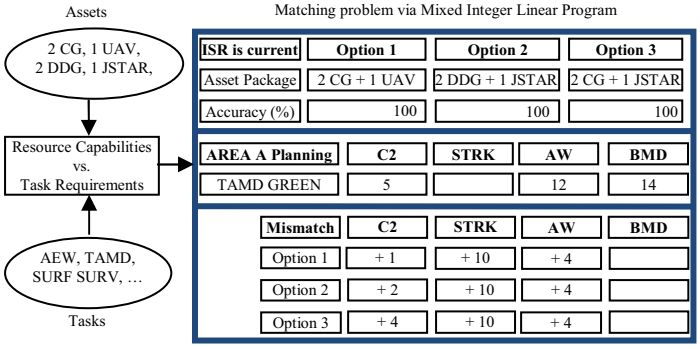

Figure 2: FOPS Planning Module

B. Mathematical formulation

Variables

$$y_{i,j} = \begin{cases} 1, & \text{if asset } j \text{ is assigned to task } i \\ 0, & \text{otherwise} \end{cases}$$

$acc(i)$ = execution accuracy of task i

$$acc(i) = \sqrt{\prod_{l=1}^{L} z_{i,l}}\,;\ \forall\ R_{i,l} > 0 \qquad (1)$$

where $z_{i,l} =$ accuracy due to resource type l for task i

From a computational viewpoint, we found it convenient to use as an objective the minimization of the sum of resource accuracies of relevant resource requirements of a user-selected task. The model used for FOPS decision aid is as follows.

$$\begin{aligned} \text{maximize} \quad & \sum_{l=1}^{L} z_{i,l} \\ \text{subject to} \quad & z_{i,l} \le \sum_{j=1}^{K} \frac{r_{j,l}}{R_{i,l}} y_{i,j},\ \forall\ R_{i,l} > 0 \\ & z_{i,l} \le 1,\ \forall\ R_{i,l} > 0 \\ & 0 < \sum_{j=1}^{K} y_{i,j} \le M,\ \forall\ \text{i} \\ & 0 < \sum_{i=1}^{H} y_{i,j} \le D_j,\ \forall\ \text{j} \\ & y_{i,j} \in Z^{+},\ \forall\ i, j \end{aligned} \qquad (2)$$

Once the asset packages are computed, the task accuracy for each asset package option is computed using (1) and displayed to the FOPS players.

C. Methods

1) Mixed Integer Linear Programming (MILP): The problem in (2) was formulated as a MILP problem. Lpsolver (http://lpsolve.sourceforge.net/5.5/) is one of many optimzation tools, which have been developed to solve MILP problems. Lpsolver was used to generate the best asset allocation for a user-slected task via the revised simplex and a branch-and-bound method. The usage of the combined algorithm guarantees an optimal solution, if it is feasible.

2) Extension of Murty's ranked assignment algorithm to MILP Problems: Murty's *N*-best assignment algorithm provides a general framework to generate *N*-best solutions to MILP problems. The key ideas of Murty's algorithm are:

- Partition the original problem into sub-problems according to the best current solution generated by Lpsolver;
- Compute optimal solution for each sub-problem;
- Select the next best solution among all generated solutions by sorting their corresponding costs.

The *N*-best assignment process works as follows (we omit task index *i* below):

- The assignment variable is comprised of positive integer variables $[y_1,y_2,\ldots,y_K]$ and real-valued accuracy variables $[z_1,z_2,\ldots,z_L]$. Since the accuracy vector depends on the assignment vector, only $[y_1,y_2,\ldots,y_K]$ is considered in Murty's problem decomposition.
- Since each component in $[y_1,y_2,\ldots y_K]$ can be any positive integer ($\le M$), the following symbols are used to denote the solution set for each sub-problem: "1" means the component's value remains the same as in the last best solution, "0" means the component's value can be any positive integer number different from the one in the last best solution, "*" means the component can take any value without any constraints. For example, to search for the first best solution, the solution set can be denoted as [*,*,…,*]. Suppose the first best solution is [2, 1, 0,…, 0]. One possible sub-problem solution set is [2, 0, *,…, *], which means that the first element is 2, the second element can be any value but 1.
- To avoid redundant assets among solutions, a pruning strategy was developed. Here is an example of a redundant asset problem: suppose two asset packages are requested. The algorithm may return the second best solution, which is nothing but the first best solution plus some redundant assets (both solutions providing 100% accuracy). Therefore, a pruning strategy is required to remove such solutions. In addition, our approach can reduce the computation time by moving the nonzero elements to the beginning of the assignment vector, thereby solving fewer sub-problems. For example, suppose the first best solution is [1, 0, 0, 2]. Then, three sub-problems with solution sets [0, *, *, *], [1, 0, *, *], [1, 1, 0, *] need to be solved. However, if we reorder the assignment vector as [1, 2, 0, 0], the two sub-problems [0, *, *, *] and [1, 0, *, *] cover the solution space for the second best solution.

The *N*-best assignment algorithm and pruning strategy are described in the APPENDIX.

IV. THE SCHEDULING MODULE

A. Problem description

For each geographic area (A or B), mission decomposition specifies: (1) the set of tasks with their resource requirements, and (2) the area-specific assets and shared assets with their resource capabilities. The objective is to determine a near-optimal schedule for pre-experiment analysis and for later model-data comparison. An asset can only be assigned to one task at any given time. There should be no more than *M* assets assigned to any task. Experiment designers provided the following information for each task: required start time, processing time, and prerequisites. All tasks must be completed within a specified time horizon (e.g. 30 hrs). The minimum task execution accuracy must be greater than a

specified threshold (e.g. 75%). The module provides experiment designers with the best asset package for each task in the mission, task execution accuracy, and a list of unallocated assets (if any) prior to experimentation. This analysis also provides insights into assets that are critical to accomplishing the mission with maximum accuracy.

B. Scheduling Algorithm

1) Task Grouping: In this experiment, all tasks are specified with their desired start time (release time). So, we first group tasks which have the same start time into a cluster. Once all tasks are grouped together according to the start time, the groups of tasks can be ranked in the ascending order of task start time. Next, the groups are scanned from the beginning of the sequence. Two neighboring groups can be combined if the latest finish times of tasks in the previous group are greater than the start times of tasks in the next group. This is because asset(s) released from tasks whose processing overlaps cannot be used by the next task group because of time conflict.

2) Pair-wise exchange (PWE): The near-optimal schedule generated by applying the Lpsolver and task grouping technique can be improved by employing the PWE method. Since a task's execution accuracy is defined as a square root of product of its resource accuracies, it is possible that there exists a task with zero accuracy. The PWE method is used to to improve task accuracy by transferring assets among tasks or by adding unassigned assets to the task. For each task, PWE checks if adding assets, which come from other tasks or the list of unused assets, improves its execution accuracy.

Table V. Addition assets increases the balance between task 1 and 2

	Accuracy Before (%)	Accuracy After (%)
Task 1	20	40
Task 2	100	60

C. COPS Risk Analysis Module

This module was implemented to assist COPS players on the consequences of redirecting assets from an ongoing task to gather information associated with either: 1) a RFI for a FOPS task, 2) a CCIR, commander critical information request, or 3) an urgent RFI from a subordinate task force. For each task for which assets are considered for removal, the COPS module shows the adjusted/reduced accuracy that would be attained and risk incurred (during the period of removal).

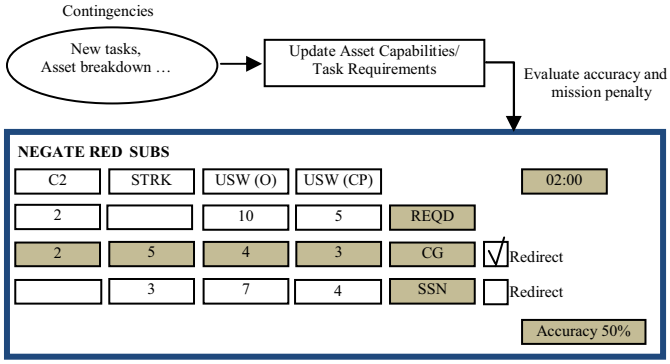

Figure 3: COPS Risk Analysis Module

V. SIMULATION RESULTS

There were 11 tasks and 25 assets in area A. The FOPS player selects a task and a list of assets from which four best asset packages need to be selected. For a given task, the interactive *FOPS Planning Module* provided four best asset packages in terms of task execution accuracy (typically in less than 3 seconds). No more than four assets were allocated to any task. It may happen that, for a given task, the selected assets may not yield a feasible solution or provide a full list of *N*-best asset packages. If less than *N* best options are computed, the reduced set is displayed to the FOPS player.

The experiment designers examined several changes to the mission planning scenario (e.g., asset types and numbers, task requirements and asset capabilities) to see if the resultant schedule, as generated via the *Scheduling Module*, could be improved and by how much. In particular, two adjustments were found to be quite useful in designing the mission planning scenario: (1) Manipulating the number of assets, and (2) Reducing selected resource requirements on some tasks. For example, interchanging an AWACS from area A and an AEF from area B gave a 3% improvement and a 5 % reduction in average mission accuracy in area A and B, respectively. Reducing the resource requirement MIW from 5 to 4 caused a 1 % improvement in the mission accuracy in area A. In summary, changing the mission planning parameters gave a minimal accuracy improvement. It was found that the division of SSNs and DDGs between the two areas had a significant impact on mission accuracy. This phenomenon is reminiscent of bin packing problems: optimal bin packing solutions depend on not only on the objects' volumes and bins' capacities, but also on the objects' shapes.

Table VI. Outputs from the FOPS Planning Module

Name of Selected Task	**Attack Air Bases**			
ISR is current	***Option 1***	***Option 2***	***Option 3***	***Option 4***
Asset Package	1 CVN	2 DDG + 1 UAV	1 AEF	1CG + 1U2
Accuracy (%)	100	100	100	100

Mismatch	**C2**	**STRK**	**AW**	**BMD**	**CMD**	**SUW**	**USW**	**MIW**	**ISR(A)**	**ISR(S)**	**ISR(G)**	**BDA**
Option1	3	1			2	5	2	1	5	5		2
Option 2			3	7	6	4	3	3	6	5	3	2
Option 3	3	2				1			4	1	1	
Option 4	1		3	7	6	4	3	3	10	7	2	

Table VII. An example of an optimal schedule (Area A)

Task Description	**Accuracy (%)**	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30
Sample Task	87	CG_1, DDG_1, UAV_2														
AEW Area A	100	AWACS_3														
TAMD GREEN	100	CG_1, DDG_1, UAV_1														
TAMD BLUE in A	87	JSTAR_1, DDG_2														
SURF SURV Area A	100	P3_1, AWACS_4														
MIW in Strait A	89				MH53_2, P3_2, P3_5											
CSG Penetrate into A	100															CVN_2, CG_2, DDG_5
DEF vs. CDCM Attack	87	CG_2														
Attack Air Bases	100	DDG_5, UAV_5														
Attack_C2 Nodes	100	SSN_3, SSN_4, U2_1, RJ_1														
Attack IADS	87	CVN_2														
Attack MSL Bases	100	UAV_2, AEF_2														

VI. SUMMARY

In this paper, asset-to-task allocation and scheduling problems were introduced and formulated as MILP problems.

A novel aspect of this work is the decomposition of the search space to obtain *N*-best optimal solutions to MILP problems. The paper also presented a strategy to prune redundant assets among asset packages. In the scheduling module, a task grouping technique was applied to reduce the size of the scheduling problem. Near-optimal schedules are generated by applying the Lpsolver and a task grouping technique. The results for scheduling problem were better than we expected with 87% as the minimum accuracy, and 94% as the average accuracy. The algorithms were implemented to support a team-in-the-loop planning experiment conducted at the Naval Postgraduate School (NPS) in March 2009. The experiment examined coordination in planning activities among three cells in an abstracted and simplified Maritime Operations Center (MOC). Our future research will explore post-experiment model-data comparisons, embedding the assignment and scheduling algorithms in an agent-based framework, and multi-level coordination issues in MHQ/MOCs.

APPENDIX

N-Best Assignment Algorithm

Step 1: Set the initial solution set as [*,*, …,*]. Solve the original problem and obtain the best solution S_1. Set m=2 and the initial solution candidate set as empty.

Step2: Re-rank the last best solution S_{m-1}, and move the P non-zero elements to the beginning of the assignment vector.

Step 3: Partition the problem into K sub-problems with solution sets constrained as:

$$[0,*,...,*],[1,0,*,...,*],...,[\underbrace{1,..,1}_{P-1},0,*,...,*]$$

Step 4: Solve the sub-problems and invoke pruning strategy to refine the solutions. Put them into the candidate solution set.

Step 5: Select the current best solution from the candidate solution set. Remove it from the set and set the last solution set as the current best solution's solution set.

Step 6: If $m = N$ or the candidate solution set is empty, stop; otherwise, set $m = m+1$ and go to step2.

Pruning Strategy

Step 1: Given a solution S_0 and its solution space, check whether the solution contains redundant assets. If not, stop. Otherwise, go to step 2.

Step 2: Remove all the redundant assets and generate a new solution S_1. Check whether the new solution $S_1$is in the solution space, if yes, stop and return S_1. Otherwise, go to step 3.

Step 3: Check whether there is a solution other than S_0 in the solution space. If none, stop and return with no feasible solution. Otherwise, go to step 4.

Step 4: Treat S_0 as the first best solution. Invoke *N*-best assignment algorithm to solve within the current solution space for the second best solution.

Table VIII. List of resource categories defined in the experiment

Abbreviation	Description
C2	Command and Control
STRK	Strike
AW	Air Warfare
BMD	Ballistic Missile Defense
CMD	Cruise Missile Defense
SUW	Surface Warfare
USW	Undersea Warfare
MIW	Mine Warfare
ISR (A)	Intelligence Surveillance and Reconnaissance (Air)
ISR (S)	Intelligence Surveillance and Reconnaissance (Surface)
ISR (G)	Intelligence Surveillance and Reconnaissance (Ground)
BDA	Battle Damage Assessment

ACKNOWLEDGMENT

The authors would like to thank the NPS and Aptima, Inc. A2C2 team members for their tireless efforts in designing the MOC-1 experiment, and for providing detailed comments and helpful suggestions in developing optimization-based decision support systems. In particular, we would like to thank Bill Kemple and K. Pfeiffer at NPS and S. Weil at Aptima, Inc.

REFERENCES

[1] K.G. Murty, "An algorithm for ranking all the assignments in order of increasing cost," *Operations Research*, vol. 16, pp. 682-687, 1968

[2] I.J. Cox and M.L. Miller, "On finding ranked assignments with application to multi-target tracking and motion correspondence," *IEEE Transactions on Aerospace and Electronic System,* vol. 32, pp. 486-489, January 1995.

[3] D.P. Bertsekas, Introduction to linear optimization, *Athena Scientific*, Belmont, MA 1997.

[4] M. Shakeri, K.R. Pattipati, V. Raghavan, A. Petterson-Hine, "Optimal and near-optimal algorithms for multiple fault diagnosis with unreliable tests," *IEEE Transactions of Systems, Man and Cybertics – Part C,* vol. 28, pp. 431-440, August 1998.

[5] G.M. Levchuk, Y.N. Levchuk, Jie Luo, Fang Tu, and K.R. Pattipati, "A library of optimization algorithms for organizational design", *Proceedings of the 2000 Command & Control Research & Technology Symposium*, NPS, Monterey, CA June 2000.

[6] R.P. Popp, K.R. Pattipati, and Yaakov Barshalom, "m-Best S-D assignment algorithm with application to multiple target tracking," IEEE Transaction on Aerospace and Electronic Systems, vol. 37, pp. 22-39, January 2001.

[7] G.M. Levchuk, Y.N. Levchuk, Jie Luo, K.R. Pattipati, and D.L. Kleinman, "Normative design of organizations – part I: mission planning," *IEEE Transactions of Systems, Man and Cybertics – Part A*, vol. 32, pp. 346-359, May 2002.

[8] H.W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics*, vol. 52, pp. 7-21, October 2004.

[9] R.M. Freund, "Benders' decomposition methods for structured optimization, including stochastic optimization," *Massachussets Institue of Technology*, 2004.

[10] D.L Kleinman, P. Young, and G.S. Higgins, "The DDD-III: A tool for empirical research in adaptive organizations," *Proc. Command and Control Research and Technology Symposium*, Monterey, CA, June 1996.

[11] S.G. Hutchins, W.H. Kemple, D.L. Kleinman, S. Miller, K. Pfeiffer, S. Weil, Z. Horn, M. Puglisi, and E. Etin, "Maritime headquarters with maritime operations center: a research agenda for experimentation," unpublished.