Normative Design of Organizations—Part I: Mission Planning

Georgiy M. Levchuk, Yuri N. Levchuk, Jie Luo, Krishna R. Pattipati, *Fellow, IEEE*, and David L. Kleinman, *Fellow, IEEE*

***Abstract*—This paper presents a design methodology for synthesizing organizations to execute complex missions efficiently. It focuses on devising mission planning strategies to optimally achieve mission goals while optimally utilizing organization's resources. Effective planning is often the key to successful completion of the mission, and conversely, mission failure can often be traced back to poor planning. Details on subsequent phases of the design process to construct the mission-driven human organizations are discussed in a companion paper.**

***Index Terms*—Mission decomposition, mission planning, organizational design, resource allocation, scheduling.**

I. Introduction

A. Motivation

CHANGING patterns of today's world impose new requirements for many modern organizations, ranging from military establishments to agile manufacturing systems and commercial enterprises. With the benefit of new technologies now under development, the competition will be won by an organization that will best utilize both its resources and its critical information to achieve its goals. This implies the need for much greater emphasis on realistic modeling of distributed organizations in which the human participants are the focus.

In large-scale organizations that involve humans, decision-making and operational functions are distributed among team members who coordinate their actions in order to achieve their common goal. Since the processing capabilities of a human are limited, the distribution of *information*, *resources*, and *activities* among decision-makers (DMs) in an organization must be set up to guarantee that the decision-making and operational load of each DM remains below the DM's capacity thresholds. In a highly competitive and distributed environment, a proper balance among information acquisition, decision hierarchy, and resource allocation, in short, *a proper organizational structure and its processes*, is critical to superior organizational performance.

Manuscript received January 2, 2001; revised May 5, 2001. This work was supported by the Office of Naval Research under Contracts N00014-93-1-0793, N00014-98-1-0465 and N00014-00-1-0101. This paper was recommended by Associate Editor S. Ghosh.

G. M. Levchuk, J. Luo, and K. R. Pattipati are with the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, CT 06269 USA (e-mail: krishna@engr.uconn.edu).

D. L. Kleinman is with the Naval Post Graduate School, Monterrey, CA 93943 USA (e-mail: kleinman@nps.navy.mil).

Y. N. Levchuk is with Aptima Inc., Woburn, MA 01801 USA (e-mail: levchuk@aptima.com).

Digital Object Identifier 10.1109/TSMCA.2002.802819

Over the years, research in team decision-making has demonstrated that an organization operates best when its structure and processes match the corresponding mission environment [43], [49]–[51]. Consequently, it has been concluded that the *optimality* of an organizational *design* ultimately depends on the actual mission structure and on the key attributes of the environment in which the organization operates [6]. This premise has led to the application of systems engineering techniques to the process of designing human organizations [37], [38], [51]. The systems engineering approach to organizational design is as follows. First, a quantitative model describing the mission and the organizational constraints is built. Next, different criteria used to judge the optimality of an organization are combined into a (possibly nonscalar) objective function. Finally, an organizational design is generated to optimize the objective function.

In this paper, we present a methodology for *modeling* missions and for *synthesizing* the concomitant *optimal* organizations. We introduce a three-phase iterative optimization process that derives an optimized organizational design for a given mission structure and organizational constraints. In the first (mission-planning) phase of our design process, the optimal allocation of mission tasks to organization's platforms (physical resources) is determined to optimize the mission schedule. In the second phase, a three-way DM-platform-task allocation is derived to minimize the coordination and workload overhead and its impact on the mission schedule. In the third phase, other dimensions of organizational structure (e.g., information acquisition and communication structures, decision hierarchy) are optimized to fulfill the design objectives. Following a description of our modeling and design methodology, the paper focuses on the development of algorithms for mission-planning phase. The algorithms to optimize other dimensions of organizational design are presented in a companion paper.

B. Related Research

Over the past few years, mathematical and computational models of organizations have attracted a great deal of interest in various fields of scientific research (see [43] for review). The mathematical models have focused on the problem of quantifying the structural (mis)match between organizations and their tasks. The notion of structural congruence has been generalized from the problem of optimizing distributed decision-making in structured decision networks [51] to the multiobjective optimization problem of designing optimal organizational structures to complete a mission, while minimizing a set of criteria [37], [38]. As computational models of decision-making in organizations began to emerge [9], [55], [64], the study of social networks (SSN) continued to focus on examining a network structure and its impact on individual, group, and organizational behavior [67]. Most models, developed under the SSN, combined formal and informal structures when representing organizations as architectures (e.g., see the virtual design team [42] and ORGAHEAD [9]), and in many empirical studies, the informal subsumed the formal. In addition, a large number of measures of structure and of the individual positions within the structure have been developed [59], [65], [66].

Application of systems engineering techniques to the process of designing human organizations led to several graph-decomposition and combinatorial optimization algorithms to synthesize congruent organizational structures (i.e., structures that are in some sense "matched" with their mission task environment) [37], [38], [51] and to several graph-theoretic measures of task complexity [36]. Potential benefits of a structural match predicted by the normative model, as well as the ability of a proposed design process to find this match, have been tested empirically in a computer-mediated team-in-the-loop experiment with human DMs in a distributed dynamic decision-making (DDD-III) simulator of Joint C^2 scenarios (see [31] for a description of DDD-III). DDD-III and other similar simulators allow one to compare the performance of different organizational designs for a specific mission, and to test the utility of the design procedure. Findings from recent experiments on DDD-III simulator gave the empirical validation of our modeling and design methodology. It clearly showed the advantages of structural optimization, since the model-driven nontraditional architectures outperformed their traditional counterparts. The experimental results showed that not only were the system engineering methods successful in constructing the desired architectures, but architecture type differently affected team processes, as was hypothesized by the design [18], [26].

C. Organization of the Paper

The paper is organized as follows. Our three-phase organizational design process is outlined in Section II. Section III defines the organization and mission parameters, providing a rationale for various mission decomposition techniques and mission planning strategies. The problem of allocating resources to tasks (Phase I) is discussed in Section IV. We present a mathematical problem formulation and several heuristic methods to solve this problem, and give performance analyses of these algorithms. Algorithm performance and simulation results are given in Section V. The paper concludes with a summary and future extensions in Section VI.

II. Organizational Design Process

The notion of optimality is subjective [30]. Moreover, different aspects of organizational performance are deemed important when assessing the efficacy of an organization. Hence, the organizational design problem is inherently multiobjective, and the correct choice of optimization criteria is critical to generating the optimal design.

Modern man–machine systems are made exceedingly complex by the many human and technological elements involved. The sources of this complexity include *dimensional complexity* (processes and interactions on many levels), *uncertainty*, and *computational complexity* [12], [41], [44]. Even for a smaller size organization facing a mission that consists of a small number of tasks, there can be an enormous number of possible solutions to the organizational design problem (and optimization can yield significant improvements). In general, all the existing methods of multiobjective combinatorial optimization problems are NP hard (optimal algorithms take exponential time [24]).

One way of simplifying the search for the optimal organizational design is to exploit the *connection* between multidimensionality of organizational structure and the composition of a concomitant multivariable objective function. In general, the objective function combines variables representing both mission objectives and design parameters (e.g., decision-making workload, resource utilization, coordination, etc. [14], [28], [30], [39], [64]). Each dimension of the organizational structure stipulates a corresponding portion of the design parameters. For example, DM-platform allocation and mission schedule define the operational workload of a DM, while information access structure, allocation of decision variables, and communication structure stipulate a decision-making workload of a DM.

The relative weights of the optimization criteria that determine organizational performance can be represented via weighting coefficients assigned to each component in the objective function. Therefore, in theory, we can build an organizational structure by *iteratively optimizing* different structural dimensions, beginning with those dimensions that delineate the heaviest portion of the objective function. For example, an organizational strategy determines the mission processing schedule as well as the individual operational workload of a DM. Consequently, it generally specifies a large portion of parameters in the multivariable objective function. Each subsequent dimension is optimized subject to a fixed structure on those dimensions that have been optimized already. The iterative application of optimization process allows one to simultaneously optimize multiple dimensions [37].

Following the above logic, our organizational design methodology integrates various algorithms that optimize different dimensions of an organizational structure. For a given mission structure, an organization is designed via the following three phases (see Fig. 1).

Phase I: The first phase of the design process determines the task-platform allocation and task sequencing that optimize mission objectives (e.g., mission completion time, accuracy, workload, resource utilization, platform coordination, etc.), taking into account task precedence constraints and synchronization delays, task resource requirements, resource capabilities, as well as geographical and other task transition constraints. The generated task-platform allocation schedule specifies the workload of each resource. In addition, for every mission task, the first phase of the algorithm delineates a set of nonredundant resource packages capable of jointly processing a task. This information is later used for iterative refinement of the design, and, if necessary, for on-line strategy adjustments.

Phase II: The second phase of the design process combines platforms into nonintersecting groups, to match the operational

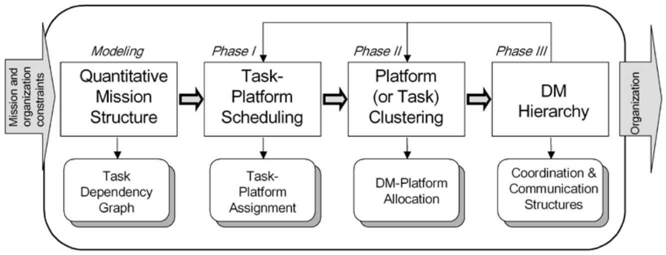


Fig. 1. Three-phase organizational design process.

expertise and workload threshold constraints on available DMs, and assigns each group to an individual DM to define the DM-resource allocation. Thus, the second phase delineates the DM-platform-task allocation schedule and, consequently, the individual operational workload of each DM.

Phase III: Finally, Phase III of the design process completes the design by specifying a communication structure and a decision hierarchy to optimize the responsibility distribution and inter-DM control coordination, as well as to balance the control workload among DMs according to their expertise constraints.

Each phase of the algorithm provides, if necessary, a feedback to the previous stages to iteratively modify the task-resource allocation and DM-platform-task schedule. Phase I of our design process essentially performs mission planning, while Phases II and III construct the organization to match the devised courses of action.

III. Mission Planning

A mission analysis that details required courses of action by specifying a sequence of tasks, defining resource-to-task allocation, and a time-line for all task activities constitutes a *mission plan*. Planning problem is investigated in artificial intelligence and behavioral science area (see [2]). Planning models in military human behavior representation have been extensively studied and several specific planning tools have been developed, including adaptive combat modeling [22], commander's visual reasoning tool [4], dismounted infantry computer-generated force [27], computer-controlled hostiles for SUTT (small unit tactical trainer) [53], fixed-wing attack-soar and soar-intelligent forces [32], [33], man–machine integrated design and analysis system [3], naval simulation system [61], automated mission planner [35], to name a few.

A. Mission Structure

A fundamental question underlying a distributed organizational design—"*who should do which part of the mission?*"—implies that the mission must be *decomposable* into a set of *entities*. These entities are generally referred to as tasks.

Definition 1: A *Task* is an activity that entails the use of relevant resources (provided by organization's platforms) and is carried out by an individual DM or a group of DMs to accomplish the mission objectives.

Every task in itself represents a "small mission," and can oftentimes be further decomposed into more elementary tasks. A *mission decomposition diagram* can be built to represent a hierarchical structure among the mission tasks. Different decomposition techniques (e.g., goal decomposition, functional decomposition, domain decomposition) represent different starting points of defining tasks and provide different task types required to complete the mission. The designer's choice of a particular decomposition technique and model granularity (number of tasks in the mission decomposition) must be contingent on the computational efficiency of the design process and its supporting algorithms.

Task attributes quantify various properties of the mission tasks that detail the specifics of task execution. They provide quantitative characteristics for the mission structure and specify the implications of commitment to task processing on both machine and human resources of an organization. In our model, we characterize every mission task T_i by specifying the following basic attributes:

1) estimated processing time t_i ($i = 1, \ldots, N$, where N is the number of tasks);
2) geographical constraint vector (e.g., the "location" (x_i, y_i) in a state space that specifies the concomitant "distance" d_{ij} to be traveled between tasks T_i and T_j);
3) resource requirement vector $[R_{i1}, R_{i2}, \ldots, R_{iL}]$, where R_{il} is the number of units of resource l required for successful processing of task T_i ($l = 1, \ldots, L$, where L is the number of resource types).

We illustrate our modeling paradigm and organizational design process via an example of a joint-task-force scenario as operationalized in the distributed dynamic decision-making (DDD-III) team-in-the-loop real-time simulator. A detailed description of this empirical research tool is provided in [31].

Example: Experiment With DDD-III Simulator: A joint group of Navy and Marine Forces is assigned to complete a military mission that includes capturing a seaport and airport to allow for the introduction of follow-on forces. There are two suitable landing beaches designated "north" and "south," with a road leading from the north beach to the seaport, and another road leading from the south beach to the airport (a mission geographic layout is shown in Fig. 2). From intelligence sources, the approximate concentration of the hostile forces is known, and counter-strikes are anticipated. The commander devises a plan for the mission that includes the completion of tasks shown in Fig. 2. The following eight resource requirements/capabilities are modeled:

1) AAW (anti-air warfare);
2) ASUW (anti-surface warfare);
3) ASW (anti-submarine warfare);
4) GASLT (ground assault);
5) FIRE (artillery);
6) ARM (armor);
7) MINE (mine clearing);
8) DES (designation).

A variety of modeling techniques [e.g., object-oriented modeling (OOM); entity relationship (ER) [17]; integrated definition (IDEF) modeling [70], [71]; relational database design (RDD)] can be applied to capture the internal structure of a mission.

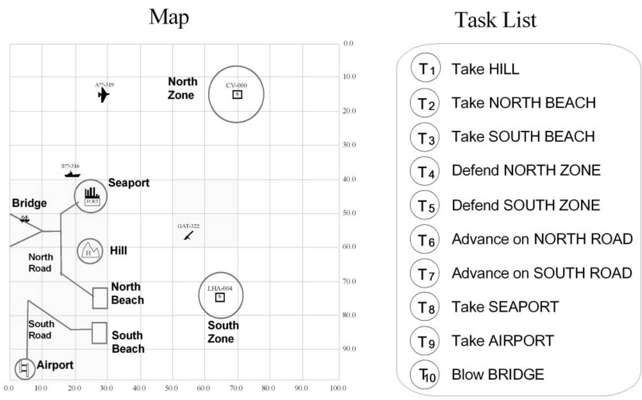


Fig. 2. Geographical constraints and mission tasks for an experiment with DDD-III simulator.

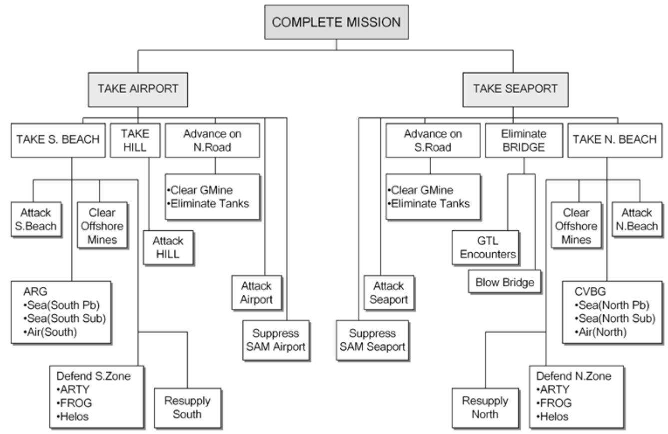


Fig. 3. Example of mission goal decomposition for an experiment with DDD-III simulator.

One of the most popular modeling methodologies employs a graph formalism to describe the mission structure. The graph formalism is used to construct the dependency among the tasks of a mission. Examples of different mission decomposition techniques used to design the organizational structure for an experiment with DDD-III simulator (see [39] for details) are shown in Figs. 3 and 4. Task parameters are shown in Table I.

Definition 2: A *Task Graph* is a dependency diagram that details the following interrelationships between tasks:

1) task precedence;
2) inter-task information flow;
3) input–output relationships between tasks.

A directed acyclic task-precedence graph represents the plan to execute the mission. The examples of task graphs and inter-task dependencies for an Experiment with DDD-III simulator [39] are shown in Figs. 5–7.

B. Organizational Constraints and Design Output

In defining an organization, we differentiate between two classes of *entities*: 1) decision-makers (DMs) and 2) resources. Organization's resources that represent nonhuman physical assets are called *platforms*.

Definition 3: A *Decision-Maker (DM)* is an entity with information-processing, decision-making, and operational capabilities that can control the necessary resources to execute mission tasks, provided that such an execution will not violate the concomitant capability thresholds.

A maximal number of available DMs is specified. An example of DM responsibilities is shown in Fig. 8.

Definition 4: A *Platform* is a physical asset of an organization that provides resource capabilities and is used to process tasks. For each platform P_m $(m = 1, \ldots, K)$, we define its maximal velocity v_m and its resource capability vector $[r_{m1}, r_{m2}, \ldots, r_{mL}]$, where r_{ml} specifies the number of units of resource type l available on platform P_m. The platform parameters are illustrated in Table II.

Definition 5: An *Organization* is a *team* of human DMs, who coordinate their information, resources, and activities in order to achieve their common goal in a complex, dynamic, and uncertain mission environment.

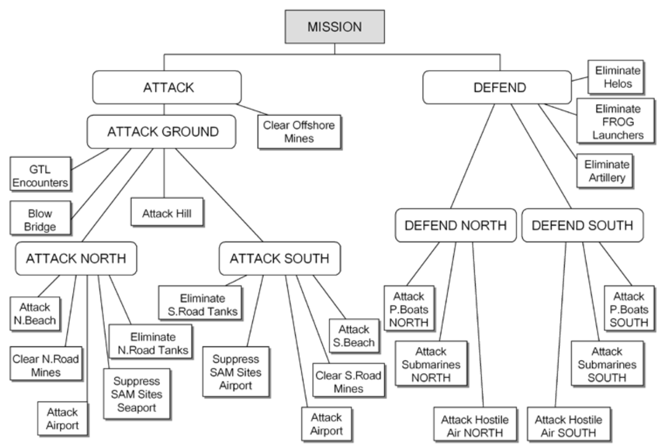


Fig. 4. Example of combined functional and domain decomposition for an experiment with DDD-III simulator.

TABLE I
ILLUSTRATION OF MISSION TASK PARAMETERS

Resource Requirements:

ID	Task Name	AAW	ASUW	ASW	GASLT	FIRE	ARM	MINE	DES	Locations		Time
1	Take HILL	0	0	0	10	14	12	0	0	24	60	10
2	Take N. Beach	0	0	0	10	14	12	0	0	28	73	10
3	Take S. Beach	0	0	0	10	14	12	0	0	28	83	10
4	Defend N. Zone	5	0	0	0	0	5	0	0	28	73	10
5	Defend S. Zone	5	0	0	0	0	5	0	0	28	83	10
6	Advance N. Road	0	0	0	0	0	10	5	0	25	45	10
7	Advance S. Road	0	0	0	0	0	10	5	0	5	95	10
8	Take SEAPORT	0	0	0	20	10	4	0	0	25	45	15
9	Take AIRPORT	0	0	0	20	10	4	0	0	5	95	15
10	Blow BRIDGE	0	0	0	8	6	0	4	10	5	60	20

The key *attributes* in modeling a DM are the individual DM *thresholds* with respect to particular DM activities (e.g., information processing and operational load thresholds). These thresholds quantify human limitations and necessitate the need for distribution and decentralization within a human organization.

As a consequence of decentralization in large-scale systems, each DM only has access to a portion of organization's resources and possibly to the information available to the team. The total decision-making and operational load is partitioned

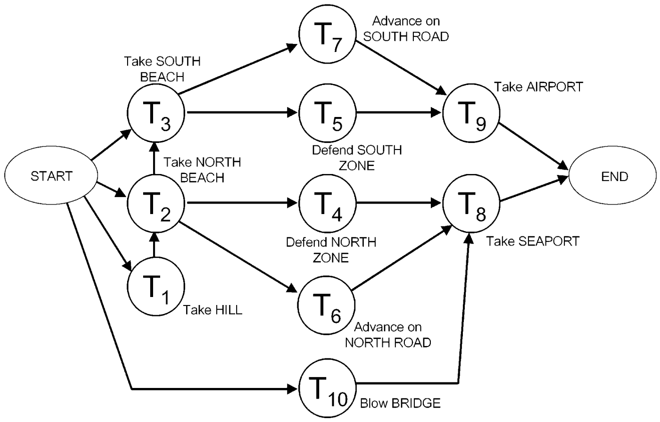


Fig. 5. Primary mission task graph for an experiment with DDD-III simulator.

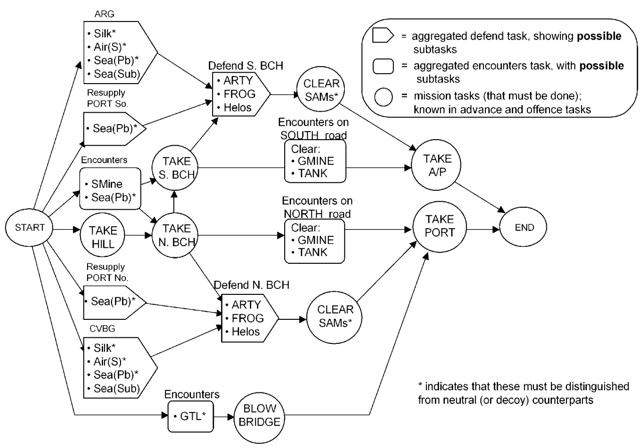


Fig. 6. Expanded mission task graph for an experiment with DDD-III simulator.

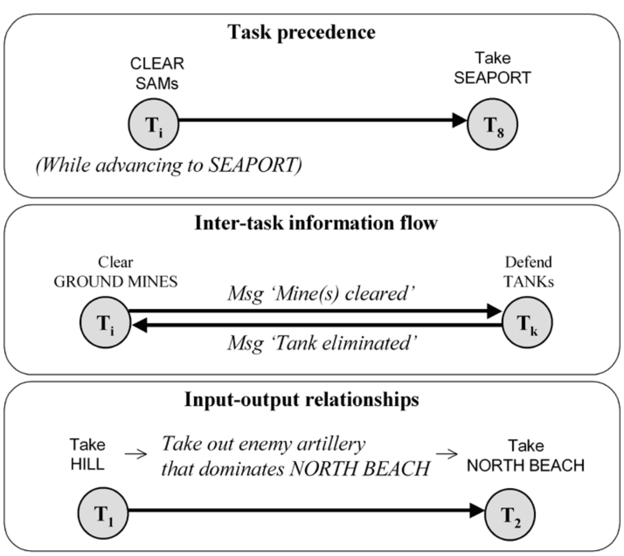


Fig. 7. Examples of interrelationships among tasks for an experiment with DDD-III simulator.

among DMs by decomposing a mission into tasks and assigning these tasks to individual DMs who are responsible for their planning and execution. An overlap in task processing (wherein two or more DMs share responsibility for a given task while each possesses the capability to individually execute a task) gives the team a degree of freedom to adapt to uneven demand by redistributing the load. The critical issues in team *task processing* are: *what* should be done, *who* should do what, and *when*. These questions are generally answered by *mission planning* that corresponds to mission-modeling phase followed by Phase I of our design process outlined in Fig. 1.

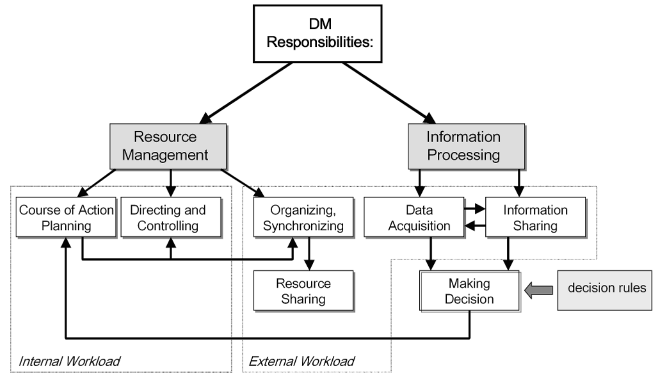


Fig. 8. Example of DM responsibilities.

TABLE II
ILLUSTRATION OF PLATFORM PARAMETERS FOR EXPERIMENT WITH DDD-III SIMULATOR

Resource Capabilities:

ID	Platform Name	AAW	ASUW	ASW	GASLT	FIRE	ARM	MINE	DES	Velocity
1	DDG	10	10	1	0	9	5	0	0	2
2	FFG	1	4	10	0	4	3	0	0	2
3	CG	10	10	1	0	9	5	0	0	2
4	ENG	0	0	0	2	0	0	5	0	4
5	INFA	1	0	0	10	2	2	1	0	1.35
6	SD	5	0	0	0	0	0	0	0	4
7	AHI	3	4	0	0	6	10	1	0	4
8	CAS1	1	3	0	0	10	8	1	0	4
9	CAS2	1	3	0	0	10	8	1	0	4
10	CAS3	1	3	0	0	10	8	1	0	4
11	VF1	6	1	0	0	1	1	0	0	4.5
12	VF2	6	1	0	0	1	1	0	0	4.5
13	VF3	6	1	0	0	1	1	0	0	4.5
14	SMC	0	0	0	0	0	0	10	0	2
15	TARP	0	0	0	0	0	0	0	6	5
16	SAT	0	0	0	0	0	0	0	6	7
17	SOF	0	0	0	6	6	0	1	10	2.5
18	INF(AAAV-1)	1	0	0	10	2	2	1	0	1.35
19	INF(AAAV-2)	1	0	0	10	2	2	1	0	1.35
20	INF(MV22-1)	1	0	0	10	2	2	1	0	1.35

In general, DMs are provided with limited resources with which to accomplish their objectives. The distribution of these resources among DMs, and the assignment of these resources to seek information and to process tasks are key elements in an organization's design. Team members must dynamically coordinate their resources to process their individual tasks while assuring that team performance goals are met. The critical issues in team *resource allocation* are: *who* should own or transfer a specific resource, *when*, and *for how long*. These questions are answered in Phase II of our design process.

In addition to assigning to each DM his share of information, resources, and activities, the organizational design must explicate a decision *hierarchy* among DMs that designates their *control responsibilities* (through command authority) and that regulates the inter-DM *coordination* (by assigning the responsibility of resolving decision ambiguities among coordinating DMs). The organizational design can also specify a *communication structure* among DMs to facilitate coordination and distributed information processing required for completing the mission. A communication structure and a decision hierarchy are devised in Phase III of our design process.

IV. Phase I: Resource-Task Allocation

A. Scheduling in Organizational Design: Motivation and Problem Definition

A successful scheduling of tasks, obtained from mission decomposition, to available organizational resources (platforms) under resource requirement and task inter-dependency constraints is a key determinant of organizational performance. Low computational complexity of algorithms for solving this NP-hard problem is a highly desirable feature. In the following sections, we present polynomial list scheduling algorithms and local search techniques to obtain an efficient near-optimal platform-task assignment for the Phase I of our design procedure.

Conceptually, the scheduling phase of the organizational design process is as follows. A set of tasks with specified processing times, resource requirements, locations, and precedence relations must be executed by a set of platforms with given resource capabilities, ranges of operation, and velocities. Tasks are allocated to groups of platforms in such a way that, for each such platform package to task assignment, the vector of task's resource requirements is componentwise less than or equal to the aggregated resource capability of the platform group. The task processing can begin only when the processing of all its predecessors is completed and all platforms from the group assigned to this task have arrived at the appropriate location. In our model, we assume that a resource can only process one task at a time. Platforms are to be routed among the tasks so that the overall *mission completion time* (i.e., the completion time of the last mission task) is minimized. An output of the scheduling phase specifies a platform-task assignment for our organizational design, delineating task start times, and platform-task routing.

B. Mathematical Formulation of the Scheduling Problem

The scheduling problem associated with the Phase I of our three-phase organizational design process is defined by the following variables:

Assignment variables:

$$w_{im} = \begin{cases} 1, & \text{if platform } P_m \text{ is assigned to task } T_i \\ 0, & \text{otherwise.} \end{cases}$$

Traversing variables:

$$x_{ijm} = \begin{cases} 1, & \text{if platform } P_m \text{ is assigned to process task } T_j \\ & \text{after processing task } T_i \\ 0, & \text{otherwise.} \end{cases}$$

Objective function:

$Y =$ mission completion time

(time when the last task is completed).

The following parameters are used:

$$a_{ij} = \begin{cases} 0, & \text{if task } T_i \text{ must be completed} \\ & \text{before task } T_j \text{ can start} \\ 1, & \text{otherwise.} \end{cases}$$

T = mission completion time found using a heuristic algorithm (or set to infinity)—the upper bound on mission completion time.

T_0 = task that serves as "start–finish" or "depot") task. It indicates the initial location of organization's platforms before the mission execution is initialized.

The objective is to minimize the mission completion time. Following [40], the problem assumes the following form:

$$\min Y$$
$$\begin{cases} \sum_{j=0}^{N} x_{ijm} - w_{im} = 0, \quad i = 1, \ldots, N;\ m = 1, \ldots, K; \\ \sum_{j=0}^{N} x_{jim} - w_{im} = 0, \quad i = 1, \ldots, N;\ m = 1, \ldots, K; \\ \sum_{i=0}^{N} x_{i0m} = \sum_{j=0}^{N} x_{0jm} = 1; \\ s_i - s_j + x_{ijm} \cdot \left(\dfrac{d_{ij}}{v_m} + a_{ij} \cdot T \right) \le a_{ij} \cdot T - t_i \\ \qquad\qquad i, j = 1, \ldots, N;\ m = 1, \ldots, K; \\ \sum_{m=1}^{K} r_{ml} \cdot w_{im} \ge R_{il}, \quad i = 1, \ldots, N;\ l = 1, \ldots, L; \\ s_i - Y \le -t_i, \qquad i = 1, \ldots, N; \\ 0 \le Y \le T;\ s_i \ge 0;\ x_{ijk}, w_{ik} \in \{0, 1\}. \end{cases} \tag{1}$$

This is a mixed-binary (i.e., containing continuous and binary variables) linear programming (MIP) problem (which is proven to be NP-hard). Moreover, even relaxing the constraints on the binary variables w_{im} and x_{ijm} (that is, making them real numbers in the [0, 1] range) produces a linear programming problem (LP) with the number of variables equal to $K(N+1)^2 + N + 1$, the number of equality constraints equal to $2K(N+1)$, and the number of inequality constraints equal to $KN(N-1) + L(N+1)$. This makes it hard to find solutions to even average-sized scheduling problems.

C. Related Research

The scheduling problem arising in organizational design extends to a large set of well-known problems. When there exists only one platform, it is related to the traveling salesman problem (TSP) and its extensions (such as time-dependent TSP, TSP with precedence relations, etc. (for a review, see [34], and for recent results, see [20], [23], [48], and [69]). When any platform can process any task, the problem simplifies to multiple TSP with precedence relations. If, in addition, the processing of a task can be separated in time among different platforms, our problem is related to the vehicle routing problem and its extensions (for a review, see [25] and [45], for the latest results, see [15] and [21]). In the case when travel times among task locations are much smaller than the task processing times (and therefore can be ignored), our problem reduces to a multiprocessor scheduling problem with precedence constraints (for a review, see [11] and [16]; for recent studies see [5], [10] and [63]). For a review of general scheduling problems, see [16] and [52].

Other variations of problem formulation are possible. For example, there may exist a delay between processing of two tasks on the same platform (“adjustment” delay). The opposite of this situation is when the delay occurs only when tasks are processed on different platforms (communication delays) with no delay for processing by the same platform. This has relevance in multiprocessor scheduling with inter-processor communication delays [5]. Another variation is the existence of time windows for processing each task (that is, the earliest start times, called *release times*, and the latest end-times, called *deadlines*, define opportunity windows for tasks). In this case, the objective function involves the minimization of earliness-tardiness penalties (that is, the penalties resulting from processing tasks outside of their time-windows). In our problem, we assume that task-processing times are fixed. In the real world, situations may arise when task-processing times depend on the amount of resources allocated to them. The objective then is to achieve a tradeoff between processing tasks as fast as possible and using as little resources as possible. Another complication is that a task can begin to be processed when the assigned platforms are within a specified distance of this task (depending on the task and ranges of platforms). In this case, the problem assumes the form of the shortest covering path problem [13]. Other realistic constraints, such as the ability of tasks to move during the mission, and platforms having expendable resources (such as fuel, firepower, supplies, etc.) can be included.

Since the static scheduling problem is NP-hard in its general forms [24], the research in this area has mainly focused on obtaining heuristic scheduling algorithms with good performance. Most of the heuristic scheduling methods can be classified as either a *clustering algorithm* or a *list-scheduling algorithm*. *Local search* techniques are used to further improve the quality of the schedule.

The clustering algorithm divides the task set into clusters to be assigned to the processing elements. This method can be used only when there is no resource sharing in task processing (that is, no multiplatform task processing). A list scheduling method assigns priorities to tasks. They are executed according to the priorities and precedence constraints. A local search technique improves the quality of the schedule by task reassignments and shifting tasks in the processing sequence or in the *critical path (CP)* (longest processing sequence) [19], [23].

List scheduling algorithm selects a *ready* node (a node becomes ready when all its predecessors are executed) according to the task priority information. The following typical methods for deciding task priority coefficients were developed: *level assignment (LA), critical path (CP),* and *weighted length (WL)* [60]. This will be explained in more detail in Section IV-D.

When a task is selected, it is to be assigned to platforms for processing. In our case, task-resource requirement vector results in multiplatform task processing. As will be shown in Section IV-D, this problem can be formulated as a multidimensional knapsack problem. The one-dimensional knapsack problem was shown to be NP-hard, but pseudopolynomial algorithms exist for this problem (see [46] for a review).

D. Multidimensional Dynamic List Scheduling Method

The multidimensional dynamic list scheduling (MDLS) finds the platform-task allocation and mission schedule by sequentially assigning tasks to platforms until task set is exhausted. MDLS heuristic has two main steps:

Step 1: Select the task to be processed.

Step 2: Select the group of platforms to be assigned to it for processing.

Task Selection: In the first step, a ready task is selected (a task becomes *ready* when all its predecessors have been completed). The selection is determined by the current assignment information and precedence structure. The selection is made according to the priority coefficients assigned by using one of the three algorithms: *1) critical path (CP); 2) level assignment (LA);* or *3) weighted length (WL)* [60]. The complexity of calculating task priority coefficients is the same for each algorithm, and it is O(M), where M is equal to number of edges in the task precedence graph.

Critical Path Algorithm (CP): Many of the earlier and classical task allocation schemes are based on a CP heuristic. The idea is that tasks on the CP determine the shortest possible execution time for the mission. Furthermore, tasks on the CP must be executed in sequence. Therefore, one may identify the length of the CP for each graph node (task), rank the tasks in the decreasing order of CP lengths, and assign them to platforms on the basis of the priority list scheduling method. CPs and their lengths $CL(i)$ are calculated for each task given the task precedence graph and the task processing times (see *Exit Path* algorithm in [60]). In the list-scheduling algorithm, a ready task is selected with the largest $CL(i)$. When ties occur, a task with the largest number of direct successors is chosen (or ties are broken arbitrarily).

Level Assignment Algorithms (LA): Levels are defined for each task based on the task precedence graph in a sequential manner. All predecessors of a task can be located only on lower levels (no task can have a direct successor in the same or lower level) with one immediate predecessor located at the previous level. The LA algorithm assigns tasks level by level. At each level, tasks are assigned in the decreasing order of their processing times [60] (this is called *Heavy Node First* algorithm) or in the decreasing order of their CP lengths $CL(\cdot)$ [1].

Weighted Length Algorithm (WL): A major flaw in the CP algorithm is that it does not take into account the structure of subtrees in the node’s neighborhood. The success of scheduling depends on the efficiency of balancing the load of task processing among platforms. It is clear that the more tasks are ready for processing at the current time, the more efficient the load balancing would be. Therefore, tasks with large numbers of direct successors (called *control nodes*) should have a higher priority of processing.

The WL algorithm is an extension of the CP method in which the rank of a node depends on its processing time, a branching factor, number of direct successors, and their weights. In WL, tasks are assigned priority coefficients according to [60]

$$WL(i) = t_i + \max_{j \in OUT(i)} WL(j) + \frac{\sum_{j \in OUT(i)} WL(j)}{\max_{j \in OUT(i)} WL(j)} \qquad (2)$$

where $OUT(i)$ is the set of direct successors of task T_i in the task precedence graph. Here, the third coefficient is the sum of WLs of the children of task T_i normalized by the maximum WL among them. The WL algorithm selects a ready task with the largest $WL(i)$. If ties occur, a task with the largest $CL(i)$ is chosen (or ties are broken arbitrarily).

Another variation of this approach is to assign priorities according to weighted CP length

$$WCP(i) = CL(i) + \max_{j \in OUT(i)} CL(j) + \frac{\sum_{j \in OUT(i)} CL(j)}{\max_{j \in OUT(i)} CL(j)}. \quad (3)$$

In list scheduling based on LA, tasks are scheduled level-by-level. Thus, we can decrease the complexity of selecting the task to be processed by using a heap implementation (storing tasks in the same level as heaps according to their priorities).

Platform Group Selection: In the second step, a group of platforms is chosen for processing a selected task. A task is assigned to groups of platforms in such a way that the vector of task's resource requirements is componentwise less than or equal to the aggregated resource capability vector of the group of platforms assigned to it. An assignment is considered whenever a task (or a group of tasks) is completed. At that time, all of the platforms processing the completed task become *free*.

A major question is how to distribute the processing of a task under resource requirement constraints among available platforms to achieve minimal execution time of the mission. We obtain a tradeoff between the following objectives:

1) minimization of task's completion time;
2) minimization of allocated resources assigned to this task that may be needed to process remaining tasks.

Accordingly, the objective function $\sum_{m=1}^{K} c_{mi} \cdot w_{im}$ is minimized subject to resource requirement constraints

$$\sum_{m=1}^{K} r_{ml} \cdot w_{im} \geq R_{il}, \qquad l = 1, \ldots, L. \quad (4)$$

The coefficients of the objective function c_{mi} define the *cost* of assigning platform P_m to process task T_i. The group with minimal aggregated cost is selected. The problem becomes

$$\min \sum_{m=1}^{K} c_{mi} \cdot w_{im}$$
$$\begin{cases} \sum_{m=1}^{K} r_{ml} \cdot w_{im} \geq R_{il}, \quad l = 1, \ldots, L; \\ w_{im} \in \{0, 1\}. \end{cases} \quad (5)$$

Reducing and relaxing resource constraints produces a problem which is NP-hard (equivalent to a single 0–1 knapsack problem)

$$\min \sum_{m \in free} c_{mi} \cdot w_{im}$$
$$\begin{cases} \sum_{m \in free} w_{im} \cdot \sum_{l=1}^{L} \min(r_{ml}, R_{il}) \geq \sum_{l=1}^{L} R_{il} \\ w_{im} \in \{0, 1\}. \end{cases} \quad (6)$$

The knapsack-structure of the relaxed problem [45] is used in deriving a heuristic algorithm to find the group of platforms to be assigned to process a task. Following the ideas of *greedy* algorithms for knapsack problems, the assignment group is found by selecting platforms in the increasing order of the following coefficients:

$$R(m) = \frac{c_{mi}}{\sum_{l=1}^{L} \min(r_{ml}, R_{il})} \quad (7)$$

until the group's aggregated resource capability vector is componentwise equal to or more than the resource requirement vector of a task. Then, the group is pruned making feasible reductions in the reverse order.

Coefficients c_{mi} determine performance of the algorithm. Accordingly, four basic platform coefficients are used

$$R^1(m) = \left(s_{l(m)} + t_{l(m)} + \frac{d_{l(m),i}}{v_m}\right) \cdot \frac{\sum_{i \in READY \setminus \{i\}} \sum_{l=1}^{L} \min(r_{ml}, R_{il})}{\sum_{l=1}^{L} \min(r_{ml}, R_{il})} \quad (8)$$

$$R^2(m) = s_{l(m)} + t_{l(m)} + \frac{d_{l(m),i}}{v_m} \quad (9)$$

$$R^3(m) = \left(s_{l(m)} + t_{l(m)} + \frac{d_{l(m),i}}{v_m}\right) \cdot \sum_{i \in READY \setminus \{i\}} \sum_{l=1}^{L} \min(r_{ml}, R_{il}) \quad (10)$$

$$R^4(m) = \frac{\sum_{i \in READY \setminus \{i\}} \sum_{l=1}^{L} \min(r_{ml}, R_{il})}{\sum_{l=1}^{L} \min(r_{ml}, R_{il})} \quad (11)$$

where $READY = \{j\text{: task } T_j \text{ is ready for processing}\}$, and

$$READY \setminus \{i\} = \{j\text{: } j \in READY, j \neq i\}.$$

When a task is assigned, platform-task related assignment and scheduling information is updated, as well as the activity coefficients of the platforms. The starting time of the selected task T_i is found to be

$$s_i = \max\left(f, \max_{m \in G(i)} \left\{s_{l(m)} + t_{l(m)} + \frac{d_{l(m),i}}{v_m}\right\}\right) \quad (12)$$

where f is the current time; $G(i)$ is the group of platforms assigned to task T_i; and $l(k)$ is the last task processed by platform P_k (see [40] for details). The multidimensional dynamic list scheduling (MDLS) algorithm is given in the Appendix.

E. Pairwise Exchange Improvement

The MDLS algorithm produces sub-optimal solutions. It is expected that the sequence with which the tasks are assigned according to MDLS is near-optimal. The pairwise exchange (PWE) method improves the solution by considering all possible task assignment sequences obtained by exchanging the task at the current place in the assignment sequence with some other

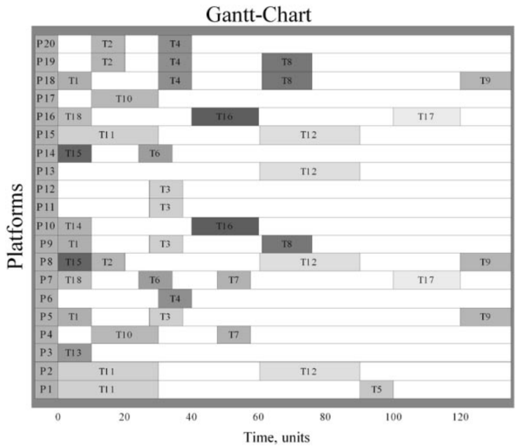


Fig. 9. MDLS results for an experiment using R^1 platform-task assignment.

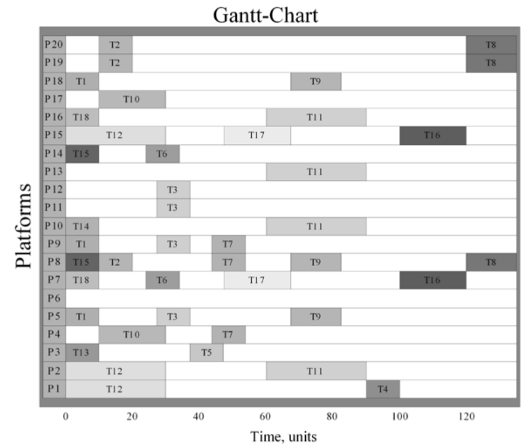


Fig. 10. Pairwise exchange results for an experiment with DDD-III simulator.

task. An exchange of tasks i_n and $i_m (n < m)$ in the sequence $\{i_1, \ldots, i_N\}$ is feasible if

a) $IN(i_m) \subset \{i_1, \ldots, i_{n-1}\}$
b) $OUT(i_n) \subset \{i_{m+1}, \ldots, i_N\}$

(where $IN(i)$ is the set of direct predecessors of task T_i and $OUT(i)$ is the set of direct successors of task T_i).

The algorithm is as follows:

for $n = 1\colon N-1$
do
Select $j \in \{n+1, \ldots, N\}$ such that the scheduling sequence $\langle i_1, \ldots, i_{n-1}, i_j, i_{n+1}, \ldots, i_{j-1}, i_n, i_{j+1}, \ldots, i_N\rangle$ is feasible and the schedule obtained using platform allocation from MDLS algorithm is the shortest one.
Then
$\langle i_1, \ldots, i_N\rangle \leftarrow \langle i_1, \ldots, i_{n-1}, i_j, i_{n+1}, \ldots, i_{j-1}, i_n, i_{j+1}, \ldots, i_N\rangle$
(permute tasks i_n and i_j in the scheduling sequence).
end for

We would like to point out that at each step of the algorithm, the sequence $i_1, \ldots, i_{n-1}$ is fixed (meaning that the scheduling and allocation for these tasks are fixed), and the search is only conducted to find the schedule for the remaining tasks.

Example (continued): For our experiment with the DDD-III simulator, MDLS algorithms based on CP and LA methods (as well as weighted CP variation of WL) produced the same optimal-length schedules (see Fig. 9). Breaking the ties by choosing a schedule with the least multiplatform task processing, the PWE procedure outputs the assignment-schedule shown in Fig. 10. These results are based on the platform-task assignment obtained by using coefficients $R^1(\cdot)$.

Fig. 11 shows the results based on platform-task assignment obtained using coefficients $R^4(\cdot)$. Although the completion time of the schedule is not optimal, this platform-task allocation utilizes resources better (six platforms are left idle during the mission).

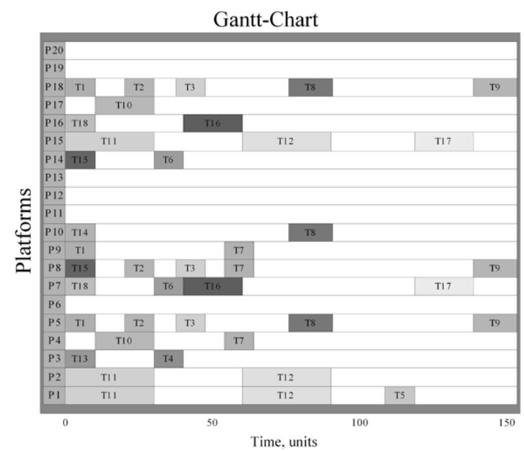


Fig. 11. MDLS scheduling results for an experiment using R^4 platform-task assignment.

V. Simulation Results

As stated earlier, the scheduling problem is NP-hard, which means that optimal solution takes exponential time in problem parameters (such as number of tasks, platforms, resources, and precedence constraints). Fig. 12 (CP = critical path, LA = level assignment, WCP = weighted CP; PWE_CP, PWE_LA, PWE_WCP = pairwise exchange based on CP, LA, and WCP) shows the box plot of optimality ratio of heuristic algorithms (ratio of objective function value obtained by heuristic algorithm to optimal solution) for a case of ten tasks (based on 1000 Monte-Carlo simulations). The best performance is obtained by list scheduling methods based on CP task selection (including weighted CP method).

Average CPU time data (for Pentium 600 MHz processor) for heuristic algorithms is shown in Figs. 13 (MDLS) and 14 (PWE). CPU times of PWE methods is approximately three times the processing time of MDLS algorithms.

As was stated earlier, the CPU time of optimal algorithm is strongly exponential. The data for ten tasks (see Fig. 15) indicate that, although the CPU time of optimal algorithm is mostly acceptable, in some cases it increases significantly due to the special structure of generated problems. Our simulations show that this behavior cannot be controlled.

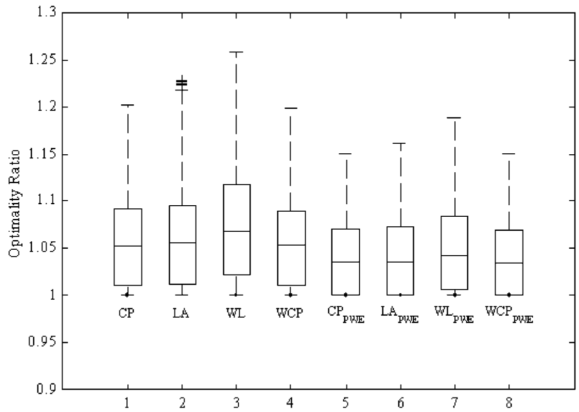


Fig. 12. Box plot of optimality ratios of heuristic algorithms. Number of tasks $= 10$.

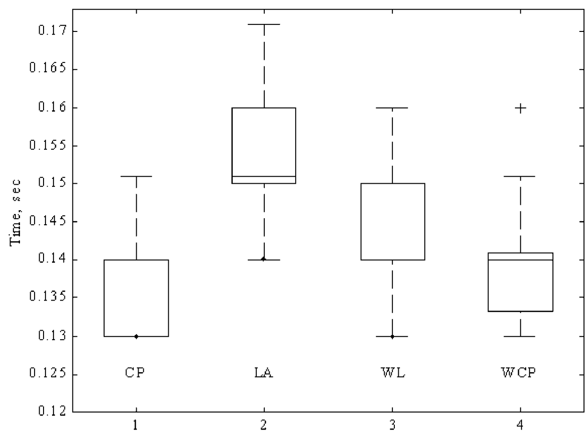


Fig. 13. Box plot of MDLS algorithms CPU times. Number of tasks $= 10$.

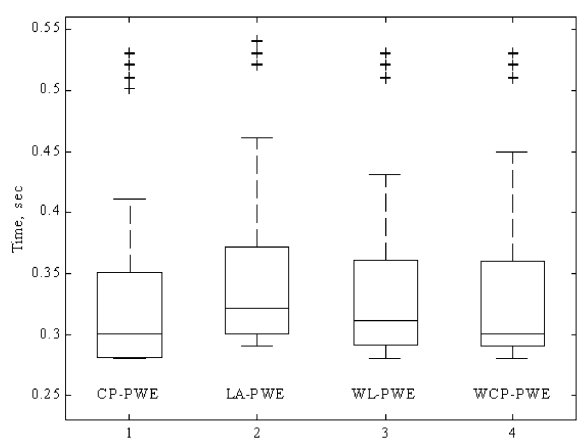


Fig. 14. Box plot of PWE algorithms CPU times. Number of tasks $= 10$.

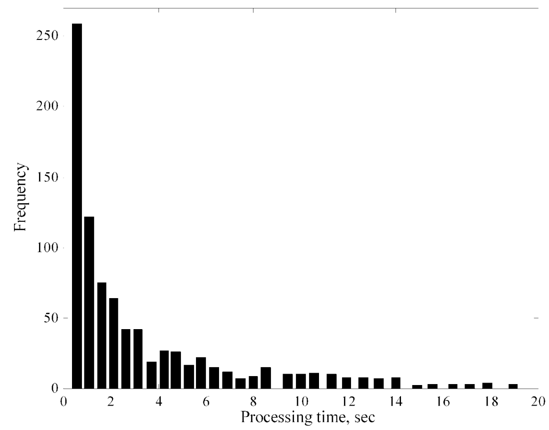


Fig. 15. Histogram of CPU time of optimal algorithm. Number of tasks $= 10$, number of simulations $= 1000$.

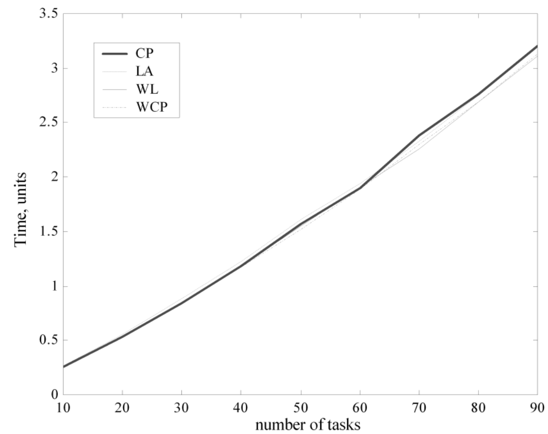


Fig. 16. Average CPU time of MDLS algorithm. Number of simulations $= 500$.

Simulation results for up to 90 tasks for heuristic algorithms only [with coefficient $R^1(\cdot)$ chosen for task selection] are shown in Fig. 16, Fig. 17, and Table III. This data shows linear relation between MDLS and PWE execution times. Instead of optimality ratio, we use scheduling length ratio (SLR) that is computed as the length of the schedule (algorithm's objective function value) normalized by the length of the CP in a task graph. It shows that improvement obtained by PWE methods is on average 10% (these results are obtained from 500 Monte Carlo simulations; number of platforms is seven; each task has no more than four direct predecessors).

The choice of a particular algorithm depends on the organization's objectives and resource constraints, and should be determined via simulations on a case-by-case basis. As can be seen from simulation results (see Table III), the best performance is obtained by MDLS and PWE methods based on CP task selection.

A platform group selection procedure identifies resource utilization. Specifically, it was found that platform-task allocation obtained using $R^4(\cdot)$ priorities produces the best resource utilization (but it extends the mission completion time). Simulations show that the use of $R^2(\cdot)$ priorities produces schedules with the shortest length, while coefficients $R^1(\cdot)$ generate the best tradeoff between the minimization of mission completion time and efficient resource utilization.

A platform group selection procedure identifies resource utilization. Specifically, it was found that platform-task allocation obtained using $R^4(\cdot)$ priorities produces the best resource utilization (but it extends the mission completion time). Simulations show that the use of $R^2(\cdot)$ priorities produces schedules with the shortest length, while coefficients $R^1(\cdot)$ generate the best tradeoff between the minimization of mission completion time and efficient resource utilization.

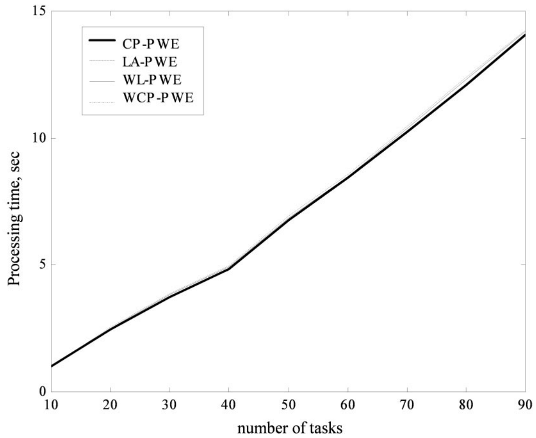


Fig. 17. Average CPU time of PWE algorithms. Number of simulations = 500.

TABLE III
OPTIMALITY OF HEURISTIC ALGORITHMS

number of tasks	CP	LA	WL	WCP
10	1.9008	1.9008	1.9256	1.9506
20	1.9375	1.9375	1.9587	1.9705
30	1.9569	1.9571	1.9784	1.9858
40	2.0106	2.0111	2.0257	2.0312
50	2.0512	2.0507	2.0614	2.058
60	2.094	2.0963	2.1101	2.1124
70	2.0701	2.0723	2.088	2.0855
80	2.1174	2.1169	2.1219	2.1206
90	2.19	2.1899	2.2109	2.2143
number of tasks	CP_PWE	LA_PWE	WL_PWE	WCP_PWE
10	1.6942	1.6942	1.6867	1.7019
20	1.7072	1.7048	1.7296	1.7259
30	1.749	1.7486	1.7661	1.7707
40	1.804	1.8047	1.8161	1.8239
50	1.8488	1.8493	1.863	1.8603
60	1.9013	1.9011	1.9156	1.9112
70	1.8877	1.8879	1.8978	1.8978
80	1.9265	1.9261	1.9409	1.9368
90	2.0078	2.0079	2.0188	2.0217

VI. SUMMARY AND FUTURE RESEARCH

In this paper, we presented guidelines for model-driven synthesis of optimized organizations for a specific mission. The primary contributions of this paper include a formal method for representing missions and human–machine organizations, a three-phase iterative design procedure to devise an optimized organizational structure and its mission processing strategy, and a description of the mission-planning phase (i.e., specifying mission task structure and defining task processing schedule and resource utilization scheme). We also presented an overview of the state-of-the-art in different domains of organizational design.

The potential of applying systems engineering approach to designing organizations is enormous, which was clearly shown by the experiments [18], [26]. This approach to designing man-machine systems allows for replacement of cumbersome centralized control with decentralized control and autonomy. Strict mathematical problem formulations provide the foundation for exploring ways to solve design problems efficiently and with the required degree of optimality to make best use of available time and computational resources. The latter is especially important for designing dynamic algorithms that help humans to adapt.

However, the field of model-based organizational design is in its infancy. The researchers lack a detailed classification of the design objectives and principles in building human–machine systems, as well as an understanding of the inner workings of a human organization. Some of these issues, including modeling a human DM as an integral part of a man–machine system and a detailed methodology for optimizing DM-resource allocation, inter-DM communication, and DM decision hierarchy, are presented in Part II of this paper. These methods, together with mechanisms for adaptation, including algorithms for on-line strategy adaptation and structural reconfiguration, form the basis for our continuing research in this area.

APPENDIX
MDLS ALGORITHM

Initialization

$$OUT(i) = \{j\colon T_j \text{ is a direct successor of } T_j\};$$
$$nOut(i) = |OUT(i)|$$
$$IN(i) = \{j\colon T_j \text{ is a direct predecessor of } T_j\};$$
$$nIn(i) = |IN(i)|$$
$$READY = \{j\colon nOut(j) = 0\}, \quad FT = \{0\}, \quad M = 0.$$

STEP 1. Completion time Update.
(skipped during initialization stage).
Pick $f = \min_{f_t \in FT}(f_t)$
$FT \leftarrow FT \setminus \{f\}$
Let F_G be the corresponding group of tasks.
$FREE \leftarrow FREE \cup G(F_G)$
for each $i \in F_G$
  for each $j \in OUT(i)$
    $nIn(j) \leftarrow nIn(j) - 1;$
    if $nIn(j) = 0$
      $READY \leftarrow READY \cup \{j\}$
    end if
  end for
end for

STEP 2. Assignment Check.
if $\forall i \in READY\ \exists s\colon \sum_{m \in FREE} r_{ml} \le R_{il}$
  GO TO Step 1.
else GO TO Step 3
end if

STEP 3. Task Selection.
if $READY = \varnothing$
  GO TO Step 1.
end if

Find the set

$$READY1 = \left\{ i \in READY \,\middle|\, \sum_{m \in FREE} r_{ml} \geq R_{il}\,,\; l = 1, \ldots, S \right\}$$

Select $i = \underset{j \in READY1}{\arg\min} \{P(j)\}$

$READY \to READY \setminus \{i\}$

STEP 4. Platform Group Selection.

Find the set

$$FREE1 = \left\{ m \in FREE \,\middle|\, \sum_{l=1}^{S} \min(r_{ml}, R_{il}) \neq 0 \right\}$$

$TG = \varnothing$

do until $\sum_{m \in TG} r_{ml} \geq R_{il}, \qquad \forall l = 1, \ldots, S$

$$n = \arg \max_{m \in FREE1} \{V_2(m)\}$$

$$FREE1 \leftarrow FREE1 \cup \{m\}$$

$$TG \to TG \cup \{n\}$$

end do

STEP 5. Platform Group Pruning.

$n = \arg \min_{m \in TG} \{V_2(m)\}$

$TG1 = TG$

while $TG1 \neq \varnothing$

$$n = \arg \min_{m \in TG1} \{V_2(m)\}$$

$$TG1 \leftarrow TG1 \setminus \{n\}$$

if $\sum_{m \in TG \setminus \{n\}} r_{ml} \geq R_{il}, \qquad \forall l = 1, \ldots, S$

$TG \leftarrow TG \setminus \{n\}$

end if

end while

STEP 6. Group Assignment.

$G(i) = TG$

$$s_i = \max\left(f,\, \max_{m \in G(i)} \left\{ s_{l(m)} + t_{l(m)} + \frac{d_{l(m),\, i}}{v_m} \right\}\right)$$

$f = s_i + t_i$

if $f \notin FT$

$$FT \leftarrow FT \cup \{f\}$$

end if

Update finishing platform groups.

GO TO Step 3.

References

[1] T. L. Adam, K. Chandy, and J. Dickson, "A comparison of list scheduling for parallel processing system," *Commun. ACM*, vol. 17, pp. 685–690, Dec. 1974.

[2] J. Allen, *Readings in Planning*. San Mateo, CA: Morgan Kaufmann, 1990.

[3] C. Banda, "Army–NASA aircrew/aircraft integration program: Phase IV A^{3}I man–machine integration design and analysis systems (MIDAS) software detailed design document," NASA Ames Research Center, Moffett Field, CA, NASA Contractor Rep. 177 593.

[4] M. J. Barnes and B. G. Knapp, "Collaborative planning aids for dispersed decision-making at the brigade level," in *Proc. 1st Annu. Symp. Advanced Displays and Interactive Displays Federated Laboratory*, Adelphi, MD, Jan. 28–29, 1997.

[5] S. K. Baruah, "The multiprocessor scheduling of precedence-constrained task systems in the presence of interprocessor communication delays," *Oper. Res.*, vol. 46, no. 1, pp. 65–72, 1998.

[6] R. M. Burton and B. Obel, *Strategic Organizational Diagnosis and Design: Developing Theory for Application*, 2nd ed. Norwell, MA: Kluwer, 1998.

[7] L. G. Bushnell, D. Serfaty, and D. L. Kleinman, "Team information processing: A normative-descriptive approach," in *Science of Command and Control: Coping With Uncertainty*, J. L. Boyes and S. J. Andriole, Eds. Washington, DC: AFCEA International Press, pp. 62–72.

[8] K. M. Carley and M. J. Prietula, "ACTS theory: Extending the model of bounded rationality," in *Computational Organization Theory*, K. M. Carley and M. J. Prietula, Eds. Hillsdale, NJ, 1994, pp. 55–88.

[9] K. M. Carley and D. M. Svoboda, "Modeling organizational adaptation as a simulated annealing process," *Sociol. Meth. Res.*, vol. 25, no. 1, pp. 138–168, 1996.

[10] L. M. A. Chan, "Parallel machine scheduling, linear programming, and parameter list scheduling heuristics," *Oper. Res.*, vol. 46, no. 5, pp. 729–741, 1998.

[11] T. C. E. Cheng and C. C. S. Sin, "A state-of-the-art review of parallel-machine scheduling research," *Eur. J. Oper. Res.*, vol. 47, pp. 271–292, 1990.

[12] C. Cooper, "Complexity in C3I systems," *Complexity Int.*, vol. 1, Apr. 1994.

[13] J. Current, Jr., H. Pirkul, and E. Rolland, "Efficient algorithms for solving the shortest covering path problem," *Transp. Sci.*, vol. 28, no. 4, pp. 317–325, 1994.

[14] K. R. Davis, "Resource constrained project scheduling with multiple objectives: A decision support approach," *Comput. Oper. Res.*, vol. 19, no. 7, pp. 655–659, 1992.

[15] Y. Dumas, J. Desrosiers, E. Gelinas, and M. M. Solomon, "An optimal algorithm for the traveling salesman problem with time windows," *Oper. Res.*, vol. 43, no. 2, pp. 367–371, 1995.

[16] H. El-Rewini, *Task Scheduling in Parallel and Distributed Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1994.

[17] R. Elmasri and S. E. Navathe, *Fundamentals of Database Systems*. Redwood City, CA: Benjamin Cummings, 1994.

[18] E. E. Entin, "Optimized command and control architectures for improved process and performance," in *Proceedings of the 1999 Command and Control Research and Technology Symposium*. Newport, RI: NWC, June 1999.

[19] G. Finn and E. Horowitz, "A linear time approximation algorithm for multiprocessor scheduling," *BIT*, vol. 19, pp. 312–320, 1979.

[20] M. Fischetti, P. Toth, and D. Vigo, "A branch-and-cut algorithm for the symmetric generalized traveling salesman problem," *Oper. Res.*, vol. 45, no. 3, pp. 378–394, 1997.

[21] M. L. Fisher, K. O. Jornsten, and O. B. G. Madsen, "Vehicle routing with time windows: Two optimization algorithms," *Oper. Res.*, vol. 45, no. 3, pp. 488–492, 1997.

[22] M. D. Fisk, "Marine corps modeling and simulation management office," in *Proc. 6th Annu. Modeling and Simulation Briefing to Government and Industry*, Alexandria, VA, May 22–23, 1997.

[23] P. M. Franca, M. Gendreau, G. Laporte, and F. M. Muller, "A composite heuristic for the identical parallel machine scheduling problem with minimum makespan objective," *Comput. Oper. Res.*, vol. 21, no. 2, pp. 205–210, 1994.

[24] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco, CA: Freeman, 1979.

[25] B. L. Golden and A. A. Assad, *Vehicle Routing: Methods and Studies*. New York: Elsevier, 1988.

[26] S. P. Hocevar, W. G. Kemple, D. Kleinman, and G. Porter, "Assessments of simulated performance of alternative architectures for command and control: The role of coordination," in *Proceedings of the 1999 Command and Control Research and Technology Symposium*. Newport, RI: NWC, June 1999.

[27] B. R. Hoff, "USMC individual combatants," in *Proc. Defense Modeling and Simulation Office Individual Combatant Workshop*, Alexandria, VA, July 1–2, 1996.

[28] H. Ishii, *Multi-Objective Scheduling Problems*, ser. Lecture Notes in Economics and Mathematical Systems, 1994, vol. 405, p. 386.

[29] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, NJ: Prentice-Hall, 1988.

[30] R. Keeney and H. Raiffa, *Decisions With Multiple Objectives*. Cambridge, U.K.: Cambridge Univ. Press, 1993.

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

[32] J. Laird, "SOAR/IFOR: Intelligent synthetic agents," in *Presentation to National Research Council Panel on Modeling Human Behavior and Command Decision Making*, October 17, 1996.

[33] J. E. Laird and P. S. Rosenbloom, "Integrating, execution, planning, and learning in SOAR for external environments," in *Proceedings of the Eighth National Conference on Artificial Intelligence*. Menlo Park, CA: American Association for Artificial Intelligence, 1990.

[34] E. L. Lawler, *The Traveling Salesman Problem*. New York: Wiley, 1985.

[35] J. J. Lee and P. A. Fishwick, "Real-time simulation-based planning for computer generated forces," *Simulation*, vol. 63, no. 5, pp. 299–315, 1994.

[36] Y. N. Levchuk, K. R. Pattipati, M. L. Curry, and M. Shakeri, "Design of congruent organizational structures: Theory and algorithms," in *Proc. Command and Control Research and Technology Symp.*, Monterey, CA, June 1996.

[37] Y. N. Levchuk, K. R. Pattipati, and M. L. Curry, "Normative design of organizations to solve a complex mission: Theory and algorithms," in *Proc. Command and Control Research and Technology Symp.*, Washington, DC, June 1997.

[38] Y. N. Levchuk, K. R. Pattipati, and D. L. Kleinman, "Designing adaptive organizations to process a complex mission: Algorithms and applications," in *Proceedings of the 1998 Command and Control Research and Technology Symposium*. Monterey, CA: NPS, June 1998.

[39] ——, "Analytic model driven organizational design and experimentation in adaptive command and control," *Syst. Eng.*, vol. 2, no. 2, 1999.

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

[41] A. H. Levis and M. Athans, "The quest for a C3 theory: Dreams and realities," in *Science of Command and Control: Coping With Uncertainty*, S. E. Johnson and A. H. Levis, Eds. Washington, DC: AFCEA International Press, 1988, pp. 4–9.

[42] R. E. Levitt, G. P. Cohen, J. C. Kunz, C. I. Nass, T. Cristiansen, and Y. Jin, "The 'virtual design' team: Simulating how organization structure and information processing tools affect team performance," in *Computational Organization Theory*, K. M. Carley and M. J. Prietula, Eds. Hillsdale, NJ, 1994.

[43] Z. Lin and K. M. Carley, "Proactive and reactive: An analysis of the effect of agent style on organizational decision-making performance," *Int. J. Intell. Syst. Account. Finance, Manage.*, vol. 2, no. 4, pp. 271–288, 1993.

[44] O. Maimon and D. Braha, "On the complexity of the design synthesis problem," *IEEE Trans. Syst., Man, Cybern.*, vol. 26, pp. 142–151, Jan. 1996.

[45] C. Malandraki and M. S. Daskin, "Time dependent vehicle routing problems: Formulations, properties, and heuristic algorithms," *Transp. Sci.*, vol. 26, no. 3, pp. 185–199, 1992.

[46] S. Martello and P. Toth, *Knapsack Problems: Algorithms and Computer Implementations*. New York: Wiley, 1990.

[47] I. Mayk and I. Rubin, "Paradigms for understanding C3, anyone?," in *Science of Command and Control: Coping With Uncertainty*, S. E. Johnson and A. H. Levis, Eds. Washington, DC: AFCEA International Press, 1988, pp. 48–61.

[48] A. Mingozzi, L. Bianco, and S. Ricciardelli, "Dynamic programming strategies for the traveling salesman problem with time window and precedence constraints," *Oper. Res.*, vol. 45, no. 3, pp. 365–377, 1997.

[49] J. D. Papastavrou and M. Athans, "On optimal distributed detection architectures in a hypothesis testing environment," *IEEE Trans. Automat. Contr.*, vol. 37, pp. 1154–1169, 1992.

[50] A. Pete, D. L. Kleinman, and K. R. Pattipati, "Structural congruence of tasks and organizations," in *Proceedings of the 1994 Symposium on Command and Control Research and Decision Aids*. Monterey, CA: NPS, 1994, pp. 168–175.

[51] A. Pete, K. R. Pattipati, D. L. Kleinman, and Y. N. Levchuk, "An overview of decision networks and organizations," *IEEE Trans. Syst., Man, Cybern.*, vol. 28, pp. 172–192, May 1998.

[52] M. Pinedo, *Scheduling: Theory, Algorithms, and Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1995.

[53] D. Reece, "Computer controlled hostiles for TTES," in *Proc. Defense Modeling and Simulation Office Individual Combatant Workshop*, Alexandria, VA, July 1–2, 1996.

[54] A. Reibman and L. W. Nolte, "Design and performance comparison of distributed detection networks," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-23, pp. 78–79, Nov. 1987.

[55] F. Roberts, *Measurement Theory, With Applications to Decision-Making, Utility and the Social Sciences*. Reading, MA: Addison-Wesley, 1979.

[56] F. J. Roethlisberger and W. J. Dickson, *Management and the Worker*. Cambridge, MA: Harvard Univ. Press, 1939.

[57] A. P. Sage, "Behavioral and organizational considerations in the design of information systems and processes for planning and decision support," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-11, pp. 640–678, Sept. 1981.

[58] S. F. Sampson, "A novitiate in a period of changes: An experimental and case study of relationships," Ph.D. dissertation, Cornell Univ., Ithaca, NY.

[59] W. R. Scott, *Organizations: Rational, Natural, and Open Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1981.

[60] B. Shirazi, M. Wang, and G. Pathak, "Analysis and evaluation of heuristic methods for static task scheduling," *J. Parallel Distrib. Comput.*, vol. 10, pp. 222–232, 1990.

[61] W. K. Stevens and L. Parish, "Representation of unit level behavior in the naval simulation system (NSS)," in *Proc. Defense Modeling and Simulation Workshop on Unit Level Behavior*, Alexandria, VA, Aug. 7–8, 1996.

[62] L. V. Tavares and J. Weglarz, "Project management and scheduling: A permanent challenge for OR," *Eur. J. Oper. Res.*, vol. 49, no. 1–2, 1990.

[63] S. L. Van de Velde, "Duality-based algorithms for scheduling unrelated parallel machines," *ORSA J. Comput.*, vol. 5, no. 2, pp. 182–203, 1993.

[64] P. Vincke, *Multicriteria Decision-Aid*. New York: Wiley, 1992.

[65] S. Wasserman and K. Faust, *Social Network Analysis: Methods and Applications* Cambridge, U.K., 1994.

[66] B. Wellman, "Network analysis: Some basic principles," *Sociological Theory*, vol. 1, pp. 155–200, 1991.

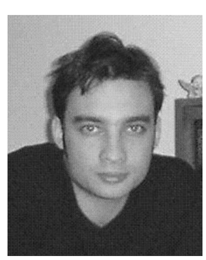
[67] B. Wellman and S. Berkowitz, *Social Structures: A Network Approach*. Norwood, NJ: Ablex, 1988.

[68] H. C. White, S. A. Boorman, and R. L. Breiger, "Social structure from multiple networks: A statistical perspective," *Neural Comput.*, vol. I, pp. 425–464, 1976.

[69] G. Zweig, "An effective tour construction and improvement procedure for traveling salesman problem," *Oper. Res.*, vol. 43, no. 6, pp. 1049–1057, 1995.

[70] National Institute of Standards and Technology (NIST), "Federal information processing standards publication i83: integrated definition for function modeling (IDEFO)," National Technical Information Services, Dec. 1993.

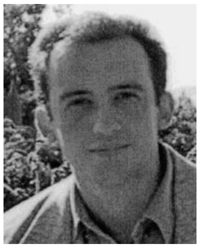
[71] National Institute of Standards and Technology (NIST), "Federal information processing standards publication i84: integrated definition for information modeling (IDEF1X)," National Technical Information Services, Dec. 1993.



Georgiy M. Levchuk was born in Kiev, Ukraine, on May 27, 1973. He received the B.S. and M.S. degrees in mathematics (with highest honors) from the Kiev Taras Shevchenko University in 1995, and is currently pursuing the Ph.D. degree in electrical engineering at the University of Connecticut, Storrs.

Since 2002, he has been with Aptima, Inc., Woburn, MA. His research interests include global, multiobjective optimization and its applications in the areas of organizational design and adaptation, and network optimization. Prior to joining the University of Connecticut, he held a Research Assistant position at the Institute of Mathematics, Kiev, and a Teaching Assistantship at Northeastern University, Boston, MA.

Mr. Levchuk received honorable mentions at both the 1999 and 2000 International Command and Control Research Symposia.

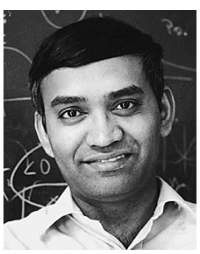


Yuri N. Levchuk was born in Kiev, Ukraine, on September 23, 1967. He received the B.S. and M.S. degrees in mathematics from the Kiev Shevchenko University in 1990, and the M.S. degree in mathematics from the University of Connecticut, Storrs, in 1997. He is currently pursuing the Ph.D. degree in electrical engineering at the University of Connecticut.

His research interests include multiobjective optimization of organizations and processes and its applications in the areas of human decision-making, organizational human-in-the-loop design, and adaptation. Since 1998, he has been with Aptima, Inc., Woburn, MA. Prior to joining Aptima, he was a Research Assistant at the Cyber Lab (the University of Connecticut) and a Research Assistant at Mathematics Research Lab (Kiev Shevchenko University).

Mr. Levchuk won the Best Technical Paper Award at the 1997 Command and Control Symposium, Washington, DC. Some of his other awards include a U.S. Government sponsored trip to Stockholm, Sweden, an honorable mention at 1996 and 1999 C2 symposiums, and an Academic Merit Scholarship from the University of Connecticut in 1993–1994.

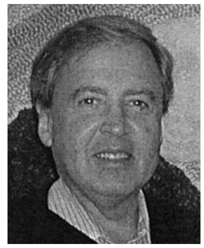
Jie Luo received the B.S. and M.S. degrees in electrical engineering (with honors) from Fudan University, Shanghai, China, in 1995 and 1998, respectively. He is currently pursuing the Ph.D. degree at the Department of Electrical and Computer Engineering, the University of Connecticut, Storrs. His research interests are in the general areas of wireless communication and signal processing.



Krishna R. Pattipati (S'77–M'80–SM'91–F'95) received the B.Tech. degree in electrical engineering (with highest honors) from the Indian Institute of Technology, Kharagpur, in 1975, and the M.S. and Ph.D. degrees in systems engineering from the University of Connecticut, Storrs, in 1977 and 1980, respectively.

From 1980 to 1986, he was with Alphatech, Inc., Burlington, MA. Since 1986, he has been with the University of Connecticut, where he is a Professor of Electrical and Computer Engineering. His current research interests are in the areas of adaptive organizations for dynamic and uncertain environments, multiuser detection in wireless communications, signal processing and diagnosis techniques for power quality monitoring, multiobject tracking, and scheduling of parallelizable tasks on multiprocessor systems. He has published over 230 articles, primarily in the application of systems theory and optimization (continuous and discrete) techniques to large-scale systems. He has served as a Consultant to Alphatech, Inc. and IBM Research and Development, and is a Cofounder of Qualtech Systems, Inc., a small business specializing in advanced integrated diagnostics software tools.

Dr. Pattipati was selected by the IEEE Systems, Man, and Cybernetics Society as the Outstanding Young Engineer of 1984, and received the Centennial Key to the Future Award. He was elected a Fellow of the IEEE in 1995 for his contributions to discrete-optimization algorithms for large-scale systems and team decision-making. He has served as the Editor-in-Chief of IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B: CYBERNETICS during 1998–2001, Vice-President for Technical Activities of the IEEE SMC Society (1998–1999), and as Vice-President for Conferences and Meetings of the IEEE SMC Society (2000–2001). He received the Andrew P. Sage Award for the Best SMC Transactions Paper for 1999. He also won the Best Technical Paper Awards at the 1985, 1990, and 1994 IEEE AUTOTEST Conferences, and at the 1997 Command and Control Conference.



David L. Kleinman (S'62–M'68–SM'93–F'94) received the B.E.E. degree from Cooper Union, New York, NY, in 1962, and the S.M. and Ph.D. degrees in electrical engineering from Massachusetts Institute of Technology (MIT), Cambridge, in 1963 and 1967, respectively.

From 1967 to 1971, he was with Bolt Beranek and Newman, Inc., Cambridge, where he was a pioneer in the application of modern control and estimation theory to develop and validate an analytical model for describing human control and information processing performance in manned vehicle systems. From 1971 to 1973, he established and directed the Cambridge office of Systems Control, Inc., where he led applied research projects in both manual and automatic control. In 1973, he joined the Department of Electrical and Systems Engineering at the University of Connecticut, Storrs, and founded the CYBERLAB—a laboratory for empirical research in cybernetic systems. He is currently a Professor Emeritus at the University of Connecticut and also holds a part-time appointment with the Command and Control Academic Group at the Naval Postgraduate School, Monterey, CA. In addition, he is a Cofounder and Chairman of the Board of Aptima, Inc. and Qualtech Systems, Inc.

Dr. Kleinman's efforts in human decision-making have involved modeling human/team performance and response characteristics in real-time multitask sequencing and resource allocation environments. His current research, which is funded through grants from the Office of Naval Research, is in the area of team decision-making and coordination in dynamic task environments, and structural adaptation processes in human organizations. He was a Technical Committee Chair (Human Decision-Making), and the Vice-President of Finance and Long-range Planning, for the IEEE SMC Society. He was the Program Chairman for the 1989 IEEE International Conference on Systems, Man, and Cybernetics. In 1993, he was elected a Fellow of the IEEE.