Dynamic and Distributed Allocation of Resource Constrained Project Tasks to Robots

Sanem Sariel[1] and Tucker Balch[2]

[1] Istanbul Technical University, Computer Engineering Dept., Istanbul TURKEY
`sariel@cs.itu.edu.tr`
[2] Georgia Institute of Technology, College of Computing, Atlanta, GA, USA
`tucker.balch@cc.gatech.edu`

Abstract. In this work, we propose a dynamic task selection scheme for allocating real-world tasks to the members of a multi-robot team. Tasks in our research are subject to precedence constraints and simultaneous execution requirements. This problem is similar to the Resource Constrained Project Scheduling Problem (RCPSP) in operations research. Particularly, we also deal with the missions that may change their forms by introducing new online tasks during execution making the problem more challenging besides the real world dynamism. Unpredictability of the exact processing times of tasks, unstable cost values during runtime and inconsistencies due to uncertain information form the main difficulties of the task allocation problem for robot systems. Since the processing times of the tasks are not exactly known in advance, we propose a dynamic task selection scheme for the eligible tasks instead of scheduling all of them to eliminate the redundant calculations. In our approach, globally efficient solutions are attained by the mechanisms for forming priority based rough schedules by tentative coalition commitments and selecting the most suitable tasks from these schedules. The approach is distributed and computationally efficient.

1 Introduction

In this work, we propose the *Dynamic Priority-based Task Selection Scheme* (DPTSS) embedded in our framework, **D**istributed and **E**fficient **M**ult**i** **R**obot - **C**ooperation **F**ramework (DEMiR-CF), for allocating complex tasks with precedence constraints and simultaneous execution requirements by a multi robot team. Robustness is provided through the integrated *Plan B Precaution Routines* [1]. DEMiR-CF is evaluated in three different domains, [2], [3], [4]. In this article, we present the formal details of our task allocation approach and the simulation scenarios on the US NAVY's simulator for dynamic tasks and events.

M+ [5] is one of the earlier cooperation schemes addressing many real time issues including plan merging paradigms. One of the latest works, Zlot's [6] task-tree auction method combined with the combinatorial auction based task allocation scheme, TraderBots [7], is suitable for the complex tasks represented as and/or trees. Lemarie et al. proposes a task allocation scheme for multi-UAV cooperation by balancing workloads [8]. Gancet [9] proposes a coordination framework addressing the planning and allocation issues. These systems use the auction based task allocation approach which is scalable and robust. However as Dias et al. report, still there are not certain procedures for re-planning, changing decomposition of tasks, rescheduling during execution [10]. Our main objective is to design the certain components in an integrated cooperation framework to deal with these issues and make it usable for as many domains as possible.

We formulate the general multi-robot multi task allocation problem as a Resource Constrained Project Scheduling Problem (RCPSP) [11]. Unpredictability of the exact processing times of tasks, the unstable cost values during runtime and the inconsistencies due to the uncertain information form the main difficulties of the task allocation problem for the robot systems. To cope with these issues, we propose a dynamic task selection scheme for the eligible tasks instead of scheduling all of them to eliminate the redundant efforts. Particularly, we also deal with the real-world missions that may change their forms by introducing new online tasks during the execution which makes the problem more challenging besides the real world dynamism. Our generic task representation is suitable for multi-robot teams and relaxes many assumptions for the real world tasks. DPTSS provides a way to find a solution to the problem from a global perspective by the mechanisms for forming priority based rough schedules and selecting the most suitable tasks from these schedules. Rough schedules are formed by the tentative coalition commitments which are agreed upon by the robots for the tasks with simultaneous execution requirements. Therefore since the allocations are not made from scratch, the scheduling costs are reduced and the communication requirements are kept at minimum as much as possible.

2 Problem Statement

We formulate the multi-robot task allocation problem for complex missions as a version of the well known NP-Hard Resource Constrained Project Scheduling Problem (RCPSP) in operations research [11]. The adapted version of the formulation for our multi robot task allocation problem on project tasks is given as follows. A complex mission consists of a set of tasks $T = \{t_1, ..., t_n\}$ which have to be performed by a team of robots $R = \{r_1, ..., r_m\}$. The tasks are interrelated by two type of constraints. First, the precedence constraints are defined between activities. These are given by the relations $t_i \prec t_j$, meaning that the task t_j cannot start before the task t_i is completed. Second, a task t_i requires a certain set of capabilities $reqcap_i$ and certain number of robots (resources) $reqno_i$ to be performed. We relax the limitation on $reqno_i$ by allowing its change during the task execution based on the requirements which provides a more realistic way of representing the real-world tasks. Therefore different alternative solutions may be found to allocate the tasks to the robots based on the environmental factors. Based on the given notation, the Scheduling Problem (ScP) is defined as determining starting times of all the tasks in such a way that: the total $reqno_i$ for each task t_i is less than or equal to the number of available robots ($R_{Sj} = \cup r_j$) with $reqcap_i \subseteq cap_j$ (Condition-1, $C1$). The given precedence conditions (Condition-

2, $C2$) are fulfilled , and the makespan $C_{max} = max(C_i),\ 1 \leq i \leq n$ (Objective, O) is minimized, where $C_i = S_i + p_i$ is assumed to be the completion of task t_i, where S_i is the actual starting time and p_i is the actual processing time respectively. It's not always possible to expect the exact processing times (p) of the tasks of real world missions in which robots involve. However to form a complete schedule, it is necessary to make an approximation in terms of the best knowledge available. Since the schedules are subject to change, we propose a way to allocate the tasks incrementally to the robots without ignoring the overall global solution quality instead of scheduling all the tasks. Therefore the main objective becomes determining which robot should do in a precedence and resource feasible manner whenever a new task needs to be assigned, instead of scheduling all the tasks from scratch. Although it is not a concern during the assignments are made, preemption (*i.e.* yielding) is possible to maintain the solution quality and to handle the failures during the execution. The main problem that we try to solve is given as follows: The Selection Problem (SlP) is determining the next action to select (either being idle or executing a task) for each robot in such a way that the $C1$ and the $C2$ are fulfilled and the O is minimized.

Missions can be represented by directed acyclic graphs (DAG) where each node represents a task (with requirements) and the directed arcs (conjunctive arcs) represent the precedence constraints among them. A sample graph for a small size mission for moving the boxes to a stamping machine and dropping them in a given order, then cleaning the room is given in Figure 1. Before dropping boxes into the mailbox, they should first be moved to the stamping machine. The room can only be cleaned after both boxes are moved. Since the box 1 is heavy, two robots ($reqno$) are needed to move and drop the box. Although this graph shows the relationships on the dependencies among tasks, it does not show which robot performs which task in sequence.

The following definitions are needed for our formulation to the solution. Intuitively, robots do not deal with the ineligible tasks (T_ϕ) as a union of tasks that are already achieved or that are not eligible from the capabilities perspective. The eligible tasks ($T_{Ej} = T \setminus T_\phi$) for the robot r_j consists of only the considerable tasks that are neither in execution (T_{ie}) nor achieved. P_i is defined as the set of all predecessor tasks of the task t_i. We define an active task set as: $T_{Aj} = \{\{t_i\} \mid reqcap_i \subseteq cap_j,\ P_i \text{ is completed},\ 0 < i \leq n\}$, $(T_{Aj} \subseteq T_{Ej})$, whereas an inactive task set $T_{Ij} = T_{Ej} \setminus T_{Aj}$ contains the tasks for which the robot r_j, $reqcap_i \subseteq cap_j$, but the precedence constraints are not satisfied yet. Incremental allocation is achieved in our system by means of the dynamic selection of a suitable task from T_{Aj} by taking into consideration of the T_{Ej}.

We call a multi-robot group (sub-team) formed to execute a particular task simultaneously and synchronously as a coalition [12]. In this research, we particularly deal with the types of tasks that require same type of capabilities within a coalition to execute a task although the overall mission requires a heterogeneous team and diverse capabilities. Shehory and Krauss [13] present an algorithm for coalition formation in cooperative multi agent systems. During the coalition value calculations, the capabilities of agents are taken into consideration. In multi robot systems, the cost values are a function of not only the capabilities but also the physical conditions, which change during execution. Vig and Adams [14] state the differences of the multi-robot and the multi-agent coalition formation issues from the sensor possessive point of view. Another important factor in multi-robot systems is the changing cost values during runtime.

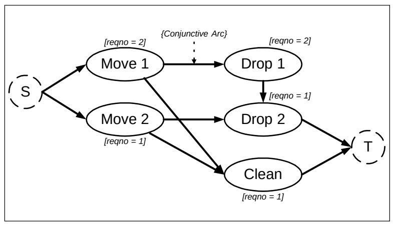


Fig. 1. The Directed Acyclic Mission Graph for dropping the stamped boxes into the mailbox in an order. The boxes (1,2) are moved to the stamping machine and then dropped. After the boxes are moved to the stamping machine, the room can be cleaned.

3 Proposed Approach

DEMiR-CF is for multi-robot teams that must cooperate/coordinate to achieve complex missions including tightly coupled tasks that require diverse capabilities and collective work [1]. It combines *auctions*, *coalition maintenance* and recovery routines called *Plan B precaution routines* to provide an overall system that finds (near-) optimal solutions in the face of noisy communication and robot failures.

3.1 The Dynamic Priority-based Task Selection Scheme (DPTSS)

In our approach, the instantaneous, precedence and resource feasible decisions are made by the robots' global time extended view of the problem from the local perspectives. While completion of the mission is the highest priority goal objective, additionally other performance objectives can also be achieved. The time extended consideration is achieved through forming the rough schedules by the robots. Since the schedules are highly probable to change in dynamic environments and furthermore robots also have the real time burdens of path planning, mapping etc., the schedules formed in our approach are tentative and constructed by computationally cheap methods.

A critical task is a task that has inflexibility from the resources point of view and the robot is suitable for that task. Level of a node (task) represents the depth of the node in the mission graph in reversed order. The level of a node is assigned as the value incrementing by one from the maximum level of the the succeeding nodes (connected by the conjunctive arcs). The coalition reservation tables are built for the critical tasks representing the committed robots for the execution. Depending on the number of entries, the possibility of mission completion can be attained. The reservation tables present the future commitments although they are roughly determined. Each robot generates its rough schedule as a dynamic priority queue by considering critical task set (T_C), the coalition reservation entries, the eligible tasks (T_E), the conjunctive arcs and the requirements. Since each robot r_j has different capabilities, the eligible sets T_{Ej} and the priority queue entries may be different. Sometimes the uncertain information (*e.g.* related to a local online task) or the unexpected events (*e.g.* detection of the fuel leakage) may result in this difference although the capabilities are the same. The rough schedule generation is implemented by the Algorithm 1. $curcs_j$ represents the remaining capacity of robot r_j and $reqcs(i)$ represents the required capacity for task t_i in terms of the consumable resources (e.g fuel). The priority queue

Algorithm 1 Rough Schedule Generation Algorithm

```
$t_s = \phi$; $R = curcs_j$; $T_{Rj} = \phi$
$C = T_{Ej} \setminus T_{Aj}$ prioritized by the level values in descending order (the tie breaking rules: type priority and $reqno$)
for each $t_i \in C$ and $t_i \in T_{Cj}$ do
    $R = R - reqcs(i)$
    if $R < 0$ then
        unachievable = true; break
    else
        $T_{Rj} = T_{Rj} \cup t_i$
    end if
end for
if (unachievable || $R - reqcs(top(T_{Aj})) > 0$ || $top(T_{Aj}) \in T_{Cj}$ then
    $t_s = top(T_{Aj})$
end if
```

is formed by first taking into consideration of the conjunctive arcs of the task graph. If there are no online tasks, or invalidations, the order of the tasks which are connected by the conjunctive arcs remains the same in the priority queue although there may be additional intermediate entries in the queue. The dynamic task selection is implemented by by using the requirements of the rough schedule (Algorithm 2). The tie breaking rules while forming the active list (T_A) is given from the highest to the lowest importance as follows: The least flexibility (*reqno*), the level value of the node, and the id. The fundamental decision that each robot must make is selecting the most suitable action for a task from a set of active tasks (T_A) by considering T_E. The four different decisions are: keeping execution of the same task (if any), joining to a coalition, forming a new coalition to perform a free task and being idle.

In DEMiR-CF, the standard auction steps of CNP [15] are implemented to announce the intentions on the task execution and select the *reqno* number of robots for a coalition in a cost-profitable, scalable and tractable way. Additionally *Plan B* precaution routines are added to check validness in these negotiation steps. Each robot intending to execute a task announces an auction after determining the rough schedules.

Maintaining the coalition reservation entries are implemented by negotiations. The robots maintain the coalition reservation entries by proposing the coalition commitment requests to the specific robots that can execute the corresponding task. The coalition reservations only show the tentative agreements which can be canceled in future.

Each robot keeps the models of the tasks and the other robots in their world knowledge to track the internal and external inconsistencies. The complete set of precaution routines to handle several contingencies can be found in [1].

Algorithm 2 DPTSS Algorithm for robot r_j

```
Determine the $T_{Ej}$, $T_{Aj} \subseteq T_{Ej}$ and $T_{Cj} \subseteq T_{Ej}$
Maintain the coalition reservation entries for the tasks in $T_{Ej}$
Generate the Rough Schedule ($T_{Rj}$)
Select the active task $t_S$ from $T_{Aj}$ to process and perform one of the following
if $t_s \neq \phi$ then
  if $t_s$ is the current task then
    Continue to the current execution
  else
    Offer an auction for forming a new coalition or directly begin execution
  end if
else
  if $R + top(T_{ie}) \leq curcs_j$ and profitable to join a coalition then
    Join a coalition
  else
    Be idle
  end if
end if
```

4 Experimental Results

In our earlier work, we apply the rough schedule generation scheme for the MTSP (open loop-Multiple Traveling Salesman Problem) on multi-robot systems [3]. Since the rough schedules are generated tentatively, quality of the solution is improved over time if the initial quality is degraded. Furthermore, an incremental assignment approach saves a considerable computation overhead. In this work, we evaluate our approach in the US NAVY's realistic simulator [16]. Particularly in this experiment, the mission consists of the online tasks, generation time of which are not known in advance by the robots (Autonomous Underwater Vehicles). The overall mission is searching a predefined area and protecting the deployment ship from any hostile intents. The initial graph of the application

Table 1. The Cost Evaluations for the tasks of the application domain

Task Type	Cost Function	Taken Action
Search Task	Distance to the region interest points [4]	In depth analysis is needed.
Intercept Task	Expected time to achieve the task: $t_E = E[dist(r_j, t_i)]/E[speed - diff(r_j, t_i)]$	Immediate response is needed. One step auction or direct execution is applied.

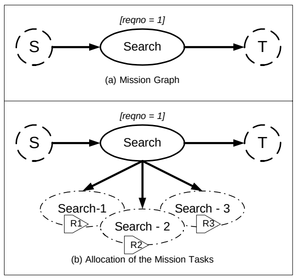
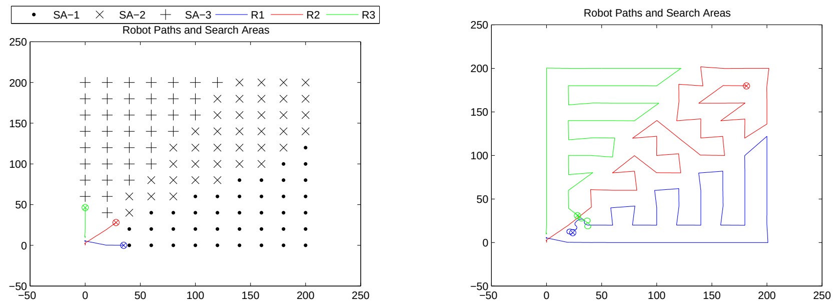


Fig. 2. Initial Mission graph consists of only Search Task.

mission is given in Figure 2. Initially the mission consists of only the Search Task. Although $reqno = 1$ for this task, since there are no other tasks and the robots have enough fuel capacities, they execute the task as a coalition and divide the area to search. The Search Task execution with three robots and the corresponding search areas are illustrated in Figure 3. The robots patrol the subareas which are determined after the negotiations [4]. Therefore, although there is only one task on the higher level, the robots create instances of the Search Task (Search 1-3) as if each instance is another separate task. If there are no hostile intentions, the robots only search the area.

Whenever a hostile diver is detected by the robots, a related interception task is generated. The execution trace after detection of the hostile diver is illustrated in Figure 4. R2 chases performing the search task and immediately switches to the Intercept Task. The hostile diver attacks to R2 by using its missiles. Therefore R2 needs to return back to the deployment area while R1 takes control of the Intercept Task. R1 can deter the diver but waits until the threat entirely disappears. The evolving mission graph is illustrated in Figure 5. The robots may need to generate local tasks (*e.g. Repair/Refuel Task,*) as in Figure 5 (d) making the graphs different even when they work cooperatively for the same objective (Figure 5 (c-d)). In Figure 5 (c), although executing the Intercept Task, R1 can make a coalition commitment assuming it will succeed in a predefined time (described as TBD), R2 cannot make any coalition commitment for the search task because its future operations depend on its recovery time.

Cost evaluation for the tasks are implemented accordingly depending on the task. While the robots try to optimize the fuel levels for the Search Task, the Intercept Task requires immediate response and time minimization (Table 1). Cost evaluation for the search task is implemented by dividing the search area



(a) (b)

Fig. 3. (a) Mission execution begins. The overall area is divided into regions related to the generated task instances. (b) Robots patrol the area in the corresponding regions.

into regions and evaluating the distance values for the interest points [4]. For the intercept task, the expected time to achieve (intercept the diver) the task is taken as the cost value. The Intercept Task is assumed to be achieved whenever the hostile threat is believed to be disappeared. The emergency tasks are directly executed. However, in this case, parallel executions may occur and should be resolved. This facility is provided in our framework by the *Plan B* precaution routines. In a sample scenario with limited communication ranges, the parallel executions arise for the Intercept Task as in Figure 6. However these inconsistencies are resolved by the *Plan B* precaution routines whenever robots enter into the communication range.

5 Conclusion

In this work, we present our dynamic and distributed task selection scheme (DPTSS) embedded in our generic cooperation framework, DEMiR-CF. The dynamic task selection scheme ensures that the instantaneous, precedence and resource feasible decisions are made by the robots' global time extended views of the problem from the local perspectives. The framework combines a distributed auction based allocation method and Plan B precaution routines to handle contingencies and real world limitations and to maintain the high solution quality with the available resources. The preliminary results on complex missions, as presented in this paper, reveal the integration of real-world task allocation and execution; immediate and effective handling of the online tasks and events and the solution quality maintenance performance of DEMiR-CF is promising.

References

1. Sariel, S., Balch, T.: A distributed multi-robot cooperation framework for real time task achievement. In: The Intl. Symp. on Distributed Autonomous Robotic Systems (DARS). (2006)

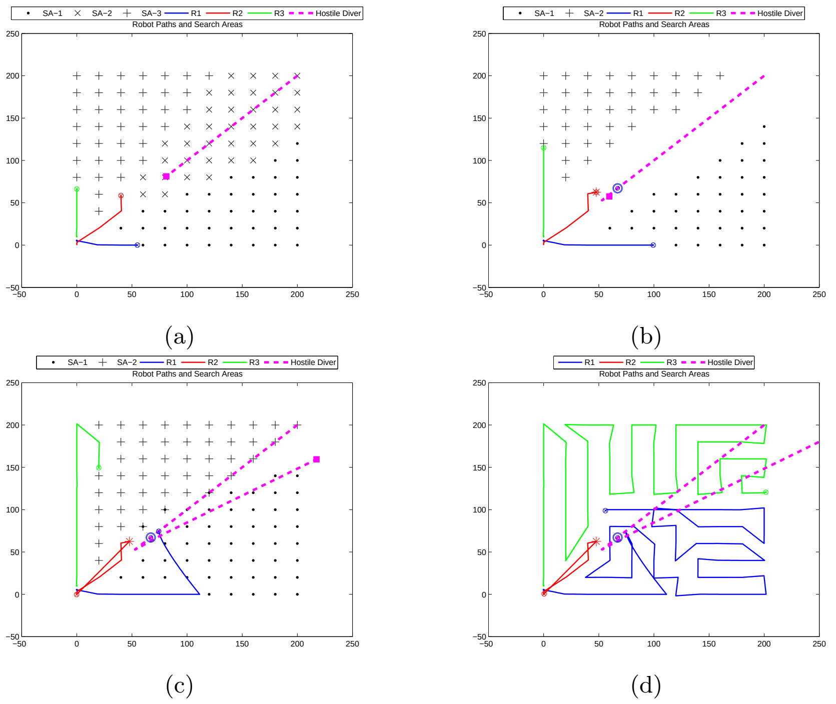


Fig. 4. A sample execution trace under highly dynamic task situations. (a) The robots begin searching the area. (b) R2 recognizes the hostile intent. After detection, the hostile vehicle attacks to R2. (c) R2 returns to the deployment ship. R1 takes control of the intercept task. (d) R1 and R3 continue to searching the area.

2. Sariel, S., Balch, T., Erdogan, N.: Robust multi-robot cooperation through dynamic task allocation and precaution routines. In: The International Conference on Informatics in Control, Automation and Robotics (ICINCO). (2006)
3. Sariel, S., Balch, T.: Efficient bids on task allocation for multi-robot exploration. In: The 19th International FLAIRS Conference. (2006)
4. Sariel, S., Balch, T., Stack, J.R.: Empirical evaluation of auction-based coordination of AUVs in a realistic simulated mine countermeasure task. In: The Intl. Symp. on Distributed Autonomous Robotic Systems (DARS). (2006)
5. Botelho, S.C., Alami, R.: M+: a scheme for multi-robot cooperation through negotiated task allocation and achievement. In: IEEE Intl. Conf. on Robotics and Automation (ICRA). (1999)
6. Zlot, R., Stentz, A.: Market-based multirobot coordination for complex tasks. Intl. J. of Robotics Research **25**(1) (2006)
7. Dias, M.: TraderBots: A New Paradigm for Robust and Efficient Multirobot Coordination in Dynamic Environments. Phd thesis, Carnegie Mellon University (2004)
8. Lemarie, T., Alami, R., Lacroix, S.: A distributed task allocation scheme in multi-uav context. In: IEEE Intl. Conf. on Robotics and Automation (ICRA). (2004)
9. Gancet, J., Hattanberger, G., Alami, R., Lacroix, S.: Task planning and control for a multi-uav system: Architecture and algorithms. In: IEEE/RSJ Intl. Conf. on

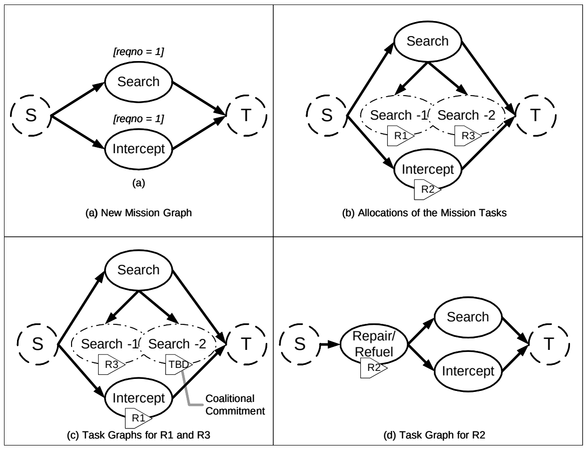


Fig. 5. Mission graph and allocations evolving through time accordingly

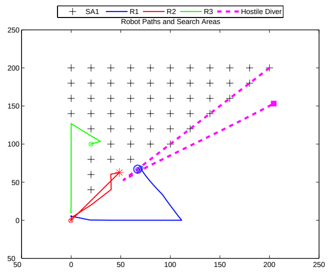


Fig. 6. Under limited communication ranges, parallel executions may occur to be resolved. R3 switches to the search task while R1 executes the intercept task.

Intelligent Robots and Systems (IROS). (2005)
10. Dias, M.B., Zlot, R.M., Kalra, N., Stentz, A.: Market-based multirobot coordination: A survey and analysis. Technical Report CMU-RI-TR-05-13, Carnegie Mellon University (2005)
11. Brucker, P.: Scheduling and constraint propagation. Discrete Applied Mathematics **123** (2002) 227–256
12. Horling, B., Lesser, V.: A survey of multi-agent organizational paradigms. The Knowledge Engineering Review **19**(4) (2005) 281–316
13. Shehory, O., Kraus, S.: Methods for task allocation via agent coalition formation. Artificial Intelligence **101** (1998) 165–200
14. Vig, L., Adams, J.A.: Issues in multi-robot coalition formation. In: Multi-Robot Systems. From Swarms to Intelligent Automata. Volume III. (2005) 15–26
15. Smith, R.G.: The contract net protocol: High level communication and control in a distributed problem solver. IEEE Trans. on Computers C- **29**(12) (1980)
16. (ALWSE:http://www.ncsc.navy.mil/Capabilities_and_Facilities/Capabilities/ Littoral_Warfare_Modeling_and_Simulation.htm)