Profiles, Proxies, and Assumptions: Decentralized, Communications-Resilient Planning, Allocation, and Scheduling*

Ugur Kuter,[1] Brian Kettler,[2] Katherine Guo,[2] Martin Hofmann,[2] Valerie Champagne,[3] Kurt Lachevet,[4] Jennifer Lautenschlager,[2] Robert P. Goldman,[1] Luis Asencios,[2] Josh Hamell[1]
[1]SIFT, LLC, [2]Lockheed Martin, ATL, [3]PatchPlus Consulting, LLC, [4]AFRL Rome
{ukuter,rpgoldman,jhamell}@sift.net,
{brian.p.kettler,jinhong.guo,martin.hofmann,jennifer.lautenschlager,luis.a.asencios.reynoso}@lmco.com,
vchampagne@patchplusconsulting.com, kurt.lachevet@us.af.mil

Abstract

Degraded communications are expected in large-scale disaster response and military operations, which nevertheless require rapid, concerted actions by distributed decision makers, each with limited visibility into the changing situation and in charge of a limited set of resources. We describe LaPlata, a novel architecture that addresses these challenges by separating mission planning from allocation/scheduling for scalability but at the cost of some negotiation. We describe formal algorithms that achieve near-optimal performance according to mission completion percentage and subject matter expert review: assumption-based planning and replanning, profile-assisted cooperative allocation, and schedule negotiation. We validate our approach on a realistic problem specification and compare results against subject matter expert solutions.

Introduction

In the U.S. Air Force, Decentralized Command and Control (C2) of large-scale, multi-day air operation campaigns (Hostage III and Broadwell Jr. 2014) follow the tenet of centralized control and decentralized execution. Recent events have led to an emphasis of decentralized C2, where communications links among forward planning and execution nodes are either interrupted by natural disaster or an adversary capable of communications denial, e.g., extensive jamming. If forward nodes were to wait for communications to be reestablished, they would miss critical time windows. It is during these moments that decentralized control, not traditional centralized control, provides a resilient C2 architecture that maintains the initiative in the contested environment.

Existing distributed and multi-agent planning systems (Torreño et al. 2017) typically focus on deterministic planning methods that distribute the input planning task among multiple agents. Decentralized C2 mission planning must involve independent planners and reasoners that are responsible for accomplishing different tasks under large-scale uncertainty, while communicating and coordinating their outcomes. Under severely degraded communications, mission planning, resource allocation, and scheduling may not converge to an optimal or even attainable solution. The large scale (a thousand missions per day), the dependencies between distributed decisions, and the high degree of configurability of resources exacerbate the plan optimization challenge. These planners must also operate under assumptions about peer decisions as resource allocation and scheduling cannot be modeled as finite, logical formalisms. On the other hand, decentralized multi-agent systems (Seuken and Zilberstein 2008) address partial-observability and uncertainty during planning, but these approaches cannot scale up to large-scale C2 operations and use closed-world formalisms.

We describe LaPlata (Living Air Operations Planning Triggered by Assessment), a novel service-oriented system designed and being developed for emerging decentralized C2 applications. LaPlata employs decentralized planning, plan coordination, adaptation, critiquing and distributed resource scheduling and allocation to address the challenges of decentralized C2. Our contributions are as follows —

- We describe our LaPlata architecture for rapid plan adaptation at multiple layers (i.e., tasks, actions, resources, and schedules for them) with minimal ripple effects. We discuss how our loosely-coupled approach to planning and allocation/scheduling negotiation results in termination under decentralized uncertainty.
- We describe LaPlata's novel assumption-based, decentralized hierarchical task network planning framework, called APA, that uniquely combines our plan generation, adaptation, and critiquing algorithms. APA manages assumptions about imperfect knowledge of dynamic demand, temporal information, and distributed constraints.
- We describe how LaPlata uses proxy knowledge to inform assumptions and estimate the status of the world and decisions at peer nodes in a decentralized auction-based resource allocation algorithm, called GAO. GAO performs global, dynamic, and distributed resource allocation under conditions of oversubscribed resources and potentially combinatorial number of scheduling options.
- We present our experimental evaluation and results that demonstrate the scalability and effectiveness of our approach for air campaigns and compare LaPlata solutions to those from subject matter experts. We evaluated LaPlata over several metrics, including planning run

*The work reported in this paper is sponsored by the Air Force Research Laboratory (AFRL) under contract FA8750-16-C-0182. Cleared for public release, no restrictions (AFRL PA 88ABW-2018-4459).

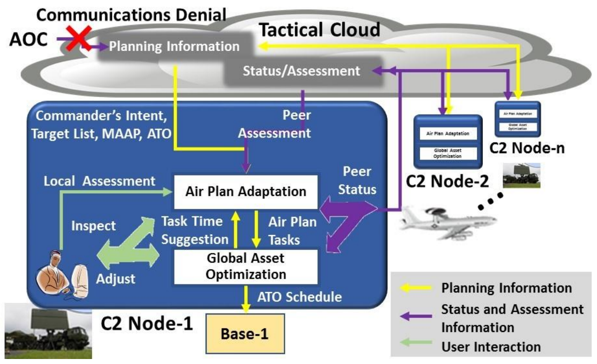


Figure 1: LAPLATA architecture.

times, scheduling and allocation success rate, and quality improvement percentages due to planning and allocation negotiations.

LAPLATA Architecture

According to the U.S. Department of Defense Dictionary of Military and Associated Terms, Command and Control (C2) is "the exercise of authority and direction by a properly designated commander over assigned and attached forces in the accomplishment of the mission." Thus, the two essential elements are, first, a commander who has the authority to assign missions and direct forces to accomplish them, and second, a system through which the commander can communicate with his or her forces and control their actions to achieve that mission. This paper focuses on the latter requirement and describes our novel decentralized C2 system, called LAPLATA, for military air operations.

Traditionally, C2 systems are centralized and have unitary allocation of decisions, resource allocations, constrained patterns of interaction, and tightly-controlled information flows. LAPLATA allows for a variable number of C2 nodes connected by a tactical cloud (not described here), which performs best effort status-sharing between nodes. Figure 1 shows the high-level illustration of our LAPLATA architecture. In LAPLATA, each C2 node contains an AIR PLAN ADAPTATION (APA) component for task planning and a GLOBAL ASSET OPTIMIZATION (GAO) component for resource allocation and scheduling. An Assessment Trigger component (not described here) interprets changing world and resource status and triggers plan adaptation.

The initial state of LAPLATA specifies a summary of available resources (e.g., available aircraft, sorties, sortie times and bases, and so on). With this input, LAPLATA's APA component starts planning a set of mission tasks, such as strikes against a set of targets and supporting tasks. Supporting tasks notably include Suppression of Enemy Air Defenses (SEAD) tasks, required for strike missions that do not have sufficient protection from adversary air defenses and, thus, must have a SEAD escort. Each mission specifies well-defined requirements (i.e., constraints) on the task. For example, a strike mission task is always associated with a start and end time as well as whether a stealth-capable SEAD escort is required or not. In addition to constraints on the mission task, each mission target can also specify time constraints for striking that specific target in the mission, geospatial constraints for the strike around the target, the level of any possible adversarial threat around the target, and the importance of that target within the mission.

Planned tasks identify options for specific capabilities to be used by the resources (i.e., specific aircraft types and configurations) and time windows for task execution. These tasks are then handed to GAO for allocation and scheduling. GAO receives from the Tactical Cloud detailed schedules of resource availability and available resource configurations at all C2 nodes, which it converts to *profiles*: histograms of availability over time. Resources are only available at certain periodic time intervals. The GAOs at all C2 nodes collaborate on (approximately) optimally allocating tasks to resources. GAOs will use resource profiles of peer C2 nodes to infer when other nodes can supply resources for tasks that cannot be fully allocated with local resources alone.

The output of GAO is a selection among the possible options and time windows for the successful execution of a mission. If GAO cannot allocate a resource to a task in the requested time window, it looks for and suggests to APA an alternate time window. APA then attempts to replan the task for the suggested time. Once primary tasks are allocated, APA creates the supporting tasks and requests allocation for them. APA postpones tasks for which required supporting tasks cannot be resourced to the next planning period, e.g., the next day.

The planning, adaptation, and allocation cycles converge when there are no mission tasks to be adapted by LAPLATA. The final output for the planning period is a list of mission tasks with their assigned resources for allocation.

LAPLATA Algorithms

Assumption-based, Decentralized Planning

LAPLATA's AIR PLAN ADAPTATION (APA) module is used to build, extend, and repair plans in response to new goals, user input, new situation/plan assessment information, or plan failure. APA is a loosely-coupled service architecture that provides the following basic planning services to achieve its assumption-based decentralized planning capability: (1) hierarchical planning, (2) plan critiquing via counter planning, and (3) dynamic plan repair. We have described our basic formalisms and approach in APA in (Kuter, Goldman, and Hamell 2018); we summarize those details here and describe our recent work.

Figure 2 shows a high-level description of APA. APA is built on the widely-used SHOP2 system[1] for generalized Hierarchical Task Network (HTN) planning (Erol, Hendler, and Nau 1994; Nau et al. 2003; Goldman 2006). SHOP2 is a modern, general-purpose and scalable HTN planner that has performed well in a broad range of applications (e.g., (Nau et al. 2005; Musliner et al. 2011; Kuter et al. 2015)). Building on SHOP2, APA provides predictability in decentralized planning across planning agents. HTN models seamlessly include planning information of other agents for planning

[1]https://sourceforge.net/projects/shop/files/SHOP2

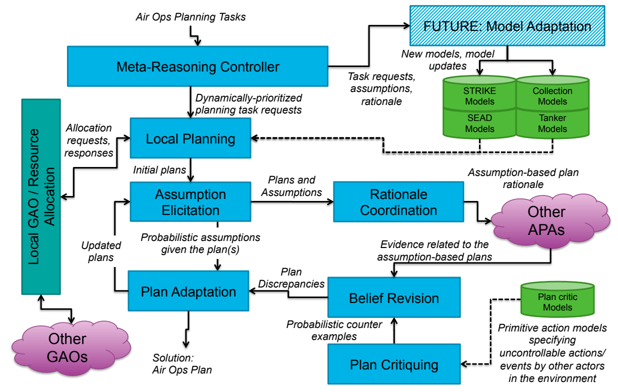

Air Ops Planning Tasks
Meta-Reasoning Controller
FUTURE: Model Adaptation
New models, model updates
Task requests, assumptions, rationale
STRIKE Models
Collection Models
SEAD Models
Tanker Models
Dynamically-prioritized planning task requests
Local GAO / Resource Allocation
Allocation requests, responses
Local Planning
Initial plans
Assumption Elicitation
Plans and Assumptions
Rationale Coordination
Assumption-based plan rationale
Other APAs
Updated plans
Probabilistic assumptions given the plan(s)
Evidence related to the assumption-based plans
Plan Discrepancies
Plan Adaptation
Belief Revision
Plan critic Models
Probabilistic counter examples
Primitive action models specifying uncontrollable actions/ events by other actors in the environment
Other GAOs
Solution: Air Ops Plan
Plan Critiquing

Figure 2: A high-level description of APA functional architecture.

and plan adaptation purposes. Local instances of APA modify fully specified plans by revisiting branching decisions at the most effective and least disruptive point in the hierarchy.

In Figure 2, SHOP2 is responsible for **localized planning** tasks. Planning agents (i.e., SHOP2 instances) coordinate their behaviors without communication by making assumptions about the constraints and goals of their peers in addition to using their own local planning constraints. These assumptions provides boundaries to a planning agent's space of plans. In APA, SHOP2 not only generates a solution plan (i.e., a sequence of actions) for a given desired task to be accomplished but it also generates a task-based plan rationale for that solution. The latter is based on the concept of a solution plan-tree in plan recognition (Geib and Goldman 2009) or in game-tree search (Russell and Norvig 2003).

A less viable alternative would be to constantly trade plans, constraints and negotiations among peers. For LaPlata, this has the major disadvantage of requiring constant, reliable communications among the planning agents. Furthermore, planning will deteriorate into a "lock-step" condition, in which decentralized planners must reason at the same time about the state of every other agent, negating many advantages of decentralized planning. Another option is for a planning agent to generate a plan, send it to the next agent in the pipeline, and so forth. This is less efficient and less robust because decentralized planning is not truly parallel and distributed. In highly complex and dynamic challenges facing LaPlata, this approach would be the worst: it would require backtracking over every agent's plans.

Plan critiquing via counter-planning uses planning techniques to try to break plans, triggering replanning to generate more robust plans that avoid vulnerabilities identified. Brittle plans may fail catastrophically, causing mission losses, and even minor failures may result in users losing faith in the system. Even if the system's plan is probabilistically the best, conditions may be encountered where the plan will break. To avoid such foreseeable, though unlikely, conditions APA uses *Murphy* counterplanning methods (Goldman, Kuter, and Schneider 2012) that attempt to "break" plans using models of potential plan contingencies (disturbances, in control-theoretic terms). Murphy can use any automated planning system to identify ways in which disturbances could cause the plan to fail in execution, and produce *counterexample traces* that demonstrate how failures could occur. Automated and human planners can use these traces as guidance in improving their plans. In that, Murphy is a planning analog to the use of model-checking systems to verify critical hardware and software systems.

Once Murphy produces probabilistic counterexamples for a plan, APA aggregates those counter examples into a *probabilistic causal model*, a directed graph in which each node represents an event (i.e., plan actions and uncontrolled contingency events) that may or may not occur (Lemmer and Gossink 2004; Kuter et al. 2004). By performing belief updates over the probabilistic causal models, APA revises the probability that an assumption is true, i.e., its confidence in that assumption, by probabilistically reasoning over Murphy's counter examples represented in the graphical model. If the confidence in an assumption drops below a given threshold, APA generates a discrepancy as the logically-negated condition of that assumption. This triggers plan adaptation in order to repair the plan and its rationale (i.e., the hierarchical plan tree). APA's plan adaptation is driven by the causal links the planner generates between tasks generated and decomposed during planning. In particular, APA traces the discrepancy to earliest task over the plan and the plan hierarchy by using the causal links in the plan hierarchy and starts to repair the subtree that is rooted at that task. Repairing that task may generate cascading conflicts to the assumptions in the rest of the original plan and APA recursively repairs those conflicts until a causally-sound plan is produced again.

Profile-assisted Cooperative Allocation

Global Asset Optimization (GAO) receives resourcing requests from APA and allocates resources to these tasks using a market-based optimization algorithm. Each GAO hosts an auctioneer agent which controls a number of local resource agents. Each resource agent represents an asset, e.g., an aircraft. When the auctioneer agent receives the tasks planned by its local APA planner, it attempts to optimally allocate the local assets it controls to these tasks. Optimization considers the priority of the task, the cost of the agent, and a number of soft constraints that express preferences.

Due to resource limitations and due to potential temporal misalignment between task requests and resource availability, some tasks remain unallocated. In this case, a GAO will first collaborate with other auctioneers to see if it is possible to borrow some of their assets and, failing that, will suggest an alternate time for the task. This is illustrated in Figure 3.

Each GAO will select one of its peers to ask for help if it failed to satisfy all the requests. The selection is based on the profiles of its peers that summarize their resource availability to maximize the chance that the task is eventually allocated while minimizing the number of requests flowing through the network. Note that the allocation of local tasks are not committed until the GAO has received requests from its peers, and the GAO will withdraw an asset from one of its own tasks when a high priority task from a peer GAO is received, since global optimal allocation provides the highest revenue. While this process causes some iteration, the ordering of tasks induced by their priority, which largely determines task reward, ensures that the algorithm terminates quickly. Finally, if there are tasks left unallocated after this step, GAO will generate and suggest to APA alternative time windows that one of its assets can accommodate.

Our winner determination (WD) algorithm, realized via greedy optimization for scalability, provides fine-grained, many-agent-to-many-task combinatorial allocation of the asset's capabilities to the capabilities required by the tasks. Our WD algorithm allocates tasks in coherent bundles to avoid infeasible solutions and enforces both hard and soft constraints. Hard constraints enforce task dependencies and temporal and spatial task requirements. Soft constraints honor task priorities, spare assets for anticipated future pop-up tasks, and minimize ripple effects of allocation changes. Our WD algorithm keeps track of expendable resources, such as munitions, across tasks assigned to a single asset. It allocates shareable resources, such as communications channels. It considers all options for configurable resources, such as aircraft with different munitions load-outs.

Because agents are cooperative, bid their true costs, and reward functions and capacities are globally known, the optimization performed by the auctioneer yields the benefits of Expressive Bidding and Allocation (Sandholm 2007), including reduced exposure risk, capability bundling, and side (capacity) constraints for bidders (resource agents). For example, the auctioneer will reduce the agent's cost when assigning the second of a pair of collocated tasks.

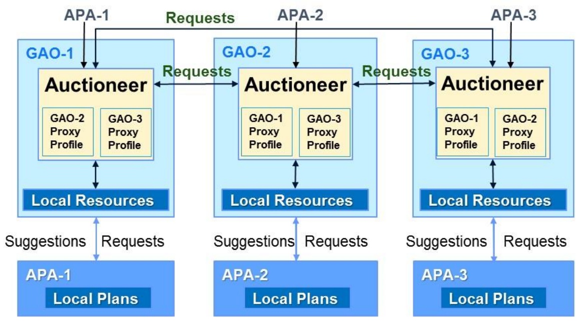


Figure 3: Distributed Resource Allocation and Scheduling: GAOs maintain profiles of peer GAO resource availability and selectively delegate task requirements to available peers.

Collaboration among peer GAOs increases global asset utilization. Borrowing assets from other GAOs is based on profiles each GAO maintains of other GAOs' assets. Static profiles are histograms of the asset types and their availability derived from the *unit contracts* of the C2 nodes. A unit contract specifies how may assets of each type are available and during which time intervals. Every GAO has a profile, p_{g_i}, for each peer g_i.

Definitions: The static asset profile for GAO $_i$ is $p_{g_i} = \{h^1_{g_i}, h^2_{g+i}, \ldots, h^N_{g_i}\}$ with N the number of asset types. $h^j_{g_i} = \{h^{j,1}_{g_i}, h^{j,2}_{g_i}, \ldots, h^{j,24}_{g_i}\}$ is a histogram built from the unit contracts of the given asset type j during a 24-hour period, our current planning horizon. The profile of asset requirements for each objective t_i is $p_{t_i} = \{h^1_{t_i}, h^2_{t_i}, \ldots, h^N_{t_i}\}$. The GAO, g_* from which a GAO will attempt to borrow assets for t_i is calculated as

$$\underset{g_k}{\text{argmax}}\{\sum_{j=1}^{N} h^j_{g_k} \cdot h^j_{t_i}\}$$

Every GAO has a profile, p_{g_i}, for each peer g_i. The profile gives the auctioneer high level knowledge of the peer GAOs despite the communication-challenged environment. Whenever communications are available, the GAOs communicate and update the profiles.

With static profiles it is likely that not all requests are satisfied by the selected peer GAO, because profiles do not take local demand for assets into account and may be out of date. In these cases, the peer passes the request to another GAO, and bookkeeping ensures that any GAO is asked at most once to allocate a task, which ensures process termination. In parallel, each GAO which fails to allocate the task attempts to suggest a different, feasible time slot. Thus, some overhead effort is expended on unsuccessful requests and on unnecessary alternative time suggestions, but in practice, the overhead is very small compared to the productive requests.

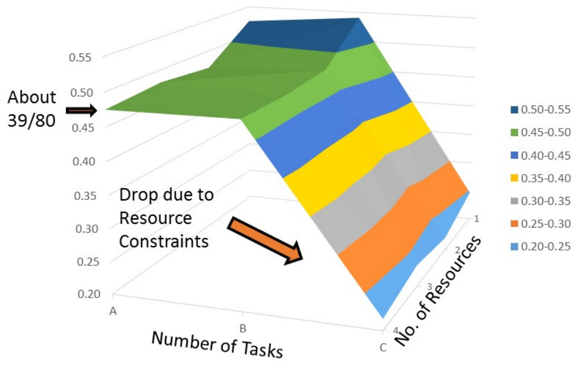
Experiments

The complexity of our domain precludes meaningful comparison to published results on standard research problems. Instead, we analyze performance of our system under varying conditions and demonstrate the performance improvements due to negotiation between the APA planner and the GAO allocator/scheduler.

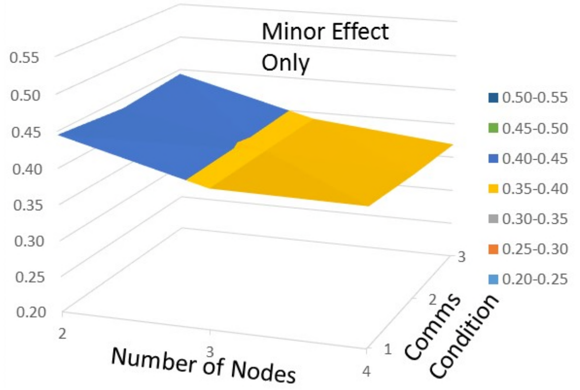
In our experiments LAPLATA plans and schedules aircraft sorties (individual flights) for each day of up to three days in advance, given a set of prioritized tasks and threats. Our subject matter experts tailored a realistic scenario to stress the LaPlata capabilities. Threats in close geographic proximity to task objectives require synchronized supporting tasks to be added and resourced. We vary four configuration items: the number of C2 nodes (from 2 to 4), the communications bandwidth between nodes (100Mb/s to 56 kb/s), the number of targets (from 24 to 250), and the number and variety of resources and their possible configurations (just enough, too few, or too many resources required for the set of tasks).

Metrics include % of input tasks planned, resourced, and scheduled and the processing time required. Failure to schedule occurs when no asset is available during the requested time. Resource scheduling is constrained by the task execution window (typically four hours) and by a window (typically 1/2 hour) for the start of asset use (the asset can be used for longer, but it can only start the task within a narrow window). We inspect whether LAPLATA preferentially resourced high priority tasks, and this was found to be true for all experiments.

Our first set of experiments reveals the sensitivity of our approach to configuration variations and to serve as a baseline for enhancements. Figure 4 shows baseline results where APA and GAO do not iterate over requested task times. As expected, scheduling success rate (%) depends



(a) Scheduling success % by number of tasks versus number of resources, averaged over the number of C2 nodes and comms conditions.



(b) Scheduling success % by number of C2 nodes versus comms conditions averaged over task and asset numbers.

Figure 4: Baseline task allocation success. Number of tasks: A: 150, B: 158, C: 250; Number of sorties (combined strike and SEAD): 1: 194, 2: 146, 3: 128, 4: 113; Comms Conditions: 1 = 100Mb/s, 2 = 1 Mb/s, 3 = 56kb/s.

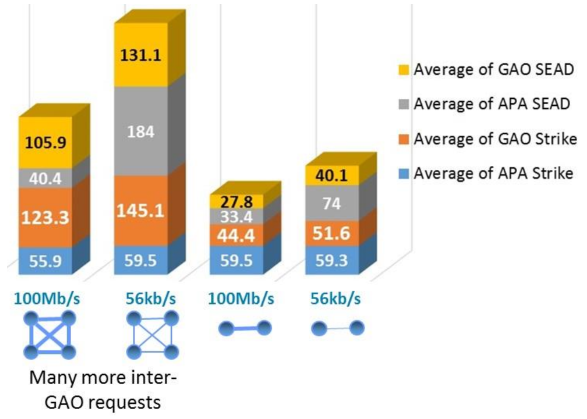


Figure 5: Processing times for planning and scheduling strike and SEAD tasks with two and four C2 nodes and 100Mb/s and 56kb/s comms conditions.

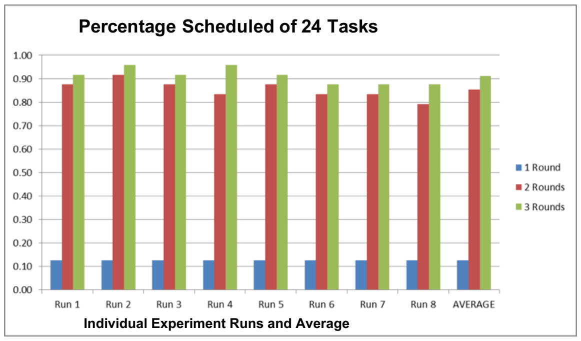


Figure 6: Scheduling success % improves with schedule negotiation between APA and GAO, shown here for a small, 24 task scenario.

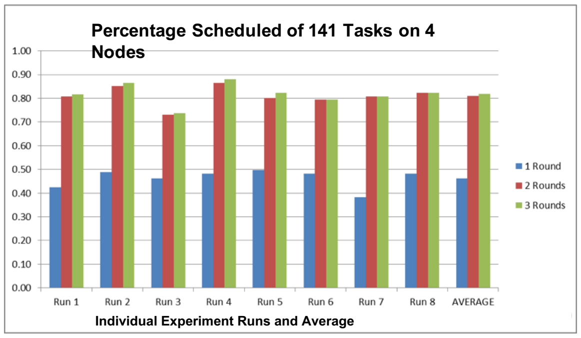


Figure 7: Scheduling success % improvement on a 141 task scenario, equivalent to case A in the baseline experiments.

strongly on the number of tasks and the available assets (Figure 4a). The roughly 50% scheduling success rate is due to time mismatches between requests and availability, caused by the separation of planning and scheduling and the lack of negotiation between APA and GAO to revise the requested time windows. When more tasks are planned with the same number of assets, scheduling success falls further, as expected.

Figure 4b shows that scheduling sensitivity to bandwidth constraints down to 56Kb/s is very low, attesting to the minimal communication needs of our approach. Overall, processing time increases with lower bandwidth and with larger numbers of nodes (Figure 5). We observe near-constant time of the APA Strike task generation step (blue), as the speed up due to additional planning nodes is offset by inter-APA coordination. For SEAD task planning, coordination overhead dominates, since SEAD tasks support multiple strike tasks being planned by all of the nodes. Thus, SEAD task planning time (gray) increases with the number of nodes and significantly increases with lower bandwidth communications. GAO processing times (orange and yellow) are dominated by the additional processing time to service peer GAO requests, especially when the number of nodes increases.

Figure 6 shows results for a set of resources that are exactly sufficient for the task needs, and where APA and GAO iterate over requested task times. In this experiment, we varied the number of interactions between APA and GAO when scheduling task varies. The scheduling success rate improves significantly move from 1 to 2 rounds of interactions, though increasing to 3 rounds of interactions yielded only slight improvements over 2 rounds. As before, scheduling success is insensitive to communications conditions, and, in this case, also the number of nodes. Figure 7 shows similar success rates for a larger, 141 task scenario

Relatively large variations between runs of the same configuration are due to a combination of factors: the tasks are stored as an unordered collection and, thus, are received in random order by LAPLATA; timing of the allocation requests depends on the number of targets and varies; and the greedy optimization in the GAO allocator is sensitive to the order in which request arrive. GAO is designed for rapid, incremental allocation changes, but we avoid, for now, changes that affect schedules that have been committed and communicated to APA and which would require replanning by APA, except when the task list changes via external input.

Conclusions and Future Work

We have developed an effective and efficient architecture and set of planning, allocation, and scheduling algorithms for a problem whose realistic complexity exceeds what approaches reported in the literature can handle. We have shown decentralized planning and scheduling to be resilient to extreme bandwidth constraints. Profile-assisted, cooperative allocation and scheduling improves on hierarchical auction approaches under communications denial, since it uses fewer messages and does not require a full auction among auctioneers. It is also more robust to communications disruptions than distributed winner determination auctions (consensus protocols), since it does not require multiple rounds of negotiations.

Future work will include scaling the algorithms to plan and schedule 1000 tasks per day, e.g., increasing the efficiency of planning and plan repair search. Another enhancement will be to enable assets to team up in the bidding stage and, instead of bidding separately, jointly bid for interdependent supported and supporting tasks. Joint bids promise to increase efficiency and reduce iterations between APA and GAO, since bids from multiple assets are pre-coordinated. This avoids the case where a supported task is allocated but has to be canceled because its required supporting task cannot be scheduled.

Another enhancement will be to make GAOs' peer profiles dynamic. The current method only considers nominal asset availability under the implicit assumption that some of these assets will remain available. The dynamic method will use actual, when known, or assumed demand on assets held by peer GAOs into account.

Finally, it will be important to consider how to explain LAPLATA's planning and scheduling rationale to a human operator and to provide means by which the operator can adjust portions of the plan.

References

Erol, K.; Hendler, J.; and Nau, D. S. 1994. HTN planning: Complexity and expressivity. In *Proceedings AAAI*.

Geib, C. W., and Goldman, R. P. 2009. A probabilistic plan recognition algorithm based on plan tree grammars. *AIJ* 173(11):1101–1132.

Goldman, R. P.; Kuter, U.; and Schneider, A. 2012. Using classical planners for plan verification and counterexample generation. In *Proceedings of AAAI Workshop on Problem Solving Using Classical Planning*.

Goldman, R. P. 2006. Durative planning in HTNs. In *ICAPS-06*, 382–385.

Hostage III, G. M., and Broadwell Jr., L. R. 2014. Resilient command and control, the need for distributed control. *Joint Force Quarterly* 74.

Kuter, U.; Nau, D.; Gossink, D.; and Lemmer, J. F. 2004. Interactive course-of-action planning using causal models. In *Proceedings of the Third International Conference on Knowledge Systems for Coalition Operations (KSCO-2004)*, 37–52.

Kuter, U.; Burstein, M.; Benton, J.; Bryce, D.; Thayer, J.; and McCoy, S. 2015. HACKAR: helpful advice for code knowledge and attack resilience. In *AAAI/IAAI-15*.

Kuter, U.; Goldman, R. P.; and Hamell, J. 2018. Assumption-based decentralized htn planning. *Hierarchical Planning 2018* 9.

Lemmer, J. F., and Gossink, D. 2004. Recursive noisy-or: A rule for estimating complex probabilistic causal interactions. *IEEE Transactions on Systems, Man, and Cybernetics, Part B* 34(6):2252–2261.

Musliner, D.; Goldman, R.; Hamell, J.; and Miller, C. 2011. Priority-Based Playbook Tasking for Unmanned System Teams. In *Infotech@ Aerospace 2011*. AIAA. 1566.

Nau, D.; Au, T.-C.; Ilghami, O.; Kuter, U.; Murdock, W.; Wu, D.; and Yaman, F. 2003. SHOP2: An HTN planning system. *JAIR* 20:379–404.

Nau, D.; Au, T.-C.; Ilghami, O.; Kuter, U.; Muñoz-Avila, H.; Murdock, J. W.; Wu, D.; and Yaman, F. 2005. Applications of SHOP and SHOP2. *IEEE Intelligent Systems* 20(2):34—41.

Russell, S., and Norvig, P. 2003. *Artificial Intelligence, A Modern Approach (Second Edition)*. Upper Saddle River, NJ: Prentice-Hall.

Sandholm, T. 2007. Expressive commerce and its application to sourcing: How we conducted $35 billion of generalized combinatorial auctions. *AI Magazine* 28(3):45–58.

Seuken, S., and Zilberstein, S. 2008. Formal models and algorithms for decentralized decision making under uncertainty. *AAMAS* 17(2):190–250.

Torreño, A.; Onaindia, E.; Komenda, A.; and Štolba, M. 2017. Cooperative multi-agent planning: A survey. *ACM Computing Surveys (CSUR)* 50(6):84.