(Preprint) AAS 24-192

INTENT SHARING FOR EMERGENT COLLABORATION IN AUTONOMOUS EARTH OBSERVING CONSTELLATIONS

Mark Stephenson*, Lorenzzo Mantovani* and Hanspeter Schaub†

The objective of the multiagent agile Earth-observing satellite scheduling problem is to maximize the global sum of values of requests imaged by a constellation while satisfying safety constraints for each satellite. Prior work has shown that individually-trained satellites with reinforcement learning-based policies operating on the same fixed-interval cadence exhibit emergent cooperative behavior when a communication step is added to share what targets have been successfully imaged; however, in cases with high satellite or low target densities, this is not sufficient to avoid duplicating efforts. In this work, novel methods for generalizing to constellations where satellites are operating on asynchronous variable intervals — a more realistic case — are explored. Also, an intent-sharing method for coordinating actions to avoid duplication of tasks in dense constellations is considered. This method maintains the advantages of the single-agent training, multi-agent deployment pipeline, including scalability and responsiveness, while improving global performance.

INTRODUCTION

Scheduling constellations of agile Earth-observing satellites (AEOSs) is a challenging problem, especially with higher densities of targets and satellites [1]. Because of the agile nature of the satellites (i.e. the ability to slew along-track), each observation request has a time window for which can be imaged, greatly increasing the set of feasible solutions when compared to traditional Earth-observing satellites (EOSs) that can only slew across-track. In this work, ideas from two previous papers are combined to create a scaleable autonomous solution to the problem: a variable-interval formulation of the agile Earth-observing satellite scheduling problem (AEOSSP) [2] and the use of single-agent policies in a multi-agent environment [3]. This paper explores how best to induce cooperation between a cluster of satellites operating on different intervals to complete a many-target imaging task, as shown in Figure 1.

Much of the existing literature treats the AEOSSP as a discrete task-based planning problem, representing possible collection events as vertices and feasible slews as edges [4]. The problem (not considering resource management, which must be handled by additional constraints) is then reduced to maximization over a directed acyclic graph, part of a well-studied class of problems [5, 6]. Mixed-integer programs (MIPs) or iterative local search (ILS) can then be used to find optimal solution. For single-satellite planning, these approaches are considered in [7] and [8], among many others.

*Ph.D. Student, Ann and H.J. Smead Department of Aerospace Engineering Sciences, University of Colorado, Boulder, Boulder, CO, 80303. AIAA Member.

†Professor and Department Chair, Schaden Leadership Chair, Ann and H.J. Smead Department of Aerospace Engineering Sciences, University of Colorado, Boulder, 431 UCB, Colorado Center for Astrodynamics Research, Boulder, CO, 80309. AAS Fellow, AIAA Fellow.

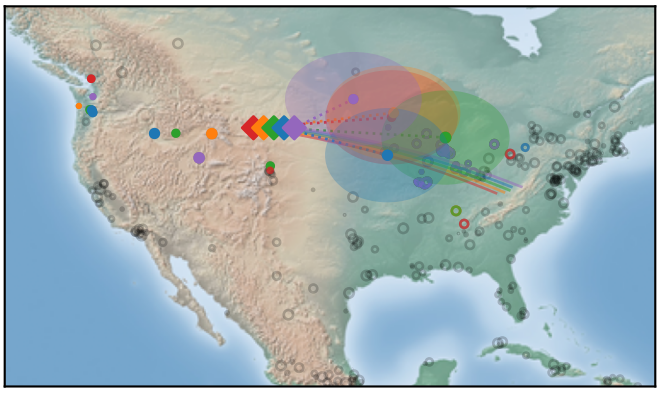

Figure 1: A formation of satellites each selecting from among upcoming point targets.

These preplanning approaches can be extended to account for multi-satellite constellation planning and request deconflicting among agents. Bianchessi considers the problem for hundreds to low thousands of requests across two satellites and multiple users, comparing a heuristic method to an optimal solution [9]. Cho develops a two-step approach to constellation scheduling, first planning downlink opportunities then request fulfillment for up to 12 satellites and 700 requests [10]. Kim introduces additional realistic constraints for imaging and a heuristic to speed planning times for a constellation [11]. Wang formulates a MIP with added cloud uncertainty for up to 300 targets [12]. Nag utilizes dynamic programming to greatly improve planning times for a two-satellite constellation [13]. Eddy introduces a set-theory-based approach to target allocation that scales better than other approaches for large constellations and request sets [14].

A common theme among these approaches is a time-expensive planning stage, often requiring tens of minutes to hours to plan for constellations with at most tens of agents and hundreds to thousands of requests. These computationally expensive and time-consuming approaches limit the possibility of replanning when the request set has changed or of onboard planning. A recently proposed alternative for satellite scheduling is the use of onboard autonomous tasking policies found with reinforcement learning. Markov decision processes (MDPs) for the single-agent, task-based AEOSSP are formulated in [15, 16, 17, 18]. In particular, [15] and [16] use high-fidelity simulation environments for training and testing. Zhao applies reinforcement learning differently, training an agent to perform the global scheduling problem for a single satellite [19]. Reference [2] builds on previous work in on-orbit autonomy by implementing variable-duration decision intervals into the MDP formulation, in order to be able to produce the same quality of solutions as MIP-based approaches. To generalize to a scalable multi-satellite constellation, Herrmann adds communication between single-agent-trained satellites to deduplicate efforts [3]. This work builds on Herrmann's paper by introducing considerations for communications when satellites are not operating on the same decision interval.

PROBLEM STATEMENT

A multi-satellite, power-constrained variation of the AEOSSP is modeled with a high-fidelity spacecraft simulation and formalized as a decentralized partially-observable Markov decision process (Dec-POMDP).

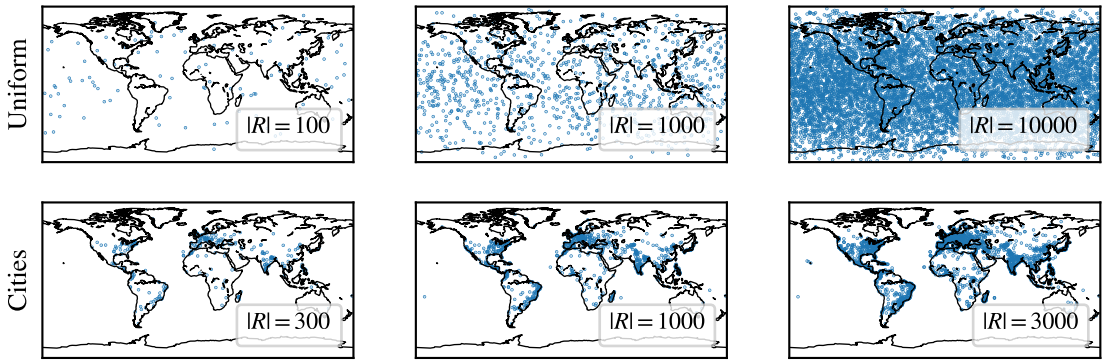

Figure 2: Examples of each target distribution over various request densities.

Agile Earth-Observing Constellation Scheduling Problem

The AEOSSP is considered for a homogeneous constellation of imaging satellites with a joint objective to maximize the cumulative value of unique imaged targets. The environment is the same as the power-limited environment described in [2], other than the addition of multiple agents; thus, a compressed development of the environment is presented here.

Image Request Model Image requests are specified as a tuple of target position and priority, $\rho_i = (\boldsymbol{r}_i, r_i) \in R$, using a point-based request model. In this work, two distributions of requests are considered (Figure 2): A uniform distribution over the surface of the Earth, $|R| \in [100, 10000]$, and a city-based distribution, $|R| \in [300, 3000]$.* For both distributions, target priorities are uniformly distributed $r_i \in [0, 1]$.

All requests start in the unfulfilled set $R = U$. When a request is imaged, it is removed from R and added to the fulfilled set F. Imaging a target $\rho_i \in U$ yields a reward equal to the priority r_i; imaging $\rho_i \in F$ yields no reward, though should an operator want to image a location multiple times, multiple requests can be made in the same location.

Target requests are subject to various dynamic constraints to be imaged. The satellite's instrument must be pointed at the target within $\delta\theta$ and track its relative motion within $\delta\dot{\theta}$ to take an image. Since the satellites are agile (i.e. capable of slewing along-track), each target is accessible for an interval of time; additional constraints define the opportunity window $[\tau^o, \tau^c] = w$ for the target; in this work, a minimum elevation angle constraint is included.

Power Constraints The satellite is modeled with a power system consisting of solar panels (normal antiparellel to the instrument), a battery, and power draws from reaction wheels, the bus, and the instrument. The satellite's power system must be maintained positive at all times, including through eclipse. While the satellite passively charges during operations, it can enter a charging mode that points the solar panels at the sun to maximize power gain.

Markov Decision Process Formulation

The problem is formalized as a Dec-POMDP, a generalization of an MDP for decentralized decision-making between multiple agents with a single goal [20]. A deterministic simulation of

*City location data from simplemaps.com.

Quantity	Normalization	Description
${}^{\mathcal{E}}\boldsymbol{\omega}_{\mathcal{BE}}$	0.03 rad/s	Body angular rate
${}^{\mathcal{E}}\hat{\boldsymbol{c}}$	-	Instrument pointing direction
${}^{\mathcal{E}}\boldsymbol{r}_{\mathcal{BE}}$	r_E	Earth-fixed position, Earth radius-normalized
${}^{\mathcal{E}}\boldsymbol{v}_{\mathcal{BE}}$	v_{orb}	Earth-fixed velocity, orbital velocity-normalized
t	t_{max}	Time through episode (completion fraction)
$\{r_m \mid m \in 1, \ldots, M\}$	-	Rewards $\in [0, 1]$ of next M targets
$\{{}^{\mathcal{E}}\boldsymbol{r}_m \mid m \in 1, \ldots, M\}$	r_E	Positions of next M targets
Equation 1	5.0	Upcoming reward density
z	z_{max}	Charge fraction
$\tau^o_{Ecl}, \tau^c_{Ecl}$	T	Next eclipse transitions, orbital period-normalized

Table 1: Elements in the observation o and their normalization constants, from [2].

the scenario is given as $G(s, a) = s'$. The elements of the Dec-POMDP are as follows:

- **State Space $\mathcal{S}$:** The space of simulator states required to maintain the Markov assumption. This consists of the satellite state spaces and the environment state, $\mathcal{S} = \mathcal{S}_1 \times ... \times \mathcal{S}_s \times \mathcal{S}_{\text{env}}$. These states include "intuitive" aspects of the state, like satellite positions and attitudes, as well as "hidden" aspects, like internal flight software states, in order to make the formulation Markov.

- **Joint Observation Space $\mathcal{O}$ and Observation Probability Function Z:** The product of the individual observation spaces of each satellite; since the satellites are homogeneous, $\mathcal{O} = \mathcal{O}_1 \times ... \times \mathcal{O}_s = \mathcal{O}^S_{\text{sat}}$. Each satellite's observation is a noiseless subset of the state space $O(s) \subset s$, so $Z(O(s)|s) = 1$. The elements included in the observation for each satellite are given in Table 1; this includes satellite states and information about the next $M = 32$ unfulfilled requests. To encode long-term reward availability, the reward density function

$$\left\{ \sum_{i \in U} \begin{cases} r_i & \text{if } t + (k-1)\Delta t \leq \tau^o_i < t + k\Delta t \\ 0 & \text{else} \end{cases} \mid k = 1, \ldots, K \right\} \tag{1}$$

  is included in the state. This allows the satellite to wait to charge until little or no reward is available, as opposed to charging during periods of high reward density due to a myopic view of the environment.

- **Joint Action Space $\mathcal{A}$:** The product of the individual action spaces of each satellite; since the satellites are homogeneous, $\mathcal{A} = \mathcal{A}_1 \times ... \times \mathcal{A}_s = \mathcal{A}^S_{\text{sat}}$. Each satellite has $N + 1$ actions:

  - $a_{\text{im},i} \times N$: Actions to attempt to image each of the $N = 32$ next unfulfilled targets, by time of next encounter. The satellite slews to point the instrument at the target and — if attitude, rate, and access constraints are met — takes an image. As such, this action does not guarantee successful imaging.
  - a_{charge}: Action to enter sun pointing mode for one minute. The satellite slews to point the solar panels at the sun to maximize power gain; if the satellite is in eclipse, the underlying simulator will not charge the battery.

- **Transition Probability Function $T(s'|s,a)$:** Transitions are generated by the deterministic simulator, leading to $T(G(s,a)|s,a) = 1$. The simulator propagates for a variable amount of time, depending on the actions taken and how the environment evolves during propagation. Propagation is halted at one of three conditions:

    1. **Imaging Successful:** If any satellite tasked with an imaging action successfully images the target, the simulation halts since that satellite's action can result in more reward.
    2. **Opportunity Window Close:** If the opportunity window closes for any satellite tasked with an imaging action, the simulation halts since the action will not result in any reward.
    3. **Maximum Step Duration Timeout:** If none of the above conditions are met for any satellite after $\Delta t = 5$ minutes of simulation time, the simulation halts.

- **Reward Function $R(s,s')$:** The agents are jointly rewarded for imaging unfulfilled target requests during a step:

$$R(s,s') = \sum_{\rho_i \in R} \begin{cases} r_i & \text{if } \rho_i \in U \text{ and } \rho_i \in F' \\ 0 & \text{else} \end{cases} \tag{2}$$

Simulation Environment

The environment is configured using `bsk-rl`[†], an open-source package for building spacecraft tasking environments for reinforcement learning (RL). The environment includes component-level dynamics models, flight-proven flight software algorithms, and models of the space environment. The package uses Basilisk[‡] for the underlying high-fidelity spacecraft simulation [21]. Agents interact with the environment via the Gymnasium API, allowing for compatibility with all major RL frameworks [22].

METHODS

Shielded RL is applied to the single-satellite case to find a per-agent policy; a complete development of the policy can be found in [2]. Central to this work is determining the best methods for sharing information between agents and retasking the agents to induce collaborative behavior; the considered methods are described in detail.

Reinforcement Learning

RL is a method of finding a mapping of states to actions ($\pi(s) = a$) to maximize a long-term reward signal in an environment with unknown and sometimes probabilistic dynamics [23]. The learning agent finds this mapping by interacting with the environment, receiving a reward signal, and updating its policy based on the reward signal and the observed state-action pairs. For continuous or sufficiently large state and action spaces, deep reinforcement learning (DRL) is often applied, which uses deep neural networks to represent the agent's policy.

To guarantee the safety of an agent, shielded RL is used in deployment [24]. When the agent selects an action, the shield checks if the action is safe with respect to the current state. If it is safe, the action is taken; if it is not, a deterministic safe response is executed instead. In this environment,

[†] https://github.com/AVSLab/bsk_rl/

[‡] http://hanspeterschaub.info/basilisk/

the only safety constraint is keeping each satellite's power system alive. The shield checks if the power level is sufficient to make it through the next eclipse; if it is not, the agent is forced to charge.

Single-Satellite Agent Generation

The policy network and shield used for agents in this paper are generated in [2]. In that paper, agents are trained in a single-satellite environment using the RLlib implementation of asynchronous proximal policy optimization (APPO) and then deployed with a shield to prevent unsafe actions [25, 26]. Experiments are performed on the state and actions space target lookaheads, M and N, and the failure penalty. The policies produced in that work are found to perform well when compared to a pseudo-optimal MIP-based solution method. On uniformly distributed targets, the agent achieves 90-95% of the optimal reward; for cities, a more challenging environment, the agent achieves 60-80% of optimality. The primary cause of suboptimality is found to be the agent selecting targets for which it has insufficient time to slew to before the opportunity window closes. This effect is seen in certain cases in this paper's multi-satellite results.

Communication Between Agents

The system contains multiple satellites sharing information among them. Therefore, a communication model is utilized where communication is assumed to be available at all times with sufficient bandwidth to transmit necessary information, similar to the free communication adopted by [3]. Communication is independent of the satellites' attitudes. These assumptions are reasonable for small satellites flying close together, with the possibility of using omnidirectional antennae. The inter-satellite communication links are used to share information about requests that have been fulfilled or are currently being pursued.

Completion Sharing vs. Intent Sharing Two communication cases are considered, each with different information shared about request status:

- **Completion Sharing** Each satellite broadcasts when it has successfully fulfilled a request, causing the other satellites to remove that request from their list of unfulfilled requests.
- **Intent Sharing** Each satellite broadcasts the request associated with its currently selected action, causing the other satellites to temporarily remove the request from the set of unfulfilled targets. If the satellite fails to complete the imaging task after communicating intent, the target is readded to each satellite's set of unfulfilled targets; if successful, the request is permanently moved to the set of fulfilled requests.

Since only unfulfilled requests appear in the observation and action spaces, satellites will not select targets that have been communicated as fulfilled or intended-to-be-fulfilled by other satellites.

Continuous Retasking vs. Conflict Retasking Two approaches are used to determine when satellites should retask (i.e. reevaluate the policy network), as opposed to continuing their current task.

- **Continuous Retasking** All satellites retask with the shielded policy network at every MDP; that is, when any satellite completes a task. Since the completion of tasks changes the environment, other satellites may have a new optimal action with respect to the current environment
- **Conflict Retasking** Satellites only reevaluate the shielded policy network when the task associated with their previous action is completed by themselves or another satellite; otherwise,

the previous action is maintained through the next step. If the previously selected request is shared as fulfilled or intended by another satellite, the satellite will retask.

Tasks are evaluated in two passes: first, actions that are being maintained from a previous step are assigned in order of satellite index. Then, tasks selected by the shielded policy network are assigned in order of satellite index. This gives a slight advantage to the first satellite in the list, since in cases where multiple satellites retask on the same step, earlier satellites get first pick of the requests.

RESULTS

The performance of the intent-sharing and completion-sharing methods for communication combined with continuous retasking and conflict retasking are compared. Additionally, a method with no communication and no retasking is added as a benchmark, where satellites act independently of each other. Therefore, the results showcase the following cases:

- **No Communication:** No sharing of fulfilled requests and retasking only when the current task is completed.
- **Intent-Conflict:** Intent sharing with conflict retasking.
- **Intent-Continuous:** Intent sharing with continuous retasking.
- **Completion-Conflict:** Completion sharing with conflict retasking.
- **Completion-Continuous:** Completion sharing with continuous retasking.

These methods are tested in along-track (spaced true anomaly) and across-track (spaced longitude of ascending node) formations with five satellites each on a 45°inclination, 500 km orbit over the city and uniform request distributions.

The performance of the methods are measured by the total reward obtained by the system, the wasted time, and the counts of unique and duplicated requests. The total reward is the sum of the rewards obtained by each satellite. The wasted time is defined as the time spent by the satellites in a given action without completing it due to retasking; it is normalized by the total simulation time — three orbits — times the number of satellites. The number of unique requests fulfilled is the number of targets that were imaged at least once by the constellation, while the number of duplicated requests is the number of times beyond the initial fulfillment that a target was imaged.

Figure 3 shows results for a formation of five satellites in an along-track configuration with 0.1° of true anomaly spacing. The results show that the Intent-Conflict method performs better than all other methods in terms of total reward, wasted time, and unique and duplicated request fulfillments. Cases with continuous retasking show higher wasted time since satellites retask every time another satellite completes an action. During this retasking process, the satellite picks a new action that can differ from the previous one, which creates a chattering behavior of constantly changing actions that wastes time since actions are infrequently completed. Conflict retasking has less chattering behavior since the satellite only retasks after finishing an action or when its current action conflicts with other satellites' actions. Completion-Conflict cases share information about imaged targets after actions are completed, which leads to satellites wasting time when two pursue the same request. Since the Intent-Conflict method involves sharing intent before taking actions, the satellites never waste time due to conflicts leading to strong overall performance. Considering a city-based request

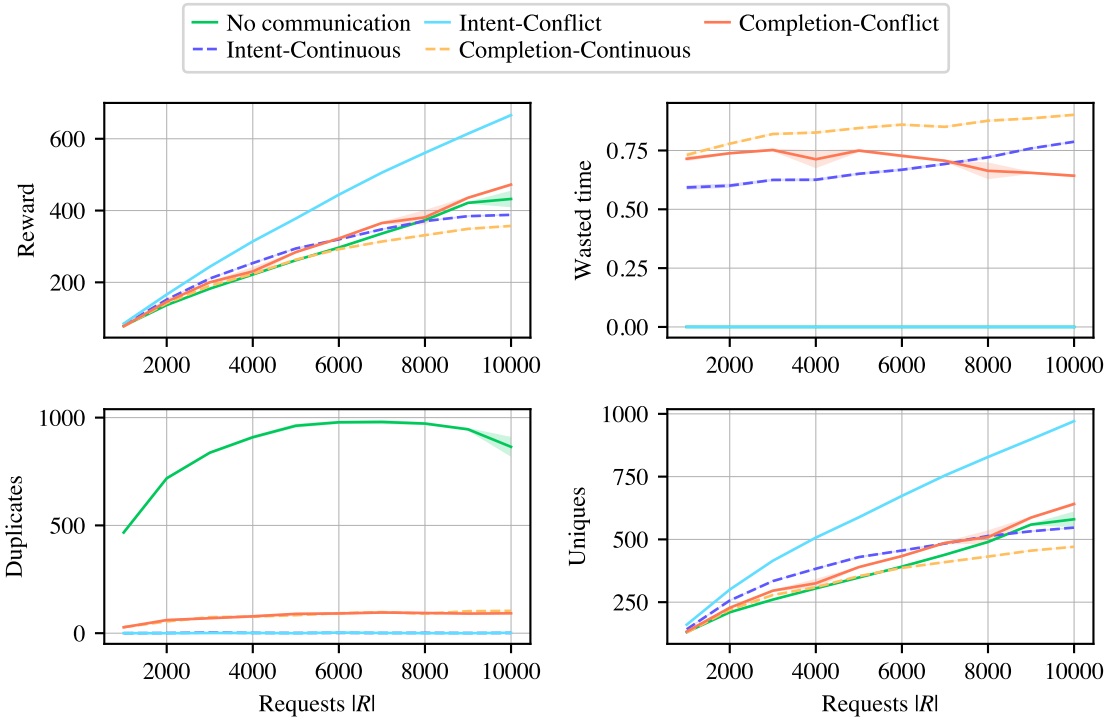

Figure 3: Performance comparison of the five different methods in the along-track formation with spacing of 0.1 degrees with uniform request distribution.

distribution instead of a uniform distribution leads to similar results, with the Intent-Conflict case again delivering the best performance as shown in Figure 4.

The chattering behavior can be partially explained by the single-agent policy's lack of experience in the domain of mid-action dynamics (i.e. it never experienced steps stopping mid-slew during training). As a result, the policy does not know that it can feasibly continue pursuing the same request and thus retasks to a later request, even though continuing to pursue the request is optimal in most cases as the environment has not significantly changed.

The No Communication method yields a higher number of duplicated requests than any other; however, the method's reward in the uniform request distribution cases (Figure 3) is as least as good as the worst communication method because of the sheer number of requests fulfilled. The dynamics of gathering many targets without coordination is seen as the number of duplicates for the No Communication case reaches a maximum close to 6,000 requests and then decreases; this is because the increased number of requests reduces the chance of two satellites selecting the same target. This implies that in certain cases, some types of communication can hurt overall performance by preventing satellites from engaging in potentially useful tasks in order to avoid duplication. In the city-based request distribution, the greedy strategy displayed by No Communication performs worse since the requests are sparser; when only clusters of requests are present, communication is a net positive to maximize diversity of tasks during those short periods of time.

In the along-track formation case, the satellite in front of the formation accesses requests first; as a result, it has a higher chance of imaging it first, collecting the reward, and tasking another request.

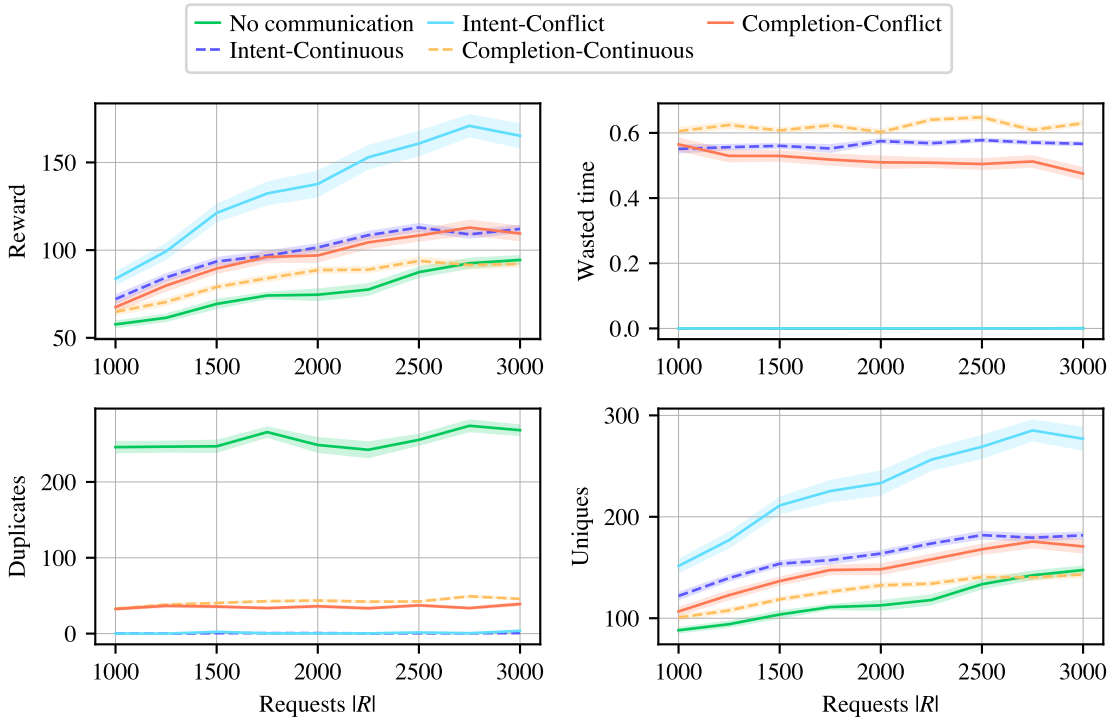

Figure 4: Performance comparison of the five different methods in the along-track formation with 0.1° spacing with city-based request distribution.

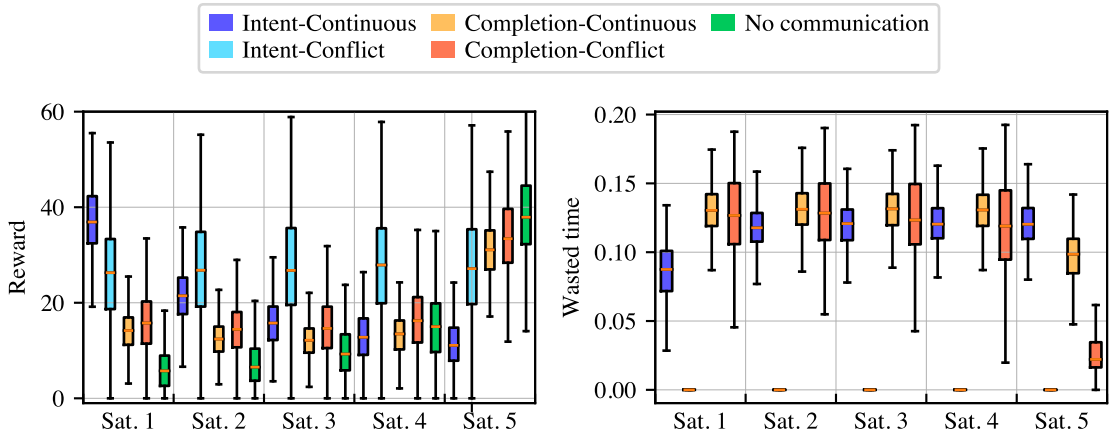

Figure 5: Reward and wasted time per satellite in the along-track formation with 0.1° spacing with city request distribution.

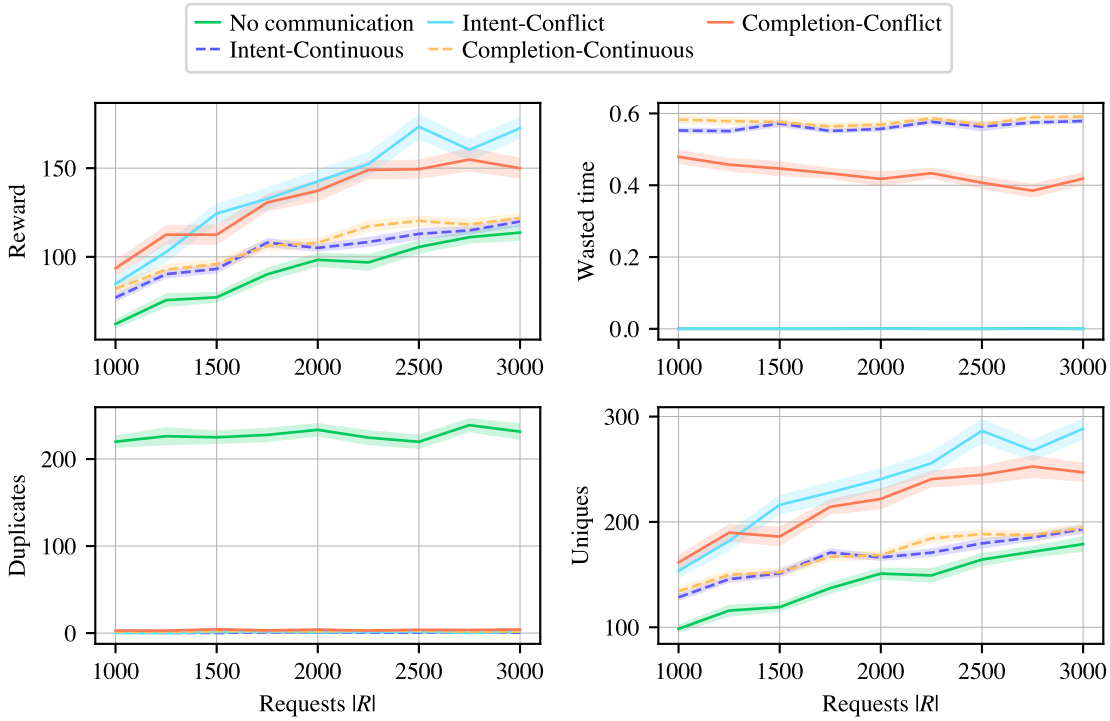

Figure 6: Performance comparison of the five different methods in the along-track formation with 2.0° spacing with city-based request distribution.

If continuous retasking is used, the other satellites retask only once the first satellite successfully images the target, resulting in more wasted time. This is corroborated by the results presented in Figure 5, which shows the reward per satellite and wasted time per satellite for both Intent-Conflict and Intent-Continuous cases. The results show that the satellite in front of the formation — "Sat. 1" — has the largest reward and the other satellites have larger wasted times in the Intent-Continuous case, while both the reward and wasted time are more evenly distributed in the Intent-Conflict case. The opposite is seen in the Completion-Conflict and Completion-Continuous cases, where the satellites in the back of the formation have the largest reward and lowest wasted time. Completion cases share the information after imaging the target, so satellites in the back have access to requests other satellites do not have anymore, preventing them from being retasked due to conflict.

Constellation Spacing Experiments

In addition to changing the number of requests, the spacing of both formations was varied to address the impact of the distance between satellites on the performance of the methods. Initially, an along-track formation similar to Figure 4 with a wider spacing is addressed. After, an across-track formation is considered where the spacing between the satellites' longitude of the ascending node was varied for a constant number of requests.

By increasing the separation between satellites, the performance of completion methods gets closer to the intent methods. Figure 6 shows the case of a formation with 5 satellites in an along-track configuration and 2.0°of spacing between the satellites. In this case, the reward as a function of the number of requests in the Completion-Conflict case overlaps with the Intent-Conflict

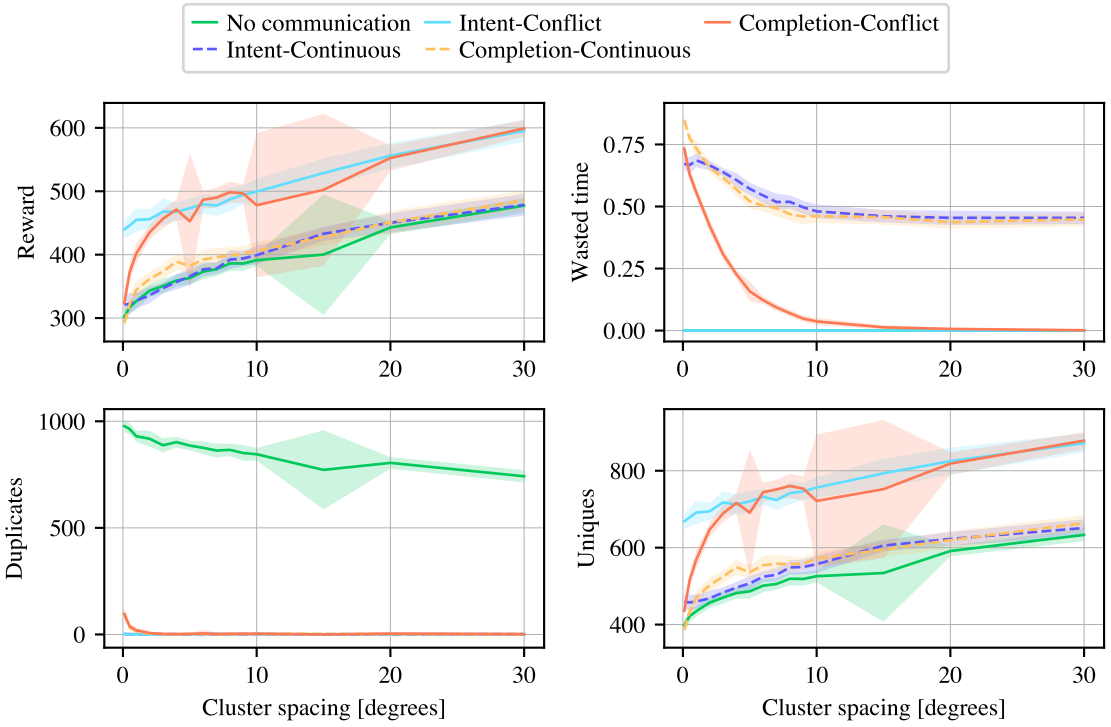

Figure 7: Along-track formation with spacing varying from 0.1° to 30.0° degrees with $|R| = 6000$ uniform request distribution.

case in some intervals. This behavior is associated with the number of duplicates obtained by the Completion-Conflict and Completion-Continuous cases, which dropped from about 40 for a 0.1°cluster separation — seen in Figure 4 — to less than 10 for a 2.0°cluster separation. This is explained by the probability of satellites selecting the same request decreasing with the increase in the angular separation between satellites, increasing the number of unique images. Therefore, the wasted time is reduced for the Completion-Conflict case, but not significantly for the Completion-Continuous and Intent-Continuous cases.

Increasing the number of requests reduces the probability of satellites selecting the same request, therefore reducing the number of duplicates and increasing overall rewards. Similarly, increasing the distance between satellites, defined by the cluster spacing in degrees, also reduces the probability of satellites selecting the same requests. Figure 7 shows the results of varying the cluster spacing from 0.1° to 30.0° for an along-track formation and 6000 request uniform target distribution. These results showcase that the number of duplicates reduces as the satellites are farther apart; the same is seen for the wasted time.

Overall, the results show that the Intent-Conflict outperforms other methods, particularly in challenging cases with smaller inter-satellite distances and higher target density. Intent-sharing communication reduces the number of duplicates, while conflict-based retasking reduces wasted time. Combining these two approaches leads to the best performance, dropping the wasted time and number of duplicates to zero and avoiding chattering behavior during retasking.

CONCLUSION

This paper presents the use of RL to solve a Dec-POMDP for the scheduling of multiple AEOSs. The novelty of this work is showing the performance of asynchronous variable decision-making intervals and the advantage of intent-sharing communication methods to create emergent collaboration in a multi-agent system, avoiding a duplication of efforts. Hence, two different retasking methods were compared — continuous retasking and conflict-based retasking — with two different information-sharing approaches — intent sharing and completion sharing. The results show that the Intent-Conflict method tends to show the best performance, while cases with continuous retasking show higher wasted time and lower rewards compared to conflict retasking. This trend is seen in both city-based targets and uniform targets. The relative performance of the methods depends on the distance between satellites; larger spacing results in fewer conflicts between satellites.

The single-satellite training of the agent is shown to not be suitable for retasking when interrupted mid-task, as this is outside the training domain. Therefore, future work is required to train an algorithm with continuous retasking capabilities to improve overall rewards. Also, an auction method could be added to improve target selection in the case of intent sharing with continuous retasking. Ultimately, an agent-count agnostic multi-agent reinforcement algorithm may be necessary to maximize coordination between the agents.

ACKNOWLEDGEMENT

This work is partially supported by a NASA Space Technology Graduate Research Opportunity (NSTGRO) grant, 80NSSC23K1182. This work is also partially supported by the Air Force Research Lab grant FA9453-22-2-0050.

This work utilized the Alpine high performance computing resource at the University of Colorado Boulder. Alpine is jointly funded by the University of Colorado Boulder, the University of Colorado Anschutz, Colorado State University, and the National Science Foundation (award 2201538).

REFERENCES

[1] X. Wang, G. Wu, L. Xing, and W. Pedrycz, "Agile Earth Observation Satellite Scheduling Over 20 Years: Formulations, Methods, and Future Directions," *IEEE Systems Journal*, Vol. 15, Sept. 2021, pp. 3881–3892, 10.1109/JSYST.2020.2997050.

[2] M. Stephenson and H. Schaub, "Reinforcement Learning For Earth-Observing Satellite Autonomy With Event-Based Task Intervals," *AAS Rocky Mountain GN&C Conference*, Breckenridge, CO, Feb. 2024.

[3] A. Herrmann, M. A. Stephenson, and H. Schaub, "Single-Agent Reinforcement Learning for Scalable Earth-Observing Satellite Constellation Operations," *Journal of Spacecraft and Rockets*, Nov. 2023, pp. 1–19, 10.2514/1.A35736.

[4] V. Gabrel, A. Moulet, C. Murat, and V. T. Paschos, "A New Single Model and Derived Algorithms for the Satellite Shot Planning Problem Using Graph Theory Concepts," *Annals of Operations Research*, Vol. 69, 1997, pp. 115–134.

[5] D. L. Applegate, R. E. Bixby, V. Chvatál, and W. J. Cook, *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, 2006.

[6] S. Augenstein, "Optimal Scheduling of Earth-Imaging Satellites with Human Collaboration via Directed Acyclic Graphs," *The Intersection of Robust Intelligence and Trust in Autonomous Systems: Papers from the AAAI Spring Symposium*, 2014, pp. 11–16.

[7] G. Peng, R. Dewil, C. Verbeeck, A. Gunawan, L. Xing, and P. Vansteenwegen, "Agile Earth Observation Satellite Scheduling: An Orienteering Problem with Time-Dependent Profits and Travel Times," *Computers & Operations Research*, Vol. 111, Nov. 2019, pp. 84–98, 10.1016/j.cor.2019.05.030.

[8] M. Stephenson and H. Schaub, "Optimal Target Sequencing In The Agile Earth-Observing Satellite Scheduling Problem Using Learned Dynamics," *AAS/AIAA Astrodynamics Specialist Conference*, Big Sky, MT, Aug. 2023.

[9] N. Bianchessi, J.-F. Cordeau, J. Desrosiers, G. Laporte, and V. Raymond, "A Heuristic for the Multi-Satellite, Multi-Orbit and Multi-User Management of Earth Observation Satellites," *European Journal of Operational Research*, Vol. 177, Mar. 2007, pp. 750–762, 10.1016/j.ejor.2005.12.026.

[10] D.-H. Cho, J.-H. Kim, H.-L. Choi, and J. Ahn, "Optimization-Based Scheduling Method for Agile Earth-Observing Satellite Constellation," *Journal of Aerospace Information Systems*, Vol. 15, Nov. 2018, pp. 611–626, 10.2514/1.I010620.

[11] J. Kim, J. Ahn, H.-L. Choi, and D.-H. Cho, "Task Scheduling of Agile Satellites with Transition Time and Stereoscopic Imaging Constraints," *Journal of Aerospace Information Systems*, Vol. 17, June 2020, pp. 285–293, 10.2514/1.I010775.

[12] X. Wang, Y. Gu, G. Wu, and J. R. Woodward, "Robust Scheduling for Multiple Agile Earth Observation Satellites under Cloud Coverage Uncertainty," *Computers & Industrial Engineering*, Vol. 156, June 2021, p. 107292, 10.1016/j.cie.2021.107292.

[13] S. Nag, A. S. Li, and J. H. Merrick, "Scheduling Algorithms for Rapid Imaging Using Agile Cubesat Constellations," *Advances in Space Research*, Vol. 61, Feb. 2018, pp. 891–913, 10.1016/j.asr.2017.11.010.

[14] D. Eddy and M. J. Kochenderfer, "A Maximum Independent Set Method for Scheduling Earth-Observing Satellite Constellations," *Journal of Spacecraft and Rockets*, Vol. 58, Sept. 2021, pp. 1416–1429, 10.2514/1.A34931.

[15] M. Nazari, A. Oroojlooy, L. Snyder, and M. Takac, "Reinforcement Learning for Solving the Vehicle Routing Problem," *NeurIPS*, 2018.

[16] A. Harris, T. Teil, and H. Schaub, "Spacecraft Decision-Making Autonomy Using Deep Reinforcement Learning," *AAS Spaceflight Mechanics Meeting*, Maui, Hawaii, Jan. 2019.

[17] A. Hadj-Salah, R. Verdier, C. Caron, M. Picard, and M. Capelle, "Schedule Earth Observation Satellites with Deep Reinforcement Learning," Nov. 2019.

[18] D. Eddy and M. Kochenderfer, "Markov Decision Processes For Multi-Objective Satellite Task Planning," *2020 IEEE Aerospace Conference*, Big Sky, MT, USA, IEEE, Mar. 2020, pp. 1–12, 10.1109/AERO47225.2020.9172258.

[19] X. Zhao, Z. Wang, and G. Zheng, "Two-Phase Neural Combinatorial Optimization with Reinforcement Learning for Agile Satellite Scheduling," *Journal of Aerospace Information Systems*, Vol. 17, July 2020, pp. 346–357, 10.2514/1.I010754.

[20] M. J. Kochenderfer, *Algorithms for Decision Making*. Massachusetts Institute of Technology, 2022.

[21] P. W. Kenneally, S. Piggott, and H. Schaub, "Basilisk: A Flexible, Scalable and Modular Astrodynamics Simulation Framework," *Journal of Aerospace Information Systems*, Vol. 17, Sept. 2020, pp. 496–507, 10.2514/1.I010762.

[22] M. Towers, J. K. Terry, A. Kwiatkowski, J. U. Balis, G. de Cola, T. Deleu, M. Goulão, A. Kallinteris, A. KG, M. Krimmel, R. Perez-Vicente, A. Pierré, S. Schulhoff, J. J. Tai, A. J. S. Tan, and O. G. Younis, "Gymnasium," Oct. 2023.

[23] R. S. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. Adaptive Computation and Machine Learning, Cambridge, Massachusetts London, England: The MIT Press, second edition ed., 2018.

[24] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, "Safe Reinforcement Learning via Shielding," Sept. 2017.

[25] E. Liang, R. Liaw, P. Moritz, R. Nishihara, R. Fox, K. Goldberg, J. E. Gonzalez, M. I. Jordan, and I. Stoica, "RLlib: Abstractions for Distributed Reinforcement Learning," June 2018.

[26] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," Aug. 2017.