(Preprint) AAS 25-033

QUANTIFYING THE OPTIMALITY OF A DISTRIBUTED RL-BASED AUTONOMOUS EARTH-OBSERVING CONSTELLATION*

Mark Stephenson†, Lorenzzo Mantovani†, Anaïs Cheval†, and Hanspeter Schaub‡

The objective of the multiagent agile Earth-observing satellite scheduling problem is to maximize the global sum of values of imaging requests imaged by a constellation. Prior work has shown that individually-trained satellites executing a deep reinforcement learning policy can best be induced to cooperate on an Earth-observing mission by sharing their intended tasks as they select them; these agents are operating on asynchronous variable-duration decision intervals. Additionally, methods for determining the global optimum for request allocation have been developed. In this work, these two developments will be combined to quantify how close to global optimality the distributed method can achieve. The methods are evaluated over variety of constellation configurations and request distributions, demonstrating that the intent sharing method is very effective at collaboratively, autonomously, and reactively scheduling observations for the constellations.

INTRODUCTION

Scheduling constellations of agile Earth-observing satellites (AEOSs) is a challenging problem, especially with higher densities of requests 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 multiagent environment [3]. This paper explores how best to induce cooperation between a cluster of satellites operating on different intervals to complete a many-request 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.

*This work is a continuation of "Intent Sharing For Emergent Collaboration In Autonomous Earth Observing Constellations," presented at the 2024 AAS Rocky Mountain GN&C Conference.

†Ph.D. Student, Ann and H.J. Smead Department of Aerospace Engineering Sciences, University of Colorado, Boulder, Boulder, CO, 80303. AIAA Member. Correspondence: Mark.A.Stephenson@colorado.edu

‡Distinguished Professor and Department Chair, Ann and H.J. Smead Department of Aerospace Engineering Sciences, University of Colorado, Boulder, Boulder, CO, 80303. AAS Fellow, AIAA Fellow.

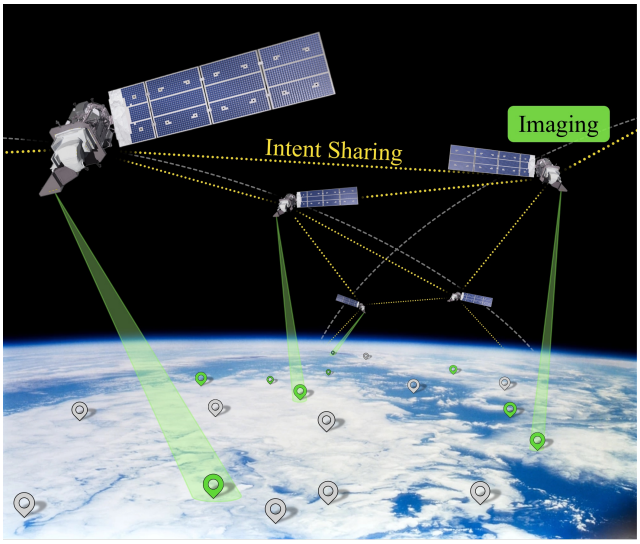


Figure 1: Concept of autonomous multiagent Earth-observing constellation tasking with intent sharing. Satellites share the request they intend to complete next in order to deconflict tasks.

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 requests [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.

Multiagent interaction has been identified as an important next step in multiagent satellite planning [20]. Asynchronous interagent reactivity can be combined with existing allocation algorithms, as demonstrated in [21]. To generalize to a scalable multi-satellite constellation using reinforcement learning (RL), reference [3] adds communication between single-agent-trained satellites to

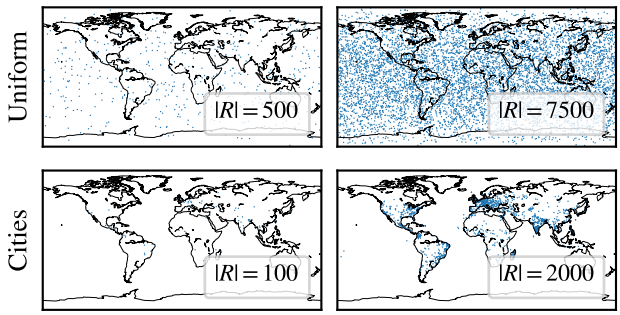


Figure 2: Examples of each target distribution over various request densities.

deduplicate efforts.

This work builds on reference [3] by introducing considerations for communications when satellites are not operating on the same decision interval. In this work, two methods are compared for solving the agile Earth-observing constellation scheduling problem (AEOCSP): a mixed-integer linear program (MILP)-based approach that yields globally optimum solutions [22] and a distributed RL-based approach that induces collaboration among agents [23]. The goal of this paper is to demonstrate that the intent sharing method from [23] can achieve near-optimal results in a multiagent system using RL when compared to the global MILP solutions, just as [2] demonstrated for a single agent. To do so, the intent sharing method is evaluated against the MILP solutions (both a 2-hour solution and an upper bound) for a variety of constellation configurations and request distributions. A string of four satellites is considered with varied spacing and a small Walker-delta constellation are both considered in order to investigate the impact of satellite spacing on the performance of the intent sharing method. The resulting performance shows that the intent sharing method is able to efficiently and effectively schedule observations for the constellation.

PROBLEM STATEMENT

The AEOCSP is considered for a homogeneous constellation of imaging satellites with a joint objective to maximize the cumulative value of unique fulfilled requests. For a comparison of methods, it is assumed that either open-loop task schedules can be uplinked to the constellation or that the constellation has intersatellite communication capabilities to autonomously coordinate closed-loop request selection.

Request Model

Image requests are specified using a point-based request model as a tuple consisting of an Earth-fixed location and a priority, $\rho_i = (\boldsymbol{r}_i, r_i) \in R$. All requests start in the unfulfilled set $R = \mathcal{U}$. When a ground location is imaged, the corresponding request is removed from $\mathcal{U}$ and added to the set of fulfilled requests $\mathcal{F}$. The value of fulfilling $\rho_i \in \mathcal{U}$ yields a reward equal to the priority r_i; fulfilling an already-fulfilled request $\rho_i \in F$ yields no reward. Should an operator want to image the same location multiple times, multiple requests can be made in that location with constraints on the availability period of each request.

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, request priorities are uniformly distributed $r_i \in [0, 1]$. The city-based distribution is intended to be representative of many Earth-observing missions, where the request type tends to be concentrated in specific areas of interest (e.g. coastline, urban areas, etc.).

Satellite Dynamics

For a satellite to fulfill a request, various dynamic requirements must be satisfied. Each satellite has a body-fixed instrument pointing in the $\hat{\boldsymbol{c}}$ direction. Being an agile satellite, an attitude control algorithm is used to orient the instrument to point at the location of the request [24]. The satellite's instrument must be pointed at the request location within $\delta\theta$ and track its relative motion within a rate threshold $\delta\dot{\theta}$ to take an image, fulfilling the request.

Because the satellites are agile (i.e. capable of slewing along-track), each request is accessible for opportunity intervals $[t^o, t^c] = o \in O_i$, as defined by quality of image constraints such as elevation angle, time of day, or lighting conditions. In this work, only a minimum elevation angle constraint $\phi_{\min}$ is considered, but generalization to other constraints is straightforward. The satellite must be able to satisfy the dynamic constraints within the opportunity interval to fulfill the request.

The combination of attitude-based pointing and limited access opportunities yields a slew time constraint between requests. The satellite must slew from its current attitude to the request attitude before the opportunity for the request closes.

Simulation Environment

Each satellite's dynamics are ultimately defined by a high-fidelity simulator of the constellation and environment. The environment is configured using BSK-RL†, an open-source package for building spacecraft tasking environments for RL [25]. The environment includes component-level dynamics models, flight-proven flight software algorithms, and models of the space environment. In particular, the attitude control system is modeled with reaction wheels driven by torque commands from a nonlinear controller; this results in flight accurate transition dynamics between requests.

The package uses Basilisk‡ for the underlying high-fidelity spacecraft simulation [26]. Agents interact with the environment via the Gymnasium API, allowing for compatibility with all major RL frameworks [27].

Optimization Objective

The objective of the problem is to maximize the sum of fulfilled requests,

$$\text{maximize} \sum_{\rho_i \in \mathcal{F}} r_i \tag{1}$$

$$\text{s.t. dynamics constraints} \tag{2}$$

subject to the previously described constraints imposed by the dynamics of the satellites and the environment in fulfilling requests.

*City location data from simplemaps.com.

†https://avslab.github.io/bsk_rl/

‡https://avslab.github.io/basilisk/

METHODS: MILP FORMULATION

A common class of methods used for satellite and constellation task sequencing are MILPs. Reference [22] describes a method for efficiently constructing a graph representation of the problem and solving it with a MILP solver. This method uses a learned transition time estimator to more accurately represent the problem and find solutions that are optimal when executed in the high-fidelity simulator. In summary, the method is as follows:

1. First, a neural network based slew duration estimator is trained using supervised learning. This estimator predicts the time it takes for the satellite to transition from one state to pointing at an upcoming request. The estimator is trained on a dataset of simulated transitions.

2. For the set of requests and horizon being planned for, the opportunities for each request are calculated. These opportunities are computed based on any constraints that must be satisfied for the request to be fulfilled. In this work, only the minimum elevation angle constraint is considered, but the method generalizes to other time-based constraints.

3. A graph of feasible transitions between discrete points in time along each opportunity is constructed. Reference [22] demonstrates how this graph can be constructed in a sparse manner to improve solution times.

4. Finally, a MILP is formulated to maximize the sum of fulfilled requests. The MILP uses the graph of feasible transitions to ensure that the solution respects the dynamics of the satellite and the environment. The MILP is solved using a commercial solver, and the solution is executed in the high-fidelity simulator.

METHODS: RL WITH INTENT SHARING

The goal of this paper is to compare a scalable RL-based approach to the optimal MILP solutions to the AEOCSP. For the RL-based approach, the problem is formalized as a MDP, RL is used to train a policy for a single-agent instance of the environment, and an intent sharing algorithm is defined to induce collaboration between agents.

Markov Decision Process

The multiagent problem is formalized as a decentralized partially-observable semi-Markov decision process (Dec-POsMDP), a generalization of a MDP for decentralized decision-making between multiple agents with a single goal [28]. Semi-Markov refers to the fact that each step has a time duration associate with it that can vary between steps [29]. A deterministic simulation of the scenario is given as the generative model $G(s, a) = s'$, as described in the simulation environment section. The elements of the Dec-POsMDP 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.

Quantity	Dim.	Normalization	Description
${}^{\mathcal{H}}\boldsymbol{\omega}_{\mathcal{BE}}$	3	0.03 rad/s	Hill-frame body angular rate
${}^{\mathcal{H}}\hat{\boldsymbol{c}}$	3	-	Hill-frame instrument pointing direction
${}^{\mathcal{E}}\boldsymbol{r}_{\mathcal{BE}}$	3	r_E	Earth-fixed position, Earth radius-normalized
${}^{\mathcal{E}}\boldsymbol{v}_{\mathcal{BE}}$	3	v_{orb}	Earth-fixed velocity, orbital velocity-normalized
r_n	N	-	Request value $\in [0, 1]$
${}^{\mathcal{H}}\boldsymbol{r}_{R_nB}$	$3N$	800 km	Hill-frame request location
θ_n	N	$\pi/2$	Angle between request and boresite
ω_n	N	0.03 rad/s	Angle rate between the request and boresite
$t_n^{\text{open}}, t_n^{\text{close}}$	$2N$	300 s	Request opportunity window

Table 1: Elements in the observation space o and their normalization constants, as in [2]. Lower half of table is provided for next N unfulfilled requests. The prepended superscript indicates the frame in which the observation is expressed.

- **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, $\mathcal{O} = \mathcal{O}_1 \times ... \times \mathcal{O}_s = \mathcal{O}_{\text{sat}}^S$. 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. The observation includes satellite states and information about the next $N = 32$ unfulfilled requests; the elements included in the observation are determined by a combination of experimentation and expert knowledge.

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

  - $a_{\text{im},i} \times N$: Actions to attempt to fulfill each of the $N = 32$ next unfulfilled requests, by time of next encounter. The satellite slews to point the instrument at the request location and — if attitude, rate, and access constraints are met — takes an image. As such, this action does not guarantee successful imaging.

- **Transition Probability Function $T(s'|s, a)$:** Transitions and transition durations are generated by the deterministic simulator. The simulator propagates for a variable amount of time, depending on the actions taken and how the environment evolves during propagation. Propagation is halted when one of two conditions is met:

  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.

  The simulator returns the next state s', the reward r, and the step duration Δt.

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

$$R(s, s') = \sum_{\rho_i \in \mathcal{R}} \begin{cases} r_i & \text{if } \rho_i \in \mathcal{U} \text{ and } \rho_i \in \mathcal{F}' \\ 0 & \text{else} \end{cases} \tag{3}$$

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 [30]. 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.

In this work, the proximal policy optimization (PPO) algorithm is used to train a policy for a single-agent instance of the environment [31]. PPO has been shown to be performant across a variety of domains, including spacecraft tasking [32]. The RLlib implementation is used due to its scalability [33]. Agents are able to learn about 10M steps of training, or 10 to 15 years of simulated on-orbit time, in about 24 hours of wall-clock time on a 32-core CPU.

Intent Sharing for Emergent Collaboration

Because multiagent RL presents a number of theoretical and practical challenges, prior work has investigated the efficacy of deploying single-agent trained policies in a multiagent environment by inducing collaboration in order to achieve scalability [3, 23]. Collaboration is induced by modifying each agent's request list based on the actions of the other agents, thus leveraging their closed-loop planning to respond to a modified environment state.

The "intent sharing" algorithm (Algorithm 1) was found to be most effective if intersatellite communication is available. In this algorithm, each agent shares the request it intends to fulfill with the other agents, in a set denoted by $\mathcal{I}$. The other agents then temporarily remove that request from their request list, as shown in Algorithm 1. If the request is successfully fulfilled, all agents permanently remove the request from their request list.

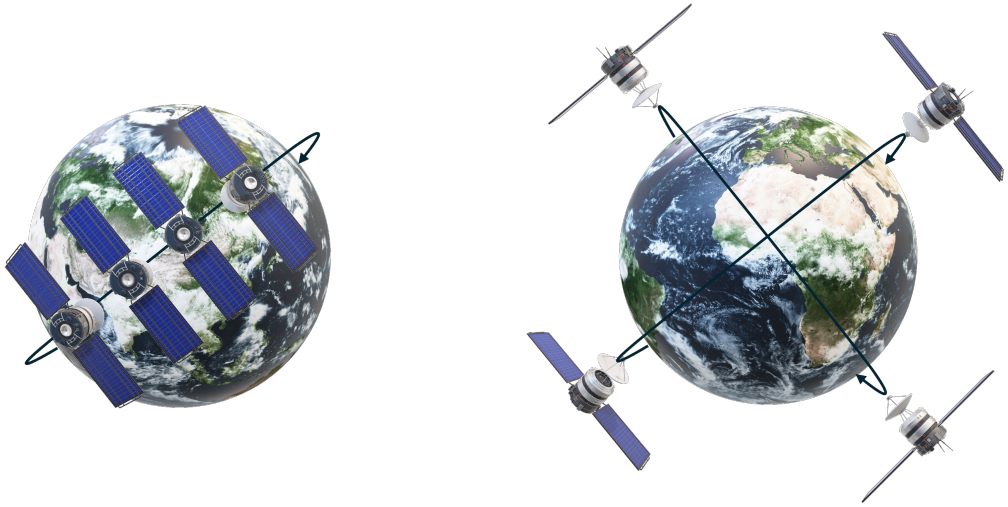
Algorithm 1 Intent Sharing

for each agent n **do**
  if current request ρ not complete (fulfilled or out of range) **then**
    assign ρ to agent n
  else current request ρ complete
    $\mathcal{I} \leftarrow \mathcal{I} \setminus \{\rho\}$
    use policy π to select request ρ' from $\mathcal{U} \setminus \mathcal{I}$
    assign ρ' to agent n
    $\mathcal{I} \leftarrow \mathcal{I} \cup \{\rho'\}$
  end if
end for

With this algorithm, satellites avoid wasting time on a task that another satellite is already planning to complete. This allows the agents to more effectively use their time and resources to fulfill requests, leading to a higher cumulative reward.

RESULTS

The performance of the intent sharing method relative to the optimal MILP solution is evaluated for two constellations—a string constellation with variable spacing and a small Walker-delta



(a) Illustration of the string constellation configurations. Four satellites in the same orbital plane with true anomaly spacing. True anomaly separation of 0.1, 1.0, and 10.0 degrees were considered.

(b) Illustration of the Walker-delta constellation configuration. Two orbital planes with two satellites each, separated by 180° in the longitude of the ascending node and true anomaly.

Figure 3: Illustrations of the two different constellation configurations used in this work.

constellation—over various request distributions.

Constellation Configurations

The string constellation contains all satellites in the same orbital plane, varying their relative true anomaly. True anomaly separation of 0.1, 1.0, and 10.0 degrees were considered, based on Reference [23]. The increase in spacing between satellites leads to a decrease in tasking conflicts. The string configuration was chosen to investigate how the RL-based policy benefits from intent-sharing in different scenarios with different levels of conflict.

A small instance of the Walker-delta constellation was also tested. Two planes with 2 satellites in each were used. The orbital planes have a 180 degrees separation in the longitude of the ascending node while satellites in the same orbital plane have a separation of 180 degrees in true anomaly. Conflict in this configuration arises when the ground path of satellites in different planes intersect. It was selected to complement the string configuration by creating zones of low and high conflict.

Performance

In each case, the performance of the intent sharing method is compared to two values: the MILP solution computed with two hours of solve time and the upper bound on optimality returned by the MILP solver. The true optimal solution falls between these bounds.

String Constellation First, the performance is evaluated on three configurations of the 4-satellite string constellation to investigate how the spacing of the satellites necessitates collaboration. The phasing of the satellites is varied at 0.1°, 1.0°, and 10.0°, with the hypothesis that closer spacing

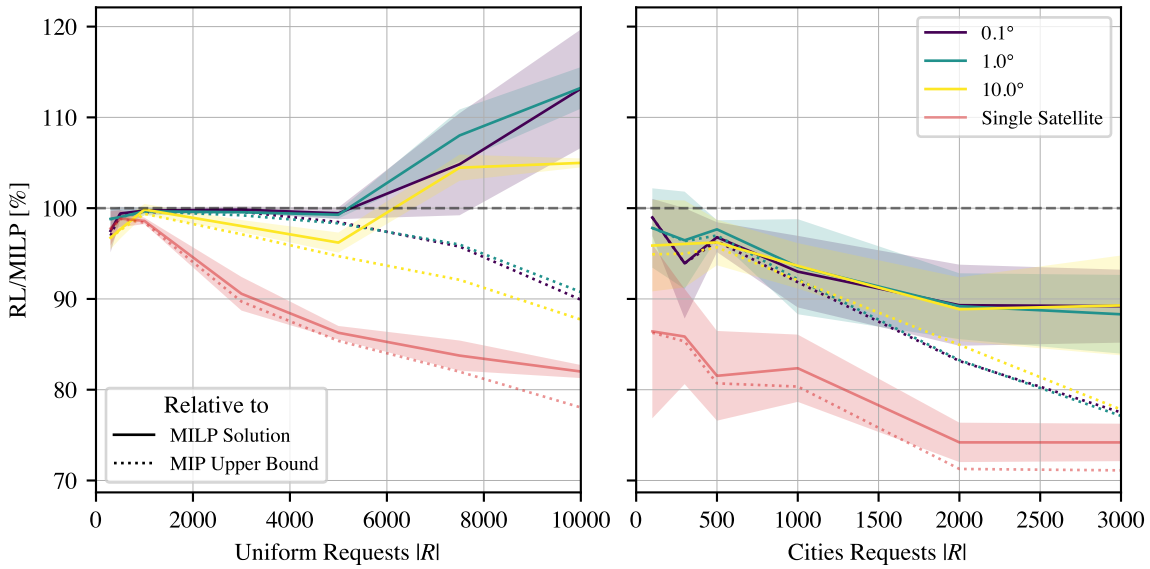


Figure 4: For the string constellation with various phasings, performance of intent sharing relative to the MILP solution across request distributions. Single-satellite RL performance is given for comparison.

will require more performant collaboration to achieve global optimality.

Figure 4 gives the results for the string constellations, compared to how optimally the policy performs in a single agent case. The intent sharing method shows strong performance across configurations and request densities. Over lower densities ($|R| \leq 5000$) of uniformly spaced requests, the intent sharing method achieves the same performance as the MILP solutions, which generally converge. At higher uniform densities, the intent sharing method is able to outperform the solution found by the MILP solver in two hours[§]; even when compared to the upper bound of optimality, intent sharing still achieves within 90% of the bound on the optimal solution. Over city-distributed requests, the performance of intent sharing is still strong relative to the MILP solutions. The intent sharing method is able to achieve within 90% of the two-hour solution across all densities, and within 77% of the upper bound.

For both request distributions, the intent sharing method performs better relative to the MILP solution than the single-agent policy performs relative to the single-agent MILP solution. This implies that the intent sharing method maintains the performance of the single-agent policy while benefiting from collaboration as a means to simplify the problem on a per-agent basis. This is in contrast with global planning methods, which tend to become less efficient as the number of agents being planned for increases.

Figure 5 (uniform requests) and Figure 6 (city requests) investigate how each satellite contributes to the total reward collected by the constellation. Both methods tend to distribute the most requests to the lead satellite when encountering a low density of requests and large satellite separation. As the density of requests increases, more requests become available for the other satellites to image, decreasing the reward-per-satellite gap.

[§]In 3/15 cases with 10,000 requests, the MILP solver did not find any reasonable solution.

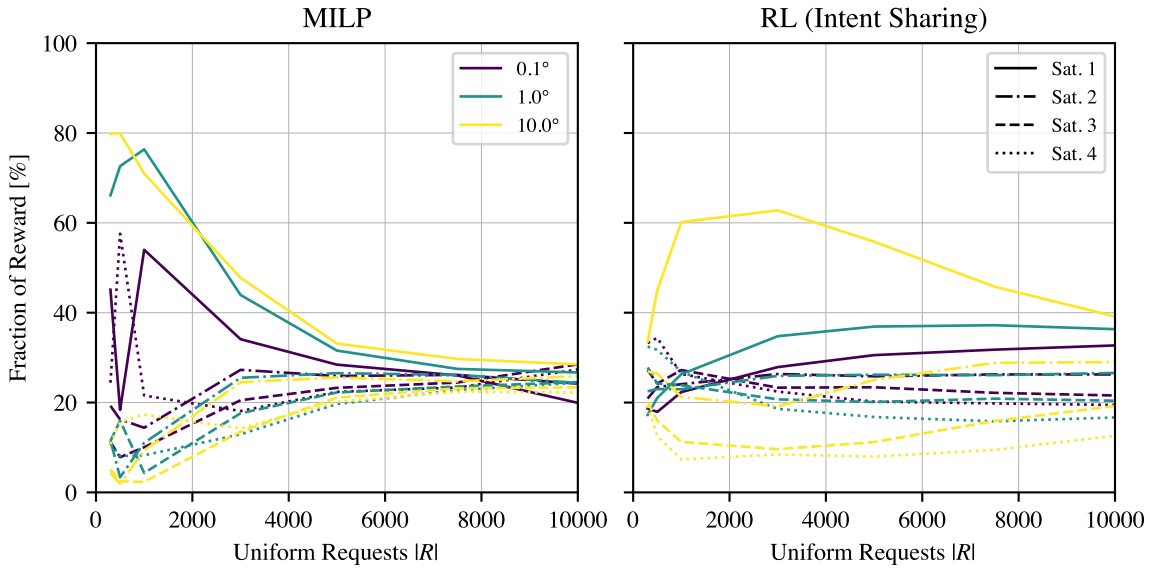


Figure 5: Fraction of reward collected by each satellite with uniformly distributed requests.

Walker-Delta Constellation The Walker-delta constellation with two planes of two satellites each is evaluated to investigate how a sparse constellation configuration affects the performance of the intent sharing method. Figure 7 gives the results of these experiments. As with the string constellation, the performance of the intent sharing method is generally strong, but it struggles most for high-density city-distributed requests. Still, the performance relative to the two-hour MILP solution remains above 75%. The higher request density caused by clustered cities is challenging for the RL-based policy, while the number of requests is still low enough for the MILP solver to find a suitable solution. Nonetheless, the intent sharing case presents a higher optimality fraction than the single satellite.

Discussion

Onboard Solution Time A key observation about the performance of the RL and intent sharing system is the low computational cost of evaluating the policy onboard the satellite. Because it consists of a single 2×2048 node fully-connected neural network, the policy can be evaluated in less than 10 ms on a modern CPU.

Relative Complexity and Scalability A limitation of the global planning techniques is that they scale poorly with constellation size and planning horizon [22]. The latter factor, planning horizon, is driven by the availability of uplink opportunities to the constellation. An important property of the intent sharing method is that the complexity of the problem does not increase with the number of agents nor the planning horizon since no global solution is required.

Communication Requirements One technical requirement of satellites operating in a distributed manner is the ability to communicate with other agents. In this work, it is assumed that intersatellite communication is available, and that the satellites can share their intended actions with each other. This is increasingly realistic as intersatellite communication technology and networks continue to grow. Even then, prior work [3] has demonstrated that full communication among all agents is not necessary for other communication-based reactive planning to be effective; in particular, only

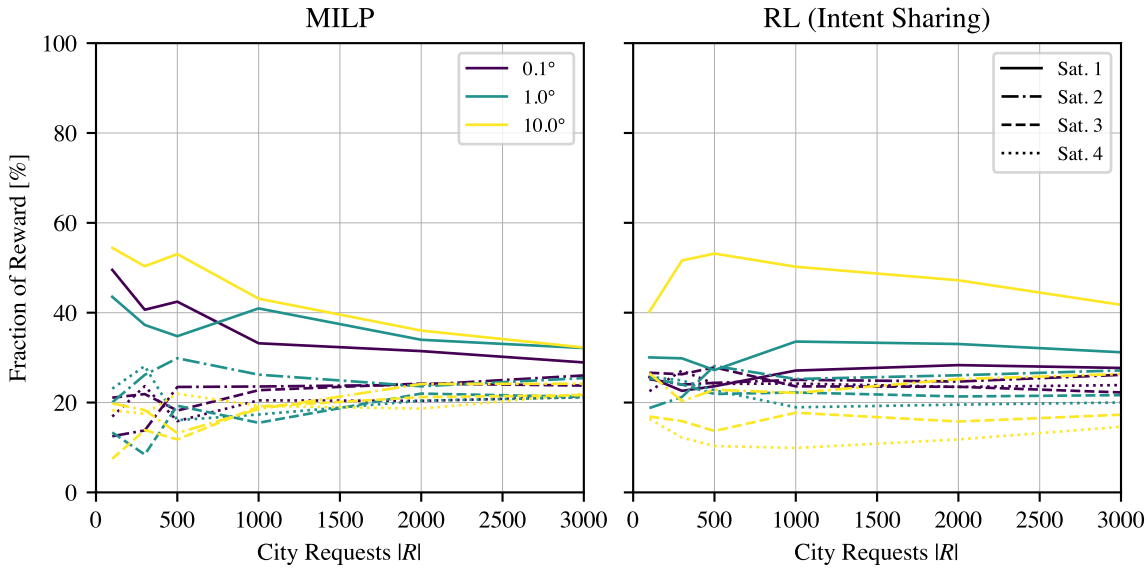


Figure 6: Fraction of reward collected by each satellite with city distributed requests.

communication among local groups of satellites is sufficient to achieve good performance.

CONCLUSION

This work demonstrates that single-agent RL policies can be effectively deployed in a multiagent environment by inducing collaboration through intent sharing, analyzing a variety of constellation configurations and request distributions. The intent sharing method is shown to induce multiagent performance that is closer to optimal than in the case with a single RL-based policy. This trend is opposite to the trend seen in global planning methods, which tend to become less efficient as the number of agents being planned for increases. As such, the intent sharing method for multiagent RL is a performant approach to achieving scalable, autonomous, and reactive tasking for Earth-observing constellations.

ACKNOWLEDGEMENT

This work is partially supported by a NASA Space Technology Graduate Research Opportunity (NSTGRO) grant, 80NSSC23K1182. This work was also partially supported by the AFRL research award 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.

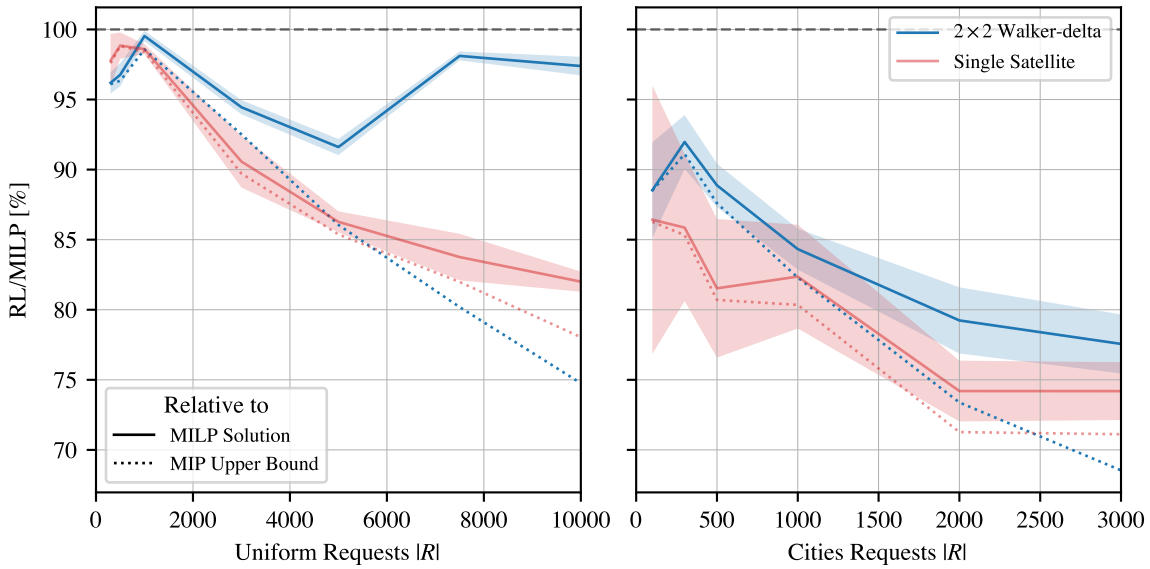


Figure 7: On the Walker-delta constellation, performance of intent sharing relative to the MILP solution across request distributions. Single-satellite RL performance is given for comparison.

[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, 10.1023/A:1018920709696.

[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, 10.48550/arXiv.1802.04240.

[16] A. Harris, T. Teil, and H. Schaub, "Spacecraft Decision-Making Autonomy Using Deep Reinforcement Learning," *AAS Spaceflight Mechanics Meeting*, Maui, Hawaii, 2019-01-13/2019-01-17.

[17] A. Hadj-Salah, R. Verdier, C. Caron, M. Picard, and M. Capelle, "Schedule Earth Observation Satellites with Deep Reinforcement Learning," Nov. 2019, 10.48550/arXiv.1911.05696.

[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] D. Selva, A. Golkar, O. Korobova, I. Cruz, P. Collopy, and O. de Weck, "Distributed Earth Satellite Systems: What Is Needed to Move Forward?," *Journal of Aerospace Information Systems*, Vol. 14, Aug. 2017, pp. 1–26, 10.2514/1.I010497.

[21] A. A. Jaramillo, B. Gorr, H. Gao, D. Selva, A. Mehta, Y. Sun, V. Ravindra, C. H. David, and G. H. Allen, "Decentralized Consensus-based Algorithms for Satellite Observation Reactive Planning with Complex Dependencies," *AIAA SciTech Forum*, Orlando, FL, AIAA, Jan. 2025.

[22] M. A. Stephenson and H. Schaub, "Optimal Agile Satellite Target Scheduling with Learned Dynamics," *Journal of Spacecraft and Rockets*, Oct. 2024, pp. 1–12, 10.2514/1.A36097.

[23] M. Stephenson, L. Mantovani, and H. Schaub, "Intent Sharing for Emergent Collaboration in Autonomous Earth Observing Constellations," *AAS GN&C Conference*, Breckenridge, CO, Februrary 2, 2024.

[24] H. Schaub and S. Piggott, "Speed-Constrained Three-Axes Attitude Control Using Kinematic Steering," *Acta Astronautica*, Vol. 147, June 2018, pp. 1–8, 10.1016/j.actaastro.2018.03.022.

[25] M. A. Stephenson and H. Schaub, "BSK-RL: Modular, High-Fidelity Reinforcement Learning Environments for Spacecraft Tasking," *75th International Astronautical Congress*, Milan, Italy, IAF, Oct. 2024.

[26] 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.

[27] 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.

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

[29] K. Menda, Y.-C. Chen, J. Grana, J. W. Bono, B. D. Tracey, M. J. Kochenderfer, and D. Wolpert, "Deep Reinforcement Learning for Event-Driven Multi-Agent Decision Processes," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 20, Apr. 2019, pp. 1259–1268, 10.1109/TITS.2018.2848264.

[30] R. S. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. Adaptive Computation and Machine Learning, Cambridge, Massachusetts London, England: The MIT Press, 2 ed., 2018.

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

[32] A. Herrmann and H. Schaub, "Reinforcement Learning for the Agile Earth-Observing Satellite Scheduling Problem," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 59, Oct. 2023, pp. 1–13, 10.1109/TAES.2023.3251307.

[33] 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," *Proceedings of the 35th International Conference on Machine Learning*, Vol. 80, June 2018, pp. 3053–3062.