Satellite Network Data Transmission Policy Dec-POMDP

1st Aidan Bagley*, 2nd Xavier O'Keefe*
*Ann & H.J. Smead Department of Aerospace Engineering Sciences, University of Colorado, Boulder, Boulder, CO, USA
aidan.bagley@colorado.edu, xavier.okeefe@colorado.edu

I. Introduction

There is growing importance for satellite networks for global communications, Earth observation, and space-based services. Despite their potential, efficient data transmission among satellite nodes towards a ground station remains a significant challenge due to limited communication resources, dynamic network topologies, and uncertain environmental conditions. In particular, routing data packets between satellites in a way that enables effective ground station transmission scheduling is a critical problem.

A major source of complexity in this problem comes from the inherent uncertainty in transmission success probabilities. Each satellite must make routing decisions without full knowledge of the goals, priorities, or current states of other satellites in the system. While centralized approaches exist, they can computationally expensive and often fall into local optima which makes decision-making difficult.

To address these challenges, this work proposes a decentralized control framework in which each satellite operates based on a local policy. Specifically, we model the problem as a Decentralized Partially Observable Markov Decision Process (Dec-POMDP), which captures both the uncertainty in communication and the limited observability of the network state.

Using this formulation, we develop local controllers that aim to efficiently route data among different satellite nodes to a ground station while minimizing the cost of latent information. Our solution approach uses policy iteration to optimize each controller's policy, and we evaluate the network of controllers across a few simulated scenarios based on the percentage of latent information successfully delivered to the ground station.

II. Background and Related Work

Satellite constellations represent coordinated groups of satellites that achieve coverage, redundancy, and performance that single satellites can not provide. These constellations are typically described by their altitude above the Earth's surface since different altitudes are tailored to specific mission objectives.

Low Earth Orbit (LEO) satellite constellations, typically operating at altitudes between 500 and 1,500 kilometers, are becoming increasingly popular due to the declining cost of launch vehicles and satellite technology. Since these constellations orbit close to Earth, they enable low-latency communication with ground users. However, only subsets of the constellation are visible to any ground user at a given time, which in turn requires dense deployments of satellites arranged in orbital shells to achieve global coverage. A prominent example is SpaceX's Starlink, which provides internet access to users on the ground via a large, distributed satellite network.

Although they have latency advantages, LEO satellites experience rapid orbital motion, with orbital periods of approximately 90 minutes. This leads to highly dynamic network topologies that create challenges for communication scheduling and task planning. As a result, efficient information routing is essential for maintaining robust and reliable service to users on the ground.

On the other end of the orbital spectrum, Medium Earth Orbit (MEO) and Geostationary Earth Orbit (GEO) satellites operate at significantly higher altitudes—between 2,000 and 35,000 kilometers for MEO, and at approximately 35,786 kilometers for GEO. These orbits offer complementary capabilities to LEO constellations, particularly in terms of broader regional coverage and longer visibility windows. This extended visibility is a key reason why Global Navigation Satellite Systems (GNSS) operate in this regime to provide continuous, global positioning coverage with high reliability. However, the increased distance from the Earth's surface introduces higher communication delay. Therefore, if quick delivery of information is critical to mission success, these latency constraints must be addressed through efficient routing algorithms.

Designing protocols to manage information flow within a satellite constellation and to send information toward ground stations involve several complex challenges. For instance, there is variable link quality between satellites due to their continuous relative motion. This issue is especially problematic in optical communication systems which necessitate precise line-of-sight pointing to successfully transmit data. Additionally, transmissions to ground stations are disrupted by atmospheric conditions and space weather. Both of which can decrease the signal to noise ratio making the information more challenging to decode.

Also, ground stations have limited and intermittent visibility windows for each satellite. These windows are often shared among multiple satellites, requiring ground stations to make scheduling decisions. Due to these factors, transmission success becomes inherently stochastic. Consequently, optimizing information routing and scheduling is a nontrivial task, requiring algorithms that can operate effectively under uncertainty.

All of these challenges have led to many different approaches to address the Satellite Data Transmission Scheduling Problem (SDTSP). The SDTSP has been proven to be NP-Hard, meaning no known algorithm can solve all instances of the problem in polynomial time. The core objective of this problem is typically to maximize the total amount of data successfully received by a ground station, subject to visibility constraints and imperfect communication.

Most existing approaches solve the SDTSP from the ground station's perspective using a centralized planning model, where all scheduling decisions are made with full global knowledge of satellite positions and communication windows.

For instance, Chen et. al [2] propose a population-based stochastic evolutionary algorithm that combines a Particle Swarm Optimizer (PSO) with genetic algorithm components such as crossover and mutation. This hybrid approach balances exploration and exploitation to avoid prematurely falling into local optima while maintaining a fast convergence. Notably, this problem is formulated from the perspective of the ground station.

Similarly, Zhang et. al [3] propose a genetic based optimization approach, but first use a classification step through a Support Vector Machine (SVM). The SVM leverages the inherit periodicity of the SDTSP to identify a more tailored optimization approach. Afterwards, they use a Non-Dominated Sorting Genetic Algorithm II (NSGA-II) to solve the multi-objective optimization problem for a pareto optimal solution. Like Chen et al., Zhang et al. operate within a centralized model focused on the ground station.

In a follow-up paper, Zhang et. al. [4] build upon their work by using a Mixed-Integer Linear Program (MILP) with an improved genetic optimization algorithm. As opposed to their previous work, this paper attempts to find solutions respecting a larger variety of constraints and demonstrate the performance of their improved genetic optimization solver to higher-quality solutions with more realistic constraints.

On the other hand, Chen et. al [5] also leverage the periodicity of the problem but formulate a Non-Uniform Time Slot Division Method (NUTSDM) to identify interested parties in a given observation window. They model the resulting interactions with a Stackelberg game model to enable sequential decision-making among the interested parties. This formulation departs from the pure centralized optimization approach by considering the behavior of multiple stakeholders.

Finally, Chen et. al [6] address this problem from a machine learning perspective. They formulate the problem as MDP with a heuristic that prioritizes relay satellites that have better access windows. Subsequently they form a Deep-Q Network (DQN) to learn an approximation of the action value function that yields a policy for communication time window selection. However, they have perfect observability of the full state-space leading to a very computationally expensive approach.

While the traditional SDTSP focuses on optimizing ground station scheduling to maximize downlink throughput, it typically assumes that satellites follow a deterministic schedule. However, in highly dynamic environments with stochastic communication success and overlapping access windows, this centralized approach may be insufficient. By extending the decision-making capabilities to satellites treating them as active agents capable of making real-time, local routing and transmission decisions, the overall system may adapt more effectively to changing conditions. This decentralized perspective may lead more robust, scalable, and cooperative communication strategies.

III. Problem Formulation

A. Decentralized POMDP Definition

We model a communication task involving a linear chain of n satellites, each with nearest-neighbor communication links and probabilistic success of transmission to a ground station. The objective is to maximize the number of information packets delivered to the ground station as efficiently as possible. Each satellite can attempt to transmit packets to adjacent satellites or directly to the ground, subject to stochastic transmission success and per-satellite storage constraints.

State Space

Let ψ be the per-satellite packet capacity and m be the total number of packets in the system. The system state at time t is denoted $s_t \in \mathcal{S}$, where s_t^i indicates the number of packets stored at satellite i. The full state space is defined as:

$$\mathcal{S} = \left\{ s \in \{0, 1, \ldots, \psi\}^n \;\middle|\; \sum_{i=1}^{n} s^i \leq m \right\} \tag{1}$$

For example, a state in a 3-satellite system with two packets at satellite 1 and one at satellite 2 would be $s = (2, 1, 0)$.

Action Space

Each satellite independently selects an action from its local action set:

$$\mathcal{A}_i = \{\text{Send Left}, \text{Send Right}, \text{Send to Ground}, \text{Wait}\} \tag{2}$$

Feasible actions are constrained by topology. For instance, the leftmost satellite cannot send left. The joint action at time t is defined as:

$$a_t = (a_t^1, a_t^2, \ldots, a_t^n) \in \mathcal{A}_1 \times \cdots \times \mathcal{A}_n = \mathcal{A} \tag{3}$$

Transition Function

The transition function $T(s' \mid s, a)$ defines the probability of transitioning from state s to state s' under joint action a. Transitions are stochastic due to probabilistic success or failure of packet transmissions.

Let:

- p_{comm} be the probability that a packet is successfully passed between satellites,
- θ^i be the probability that satellite i successfully transmits to the ground station,
- ψ be the maximum storage capacity per satellite.

Each satellite i takes action a^i at each timestep. Possible actions include:

- **Send Left**: Attempt to pass a packet to satellite $i-1$, only if $i > 1$,
- **Send Right**: Attempt to pass a packet to satellite $i+1$, only if $i < n$,
- **Send to Ground**: Attempt to transmit a packet to the ground station,
- **Wait**: Take no action.

State transitions depend on:

1) Whether the satellite has at least one packet to send, $s^i > 0$,
2) Whether the receiving satellite has storage space, $s^j < \psi$,
3) Whether the transmission actually succeeds based on a Bernoulli trial.

Since each satellite's action may succeed or fail independently, there are 2^n possible joint outcome combinations for a given joint action a. Let the outcome vector $z = (z^1, \ldots, z^n) \in \{0,1\}^n$ indicate which actions succeeded. For an action a^i, let the corresponding entry of the outcome vector z^i be equal to one if the action succeeded as intended and zero if it failed. The probability of a joint outcome given a joint action is computed:

$$P(z \mid a) = \prod_{i=1}^{n} \left(\text{succ}(a^i)^{z^i} \cdot (1 - \text{succ}(a^i))^{1-z^i} \right) \tag{4}$$

Then, let $\text{succ}(a^i)$ denote the success probability of action a^i:

$$\text{succ}(a^i) = \begin{cases} p_{\text{comm}}, & \text{if } a^i \in \{\text{Send Left}, \text{Send Right}\} \\ \theta^i, & \text{if } a^i = \text{Send to Ground} \\ 1, & \text{if } a^i = \text{Wait} \end{cases} \tag{5}$$

For each possible outcome z, the corresponding next state s'_z is computed deterministically depending on the outcome from the Bernoulli trial as follows:

- If a^i = Send Left and $z^i = 1$: decrement s^i by 1 and increment s^{i-1} by 1, subject to $s^i > 0$ and $s^{i-1} < \psi$,
- If a^i = Send Right and $z^i = 1$: decrement s^i by 1 and increment s^{i+1} by 1, subject to $s^i > 0$ and $s^{i+1} < \psi$,
- If a^i = Send to Ground and $z^i = 1$: decrement s^i by 1, subject to $s^i > 0$,
- If a^i = Wait, $z^i \coloneqq 1$: the state remains unchanged for satellite i,
- Otherwise if $a^i \in \mathcal{A}_i$ and $z^i = 0$ the state remains unchanged for satellite i aside from any successful transmission from $j \to i$.

As a result, the overall transition probability to a specific next state, s', is obtained by summing the probabilities of all outcomes, z, that lead to s'. To represent this, we use an indicator function to filter only the relevant outcomes:

$$T(s' \mid s, a) = \sum_{z \in \{0,1\}^n} \left[\prod_{i=1}^{n} \left(\text{succ}(a^i)^{z^i} \cdot (1 - \text{succ}(a^i))^{1-z^i} \right) \cdot \mathbf{1}(s'_z = s') \right] \tag{6}$$

Example: Two-Satellite Transition: To illustrate the transition function, consider a system with two satellites:

- Satellite 1 performs the action **Send Right**,
- Satellite 2 performs the action **Send to Ground**.

Let the parameters be:

- Initial state: $s = (1, 1)$,
- Maximum storage capacity: $\psi = 2$,
- Communication success probability: $p_{\text{comm}} = 0.8$,
- Ground transmission success probability: $\theta^2 = 0.6$.

Then:

$$\text{succ}(a^1) = p_{\text{comm}} = 0.8, \quad \text{succ}(a^2) = \theta^2 = 0.6$$

We enumerate all possible outcome vectors $z = (z^1, z^2) \in \{0,1\}^2$, and compute the corresponding deterministic next state s'_z and probability of each outcome:

z	Description	s'_z	Probability
(0, 0)	Both transmissions fail	(1, 1)	$0.2 \cdot 0.4 = 0.08$
(0, 1)	Sat 1 fails, Sat 2 succeeds	(1, 0)	$0.2 \cdot 0.6 = 0.12$
(1, 0)	Sat 1 succeeds, Sat 2 fails	(0, 2)	$0.8 \cdot 0.4 = 0.32$
(1, 1)	Both transmissions succeed	(0, 1)	$0.8 \cdot 0.6 = 0.48$

As an example, consider the probability of transitioning from state $s = (1, 1)$ to $s' = (0, 1)$ under the joint action $a = (\text{Send Right}, \text{Send to Ground})$.

Among all possible outcome vectors $z \in \{0,1\}^2$, only $z = (1, 1)$ results in the next state $s'_z = (0, 1)$. Therefore:

$$T((0,1) \mid (1,1), (\text{Send Right}, \text{Send to Ground})) = 0.48$$

Reward Function

The reward function encourages successful packet delivery while penalizing inefficiencies and congestion. The total reward $R(s, a, s')$ is computed as:

- **+10.0** for each packet successfully transmitted to the ground.
- **-2.0** for each failed transmission attempt (to ground or neighbor).
- **-1.0** for each transmission attempt to a neighbor (regardless of success).
- **-5.0** for each satellite exceeding its capacity in s'.

Let:

- $G(a^i, s^i, s'^i) = 1$ if satellite i successfully transmits to the ground.
- $F(a^i) = 1$ if satellite i attempts a transmission that fails.
- $P(a^i) = 1$ if satellite i attempts a transmission to a neighbor.

- $\Psi(s'^i) = 1$ if $s'^i > \psi$ and the transmission exceeded capacity.

Then the reward is given by:

$$R(s,a,s') = \sum_{i=1}^{n} \big(10 \cdot G(a^i, s^i, s'^i) - 2 \cdot F(a^i) - 1 \cdot P(a^i) - 5 \cdot \Psi(s'^i)\big) \tag{7}$$

Observation Model

While the underlying state s_t captures the full distribution of packets across the satellite network at time t, each satellite i has only limited local observability of its own state s_t^i. We formalize this partial observability through an observation model.

Observation Space: For each satellite i, the observation space $\mathcal{O}^i$ consists of locally observable information:

$$\mathcal{O}^i = \{0, 1, \ldots, \psi\} \tag{8}$$

Each observation $o_t^i \in \mathcal{O}^i$ is is the current packet count at satellite i.

The joint observation at time t is:

$$o_t = (o_t^1, o_t^2, \ldots, o_t^n) \in \mathcal{O}_1 \times \cdots \times \mathcal{O}_n = \mathcal{O} \tag{9}$$

Observation Function: The observation model is defined by the conditional probability distribution $O(o \mid s', a)$, which gives the probability of joint observation o given that the system transitioned to state s' after taking joint action a.

For local observations of packet counts, the observation likelihood is deterministic, but imperfect:

$$P(o_t^i \mid s', a) = \begin{cases} 0.8, & o_t^i = s' \\ 0.1, & o_t^i = s' - 1 \\ 0.1, & o_t^i = s' + 1 \end{cases} \tag{10}$$

Note that observations are always non-negative, but each satellite has a 20% chance of receiving an imperfect local observation. The joint observation probability is factored as the product of individual observation probabilities:

$$O(o \mid s', a) = \prod_{i=1}^{n} P(o^i \mid s', a^i, s) \tag{11}$$

This observation model reflects the limited, local information available to each satellite, consistent with the decentralized nature of the system. Importantly, satellites cannot directly observe the packet counts or transmission outcomes of others, which presents a key challenge in our decentralized controller design. While this level of observability may or may not be realistic depending on the actual network architecture, it serves to reduce the complexity of the problem, making it more tractable for computational methods.

Discount Factor

We use a discount factor of $\gamma = 0.9$ to reflect the diminishing value of delayed transmissions, promoting faster delivery of packets.

Initial State and Terminal Condition

The initial state distribution $\mu_0(s)$ specifies where the packets are located at the start. Typically, all packets begin at one or more designated source satellites.

The process terminates when all packets have been successfully delivered to the ground station, i.e.,

$$\sum_{i=1}^{n} s^i = 0 \tag{12}$$

In this terminal state, no further actions or rewards are accumulated.

B. Controller Structure

For our decentralized POMDP, we employ a finite-state controller architecture that enables autonomous decision-making by individual satellites without requiring full state observability. This approach is particularly suited to the distributed nature of the satellite network.

1) Joint Controller Architecture: The joint controller $\mathcal{C}$ comprises individual finite-state controllers for each satellite:

$$\mathcal{C} = (\mathcal{C}^1, \mathcal{C}^2, \ldots, \mathcal{C}^n) \tag{13}$$

where $\mathcal{C}^i$ is the controller for satellite i. The joint controller's behavior emerges from the simultaneous operation of these individual controllers, with interactions captured through the transition dynamics, $T(s' \mid s, a)$, of the system.

2) Individual Satellite Controller Design: Each satellite controller $\mathcal{C}^i$ is defined as:

$$\mathcal{C}^i = (N^i, n_0^i, \mathcal{O}^i, \delta^i) \tag{14}$$

where:

- N^i is a finite set of controller action nodes for satellite i
- $n_0^i \in N^i$ is the initial action node for satellite i
- $\mathcal{O}^i$ is the set of observations available to satellite i
- $\delta^i : N^i \times \mathcal{O}^i \to N^i$ is the node transition function

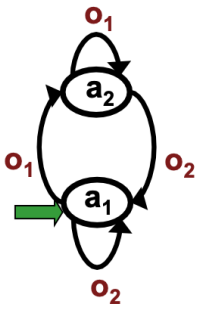


Fig. 1: Finite-State Controller Example [7]

Each controller $\mathcal{C}^i$ encapsulates a specific policy for the satellite. For instance, if C^1 starts at node $n_0^1 = a^1$ and it observes o^1, then it transitions to $n^1 = a^2$ for the next timestep.

More formally, at each time step t, satellite i occupies a controller node $n_t^i \in N^i$. The decision-making process proceeds as follows:

1) The satellite executes action $a_t^i = n_t^i$ defined by its current node
2) After action execution, it receives observation $o_t^i \in \mathcal{O}^i$ related to the real transition dynamics of the system
3) The controller transitions to the next node $n_{t+1}^i = \delta^i(n_t^i, o_t^i)$

At a high level, each controller is responsible for transitioning to the best action node given any observation.

3) Transition Mapping from Observations to Next States: The node transition function δ^i implements a mapping from the current node and observation to the next node:

$$\delta^i : N^i \times \mathcal{O}^i \to N^i \tag{15}$$

This mapping can be represented as a transition table for each controller, where each observation $o^i \in \mathcal{O}^i$ maps from the original node to a successor node. Correctly modeling these transitions is the driving factor between a good and a bad controller.

C. Optimization Objectives

The primary objective is to maximize the expected total discounted reward, which prioritizes successful packet delivery to the ground station:

$$J(\mathcal{C}) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \mid \mathcal{C}\right] \tag{16}$$

where $\gamma = 0.9$ is the discount factor, and the expectation is taken over all possible trajectories induced by the joint controller $\mathcal{C}$. The discount factor $\gamma < 1$ inherently encodes a preference for quicker packet delivery.

IV. Solution Approach

A. Algorithm

The solution approach was based upon a dec-POMDP policy iteration algorithm presented in Bernstein, et. al [1]. The authors propose a representation of multiple agents with a Moore controller where nodes are actions and transitions based on observations in the environment. These controllers are combined into one joint controller, which contains one controller for each satellite. Each controller was initialized with 2 nodes: one for when the satellite has data, and one for when the satellite does not have data. The actions for each controller are initially randomly distributed, therefore the transition mapping function, δ, is initially sub-optimal. However, this function are continuously refined through policy iteration. See Algorithm 1 for our implementation.

The exhaustive backup step is where this algorithm performs most of its work. In this step, all possible transitions and actions are evaluated for one controller, and when an improvement is found, the entries of the transition table are replaced, and the improved controller is returned. This is done for each controller. After the improved controllers are returned, the joint controller is evaluated using Equation 17 from Oliehoek et. al. [7]. This equation computes the infinite-horizon discounted reward incurred for the initial state s and the initial nodes for all controllers I, denoted by $V^m(I, s)$. The value is iteratively compared to the previous controller value terminating when difference between successive values is sufficiently small. As a result, the algorithm and returns the ϵ-optimal controller.

Algorithm 1: Decentralized POMDP Policy Iteration

```
Input: Joint controller π_0, POMDP problem, discount
       factor γ, convergence threshold ε
Output: ε-optimal joint controller π*
Initialize joint controller π_0;
Evaluate initial controller value V_0;
while improvement > ε do
    for each agent i do
        Perform exhaustive backup for agent i's
         controller;
        Update agent i's controller in the joint
         controller;
    end
    Evaluate updated joint controller value V_{k+1};
    improvement ← |V_{k+1} − V_k|;
    V_{k+1} ← V_{k+1};
    k ← k + 1;
    if improvement < ε then
        break;
    end
end
return π*;
```

Our implementation deviates from that proposed by Bernstein in that our algorithm opts for a simpler but less elegant solution to the exponential growth of controllers. Rather than pruning the controllers, our implementation enforces a maximum size constraint and replaces nodes when an improvement is found, rather than removing them altogether. Due to the simple nature of our problem this solution was able to provide sufficient performance. Another difference in our implementation is the stopping condition, where our algorithm uses a simple improvement metric to determine when to stop. The original algorithm uses a function of the iterations and maximum reward as a stopping criteria. This was not implemented because it often resulted in premature breaking due to our problem definition.

B. Verification

The controllers were evaluated using this equation in a Monte-Carlo simulation, where metrics like average reward, completion rate, and actions taken were recorded. The controllers were used to simulate an episode, and the states and actions at each time step were plotted. This allowed for manual verification of the effectiveness of the algorithm.

$$V^m(I,s) = \sum_a \pi(a \mid I) \left[R(s,a) + \gamma \sum_{s',o',I'} P(s',o' \mid s,a) \cdot \delta(I' \mid I,o') \cdot V^m(I',s') \right] \tag{17}$$

where:

$$\pi(a \mid I) \triangleq \prod_i \pi_i(a^i \mid l^i),$$
$$t(I' \mid I,o') \triangleq \prod_i t^i(l'^i \mid l^i, o'^i),$$

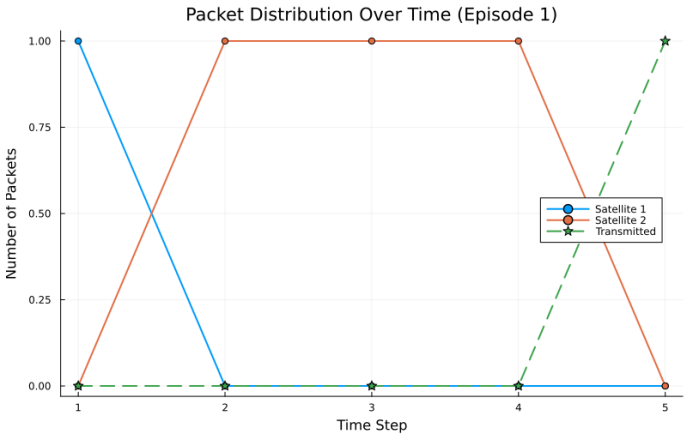
and:

- $I = (l^1, l^2, \ldots, l^n)$ is the set of current controller nodes,
- $a = (a^1, a^2, \ldots, a^n)$ is the joint action of all agents,
- $o' = (o'^1, o'^2, \ldots, o'^n)$ is the joint observation,
- $\pi^i(a^i \mid l^i)$ is the policy for agent i in node l^i,
- $\delta^i(l'^i \mid l^i, o'^i)$ is the transition function of the controller node for agent i,
- $R(s,a)$ is the immediate reward for joint action a in state s,
- γ is the discount factor,
- $P(s',o' \mid s,a)$ is the probability of transitioning to state s' and observing o' given state s and action a,
- $V^m(I,s)$ is the value function for joint controller m, initial nodes I, and state s.

V. Results

A. 2 satellites, 1 packet

The most simple case is one with 2 satellites and only one packet. Our algorithm is able to come to a solution in minutes for this example.



For this simulation satellite 1 had a 10% chance of successfully transmitting a packet to the ground station, and satellite 2 had 50% chance of successful transmission. The controller correctly routes the packet through satellite 2, and is able to get the data to the ground in 5 time steps. The satellites received accurate observations with probability 0.8. The results of Monte-Carlo simulation were:

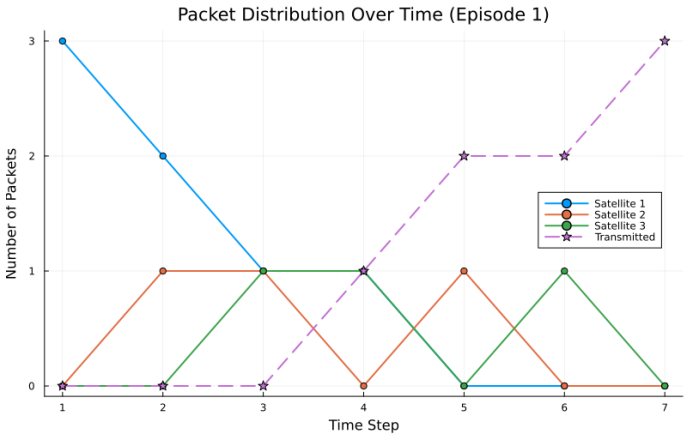
Metric	Value
Average reward per episode	7.188
Standard deviation	5.863
Episode completion rate	100.0%
Average steps per episode	3.121
Transmission efficiency	100.0%

Satellite	Ground TX Prob	TX Attempts	TX Successes (%)	Pass Successes (%)
Satellite 1	0.1	0	0 (0.0%)	1000 / 1056 (94.70%)
Satellite 2	0.5	2061	1000 (48.52%)	0 / 0 (0.0%)

These results are consistent with our coded transition probabilities, and match expectations for reward, transmission success %, and pass success %. A high reward standard deviation and low transmission success percentage make sense given the low chance of successful transmission to ground.

B. 3 satellites, 3 packets

The next level of success involved a higher number of packets, as well as an artificially imposed capacity limit of 3 packets per satellite. The network was able to successfully handle the increase in packets and satellites, but did take 2 hours to run. Again, the satellites received accurate observations with probability 0.8.



Metric	Value
Average reward per episode	-3.16
Standard deviation	29.874
Episode completion rate	78.3%
Average steps per episode	23.979
Transmission efficiency	86.1%

Satellite	Ground TX Prob	TX Attempts	TX Successes (%)	Pass Successes (%)
Satellite 1	0.1	0	0 (0.0%)	10779 / 11355 (94.92%)
Satellite 2	0.5	0	0 (0.0%)	10704 / 11268 (94.99%)
Satellite 3	0.99	2607	2582 (99.04%)	0 / 0 (0.0%)

As expected, the network moves all packets to the satellite with the highest ground transmission probability. High reward

standard deviation can be attributed to the shaping of our reward function. The episode completion rate is no longer 100%, which is a result of the imperfect observations. This simulation had a correct observation probability of 80%, which is roughly the episode completion rate. This indicates that an incorrect observation would place the network into an unrecoverable state.

C. Effects of imperfect observations and low transmission probabilities

The effectiveness of the algorithm was directly related to the accuracy of observations and the probability of successfully transmitting to ground. It was found that networks with more packets than satellites performed quite poorly when no satellite had a ground transmission probability greater than 0.7. Unsurprisingly, observation accuracy had a similar impact, where networks with imperfect observations were much more likely to get "lost" given an imperfect observation. For example, the same 2-satellite network in (A) with 3 packets and accurate observation probability 0.8 had an episode completion rate of just 8%, which was improved to 97% with perfect observations.

VI. Conclusions and Future Work

This algorithm was effectively able to compute a set of near-optimal controllers for transmitting data throughout a satellite network. We were able to handle a wide variety of transmission probabilities, varied numbers of packets, limits on data capacity at each satellite, and imperfect observations. This implementation is not without its limitations – for problems with more than 3 satellites its multi-day run time makes it quite impractical for most use cases.

Future work includes includes reducing run time required for the algorithm through the addition of pruning of the controllers. This implementation also struggles greatly with imperfect observations. In future work an information gathering action could be added to enable to network to"unstick" itself when needed.

We emphasize that although the results appear to be less than stellar, the majority of the work was in formulating this problem correctly and understanding how to implement multiple active agents.

VII. Contributions and Release

Aidan wrote the majority of the report, was instrumental in defining the problem, and coded up our early MDP solution to the problem. Xavier wrote the majority of the final code, as well as the solution approach and results section of the report. The authors grant permission for this report to be posted publicly.

A. AI Acknowledgment

We want to acknowledge that the software development for this project was significantly boosted by Large Language Models, especially Claude. While the problem formulation, solution approach, and analysis were our design, most of the lower level implementation the algorithms and solvers was aided by these LLMs. No algorithms were off-the-shelf, which is why the LLMs were very helpful.

B. Code

Code can be found in the public GitHub repository, https://github.com/xavier2933/DMU-final.git.

References

[1] Bernstein, Daniel S., Amato, Christopher, Hansen, Eric A., Zilberstein, Shlomo, Policy iteration for decentralized control of Markov decision processes, *Journal of Artificial Intelligence Research*, 2009, 34, 89–132.

[2] Chen, Hao, Zhai, Baorong, Wu, Jiangjiang, Du, Chun, Li, Jun, A Satellite Observation Data Transmission Scheduling Algorithm Oriented to Data Topics, International Journal of Aerospace Engineering, 2020, 2180674, 16 pages, 2020. https://doi.org/10.1155/2020/2180674

[3] Jiawei Zhang, Lining Xing, Guansheng Peng, Feng Yao, Cheng Chen, A large-scale multiobjective satellite data transmission scheduling algorithm based on SVM+NSGA-II, Swarm and Evolutionary Computation, Volume 50,2019,100560,ISSN 2210-6502, https://doi.org/10.1016/j.swevo.2019.100560.

[4] Jiawei Zhang, Lining Xing, An improved genetic algorithm for the integrated satellite imaging and data transmission scheduling problem, Computers & Operations Research, Volume 139, 2022, 105626, ISSN 0305-0548, https://doi.org/10.1016/j.cor.2021.105626

[5] Jing Chen, Xiaoqiang Di, Rui Xu, Hui Qi, Ligang Cong, Kehan Zhang, Ziyang Xing, Xiongwen He, Wenping Lei, Shiwei Zhang, A remote sensing data transmission strategy based on the combination of satellite-ground link and GEO relay under dynamic topology, Future Generation Computer Systems, Volume 145, 2023, Pages 337-353, ISSN 0167-739X, https://doi.org/10.1016/j.future.2023.02.016.

[6] X. Chen et al., "Data-Driven Collaborative Scheduling Method for Multi-Satellite Data-Transmission," in Tsinghua Science and Technology, vol. 29, no. 5, pp. 1463-1480, October 2024, doi: 10.26599/TST.2023.9010131.

[7] Oliehoek, Frans A., and Christopher Amato. A concise introduction to decentralized POMDPs. Vol. 1. Cham, Switzerland: Springer International Publishing, 2016.