(Preprint) AAS XX-XXX

SATELLITE NETWORK ROUTING ALGORITHM DESIGN USING MARKOV DECISION PROCESSES

Hongseok Kim,[*] and Takashi Tanaka[†]

Satellite constellation operation concept requires appropriate communication packet routing algorithms to deal with dynamic changing network topology. In this paper, we present a satellite network routing framework utilizing a Markov Decision Process (MDP) to optimize data packet transmission across satellite constellations. The framework includes methods for designing optimal pathways for the data packets from the origin to destination. And then, we also incorporate packet collision avoidance techniques including sequential, cooperative, and congestion penalty approaches. Simulation results demonstrate the effectiveness of these methods, highlighting the importance of dynamically adjusting network connectivity and state values to ensure efficient routing.

INTRODUCTION

In recent years, satellite constellations have emerged as a promising solution to overcome the inherent limitations of single low-Earth orbit (LEO) satellites. A single satellite operating in LEO often faces constraints such as long revisit times and short communication windows, limiting its continuous visibility and data transmission capabilities. To overcome this, a concept of satellite constellation is suggested, which can provide nearly continuous coverage and seamless data relay. [1] However, the challenge lies in efficiently routing data packets between satellites, especially in dynamically changing network conditions.

In this paper, we design algorithms which determines the optimal routing path that minimizes communication costs and ensures timely delivery. Specifically, we propose a novel approach using Markov Decision Process (MDP) to develop a routing algorithm tailored for satellite constellations. Unlike traditional methods, this framework offering an optimized solution for data packet transmission in dynamically changing network conditions. We also developed strategies to manage multiple data packets traversing the satellite network, addressing potential packet collisions when multiple packets converge on a single satellite. By employing a combination of sequential, cooperative, and penalty-based approaches, we aim to ensure efficient data flow in a multi-packet routing environment.

Lastly, we introduce a customizable MATLAB repository[‡] that allows researchers to apply MDP-based routing algorithms to various satellite constellations and mission-specific scenarios.

[*] Undergraduate Student, Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, Austin, TX 78712

[†] Associate Professor, Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, Austin, TX, 78712

[‡] https://github.com/redstone98/Redstone_Project

Configuration of this Research

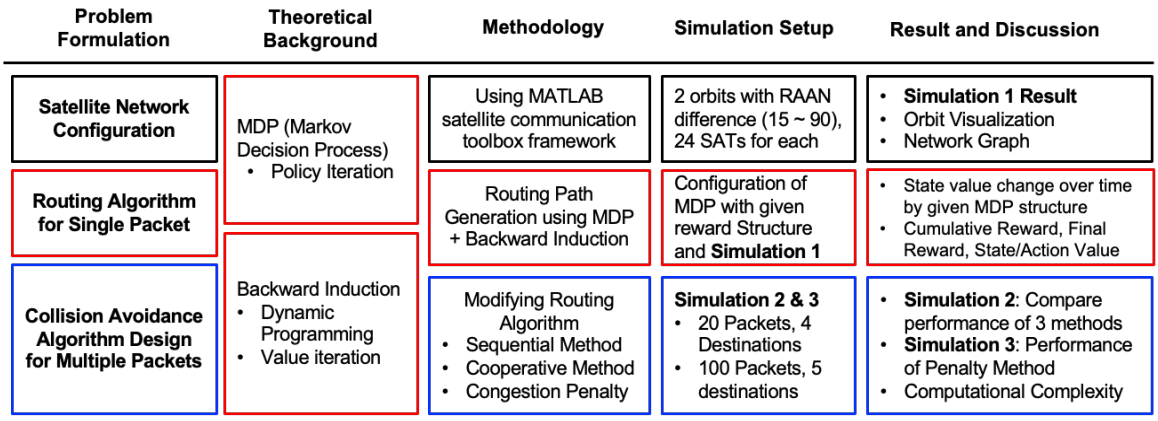


Figure 1. Configuration of this Research

Figure 1 outlines the configuration of this research conducted in this paper, divided into three key problem areas. First, Satellite Network Configuration (in black) addresses the structure and layout of the satellite constellation. Second, the Routing Algorithm for Single Packet (in red) focuses on routing paths for individual packets using a combination of theoretical approaches. Finally, the Collision Avoidance Algorithm Design for Multiple Packets (in blue) investigates techniques to handle multiple packets simultaneously without collision.

PROBLEM FORMULATION

The primary objective of this research is to develop a routing algorithm capable of efficiently transferring data packets from one ground station to another via a satellite network. The data packet, acquired from a specific ground point, must be routed through a constellation of satellites to its target destination. The key challenge is to ensure that this transmission occurs with minimal communication costs while maintaining the integrity and timeliness of the data. Reflecting this, the routing algorithm must account for the dynamic and changing nature of satellite constellations.

THEORETICAL BACKGROUND

Markov Decision Process (MDP)

Markov Decision Process (MDP) is a mathematical framework used for decision-making in situations where outcomes are partly random and partly under the control of a decision maker. [2] An MDP is characterized by a set of states (S), actions (A), transition probabilities (T), and rewards (R). [3] At any given state, an agent can take an action, leading to a reward and transitioning to the next state based on the transition probability. The goal is to maximize the cumulative reward over time by following an optimal policy, which dictates the best action to take at each state. State value represents the expected cumulative reward from that state, while the action value, represents the expected cumulative reward of taking action in given state.

Backward Induction

Backward Induction process in the MDP framework uses the Bellman Optimality Equation to compute optimal policies and value functions by working backward from a known final state. [4] The process assumes that the state value function is known at a certain time step. The Bellman Optimality Equation is then used to calculate the state value, policy function, and action value for each previous state and action combination. The aim of backward induction is to determine the optimal action for each state by choosing the action that maximizes the action value, which helps define the best policy.

METHODOLOGY

The following methodologies are adopted to develop a routing algorithm in satellite network using theoretical background introduced in the previous part. We have divided into three steps: satellite network configuration, routing algorithm design using MDP, and packet collision avoidance algorithm.

Satellite Network Configuration

In the satellites network configuration process, the orbit parameters and constellation design are established. These parameters define the positions and movements of the satellites in orbit, ensuring proper coverage and communication links between satellites

Routing Algorithm for Single Packet

We designed routing algorithm using MDP by combining policy iteration process and backward induction to design a routing algorithm. The algorithm begins with MDP policy iteration for the final timestep. A time-invariant network is assumed, and the policy iteration algorithm is applied to calculate the converged state value function at the final timestep. After the final timestep is optimized, backward induction for previous timesteps is performed. Using the state value function from the final timestep, the system works backward through each previous timestep. At each timestep, the state and action values, along with the policy function are calculated based on the current timestep's state value function. This process repeats iteratively for each prior timestep until reaching the initial timestep.

The policy distribution generated in this process tells the best action to take in every possible state and at each point in time to ensure efficient data packet transmission. Then, data packet propagator uses this policy to determine the next action based on the current state and time, deciding where the packet should go next.

Packet Collision Avoidance Algorithm for Multiple Data Packets

In the case of multiple packets influence each other, the routing algorithm is further modified using Sequential, Cooperative, and Penalty methods to handle multi agent routing scenario. [5]

Sequential Approach:

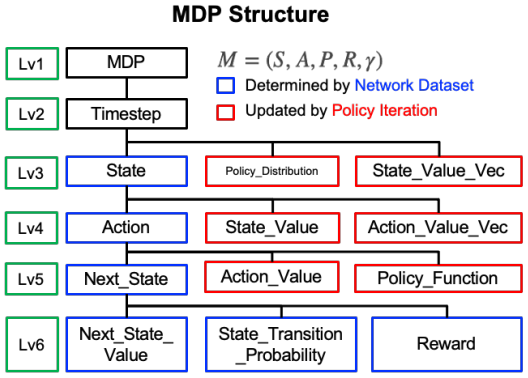
In Sequential approach, we proceed the algorithm sequentially for each packet. After packet N's path has been determined by routing algorithm, the satellite network configuration is modified. This modification involves deactivating the edges (or connections) leading to the footprint of the packet N at each timestep. This ensures that no subsequent packet will follow the same path, preventing collisions. The modified network configuration, which accounts for packet N's path, is used to compute a new path for packet N+1 using the MDP. Packet N+1 is then delivered, and the

configuration is modified again to deactivate edges for its path. This continues for all subsequent packets, ensuring that each packet avoids any overlaps or collisions with other packet.

Cooperative and Penalty Approach:

In cooperative and penalty approach, the collision avoidance is performed by adjusting policy distribution based on potential collisions. A data packet collision detection mechanism identifies any collisions by detecting data packets that are attempting to move to the same next state. If a collision is detected, the algorithm activates the collision avoidance algorithm. The Cooperative approach finds a new state-action combination that avoids collisions by re-routing one or more data packets to different paths. The Penalty deducts a penalty from the action value of the data packet whose action caused the collision. This penalty reduces the likelihood of choosing that action in future iterations, encouraging the algorithm to find better paths.[6] After this process, the updated policy distribution is generated, which provides the adjusted routing strategy for all data packets to avoid collisions and ensure efficient transmission through the network. This process continues, dynamically updating the policy based on real-time collision detection and avoidance.

SIMULATION SETUP



Transition Probability / Reward Structure

Transition Probability	Simulation 1,3	0.8
	Simulation 2	1
Reward Structure	Transition in Same Orbit	-1
	Transition in Different Orbit	-15
	Staying in Current State	-5
	Failure of Transition	-30
	Approach to Destination in Same Orbit	+100
	Approach to Destination in Different Orbit	+75

Figure 2. MDP structure organized in MATLAB and corresponding Reward Structure

For the Satellite Network Configuration, the setup uses 2 orbital planes, each with 24 satellites, and simulates different RAAN (Right Ascension of Ascending Node) configurations. We have constructed a multi-level structure for the MDP that reflects the concept described in theoretical background. Figure 2 describes the structure of MDP we designed, and corresponding transition probability and reward structure used in simulations.

We propose 3 simulations in this research. In simulation 1, we developed a platform that allows us to visualize the configuration of constellation and the corresponding network. In this simulation, we also present how different orbit configurations impact the communication network topologies and packet routing behaviors. Simulation 2 compares the performance of three routing methods, assessing efficiency and robustness under different traffic conditions. Simulation 3 focuses on the Performance of the Penalty Method for collision avoidance and further examines Computational Complexity to assess the algorithm's efficiency. Each simulation provides insights into how state values change over time, including cumulative rewards, final rewards, and state/action values.

SIMULATION RESULT AND DISCUSSION

Simulation 1 Result: Effect of Different Orbit Configuration

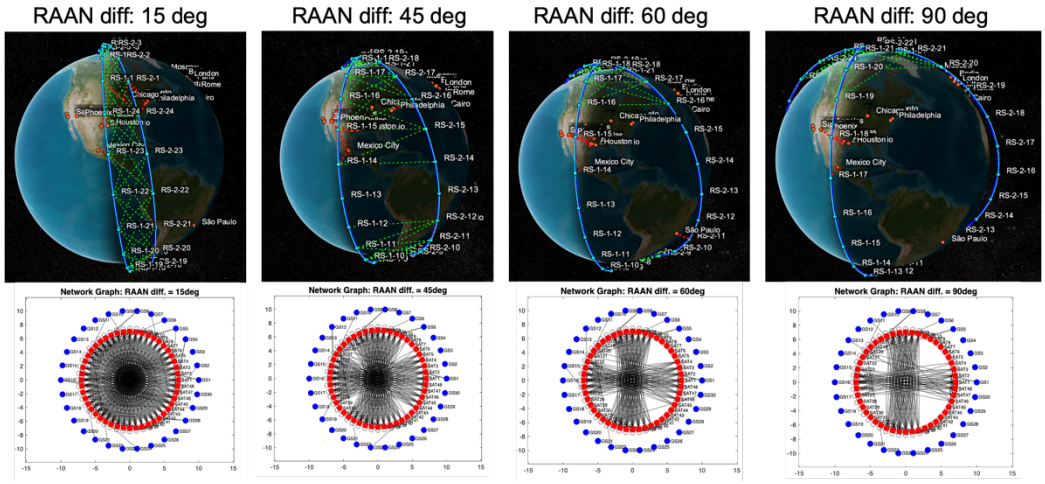


Figure 3. Simulation 1 Result: Orbit and Network Visualization

Figure 3 demonstrates how varying RAAN differences impact the orbital trajectories and network connectivity of a satellite constellation. In the 15° RAAN difference, most inter-satellite connections (represented by green lines) are maintained throughout the orbit. As the RAAN difference increases, there are noticeable inter-satellite connections over certain latitude, but no connections are observed at lower altitudes. We can observe this by diminishing density of network by increasing RAAN difference.

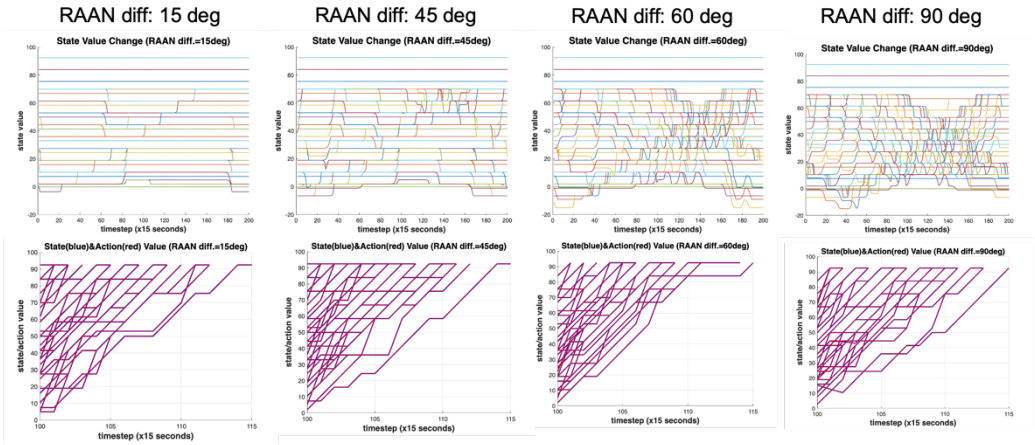


Figure 4. Simulation 1 Result: State Value Change for each State and routing result for 48 packets

Figure 4 illustrates how state value changes and packet routing behaviors are influenced by different RAAN differences in a satellite constellation. The top row of plots shows the state value changes over time for different RAAN differences. For a RAAN difference of 15°, state value changes occur only once throughout the timesteps. As the RAAN difference increases, the state value changes become more frequent and rapid. The bottom row of graphs illustrates the state value change for each of the 48 agents, all starting at the same time. Despite changes in the net-

work, the routing sequence for the data packets remains stable, regardless of the RAAN difference.

Simulation 2 Result: Compare the Sequential, Cooperative and Congestion Method

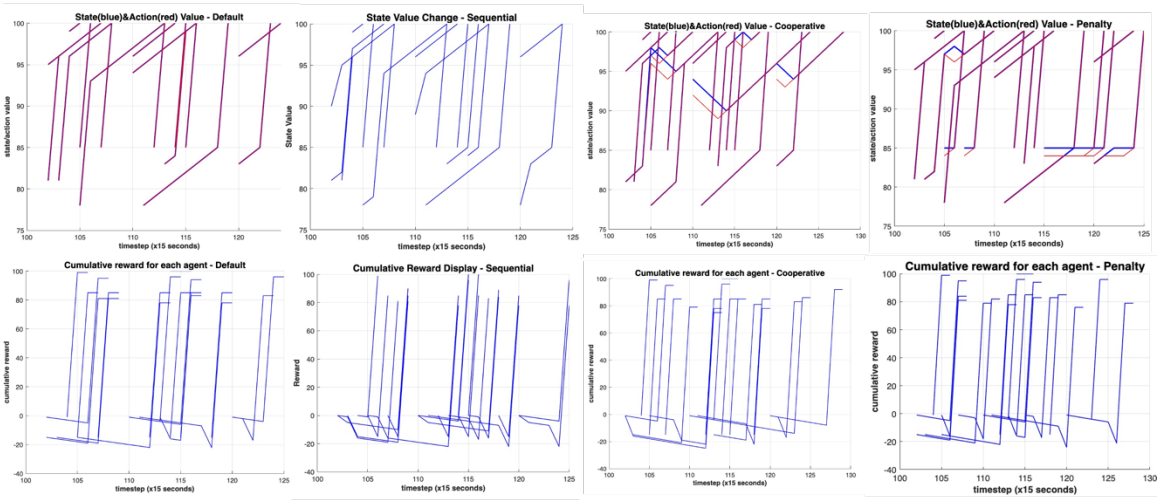


Figure 5. Simulation 2 Result: State/Action value and Cumulative Reward [Default / Sequential / Cooperative / Penalty], 20 packets and 4 destinations

Figure 5 describes the state/action value and cumulative reward across different simulation strategies: default, sequential, cooperative, and penalty. In both the default and sequential results, the state and action values are identical at each timestep, indicating that each agent takes the optimal action at every decision point. For the cooperative and penalty approaches, instances occur where the state and action values are not identical. This suggests that the agents adjust their decisions in response to external factors.

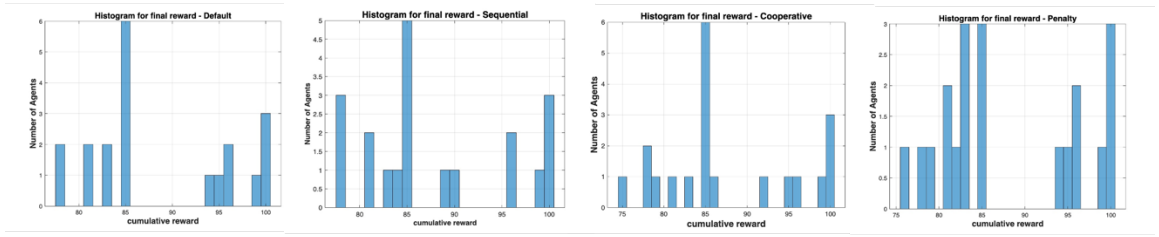


Figure 6. Simulation 2 Result: Histogram for final rewards [Default / Sequential / Cooperative / Penalty], 20 packets and 4 destinations

Figure 6 illustrate the distribution of the final cumulative rewards for each agent across the different methods: default, sequential, cooperative, and penalty. The sequential method shows a more extreme reward distribution, indicating lower fairness among agents, with some agents receiving significantly lower rewards. In contrast, the cooperative method displays a more balanced distribution, suggesting greater fairness, as rewards are more evenly spread among agents. The penalty method shows a distribution similar to the default but with a wider spread of rewards.

Simulation 3 Result: Performance of Penalty Method in Large Scale Routing

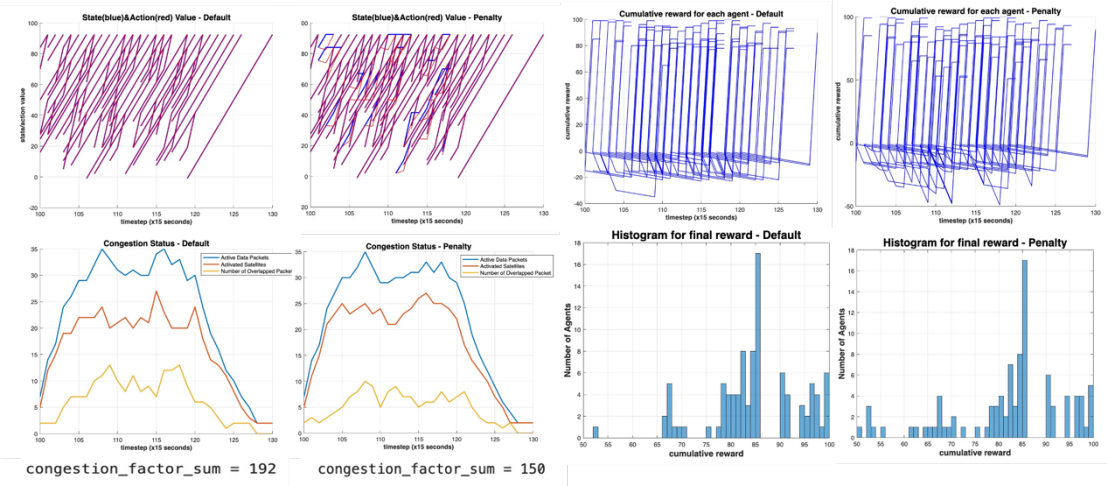


Figure 7. Simulation 3 Result: Histogram for final reward, 100 packets and 5 destinations

The results in Figure 7 demonstrate that large-scale data packet routing is feasible under the penalty method. Although this method does not completely eliminate packet collisions, it effectively reduces them without making significant changes to the action values. The state-action value plot shows multiple instances where the state and action values are not identical, indicating decision changes during the simulation. In the congestion status diagrams, the number of active satellites increases at certain timesteps, which contributes to a reduction in the number of overlapped packets. However, as a trade-off, the histogram for the final reward reveals that there are several packets with a final reward between 50 and 60 in the penalty method, whereas in the default method, only one packet falls into this reward range. This suggests that while the penalty method helps reduce congestion, it may also result in a more uneven reward distribution.

REFERENCES

[1] S. Park, G. S. Kim, S. Jung and J. Kim, "Markov Decision Policies for Distributed Angular Routing in LEO Mobile Satellite Constellation Networks," in IEEE Internet of Things Journal, doi: 10.1109/JIOT.2024.3450851.

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

[3] Richard S. Sutton and Andrew G. Barto. 2018. Reinforcement Learning: An Introduction. A Bradford Book, Cambridge, MA, USA.

[4] Kochenderfer, Mykel J., Tim A. Wheeler, and Kyle H. Wray. *Algorithms for decision making*. MIT press, 2022.

[5] Xuan, Ping, Victor Lesser, and Shlomo Zilberstein. "Communication decisions in multi-agent cooperation: Model and experiments." *Proceedings of the fifth international conference on Autonomous agents*. 2001.

[6] S. H. Q. Li, D. Calderone and B. Açıkmeşe, "Congestion-Aware Path Coordination Game With Markov Decision Process Dynamics," in *IEEE Control Systems Letters*, vol. 7, pp. 431-436, 2023, doi: 10.1109/LCSYS.2022.3189323.