# IEE 622 Project Part - 2

Divya Ramachandran

March 2025

## Introduction

The original problem in the paper proposes mathematical integer programming(IP) formulation designed to maximize the number of tasks to be executed by a satellite, with the available power at any moment along the course of an orbit as a constraint. The formulation considers task priority, minimum and maximum number of task activation, minimum and maximum execution time, and period of a given task and execution window. The power input vector was calculated on the basis of an analytical model and solar cell efficiency.

## Mathematical Formulation

| **Variables and sets** | |
|---|---|
| **Notation** | **Definition** |
| Sets | |
| S | Set of satellite subsystems |
| J | Set of tasks to be scheduled |
| T | Set of units of time |
| R | Set of available power |
| Indexes | |
| s | Subsystem index, s $\in$ S |
| j | Task or job index, j $\in$ J |
| t | Time index, t $\in$ T |
| Variables | |
| $x_{s,j,t}$ | A decision variable that takes the value of 1 for each unit of time, t $\in$ T that job j $\in$ J is executing, otherwise 0 |
| $\phi_{s,j,t}$ | An auxiliary decision variable that takes the value of 1 at each time t $\in$ T that a job j $\in$ J was initiated, otherwise 0 |

| Constants | |
|---|---|
| **Notation** | **Definition** |
| S | The number of subsystems |
| J | The number of tasks/jobs |
| T | The size of T |
| $r_t$ | The amount of power available at time t |
| $q_{s,j}$ | The amount of power consumed by executing a job j in subsystem s |
| $u_{s,j}$ | The priority of job j on subsystem s |
| $t_{s,j}^{min}$ | The minimum CPU time of job j on subsystem s |
| $t_{s,j}^{max}$ | The maximum CPU time of job j on subsystem s |
| $y_{s,j}^{min}$ | The minimum number of startups of job j on subsystem s |
| $y_{s,j}^{max}$ | The maximum number of startups of job j on subsystem s |
| $p_{s,j}^{min}$ | The minimum period of job j on subsystem s |
| $p_{s,j}^{max}$ | The maximum period of job j on subsystem s |
| $w_{s,j}^{min}$ | The start moment for the execution window of job j on subsystem s |
| $w_{s,j}^{max}$ | The finish moment for the execution window of job j on subsystem s |
| $\phi_{s,j,t}$ | An auxiliary decision variable that takes the value of 1 at each time t ∈ T that a job j ∈ J was initiated, otherwise 0 |

## Objective Function and Constraints

The objective function maximizes task scheduling by prioritizing higher-priority tasks. The energy constraint ensures the total power consumption does not exceed available energy. Constraints (2) to (6) track task initiation, ensuring the variable is correctly set when tasks start and stop. Constraints (7) and (8) limit the number of task startups within specified minimum and maximum limits. Constraints (9) to (11) ensure tasks execute within their required time windows. Constraints (12) and (13) prevent multiple task starts within the minimum period and ensure execution at least once within the maximum period. Constraints (14) and (15) control task start and stop times. Finally, constraints (16) and (17) define the decision variables $x$ and $\phi$ as binary.

$$\max_{x_{s,j,t}} \sum_{s=1}^{S} \sum_{j=1}^{J} \sum_{t=1}^{T} u_{s,j} x_{s,j,t}$$

subject to

$$\sum_{s=1}^{S} \sum_{j=1}^{J} q_{s,j} x_{s,j,t} \leq r_t, \forall t \tag{1}$$

$$\phi_{s,j,t} \geq x_{s,j,t}, \forall j, \forall t = 1, \forall s \tag{2}$$

$$\phi_{s,j,t} \geq x_{s,j,t} - x_{s,j,(t-1)}, \forall j, \forall t \geq 1, \forall s \tag{3}$$

$$\phi_{s,j,t} \leq x_{s,j,t}, \forall j, \forall t, \forall s \tag{4}$$

$$\phi_{s,j,t} \leq 2 - x_{s,j,t}, \forall j, \forall t = 1, \forall s \tag{5}$$

$$\phi_{s,j,t} \leq 2 - x_{s,j,t} - x_{s,j,(t-1)}, \forall j, \forall t \geq 1, \forall s \tag{6}$$

$$\sum_{t=1}^{T} \phi_{s,j,t} \geq y_{s,j}^{\min}, \forall j, \forall s \tag{7}$$

$$\sum_{t=1}^{T} \phi_{s,j,t} \leq y_{s,j}^{\max}, \forall j, \forall s \tag{8}$$

$$\sum_{l=t}^{t+t_{s,j}^{\min}-1} x_{s,j,l} \geq t_{s,j}^{\min} \phi_{s,j,t}, \forall t \in \{1 \ldots, T - t_{s,j}^{\min} + 1\}, \forall j, \forall s \tag{9}$$

$$\sum_{l=t}^{t+t_{s,j}^{\max}-1} x_{s,j,l} \leq t_{s,j}^{\max} \phi_{s,j,t}, \forall t \in \{1 \ldots, T - t_{s,j}^{\max} + 1\}, \forall j, \forall s \tag{10}$$

$$\sum_{l=t}^{T} x_{s,j,l} \geq (T - l + 1) \phi_{s,j,t}, \forall t \in \{T - t_{s,j}^{\min} + 2, \ldots, T\}, \forall j, \forall s \tag{11}$$

$$\sum_{l=t}^{t+p_{s,j}^{min}-1} \leq 1, \forall t \in \{1, \ldots, T - p_{s,j}^{\min+1}, \forall j, \forall s \tag{12}$$

$$\sum_{l=t}^{t+p_{s,j}^{max}-1} \leq 1, \forall t \in \{1, \ldots, T - p_{s,j}^{\max+1}, \forall j, \forall s \tag{13}$$

$$x_{s,j,t} = 0, t = 1, \ldots w_{s,j}^{\min}, \forall j, \forall s \tag{14}$$

$$x_{s,j,t} = 0, t = w_{s,j}^{\max}, \ldots, T, \forall j, \forall s \tag{15}$$

$$x_{s,j,t} \in \{0, 1\}, \forall j, \forall t, \forall s \tag{16}$$

$$\phi_{s,j,t} \in \{0, 1\}, \forall j, \forall t, \forall s \tag{17}$$

$$\tag{18}$$

# Description of the Exact Method

The implementation of the above formulation has been performed in Python Programming Language using the PuLP modeler using the default CBC(Coin or branch and cut) solver. The optimizer uses the branch-and-cut algorithm for solving the integer linear programming problem. It is an enhancement of the Branch-and-Bound method, with the addition of cutting planes to improve performance by tightening the relaxation of the problem.

# Data Generation

The existing instances in the paper are used to test the 3 scenarios of the model. We implement the code for 1U, 2U, and 3U nanosatellites as the set of satellite subsystems. The power input vector in the paper is simulated in accordance with the proposed analytical model. However, we have used a simplified version of the power input simulation for ease of computation. We used the skyfield library in Python to compute the sunlit and eclipse periods of the satellite.

### Two line element(TLE) Data

A two-line element set (TLE, or more rarely 2LE) or three-line element set (3LE) is a data format that encodes a list of orbital elements of an Earth-orbiting object for a given point in time. We load the TLE data from 'TLE.txt' file where we have saved the TLE information for 1U, 2U and 3U satellite.

### Power Input Model

The model helps determine how much power is available to the satellite at different times based on its orbit and the sun-lit condition, which is crucial for satellite scheduling and resource management.

1. **Input**

   - **satellite**: We use the EarthSatellite class to define satellite orbits and parameters for power calculation
   - **start**: Uses datetime object in python to define the starting time of calculation
   - **t**: The time is discretized into 1 minute intervals for scheduling 100 minutes of orbit period, t is an integer representing the number of minutes from the start time

2. **Power Calculation and Output**

   - The average power is assigned on the basis of the satellite type, we assume 80% efficiency for available power
   - The 'is.sunlight' function returns a boolean value indicating if the satellite is sunlit or not, available power is returned as the output at a particular timestep

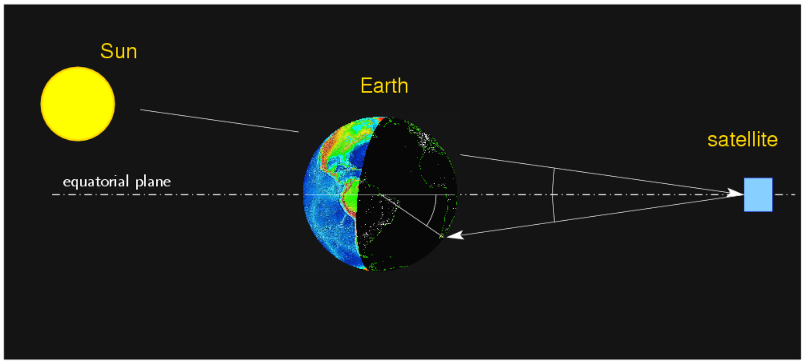


Figure 1: Eclipse period of the satellite

**Payload Scheduling Data**

| Payload | $u_j$ | $q_j$ [W] | $y_j^{min}$ | $y_j^{max}$ | $t_j^{min}$ | $t_j^{max}$ | $p_j^{min}$ | $p_j^{max}$ | $w_j^{min}$ | $w_j^{max}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2.0 | 1 | 1 | 4 | 10 | 100 | 2 | 100 | 0 |
| 1 | 2 | 1.0 | 1.3 | 3 | 3 | 3 | 100 | 2 | 100 | 0 |
| 2 | 3 | 1.0 | 0.9 | 2 | 5 | 4 | 100 | 3 | 100 | 0 |

Table 1: 1U Nanosatellite Payloads Scheduling Data

| Payload | $u_j$ | $q_j$ [W] | $y_j^{min}$ | $y_j^{max}$ | $t_j^{min}$ | $t_j^{max}$ | $p_j^{min}$ | $p_j^{max}$ | $w_j^{min}$ | $w_j^{max}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 5.0 | 1 | 3 | 15 | 3 | 40 | 2 | 100 | 0 |
| 1 | 2 | 2.0 | 1.23 | 1 | 11 | 7 | 60 | 4 | 80 | 0 |
| 2 | 3 | 1.0 | 0.8 | 2 | 5 | 3 | 80 | 2 | 100 | 0 |
| 3 | 4 | 4.0 | 1.3 | 1 | 7 | 6 | 65 | 6 | 90 | 0 |
| 4 | 5 | 1.0 | 1.5 | 1 | 10 | 4 | 70 | 2 | 90 | 0 |
| 5 | 6 | 3.0 | 1.1 | 1 | 8 | 4 | 100 | 4 | 87 | 40 |
| 6 | 7 | 1.0 | 1.1 | 2 | 7 | 6 | 100 | 2 | 60 | 0 |
| 7 | 8 | 1.0 | 0.9 | 2 | 9 | 3 | 100 | 2 | 100 | 0 |

Table 2: 2U Nanosatellite Payloads Scheduling Data

# Results

The results have been calculated considering the start time as 1 January 2025 00:00:00. The output would change significantly, even with a slight change in the date and time of the start time.

| Payload | $u_j$ | $q_j$ [W] | $y_j^{min}$ | $y_j^{max}$ | $t_j^{min}$ | $t_j^{max}$ | $p_j^{min}$ | $p_j^{max}$ | $w_j^{min}$ | $w_j^{max}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 5.0 | 3.2 | 2 | 3 | 10 | 12 | 5 | 78 | 0 |
| 1 | 2 | 2.0 | 1.23 | 1 | 4 | 2 | 50 | 4 | 50 | 0 |
| 2 | 3 | 1.0 | 0.8 | 2 | 3 | 3 | 80 | 2 | 100 | 0 |
| 3 | 4 | 4.0 | 1.3 | 1 | 7 | 6 | 65 | 6 | 90 | 0 |
| 4 | 5 | 1.0 | 1.5 | 1 | 10 | 1 | 70 | 2 | 60 | 0 |
| 5 | 6 | 3.0 | 1.1 | 1 | 8 | 2 | 100 | 4 | 87 | 40 |
| 6 | 7 | 1.0 | 1.1 | 2 | 3 | 4 | 100 | 2 | 60 | 0 |
| 7 | 8 | 1.0 | 0.9 | 3 | 9 | 3 | 100 | 2 | 40 | 0 |

Table 3: 3U Nanosatellite Payloads Scheduling Data

## Optimization Results

The objective value refers to the maximum sum of task priority times the activation variable. The optimal solution are binary integer values that indicates the different tasks and the time slots they have been scheduled.

| Satellite instance | Optimal Objective Value | Computation time (seconds) |
|---|---|---|
| 1U | 244.0 | 0.5242 |
| 2U | 752 | 0.8956 |
| 3U | 595.0 | 0.8176 |

Table 4: Optimal Objective Values and Computation time

This notation indicates that for each task, the decision variables $x_{i,j}$ are equal to 1 for time slots 5 through 65, implying that the task is scheduled during this time range

| Task | Optimal Solution |
|---|---|
| 0 | $x_{0,5-65} = 1$ |
| 1 | $x_{1,5-65} = 1$ |
| 2 | $x_{2,5-65} = 1$ |

Table 5: Optimal Solution for 1U Satellite

**Note:** The tabular results for other two instances are not given due to large space requirement, refer to the plots in the next section for optimal solution for 2U and 3U satellites.

## Optimal Solution plot

The x-axis represents the time slots, while the y-axis corresponds to the task IDs. The lines indicate the specific time slots during which each task is scheduled.

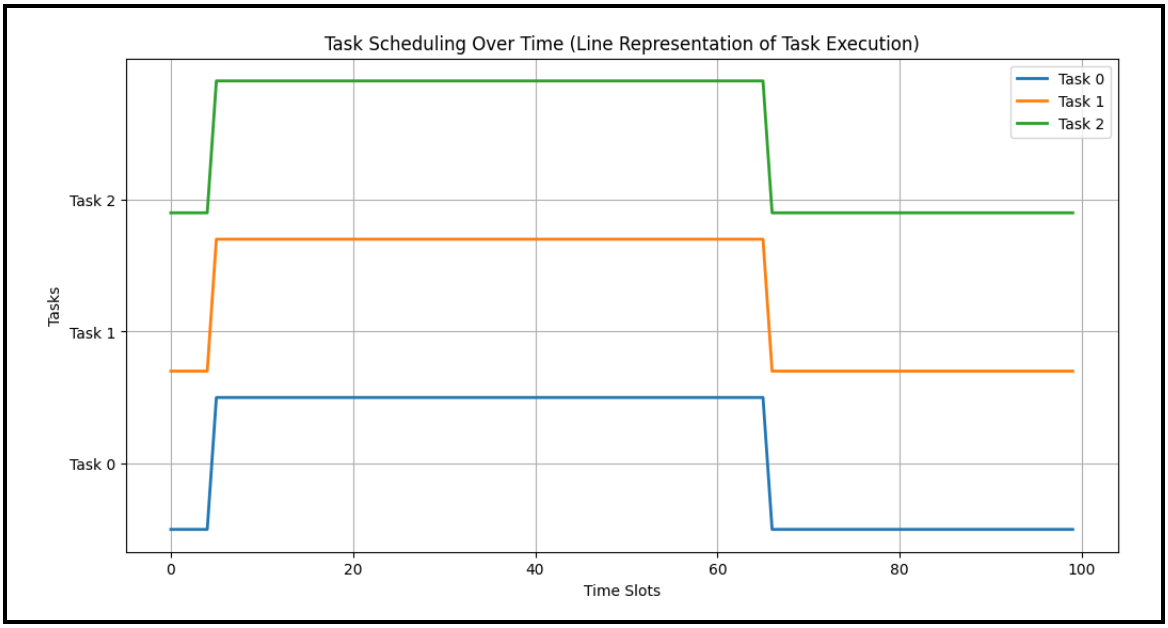


Figure 2: 1U optimimal solution

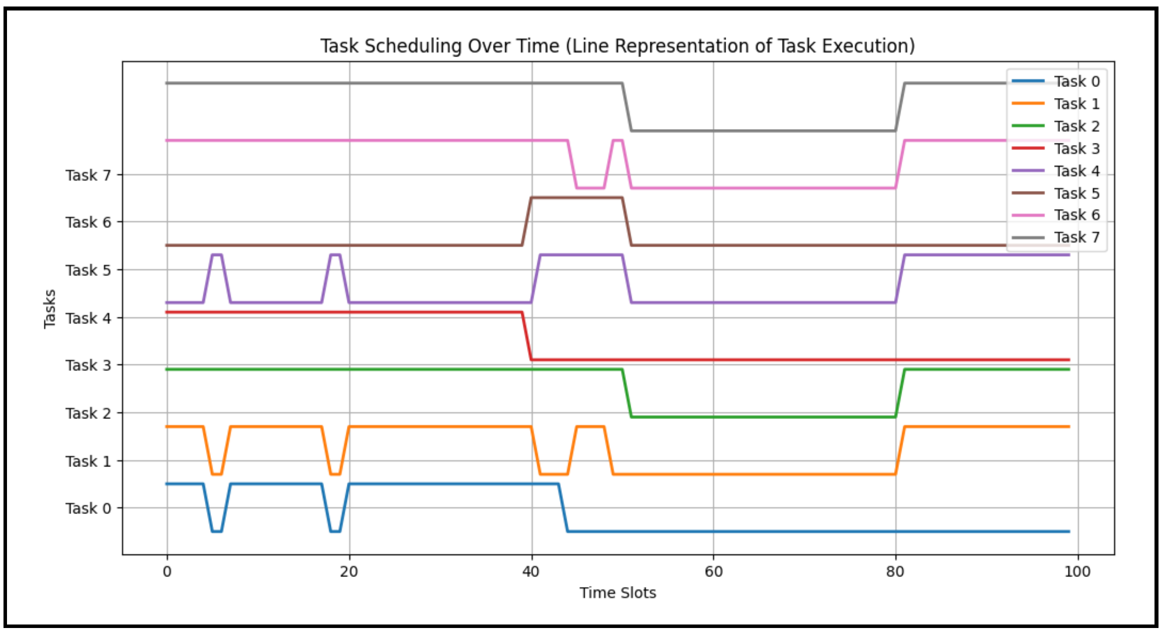


Figure 3: 2U optimimal solution

# Next Steps

In the next phase of the project, a more realistic battery model will be introduced, incorporating fuzzy constraints. To tackle the complexity of the MILP problem, Dantzig-Wolfe decomposition will be applied to exploit its unique structure, while a branch-and-price algorithm will be utilized for scheduling a large number of tasks over an extended time horizon.

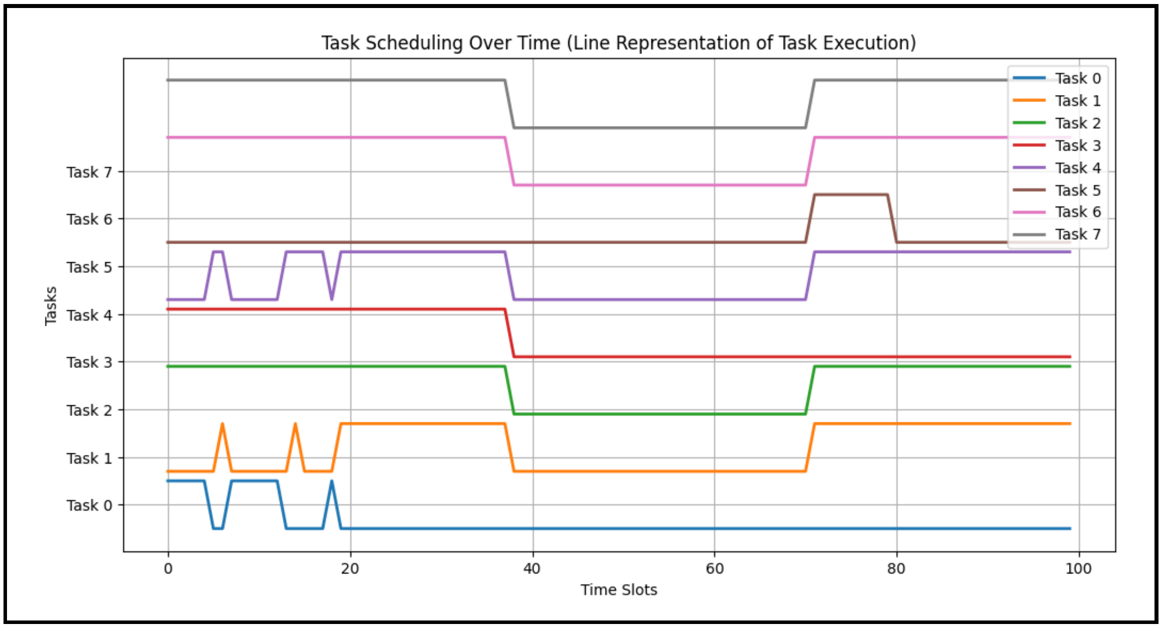


Figure 4: 3U optimimal solution