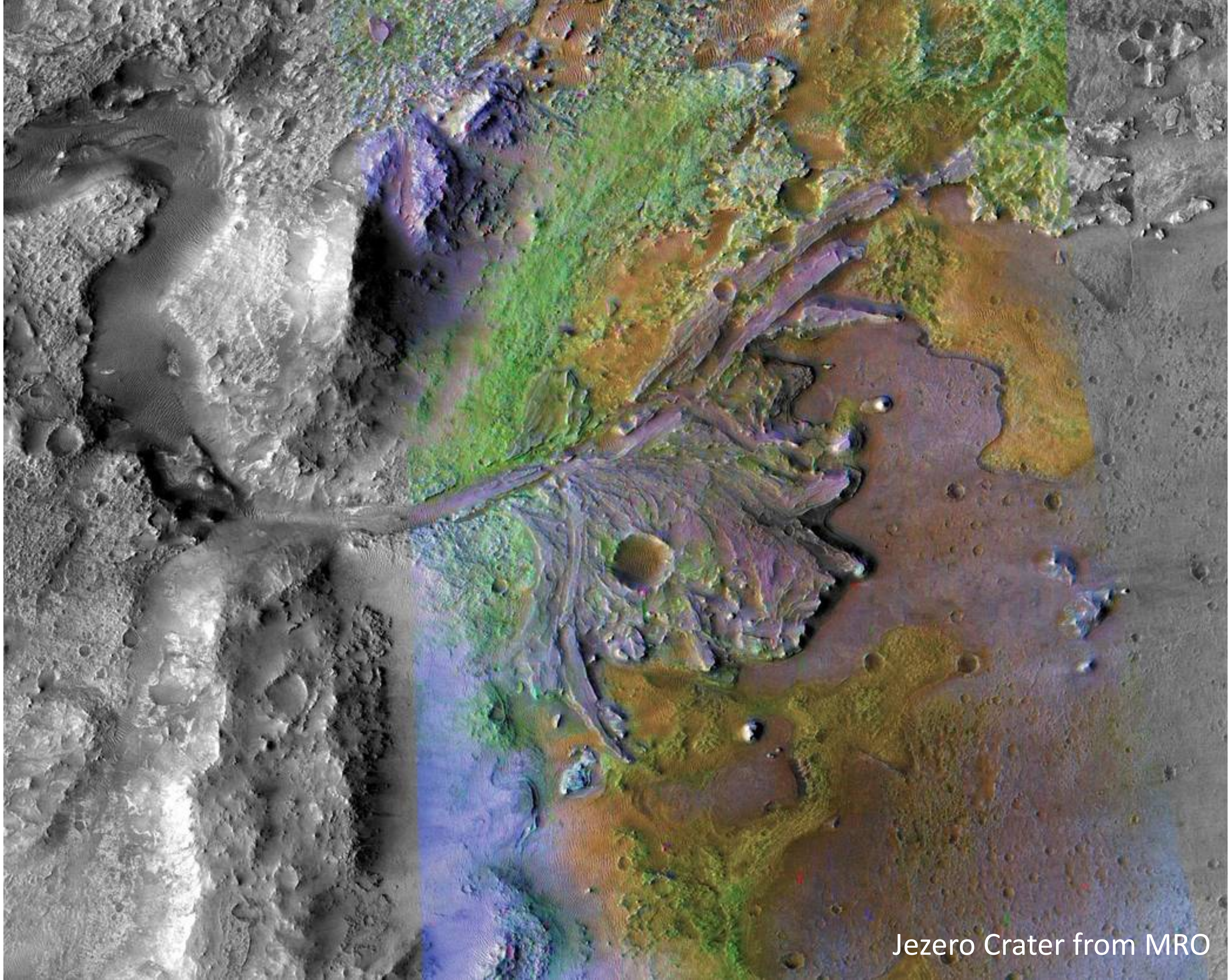
Jet Propulsion Laboratory
California Institute of Technology

Scheduling with Complex Consumptive Resources for a Planetary Rover

Wayne Chi, Steve Chien, Jagriti Agrawal
Artificial Intelligence Group
Jet Propulsion Laboratory
California Institute of Technology


Jezero Crater from MRO

Mars 2020 Onboard Scheduler

- The Mars 2020 Scheduler is a (Rabideau and Benowitz 2017)
  - single-shot, non-backtracking scheduler that
  - schedules in *priority first order* and
  - never removes or moves a main activity after it is placed during a single scheduler run.
    - Thus, each activity is scheduled such that no conflict occurs.
  - **Wakeups and Shutdowns (sleep activities) are scheduled jointly with each main activity to ensure that no constraints are violated.**
    - The algorithm is focused on local (each step) changes rather than the global (final) schedule
    - Most activities require the rover to be awake
  - Activities are not preempted
  - It does not search except for
    - Valid intervals calculations
    - Sleep and preheat scheduling

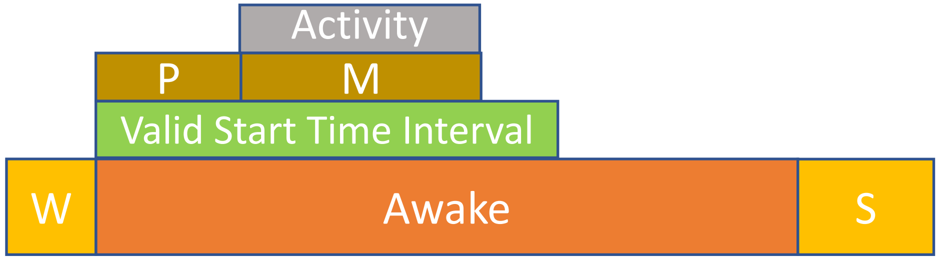
Challenges with Scheduling Wakeups and Shutdowns

- The rover gains constant energy through an MMRTG, but just being awake drains more energy than the MMRTG can supply.
  - Thus, the rover must shutdown and sleep in order to gain energy.
- Depending on an activity’s proximity to nearby wakeups and shutdowns, it may be necessary to extend an existing awake.
  - The amount of awake required by an activity varies depending on activity placement.
  - There is a minimum asleep period to prevent situations where a shutdown finishes late.
    - If something goes wrong you can miss a downlink or in the worst case end a mission
- Varying durations drastically increases difficulty in finding valid start time intervals since the algorithm must now take into account energy used as a function of activity start time.
  - Valid start time intervals are intervals in which the main activity can start and no constraints are violated.
- The most computationally expensive step of the scheduling algorithm is generating and placing wakeup and shutdowns.

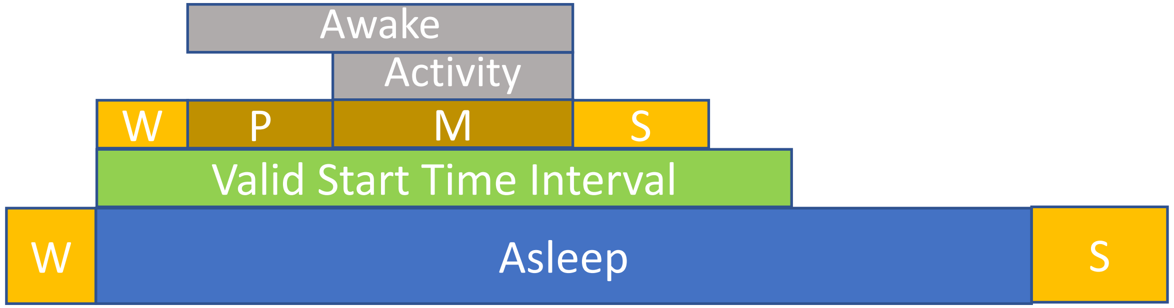
W = Wakeup | S = Shutdown

Interval Cases

1. Fully encompassed by an existing awake
   - No additional awake is needed
2. Disjoint from existing awakes
   - The duration of the awake is fixed as there is no need to extend



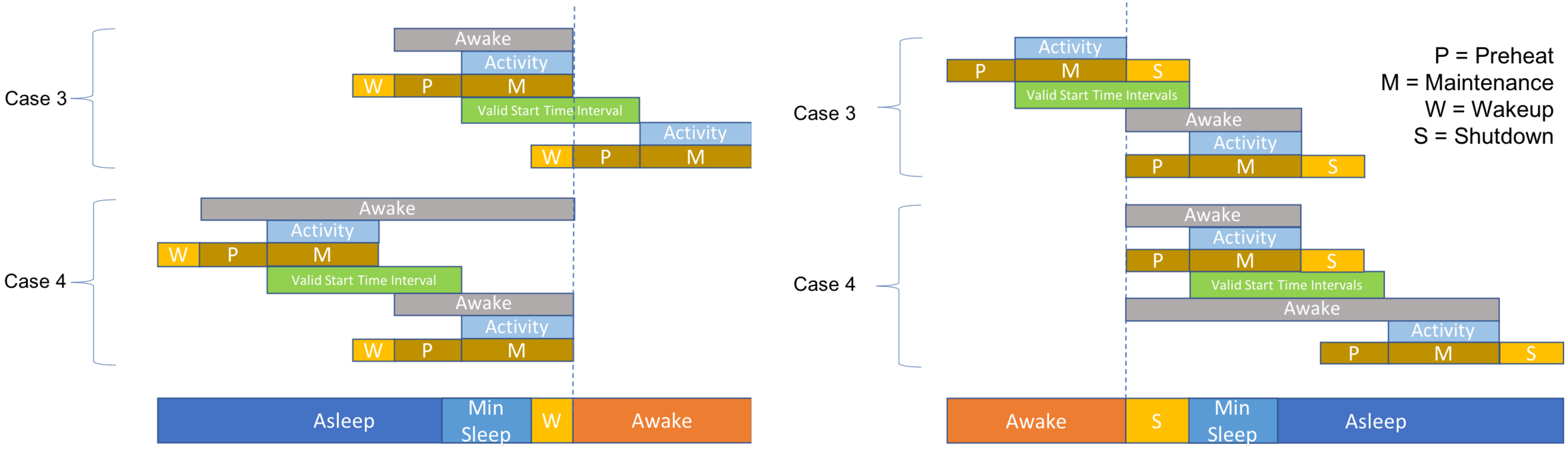
Case 1



Case 2

Interval Cases

3. Overlap with an existing awake (Straddle)
4. Overlap with a minimum asleep constraint (Stretch)



Intervals leading an awake

Intervals trailing an awake

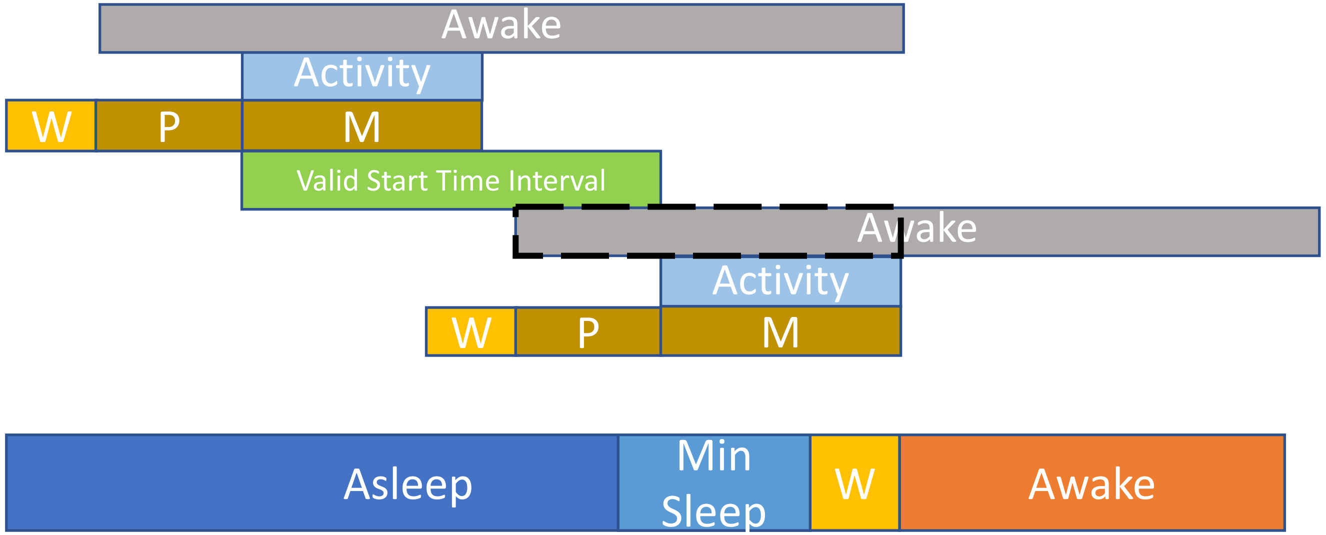
- These cases are further split into cases that *lead* or *trail* an existing awake.

Challenge

- Varying durations drastically increases difficulty in finding valid start time intervals since the algorithm must now take into account energy used as a function of activity start time.
- In cases 1 and 2, the awake duration remains constant.
  - Easy to schedule
- In cases 3 and 4, the awake duration varies depending on activity start time
  - **How do we handle this?**

Max Duration

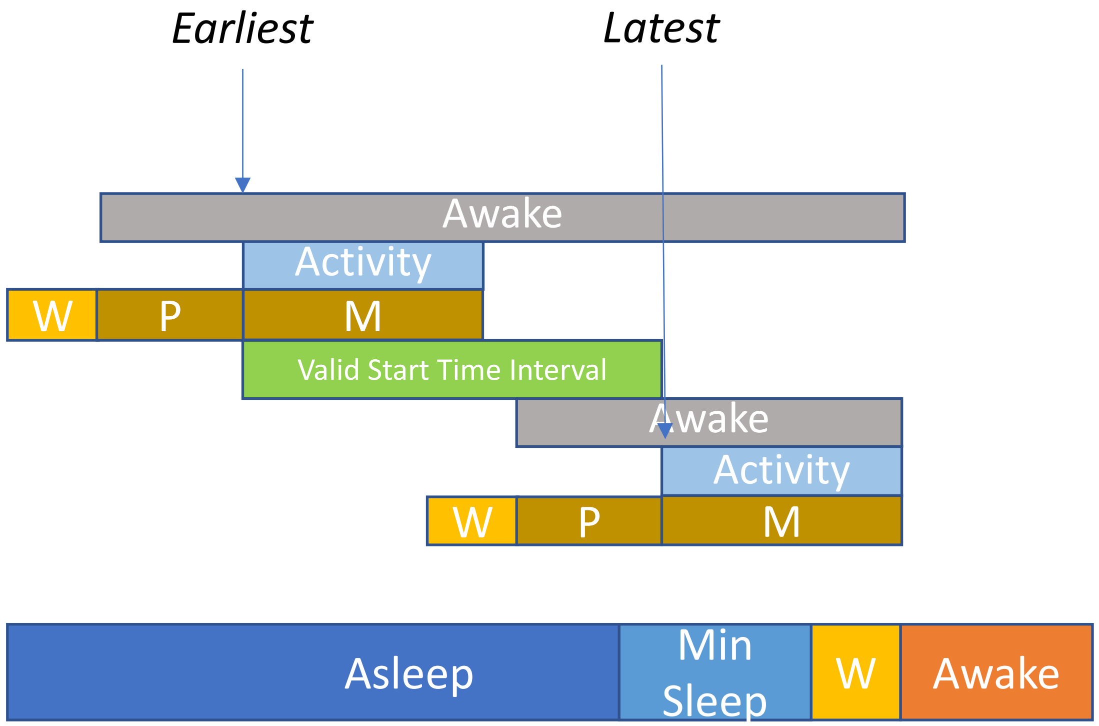
- Assume the maximum awake duration required to schedule a set of activities.
- Pros:
  - Sound
  - Simple to implement
- Cons:
  - Over-conservative – double dipping of awake periods
    - leads to incompleteness



Only the dashed box is needed, but the maximum awake period's energy is computed

Probe

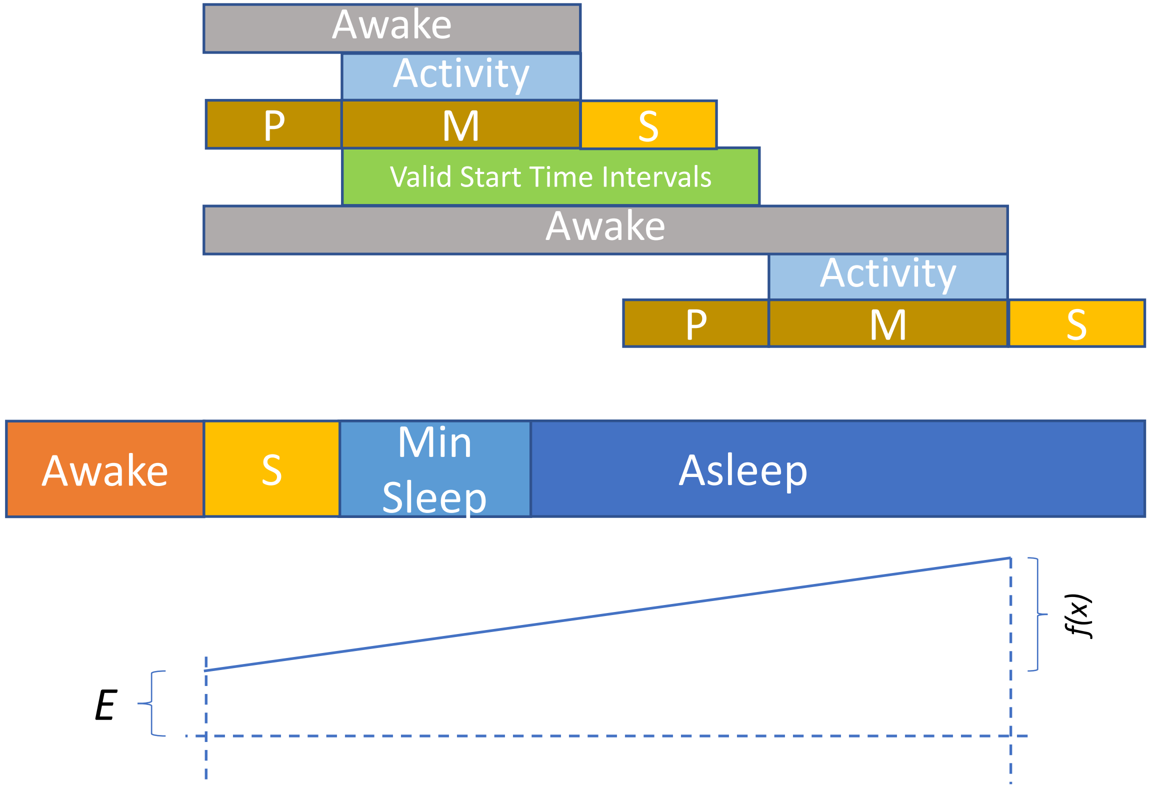
- Check if a set of activities can be scheduled at certain "probe" points.
  - Start time is fixed => awake duration is fixed
- Pros:
  - Fast algorithmically (checking a few points is faster than determining an entire valid range)
  - Simple to implement
  - Sound
- Cons:
  - Incomplete – only searches at certain points
  - Efficiency depends greatly on the heuristic for those "probe" points.
- Currently, baselined for the M2020 Onboard scheduler
  - Heuristic is to choose the point nearest to each activity's preferred time



Different probe locations can determine different fixed awake durations.

Linear

- Use the linear relationship between awake duration and energy cost to calculate the exact range of valid intervals.
  - The rover consumes $f(x)$ to stay awake.
  - All other energy costs are a constant *E.*
- Pros:
  - Sound and Complete
- Cons:
  - Difficult to implement
    - Different calculations for leading and trailing cases
    - Different calculations depending on what part of the interval you're in
  - Requires the linear relationship to be known and exist
    - A linear relationship is not always accurate

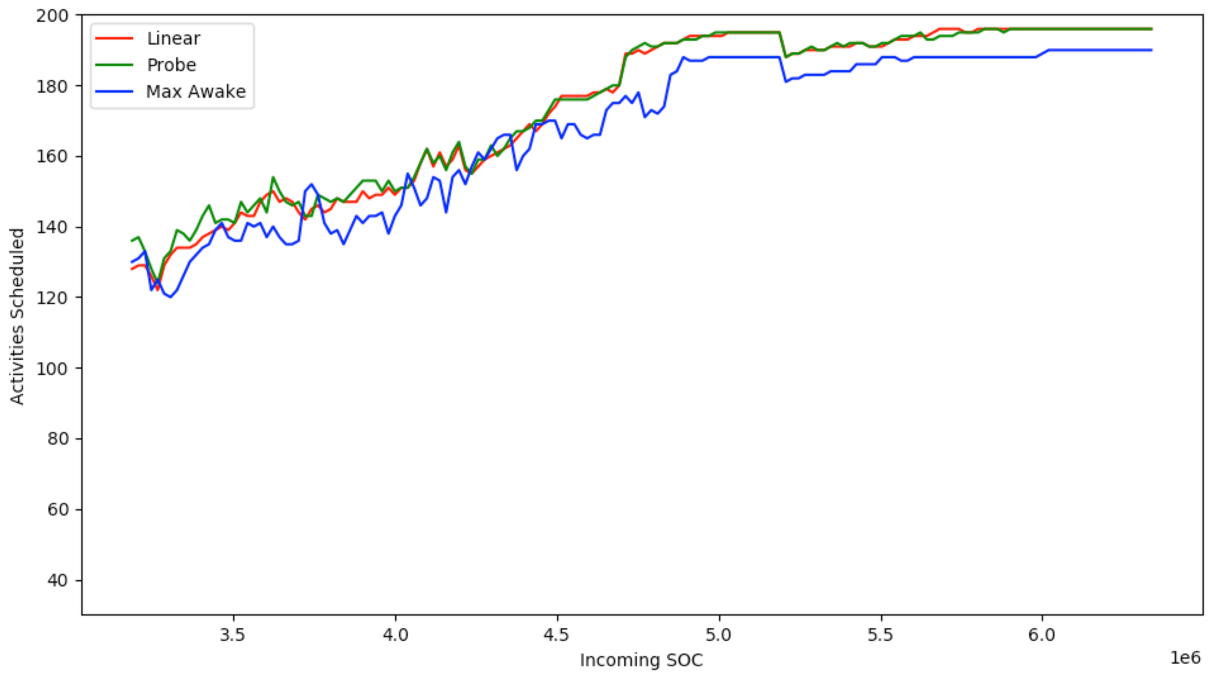


Empirical Results

- Test Input
  - 6 Plans
    - MedDrive
    - MedDrive w/ Light Constraints
    - Long Drive
    - Workspace Remote Sensing
    - Survey Remote Sensing
    - Abraded Proximity
  - Incoming SOC varies from 40% SOC to 80% SOC
    - After 80% activities rarely fail to be scheduled
  - 40% is the minimum SOC constraint

Empirical Results

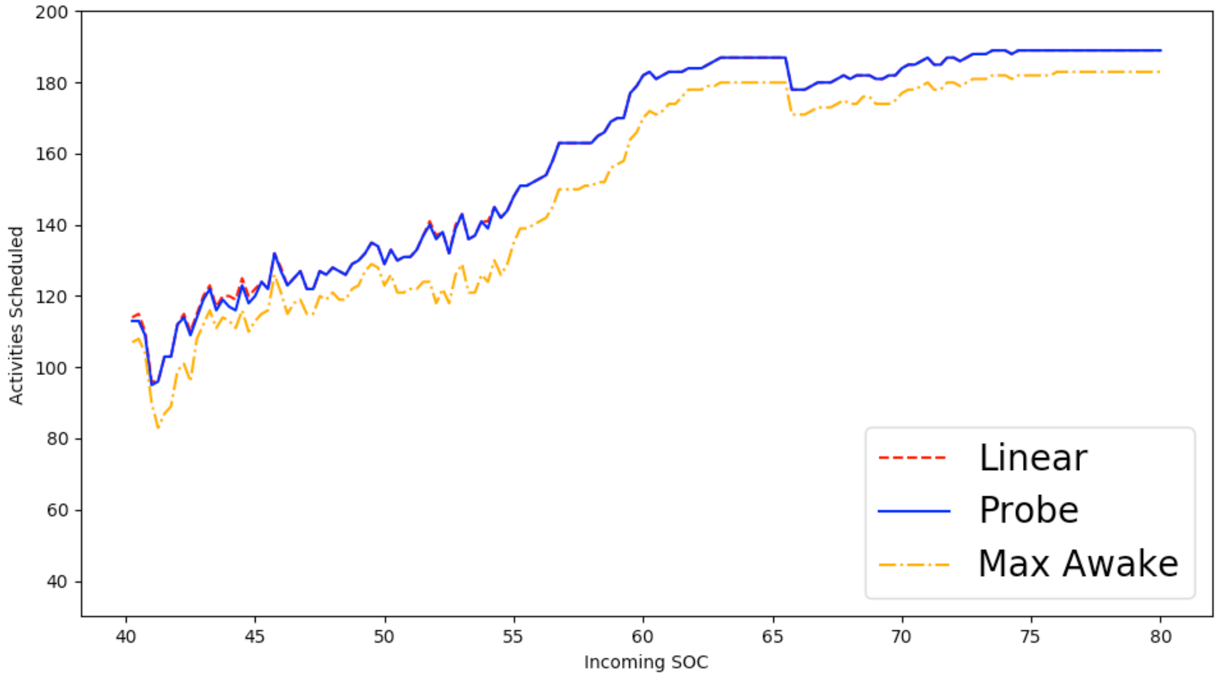
- Analyzed the number of activities scheduled as incoming SOC varies.
- Max Duration underperforms as expected.
- Probe and Linear seem to perform similarly despite the fact that Linear is complete and Probe is not.
    - Why?



Activities Scheduled with varied Incoming SOC

Reason 1 – Non Backtracking Scheduler

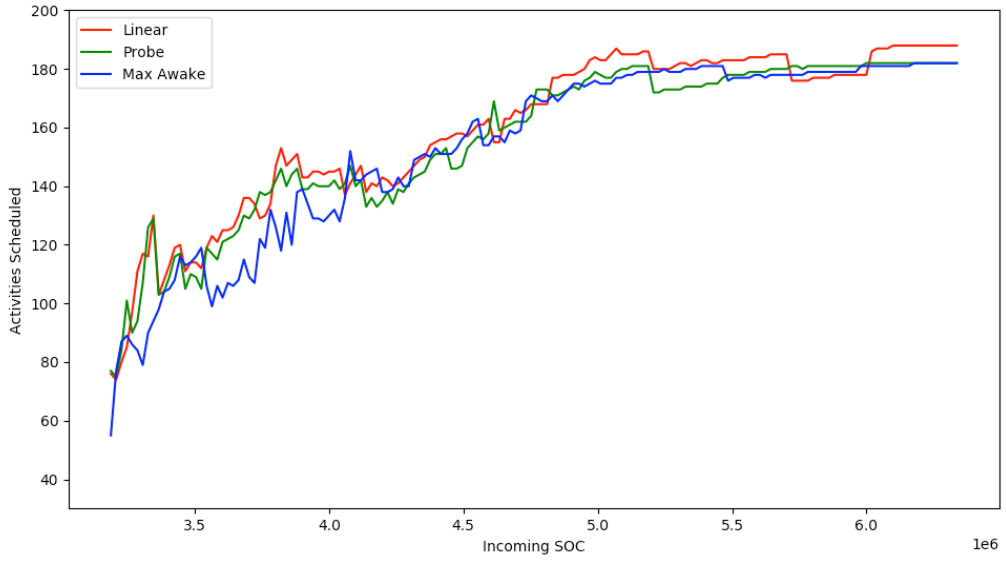
- **The scheduler is non-backtracking.**
  - **The advantages of the more complete Linear algorithm is limited to the local step (activity)**
- Partial schedule: the first *i* activities are scheduled by the same baseline algorithm, but the *i + 1* activity is scheduled with different algorithms.
  - Probe was the baseline
  - Essentially, comparing one iteration of the scheduler with one iteration.
- As expected, Max Duration performs the worst.
- Linear strictly outperforms Probe, but only slightly.



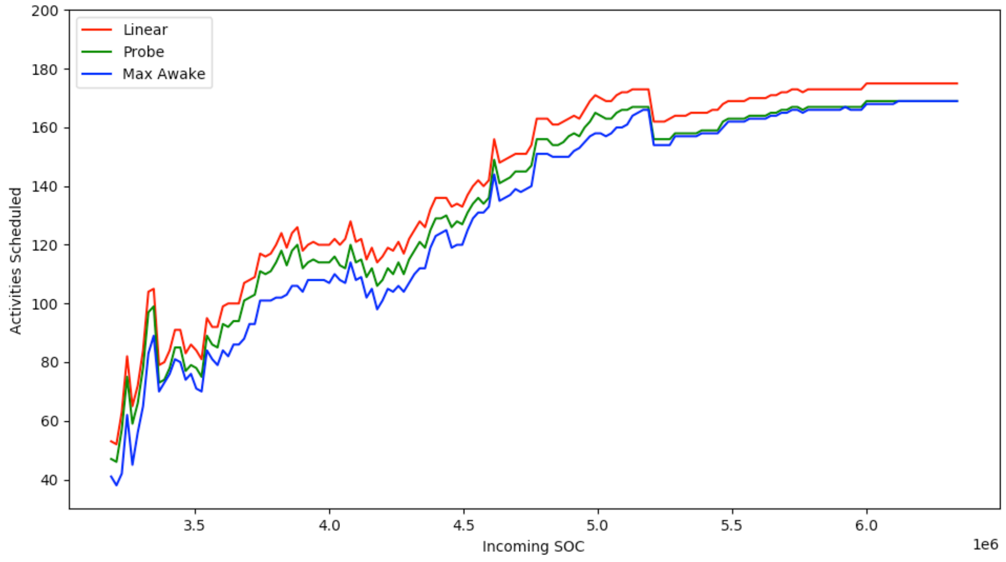
Partial Schedules

Reason 2 – Advantageous Problem Space

- **Intervals where the Linear approach provides benefit are short and sparse**
  - **Wakeups and shutdowns are only 5 and 10 minutes.**
  - **Cases 3 and 4 are rare and short**
- Increase duration of wakeups and shutdowns to 30 and 60 minutes.
- Linear algorithm starts to pull ahead
- Combined with Partial Schedules, it is clear that the Linear algorithm outperforms the Probe algorithm



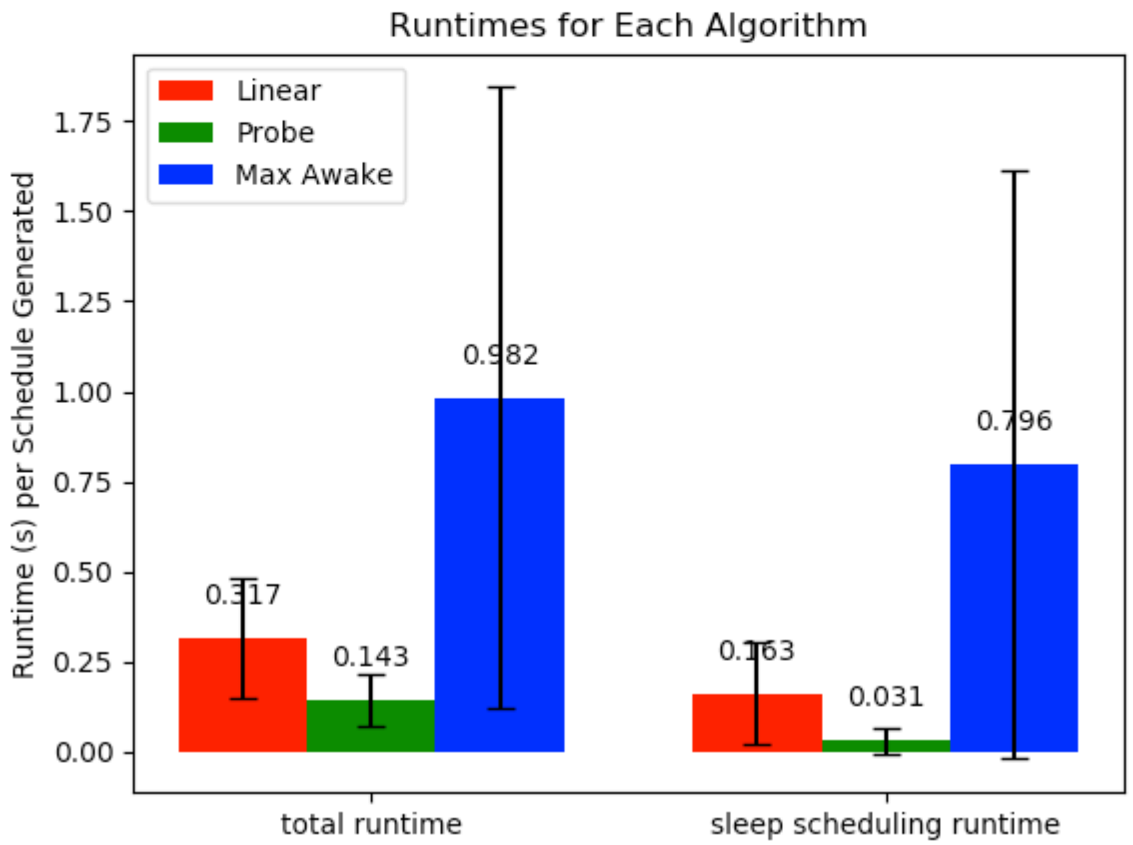
Longer Wakeups and Shutdowns



Longer Wakeups and Shutdowns w/ Partial Schedules

Runtime Analysis

- Probe runs faster than all other methods
- Max Duration performs the worst even in runtime
  - Max Duration often fails to find a place to schedule an activity, which means it spends more time searching for a valid placement while the other algorithms stop.
    - This is evidenced by the wide variance range.
- A single scheduler run can take up to 1 minute onboard. Thus, the runtime difference is substantial.



Time it takes to generate one schedule

Future Work

- Preheats and Maintenance heating were intentionally glossed over in this paper. They, however, pose a similar challenge as sleep scheduling.
  - Instruments on the rover need to meet and maintain a certain temperature threshold to operate safely.
  - Existing maintenances can be extended instead of requiring a new preheat.
  - Activities may require multiple preheats depending on thermal conditions.
- A more accurate analysis of runtimes aboard the rover. Our runtime analysis would be further substantiated if run onboard a flight-like processor.

Conclusions

- Generating and scheduling activities in the presence of consumptive regenerative resources is especially challenging when a driving factor of feasibility of placement is dependent on interactions with the existing schedule.
- Despite being a locally sound and complete algorithm, the Linear algorithm was not always able to outperform in the global problem space.
- A simple and incomplete algorithm (Max Duration) can perform sub-optimally; yet, another (Probe) can perform close to optimal.
- For M2020 use cases, Probe performs comparably to the more complete Linear Algorithm.