AVNIK DEFENSE SOLUTIONS, INC.

Modeling and Simulation for Army Sustainment: System Redundancy Analysis and Its Effect on Operational Availability

Tate Fleming,
Logistics Engineering Laboratory (LogLab)

AGENDA

1. Discrete-event simulation
2. System redundancy/M-of-N
3. Experiment (purpose, use case, setup, ideas, model creation)
4. Results and analysis
5. Summary and path forward
6. Questions

What is discrete-event simulation?

- Software engine which keeps track of a system timeline and events occurring along the timeline.
- Method to use computing power to handle complex situations – situations where using equations alone break down and can no longer adequately assess.

History

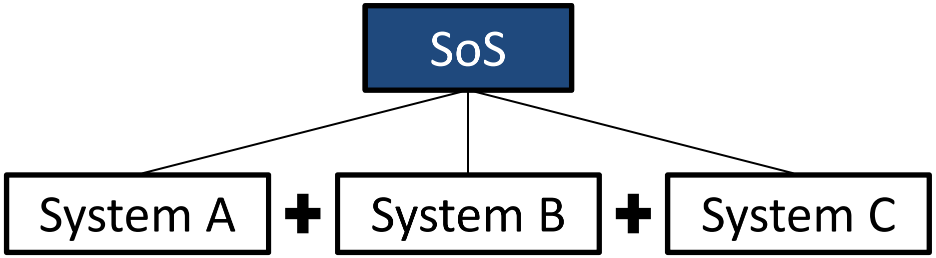
- Discrete-event simulation was invented in 1946 by physicists trying to understand the behavior of neutrons.
- By the 1970's, discrete-event simulation was widely used by tech giants and government agencies such as IBM and the Naval Air Missile Test Center.
- Discrete-event simulation is not new!

Today

- What's new is simulating Army system operations to analyze, learn, and pinpoint the variables which drive sustainment – both availability and affordability.
- Additionally, new techniques include creating a feedback loop showing how system design decisions made today will impact sustainment 10-20-50 years down the road when these new systems are operating and being supported.

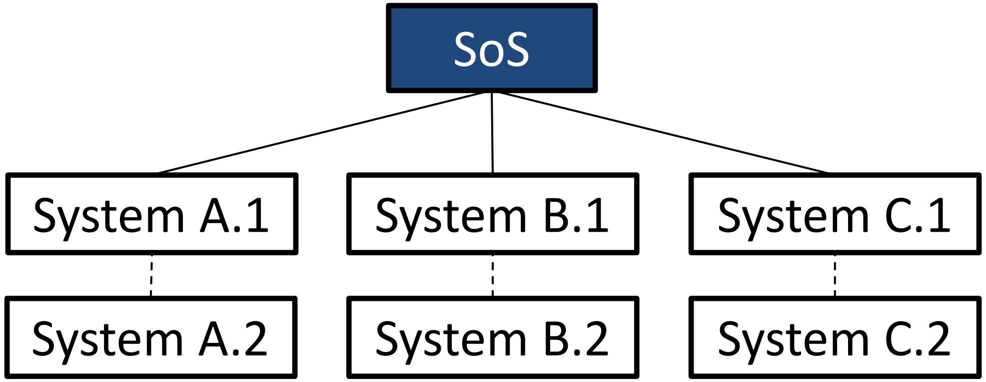
Discrete-event simulation is a trusted method to analyze extremely complex problems.

Simple System of Systems (SoS)



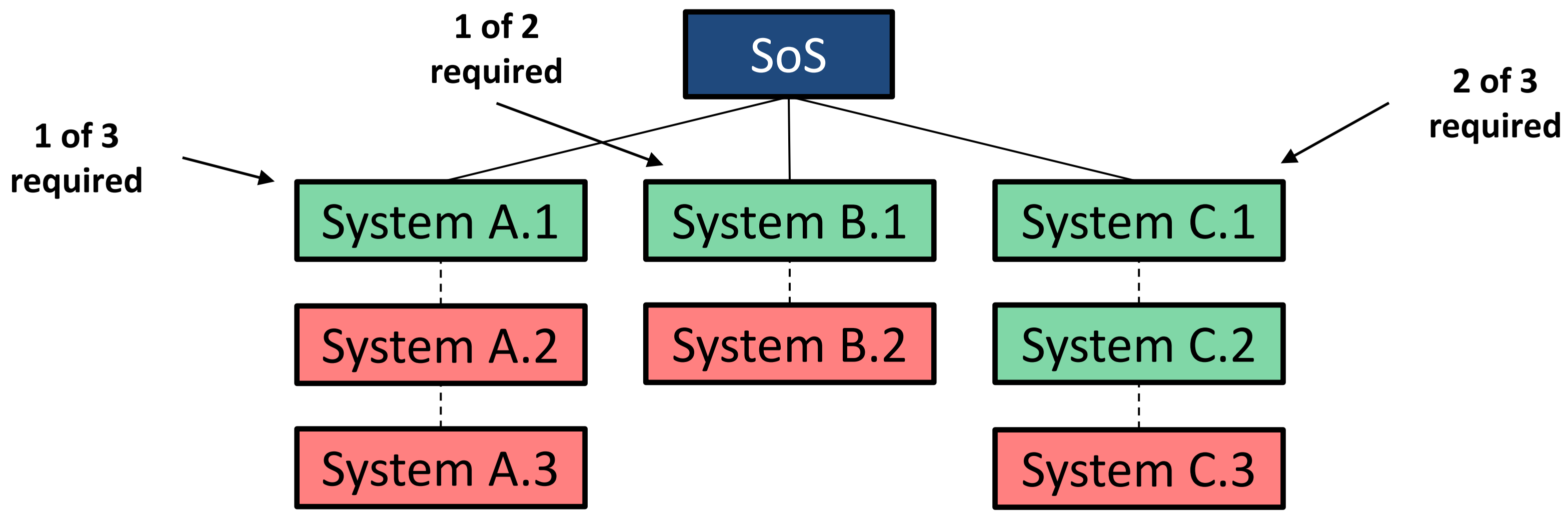
- The failure rate of the SoS is the sum of its system's failure rates.
- A technique to improve SoS reliability is to introduce system redundancy.

SoS with Redundancy



SYSTEM REDUNDANCY: M-OUT-OF-N SYSTEMS

SoS with M of N systems: (M) number of systems needed out of (N) total systems

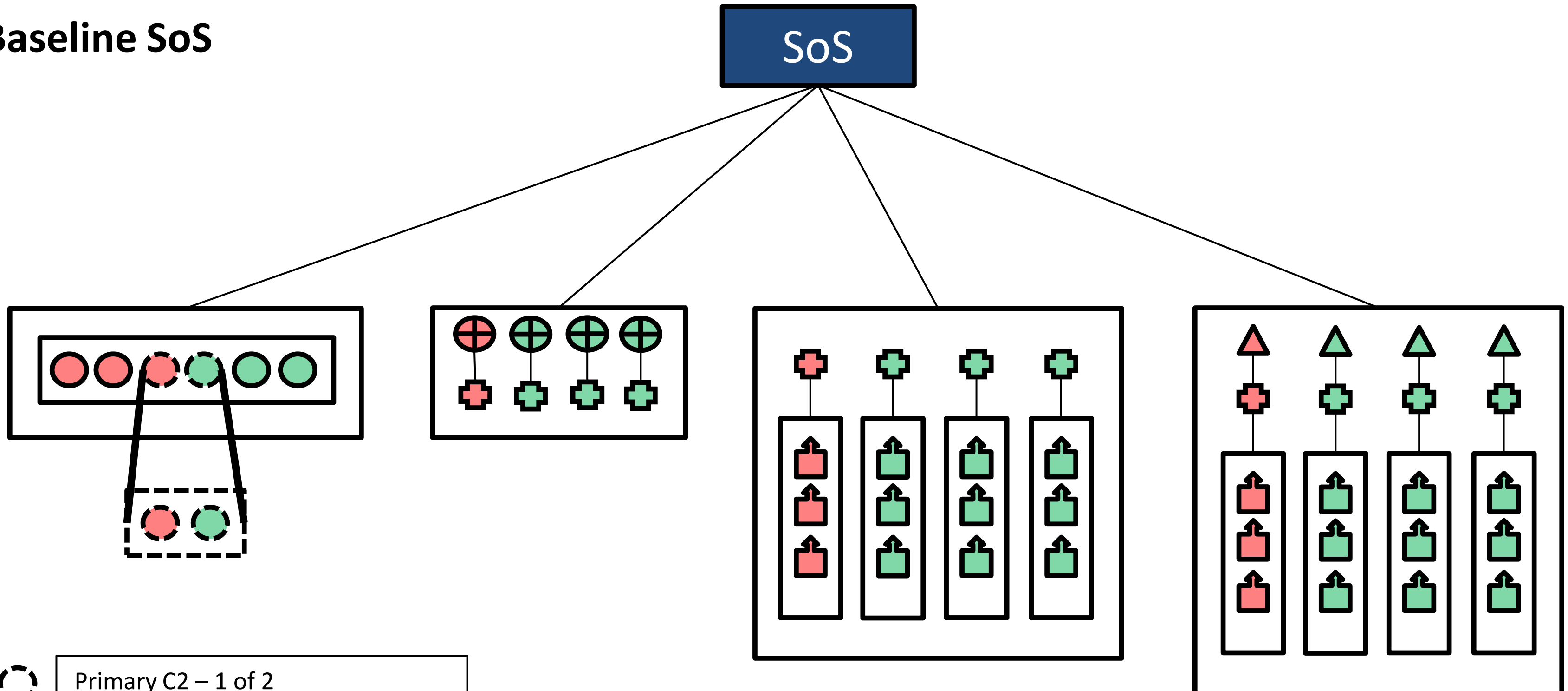


Difficult to calculate many RAM metrics for an SoS with this complexity

- Multiple M of N systems required for operability
- System types have unique failure rates
- Failure rates have unique Mean Time To Repair (MTTR)
- Many other potential intricacies

Discrete-event simulation is essential for determining RAM metrics with complex systems.

Baseline SoS



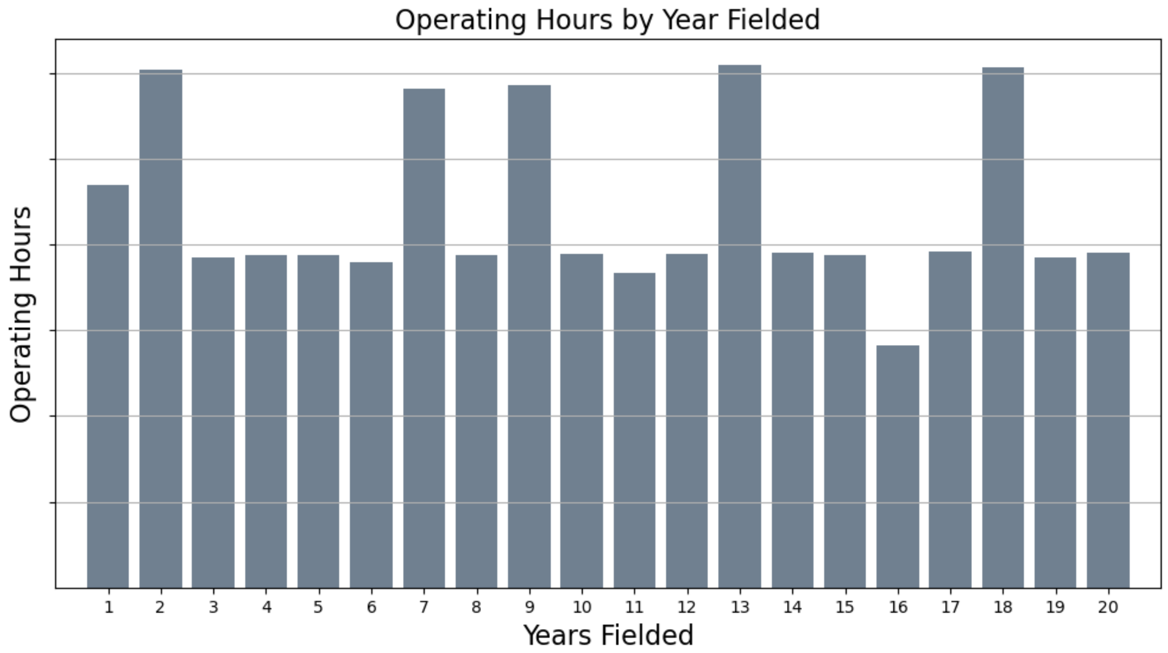
- Primary C2 – 1 of 2
- Secondary C2 – 3 of 6
- Primary Radar – 3 of 4
- Secondary Radar – 3 of 4
- Network Relay – 6 of 12
- Launcher – 18 of 24

- Required to be operational (green)
- Able to be non-operational (across all systems) (red)

Additional requirement: 18 out of 24 Launchers need to be operational.

Model inputs

- System level reliability data
- MTTR
- Admin Log Delay Time
- Sparing rates
- Maintainers
- OPTEMPO
- Scheduled maintenance



System failure rates

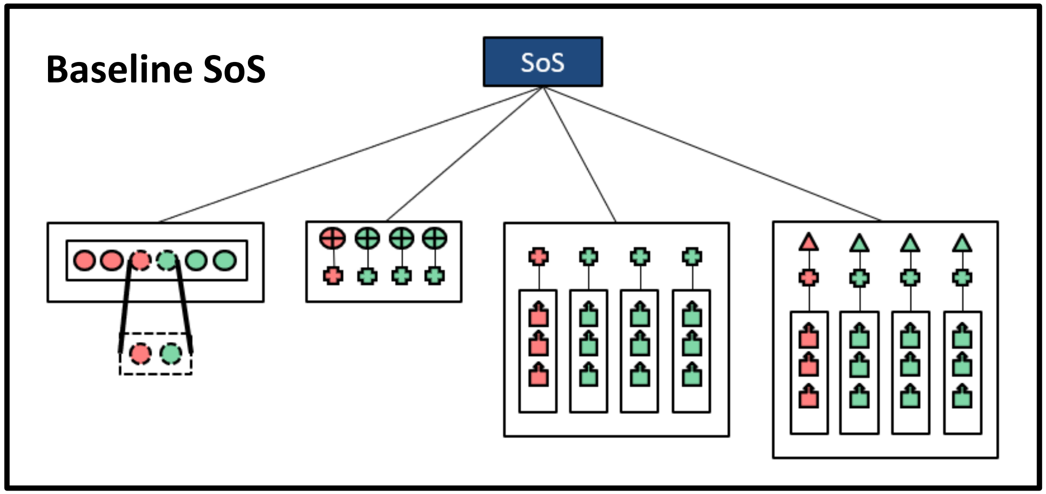
- Primary/Secondary C2
- Launcher
- Primary Radar
- Secondary Radar
- Network Relay

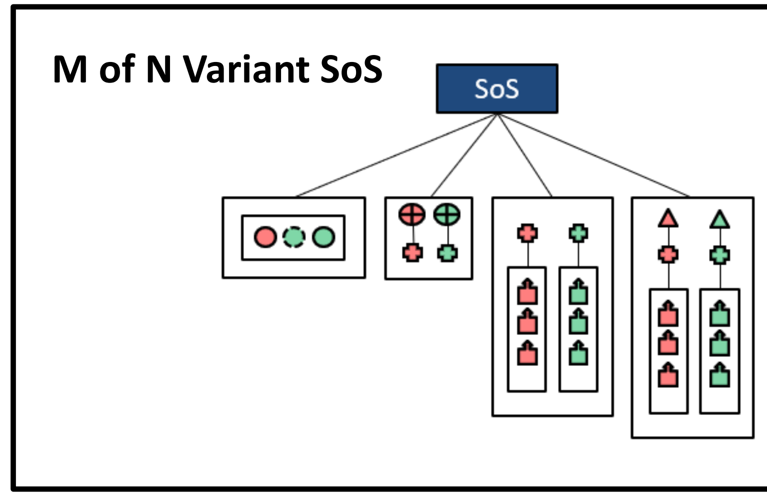
Descending failure rate

Various software tools developed by the LogLab automate data injections, input verification, testing, running the model, and results validation.

EXPERIMENT: PURPOSE

Purpose: Show the effects on SoS operational availability (A_o) due to varying system M of N requirements.





Benefits to this analysis:

- Inform SoS requirements decisions
- Determine SoS A_o given various configurations
- Identify SoS reliability drivers

Approved for Public Release

Experiment Setup: Create separate models for every M of N system requirement combinations.

$$Number\ of\ combinations = \prod_{System\ Types} \sum_{n=0}^{\#\ Systems} n$$

$$Number\ of\ combinations = 3 \cdot 21 \cdot 10\ \cdot 10 \cdot 10 \cdot 9$$

Primary C2, Secondary C2, PRadar, SRadar, Launcher Groups, Launchers

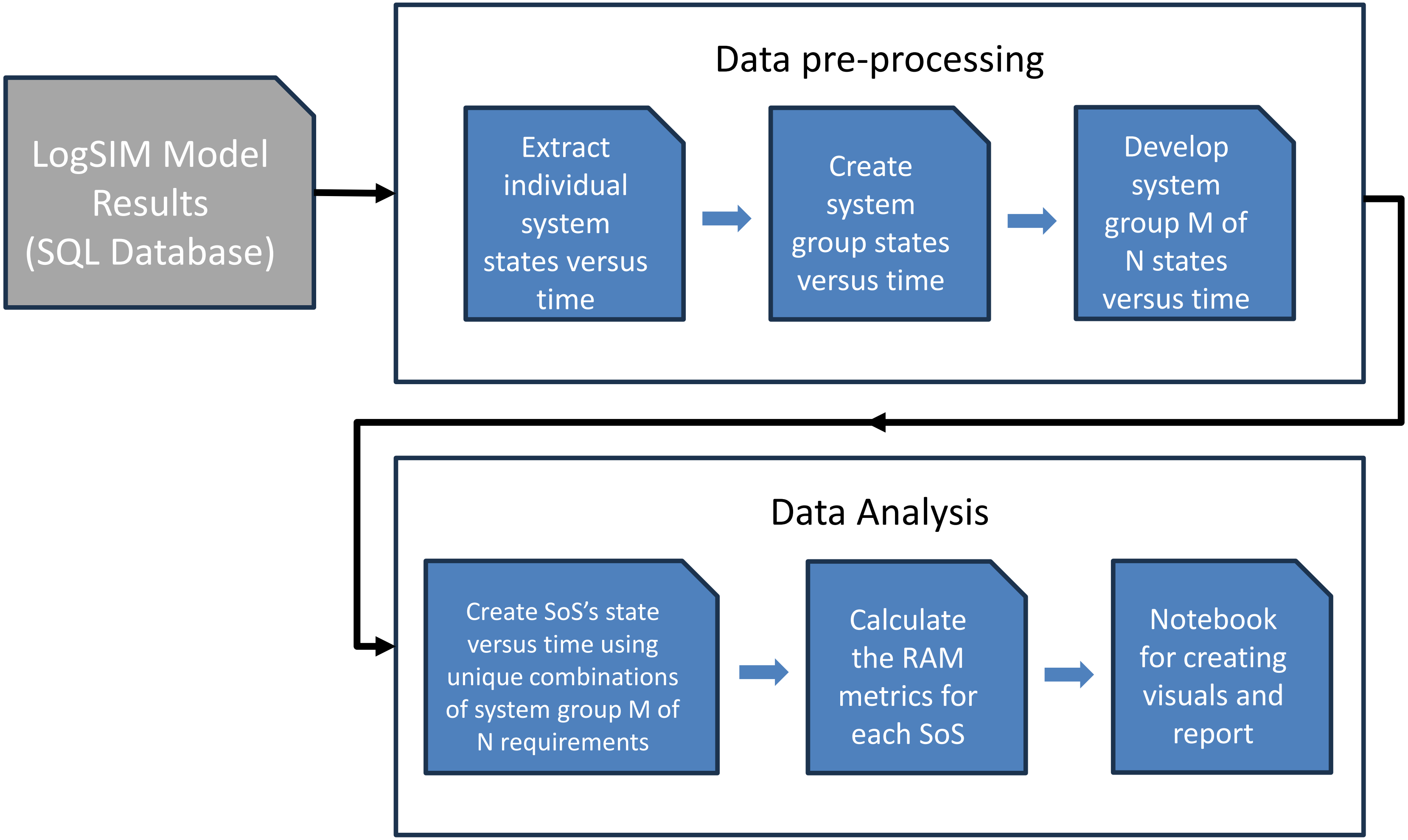
$$Number\ of\ combinations = 567{,}000\ \text{unique models}$$

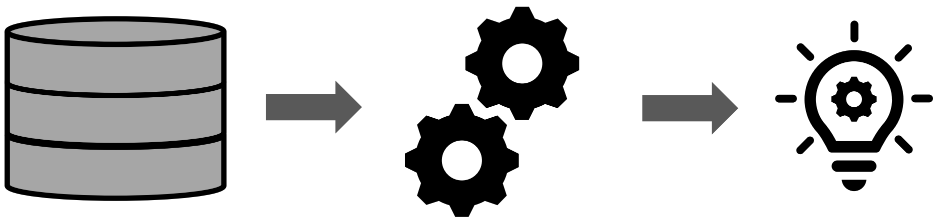
Initial method: Create all combinations as individual LogSIM models

- Relatively straight forward to conduct the experiment
- Tedious to create all required models (even with automated injection tools)
- Significant computational resources and time

Implementing data science techniques can streamline this experiment.

Data Science Approach: Create one LogSIM model with the maximum N of each system requirements and distill system events data for every combination.





Three (3) SoS Features:

1. Cost

- Apply total SoS cost based on N total systems in M of N combinations
- Unique costs for each system type

2. Total Coverage Area

- Apply total SoS coverage area based on N total systems in M of N combinations compared to maximum N systems used

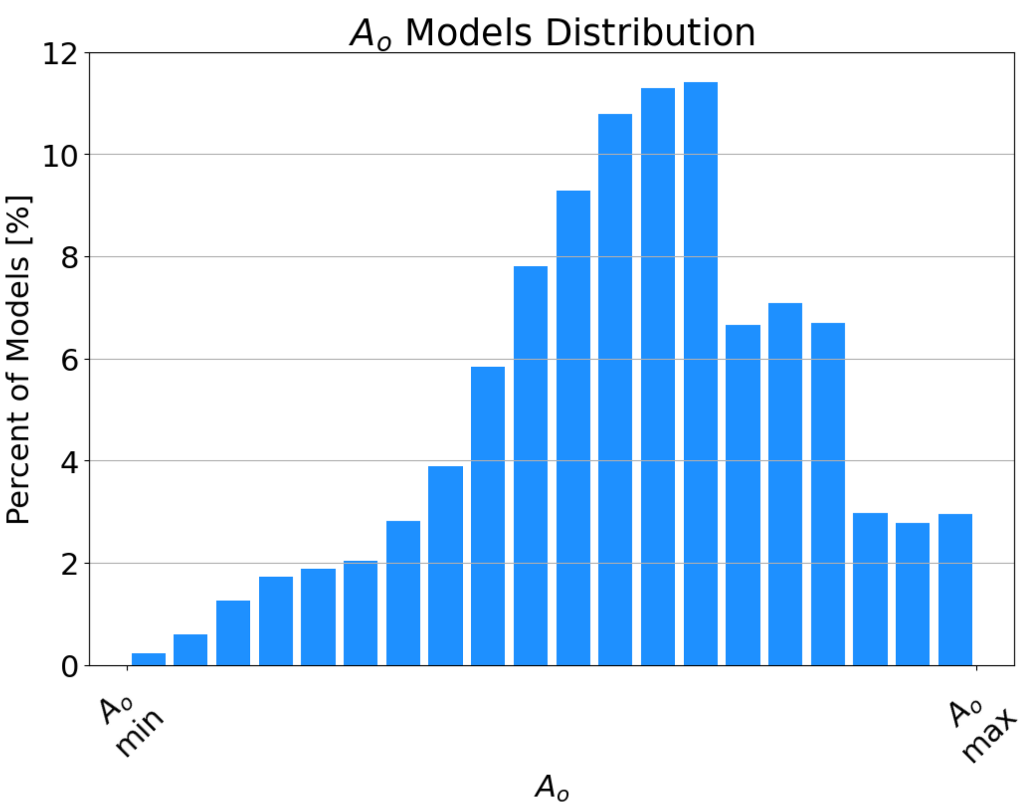
3. Required Coverage Area

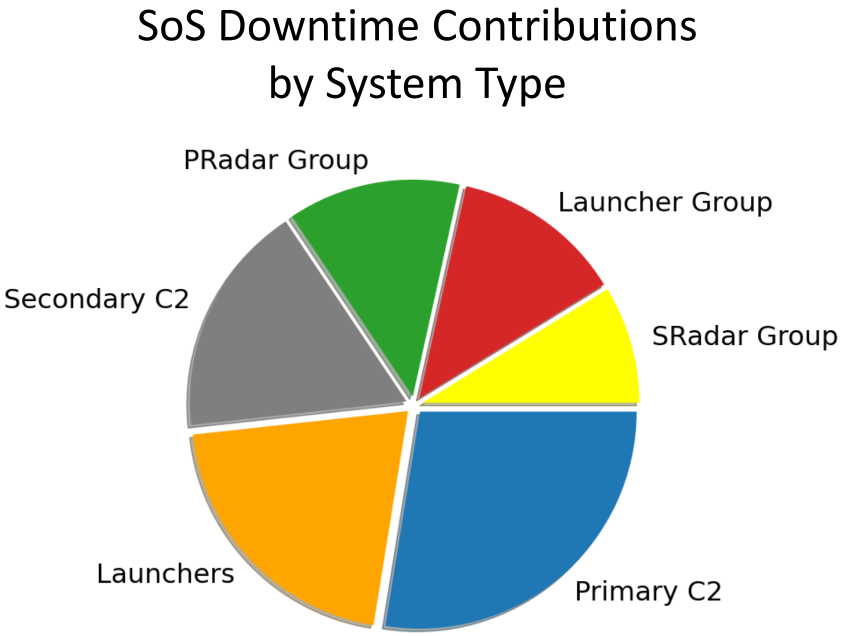
- Apply SoS required coverage area based on M required systems in M of N combinations compared to maximum N systems used

EXPERIMENT RESULTS: METADATA

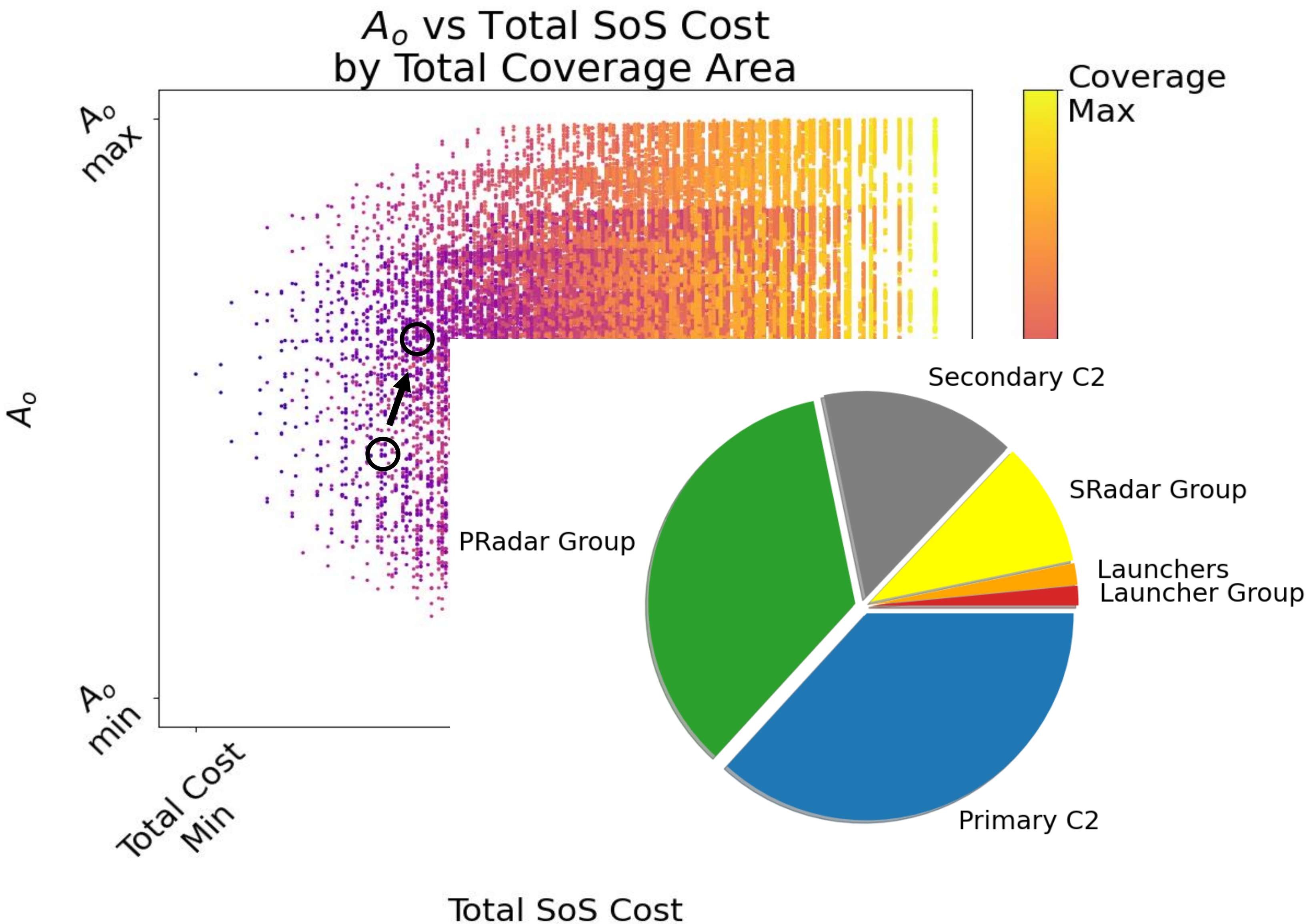
Overview of Data

- 161,671 models (M of N combinations)
- A_o distribution
- SoS downtime contributors

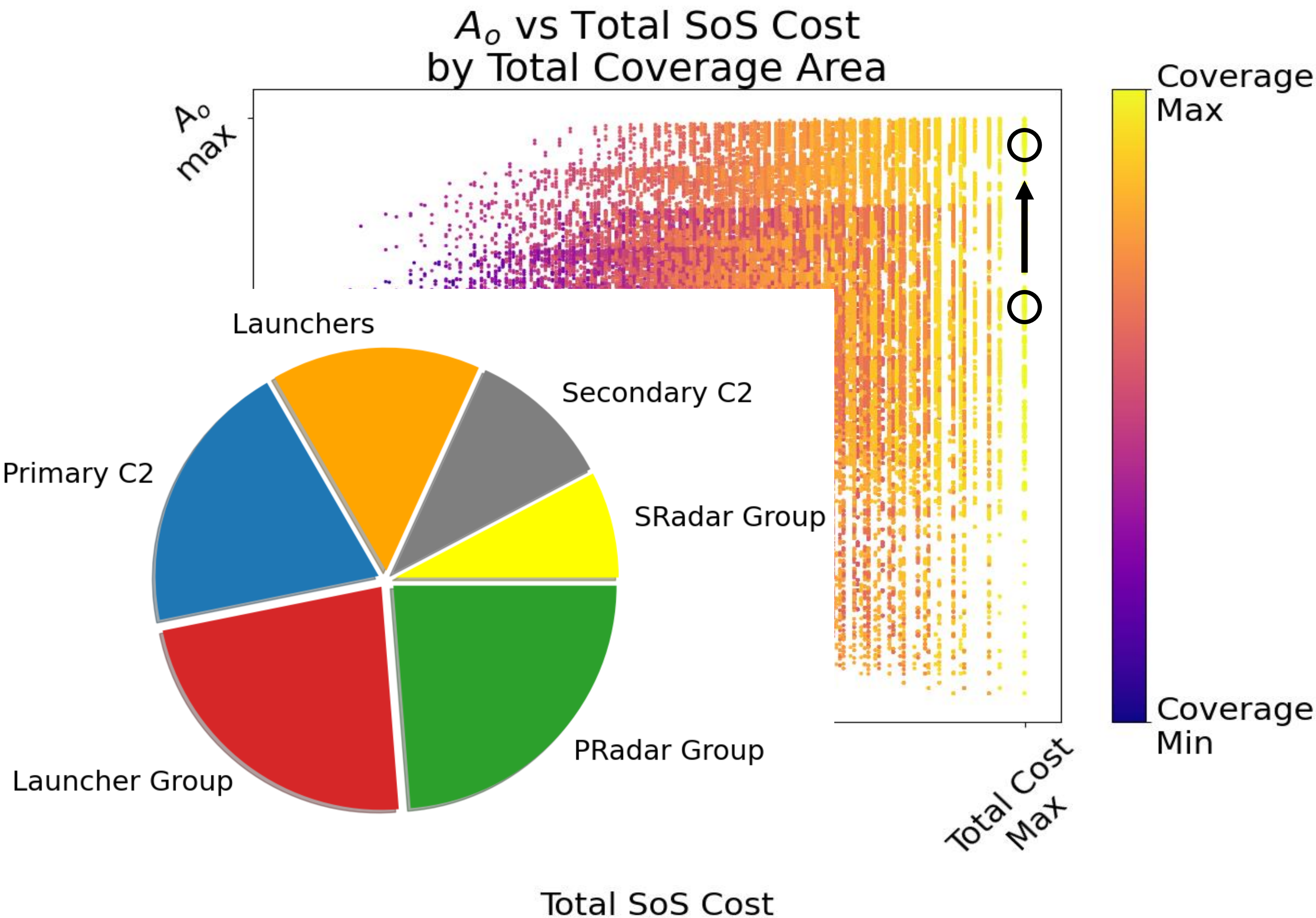




EXPERIMENT RESULTS: A_O



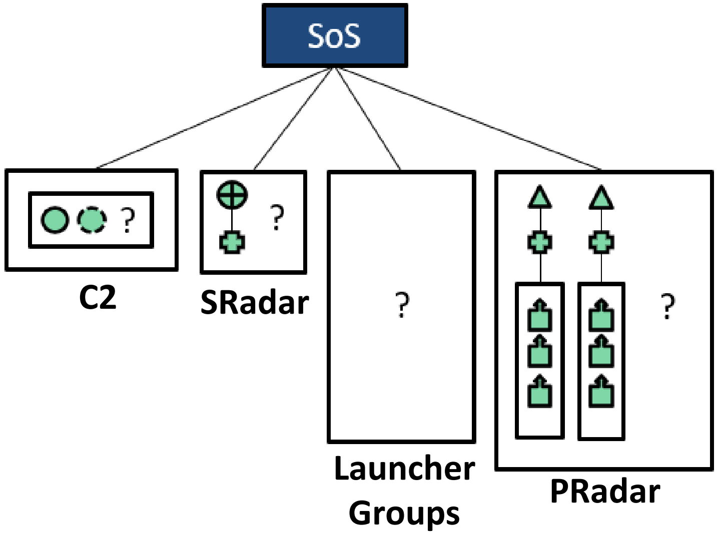
EXPERIMENT RESULTS: A_O

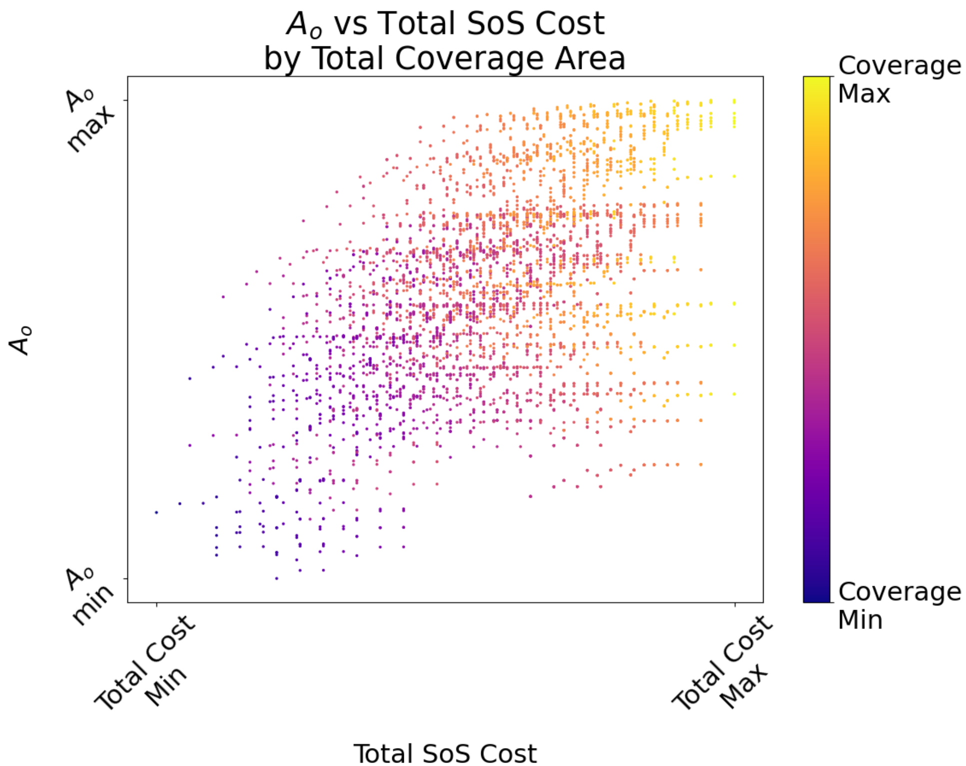


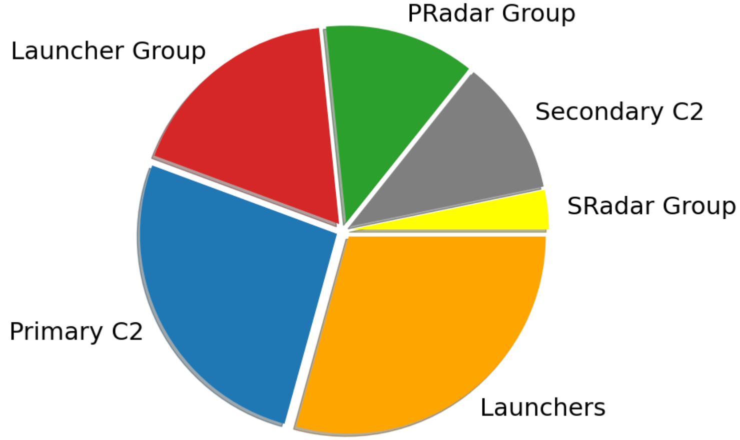
EXPERIMENT RESULTS: SCENARIO ANALYSIS

Scenario with fixed M requirements

- Over 4,000 M of N combinations
- A_o vs Cost/Coverage
- System Type A_o drivers



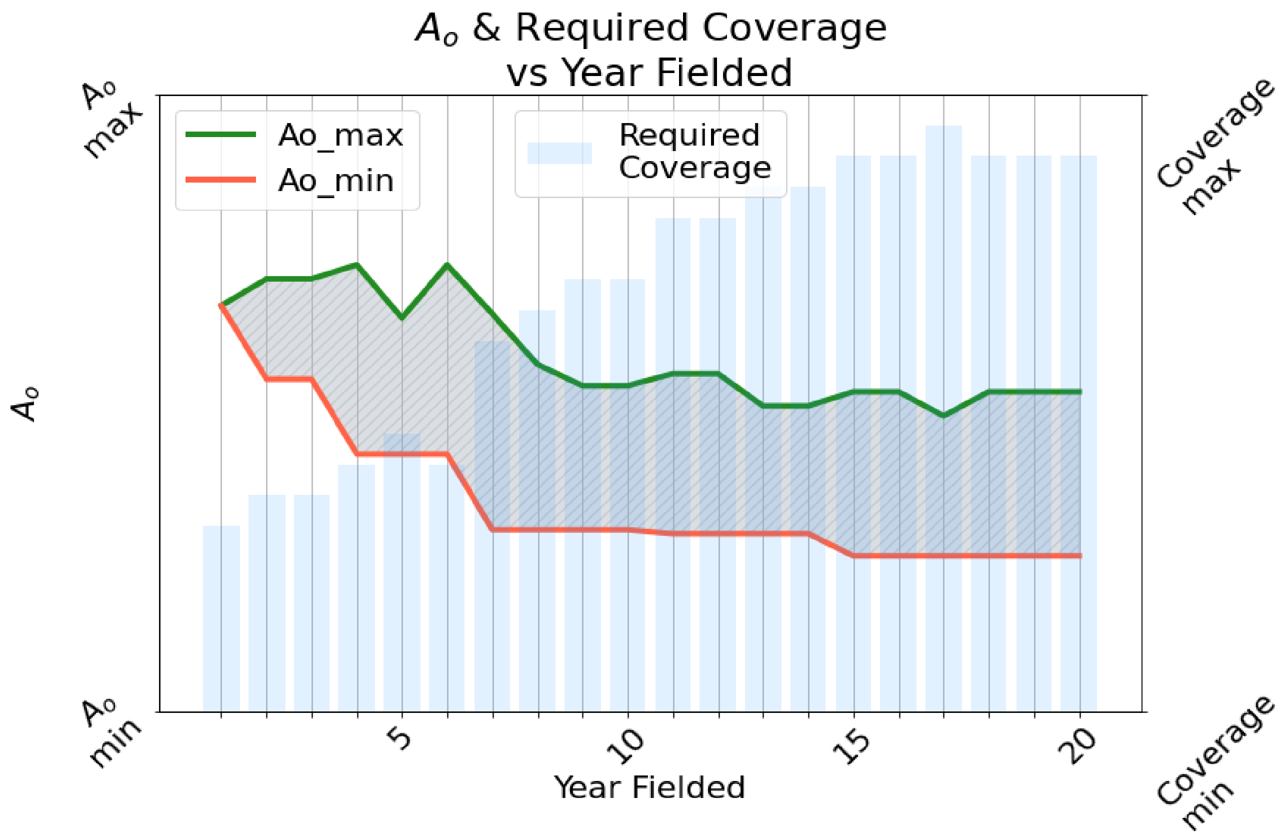




EXPERIMENT RESULTS: LIFECYCLE VARIANCE

Lifecycle SoS with Varied Requirements

- Required Coverage by the SoS ramps up over time
- Resulting A_o values that satisfy required coverage



Given certain constraints, a lifecycle A_o can be developed by querying resulting database.

PATH FORWARD

Summary

- LogSIM is essential for determining RAM metrics with complex SoS
- M of N requirements can significantly impact SoS A_o
- A data science approach streamlined experimentation and developed a database for simplified analysis

Next Steps

- Include data by time interval rather than aggregate values
  - Incorporate time-dependent system failure rates (environmental impacts, degradations/improvements over time, etc.)
- Improve model and software efficiency/resource usage
  - Exclude unnecessary data from model and post-processing
- Expand analysis efforts
  - Include additional RAM metrics
  - Explore grouped subsets
  - Increase size of datasets (fleet-wide)

QUESTIONS

BACKUP SLIDES

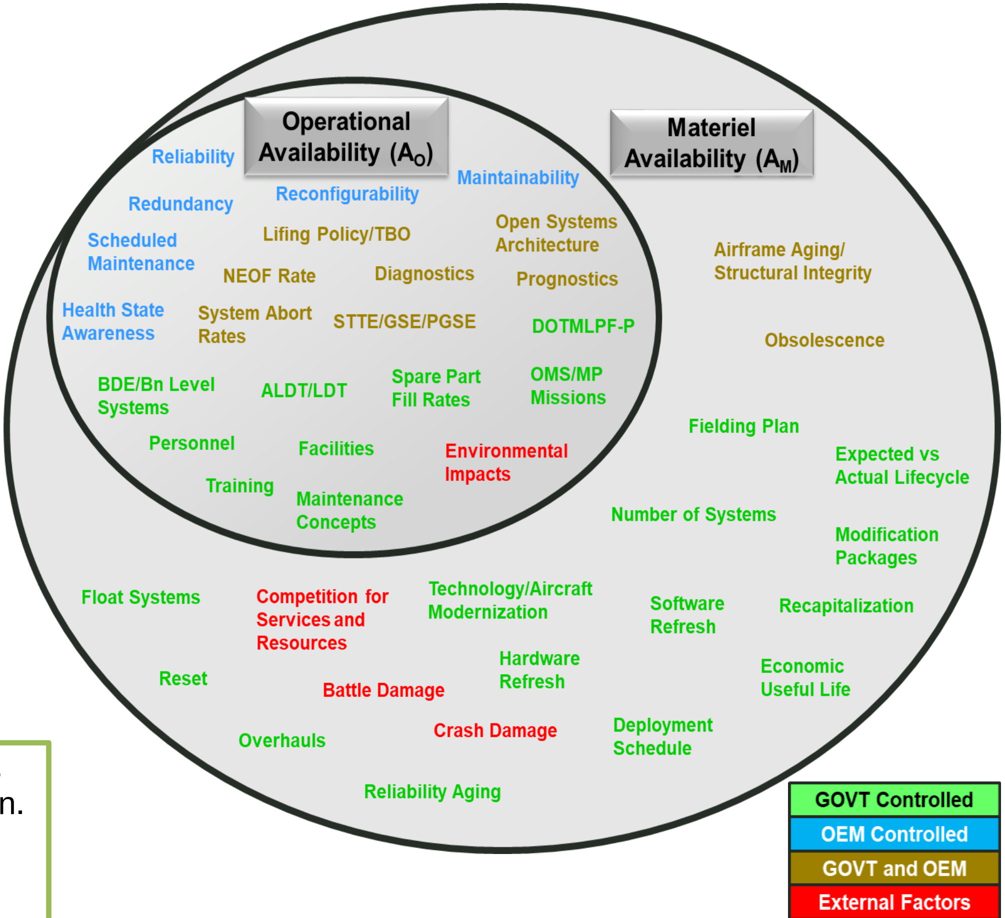
AVAILABILITY METRICS

Materiel Availability (A_M):

- “The percentage of the total inventory of a system operationally capable, based on materiel condition, of performing an assigned mission. This can be expressed mathematically as the number of operationally available end items/total population.”*
- Available Systems / Total Systems (Fleet; **Life cycle**)

Operational Availability (A_o):

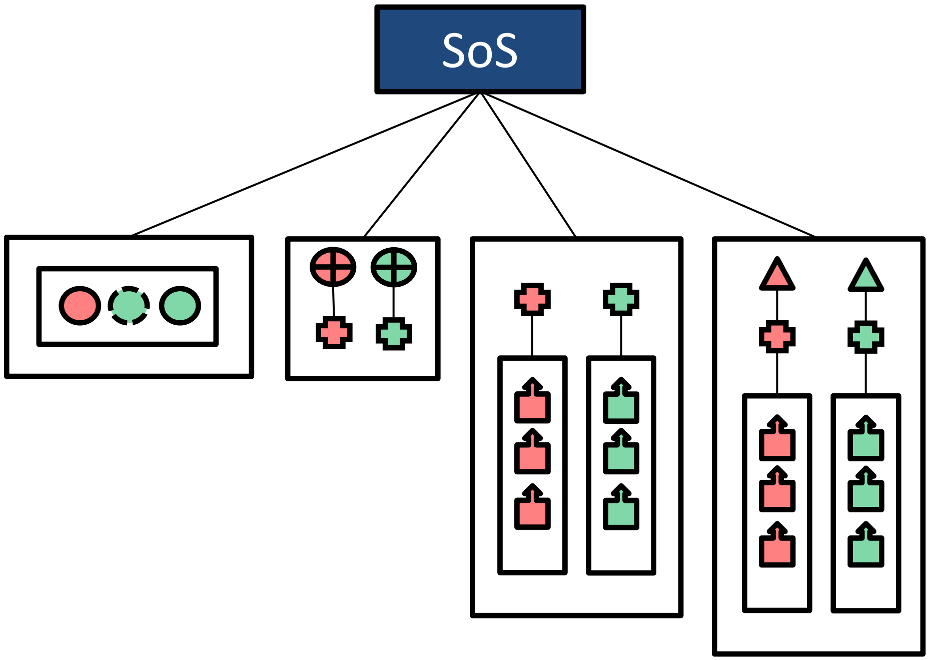
- “The degree to which one can expect a piece of equipment or weapon system to work properly when it is required, that is, the percent of time the equipment or weapon system is available for use. A_o represents system uptime and considers the effect of reliability, maintainability, and Mean Logistics Delay Time (MLDT). It is the quantitative link between readiness objectives and supportability.”*
- System Uptime / Total Time (Battalion; **Month**)



- A_M is a fleet-wide metric over a life cycle. Largest impacts are reliability aging, refresh, and technology modernization.
- A_o is an operational metric tied to a mission. Largest contributors are reliability and spare parts availability.

* https://www.dau.edu/glossary/Pages/Glossary.aspx
≠ Definition of Readiness, Operational Readiness has no official definition

EXPERIMENT: EXAMPLE M OF N VARIANT

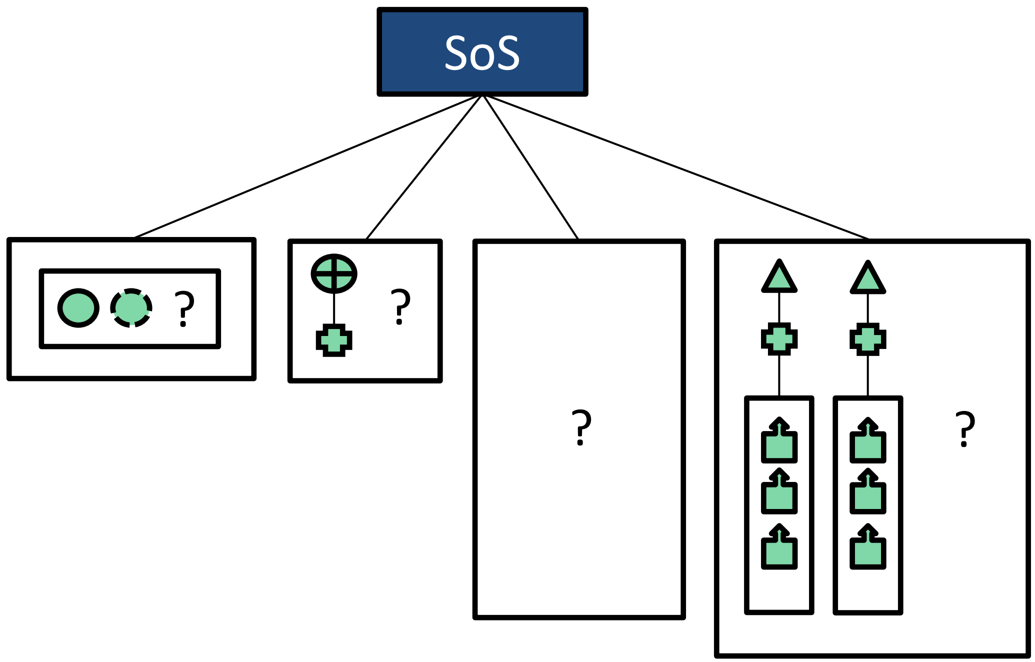


- Primary Radar – 3 of 4
- Network Relay – 6 of 12
- Primary C2 – 1 of 2
- Secondary C2 – 3 of 6
- Secondary Radar – 3 of 4
- Launcher – 18 of 24

- Green: Required to be operational
- Red: Able to be non-operational (across all systems)

Additional requirement: 18 out of 24 Launchers need to be operational.

EXPERIMENT: EXAMPLE M OF N VARIANT



- Primary Radar – 3 of 4
- Network Relay – 6 of 12
- Primary C2 – 1 of 2
- Secondary C2 – 3 of 6
- Secondary Radar – 3 of 4
- Launcher – 18 of 24

- Required to be operational
- Able to be non-operational (across all systems)

Additional requirement: 18 out of 24 Launchers need to be operational.