Automatic, Intelligent Commercial SSA Sensor Scheduling

AMOS 2019 September, 2019

Presented by: Dick Stottler
stottler@StottlerHenke.com
650-931-2714

Overview

Project Goals

Covariance/Complementary Observations/Experiment Results

Commercial SSA Sensors/Capabilities/Advantages

Space Application Scheduling Background

Bottleneck Avoidance Algorithm Applied to Space Applications

Commercial SSA Sensor Scheduling Experiments/Results

Unified Data Library (UDL) Integration

Future Work

Conclusions

Project Goals

Ultimately provide best SSA info from large # of commercial sensors

Determine beneficial use cases

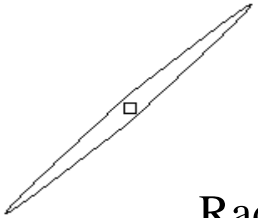
Work out integration mechanisms

Develop commercial SSA sensor optimization algorithm

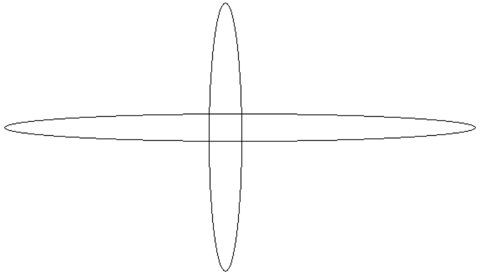
- Variety of use cases and time scales
- 24-hour schedule
- Catalog Maintenance
  - Maintaining orbital parameters
  - Searching for new objects
  - Finding newly lost objects
- Space Object Identification (SOI) Information
- Quick Reaction, i.e. tens of seconds to a few minutes

Determine capabilities/capacities of SSN Sensors

Covariance/Complementary Observations



Radar and optical covariance examples



Combining covariances at a very acute angle.

Combining covariances from orthogonal measurements

Plus nonlinear orbital propagation

Experiment Results: 3x reduction in location errors

Spatial Covariances **3x** better, Standard deviations **7x** better

3160 Deep Space Objects Final 10 Day Position/Velocity Variances

	Prototype Average	Current Method Average	Current/ Prototype	Prototype Std Deviation	Current Std Dev.	Current/ Prototype
X $(km)^2$	3.36	14.43	4.3	3.26	45.27	13.9
Y	3.66	14.39	3.9	8.04	45.32	5.6
Z	3.11	8.08	2.59	4.46	21.32	4.8
Vx $(km/s)^2$	0.00148	0.00577	3.88	0.00130	0.00662	5.1
Vy	0.00151	0.00577	3.82	0.00131	0.00662	5.1
Vz	0.00141	0.00566	4.01	0.00130	0.00662	5.1

674 GEO Final 10 Day Position/Velocity Variances.

	Prototype Average	Current Method Average	Current/ Prototype	Prototype Std Deviation	Current Std Dev.	Current/ Prototype
X $(km)^2$	4.95	29.73	6.0	2.17	72.60	33.5
Y	5.22	27.82	5.3	4.31	67.74	15.7
Z	4.13	9.17	2.22	5.78	16.11	2.8
Vx $(km/s)^2$	0.00191	0.00807	4.23	0.00228	0.00732	3.2
Vy	0.00193	0.00806	4.17	0.00229	0.00735	3.2
Vz	0.00168	0.00783	4.66	0.00231	0.00734	3.2

X-Y Covariances **5-6x** better, Standard deviations **25x** better

Commercial SSA Sensors Capabilities

Gov. will not directly use comm. SSA sensor data in its orbital parameter calculations (each SSN sensor must be certified, for now) but some government tasking is:

Searching for lost objects, providing orbital params for SSN gov. sensors to re-acquire

Searching volumes of space for new objects

Other Tipping and cueing

On the fly (short lead-time items) tasking (which could be volume/time based)

High priority objects (could be volume/time based, to avoid classification issues)

Maneuver detection / Propulsion Detection

Post-Launch Observations

Unclassified sensors could occasionally be tasked with Classified objects

Space Object Identification (SOI): Images, Light Curves (to derive rotational and other movement frequencies), and Passive RF Signals and their Timing

Track Maintenance for low priority, unclassified objects, e.g. debris and commercial and university satellites

Commercial Sensor Advantages

1000+ telescopes/sensors across 100+ sites: persistent GEO and LEO coverage

Immediately responsive (tens of seconds to a few minutes)

Real-time data: see what's happening in GEO in real-time (5 minutes after tasking)

Very low $/observation or $/FOV; great $ efficiency

Subscription model – continuous improvement in accuracy and info. extraction

No requirements so didn't stop when they were met

Extracting the maximum information angles/brightness/dim objects

Observe behaviors (including light curves) can tell if 3-D stabilized/spin stabilized/tumbling

Burns and burn size, Slot changes, Catastrophes, Objects deployed from satellites (or broken off)

Help operators locate satellites in response to immediate requests

Observe anomalous satellites in response to immediate requests

Want the space object observation data, not to acquire/own/maintain sensors

Space Application Scheduling

Very easy to build bad scheduler, hard to build good one

Scheduling with resource assignment is NP Complete (exponential time)

- Takes exponential time to guarantee an optimal solution
- $(4 \text{ meaningful options per each choice})^{1000 \text{ Decisions}}$
- $4^{1000} = 2^{2000} = 10^{200} >> 10^{80}$ = # particles in the universe

Can't guarantee optimal solution, every scheduling algorithm is different and produces different answers, some good, some bad, some fast, some slow, slow not necessarily producing better schedules

Search Alg.: Genetic Algs, Sim. Annealing, A*, Heuristic Search, Iterative Repair

Operations Research: Linear Programming, Branch and Bound, Hill-Climbing, Mixed Integer, Usually these must oversimplify the problem

Common Bad Algorithm: Priority Order, Greedily Pick Resource

- Other ways to guarantee high priority tasks, e.g. swap out lower Priority at the end

Near Linear Algorithms (Global Info./Visual Cortex) vs Search vs OR

Bottleneck Avoidance Algorithm Motivation

Human experts are currently very successful at building highly optimal schedules

Very specialized: requires lots of training and experience

Building a schedule manually requires a great deal of time and effort

Opportunity: apply automated techniques that mimic experts' processes and leverage existing knowledge

Bottleneck Avoidance Overview

Goal: schedule the least flexible tasks first; leave room for more flexible tasks to schedule later

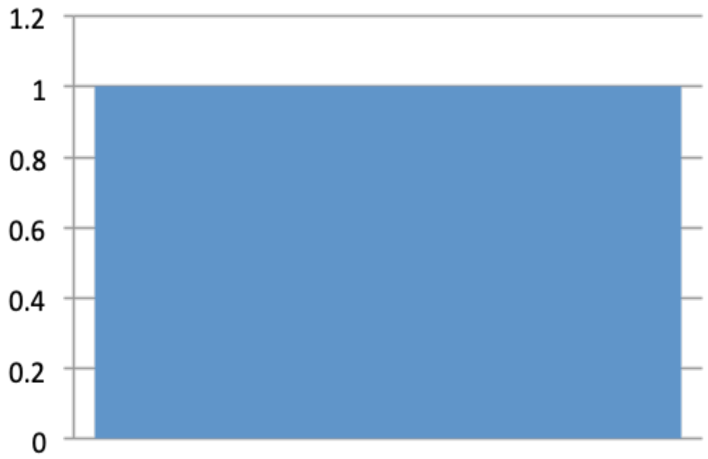
Track the actual allocations of scheduled tasks and the *probabilistic* allocations of unscheduled tasks

At each scheduling step, find “bottlenecks” – spots with the greatest resource contention

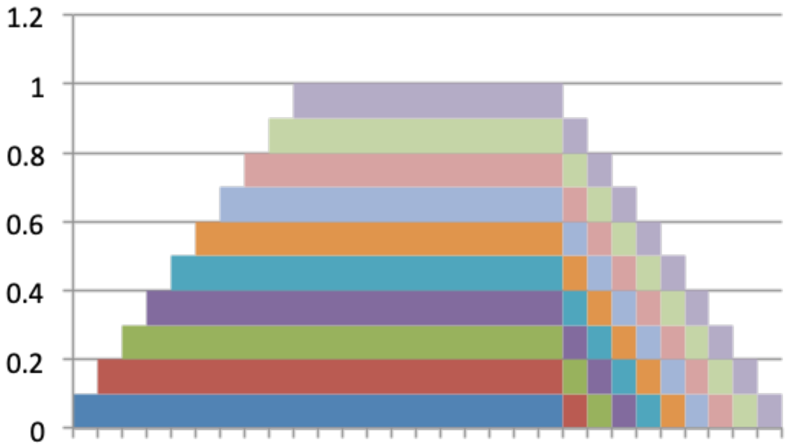
Schedule tasks away from these peaks to reduce contention

Predicting the Allocation of an Unscheduled Task

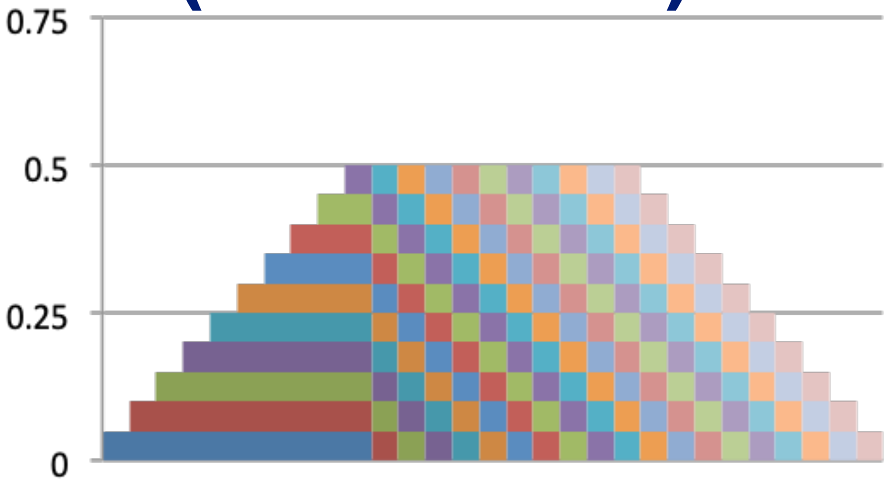
Inflexible task (e.g., LEO) : one possible assignment with 100% probability



Flexible task (MEO/GEO): 10 possible assignments, each with 10% probability



Predicting the Allocation of an Unscheduled Task (continued)



Very flexible tasks

- never reach 100% probability (20 possible allocations, each at 5%)

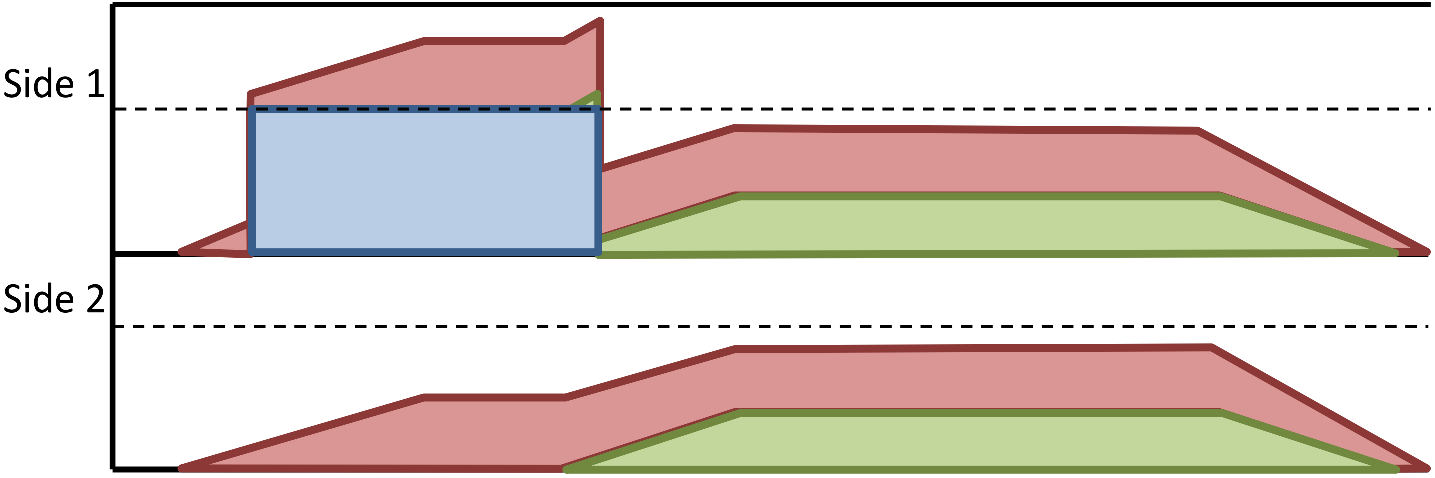
Calculate a trapezoid for each task

Divide across all possible resources (4 possible resources = 25% to each)

Probabilistic Bottleneck Model

First step: Sum the predicted allocation for all tasks

Simple example using three tasks. Blue is fixed in time (LEO support). Red and green are more flexible (MEO/GEO support).



Schedule Processing Order

BA is a one-pass algorithm without backtracking

The order in which tasks are processed is *very* important

Schedule an inflexible task early because, otherwise, its required resources may be allocated to a different task

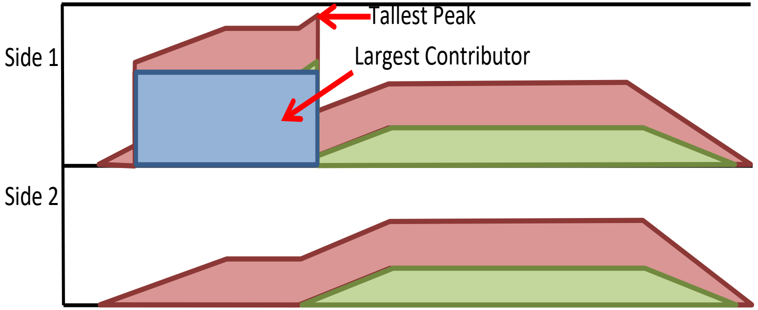
Each step attempts to reduce bottlenecks (peaks of predicted resource contention)

To find the next task to schedule:

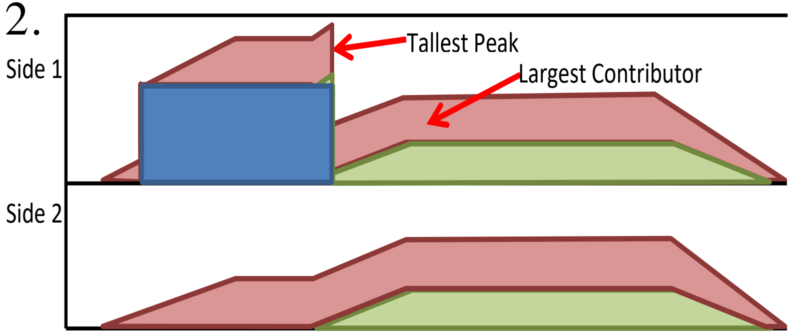
1. Find the tallest predicted usage peak or bottleneck that has at least one unscheduled task
2. Find the unscheduled task that contributes the most to the peak (the task that is most likely to schedule there)

Processing Order Example

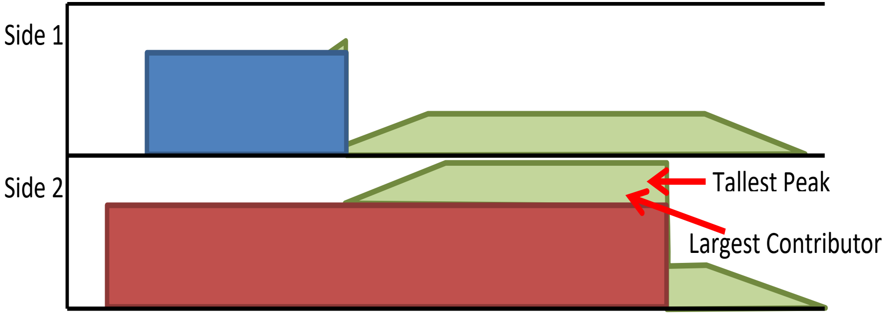
1.



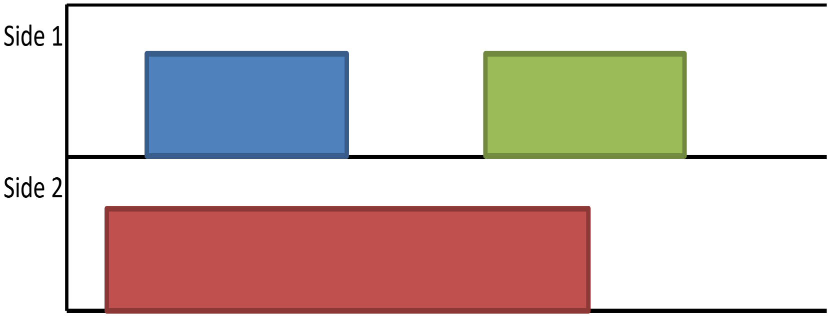
2.



3.



4.



Temporal Allocation and Resource Selection

Once a task is selected for processing, scheduling involves:

1. Finding an open combination of resources that satisfies the task's requirements
2. Allocating the task temporally on those resources

Bottleneck Avoidance attempts to minimize resource contention at each step

Schedule a task *away* from bottlenecks, such that its new allocation minimizes all peaks

Comm. SSA Sensor Sched. Experiments/Results

Appears to Produce Optimal Results

Determine Single Observation Persistence

- 4 & 8 min (LEO/nonLEO) observations
- 17,000+ TLEs, 528 sensors/93 sites/1.5M Visibilities
- 2.5 minutes (single threaded Java on desktop) scheduling -> all tasks scheduled
- 8/16 min. obs, vast majority scheduled

Algorithm Runtime Reductions:

- Parallelize Java on desktop: 1 minute
- Single Thread C++: < 1minute
- Optimize single thread C++: 8 seconds
- Parallelize optimized C++: 4 seconds

Press Commercial Capacity (3 obs):

- 17K TLEs, 52K tasks/1.5 Vizs/2-4 & 4-6 min observations
- Observations separated by > 4 hours
- 4 seconds scheduling -> 99% tasks scheduled

Quick Reaction:

- 100 new immediate requests added to above 52K schedule
- All rescheduled within 0.1 seconds (1 millisecond each)
- 33% bumped tasks rescheduled within 23 milliseconds

Unified Data Library (UDL) Integration

- UDL is a central repository of SSA data
- UDL jointly funded by AFRL/CAMO and SMC/DPMO
- Increase exposure of commercial space data
- Enable access to academic, gov. and commercially-gathered satellite data sets
- Variety of data access methods (batch, query, streaming, archive)
- Most commercial SSA sensor data providers represented
- Access to commercial observations is dependent on data purchases or affiliation to an effort that has purchased data
- Streamlines data distribution and data integration for end users or applications
- Can add specifically assigned tasks in real-time for real-time monitoring/execution by commercial SSA sensor owners
- Combined with SMC's SSA marketplace will enable real-time transactions & distribution of data. SSA marketplace will be online Fall of 2020

Future Work

More diverse tasking (e.g. searching, SOI, sensor quality)

More testing with more diverse realistic scenarios

Improved deliberate and quick-reaction algorithms

Integrate Current/Forecast Weather

UDL Integration

Government agency querying capability

Satellite Constellation Scheduling

- ISR Collections
- Support Communications

Conclusions

Commercial SSA Sensors are quite numerous and capable and constantly improving

Commercial SSA Sensors offer near-real-time tasking and data/visualization

Commercial SSA sensor data are readily available

High quality space scheduling algorithms exist to quickly take full advantage of SSA Sensors