Improved Space Surveillance Network (SSN) Scheduling using Artificial Intelligence Techniques

Richard Stottler
Stottler Henke Associates, Inc., San Mateo, CA 94002

ABSTRACT

There are close to 20,000 cataloged manmade objects in space, the large majority of which are not active, functioning satellites. These are tracked by phased array and mechanical radars and ground and space-based optical telescopes, collectively known as the Space Surveillance Network (SSN). A better SSN schedule of observations could, using exactly the same legacy sensor resources, improve space catalog accuracy through more complementary tracking, provide better responsiveness to real-time changes, better track small debris in low earth orbit (LEO) through efficient use of applicable sensors, efficiently track deep space (DS) frequent revisit objects, handle increased numbers of objects and new types of sensors, and take advantage of future improved communication and control to globally optimize the SSN schedule.

We have developed a scheduling algorithm that takes as input the space catalog and the associated covariance matrices and produces a globally optimized schedule for each sensor site as to what objects to observe and when. This algorithm is able to schedule more observations with the same sensor resources and have those observations be more complementary, in terms of the precision with which each orbit metric is known, to produce a satellite observation schedule that, when executed, minimizes the covariances across the entire space object catalog. If used operationally, the results would be significantly increased accuracy of the space catalog with fewer lost objects with the same set of sensor resources. This approach inherently can also trade-off fewer high priority tasks against more lower-priority tasks, when there is benefit in doing so. Currently the project has completed a prototyping and feasibility study, using open source data on the SSN's sensors, that showed significant reduction in orbit metric covariances. The algorithm techniques and results are discussed along with future directions for the research.

1. INTRODUCTION

Tracking of objects in earth orbit is an extremely important task for maintaining the safety and viability of manned and unmanned spacecraft. Space Track (http://www.space-track.org) provides two line elements (TLEs) for the approximately 20,000 manmade objects tracked in earth orbit that comprise the Space Catalog. The sensors used to track these objects are mechanical and phased array radar and optical telescopes. Unfortunately, insufficient resources exist to easily make enough observations to track every object's orbit to desired accuracy. This problem is exacerbated by the current space object tracking sensor-scheduling process. This process takes as input the current Space Catalog. Based on the characteristics of the object, the specifics of each sensor, and the geometry of each object's orbit with respect to each sensor site, a maximum probability of detection is calculated. Based on each object's priority, number of observations needed, and the maximum detection probabilities, each sensor is tasked to observe each object a specific number of times. A scheduler at each sensor site determines which opportunities and when within each opportunity to make observations (i.e. it schedules), without reference to what other sensors are observing the same object (and when) and which orbit metrics will contain the most and least errors. It does not necessarily schedule the same time or even the same opportunity that corresponds with the maximum probability of detection. There are issues with the current process:

1. The schedule is not globally optimized since scheduling occurs only locally without reference to what is assigned or scheduled at other sensor sites.
2. A globally optimized schedule would typically have more observations and a better number of observations for each object, with the same resources.
3. By considering the effects of other observations of the same object by other sensors, specific sensors at specific times can be chosen that provide complementary observations of the object, which will reduce the object's orbit metric error covariance matrix.

A scheduling algorithm was developed that takes as input the space catalog and the associated covariance matrices and produces a globally optimized schedule for each sensor site as to what objects to observe *and when*. This

algorithm is able to schedule a better number of observations of each object with the same sensor resources and have those observations be more complementary, in terms of the precision with which each orbit metric is known, to produce a satellite observation schedule that, when executed, minimizes the covariances across the entire space object catalog. The results would be increased accuracy of the space catalog with fewer lost objects with the same set of sensor resources. The algorithm has been prototyped and tested in simulation and the resulting orbit metric covariances calculated. These results are presented in this paper.

2. DEEP SPACE OBJECTS

The Space Catalog is generally divided between Deep Space (DS) and Low Earth Orbit objects with the dividing line being a period of 225 minutes. Due to their greater distance, DS objects require sensors of greater sensitivity and precision, and longer tracking time, and there are fewer of these sensors, hence the heightened need for highly efficient and complementary scheduling. One especially important part of the DS region is the Geosynchronous (GEO) belt. (GEO can also refer to "Geosynchronous Earth Orbit"). GEO objects orbit once per day, meaning they are essentially fixed in longitude. This means that they have LOS to a large, relatively unchanging fraction of the earth all the time, but at a long range. Non-GEO DS objects tend to have long time windows of LOS to specific locations and the set of those locations changes relatively slowly over time. Sensor requests typically inherit the constraints for the types of orbiting object to which they relate.

3. SENSOR COVARIANCE BACKGROUND

An orbiting object's state at any given instant can be uniquely defined by either 6 orbital parameters (Eccentricity, Semimajor Axis, Inclination, Longitude of ascending node, Argument of periapsis, and Mean anomaly at epoch) or equivalently as its 3-D position and velocity vectors (6 total parameters). Radar systems tend to provide accurate range and range rate measurements (2 parameters) but are much less accurate as to angle and angular rate measurements (4 parameters with relatively high error). Meanwhile optical sensors provide good angle (2 parameters) and angular rate (2 parameters) but no range or range rate (2 parameters missing). For each sensing opportunity, a 6 x 6 covariance matrix can be calculated based on the geometry of sensor and sensed object and the accuracies of the sensor in different dimensions; the covariance matrix describes the probable volume of space for the sensed object (or, equivalently, the variances and covariances of the errors in measurement). These errors in the orbit metrics can be best reduced by complementary measurements. The degree to which potential additional measurements are complementary can be determined by calculating the covariance matrices for those candidate measurements and comparing how complementary they are to the existing covariance matrix. In general, if the first measurement was optical, then for another optical measurement to be complementary would require it to either be from an instrument far enough away (and not oriented in line with the previous sensor and sensed object) to achieve a cross-fix to the desired accuracy or to be at a substantially different part of the orbit.

So given limited sensor resources and given that the large majority of measurements are made with optical sensors [1] with infinite range error, where specifically in the orbit are two measurements complementary to most reduce the error in an object's position? First, consider what is currently done. Typically, objects are scheduled to maximize their probability of detection. The time usually occurs when the object is opposite the Sun from the sensor's perspective, essentially local midnight. This is illustrated in Fig. 1 below with the object to the far right.

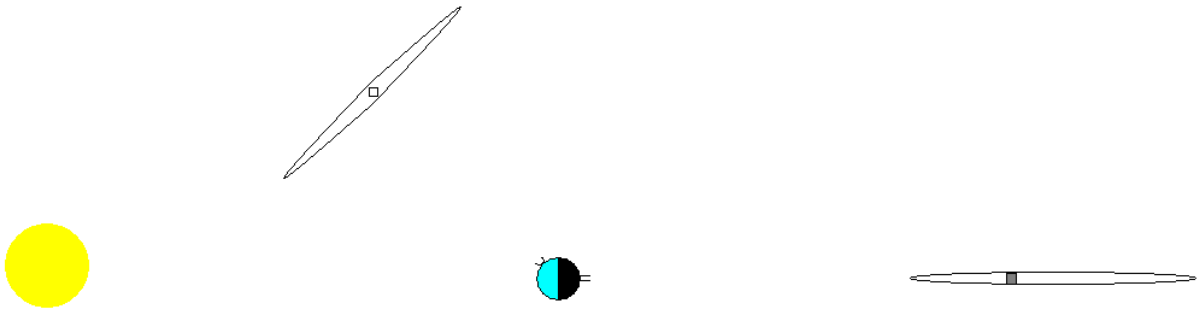

Fig. 1. Radar and optical covariance examples.

Obviously one optical measurement like this gives no information about the object's orbit's size (semi-major axis) or eccentricity. (For comparison, a radar measurement of an object, during the day, is shown to the upper left.) The semi major axis is related to the period of the orbit. So if another measurement is taken on a later night at the same place in the orbit, the period of the orbit and thus the semi-major axis can be determined. But almost no information about the eccentricity can be derived. Similarly, although the angular error is fairly low it is not zero. Minor errors in angular velocity will mean the angle of the orbit with respect to the measurement will not be precisely known as illustrated in Fig. 2 below where the measurement was taken where the orbits cross on the reader's side. If optical measurements are taken at the same point in the object's orbit, the object's location is fairly precisely know at that point and exactly opposite that point but not known with much precision 90 degrees from that point (where both the angle and eccentricity errors will be greatest.) Both of these issues point to taking complementary measures, optimally 90 degrees from the last measurement.

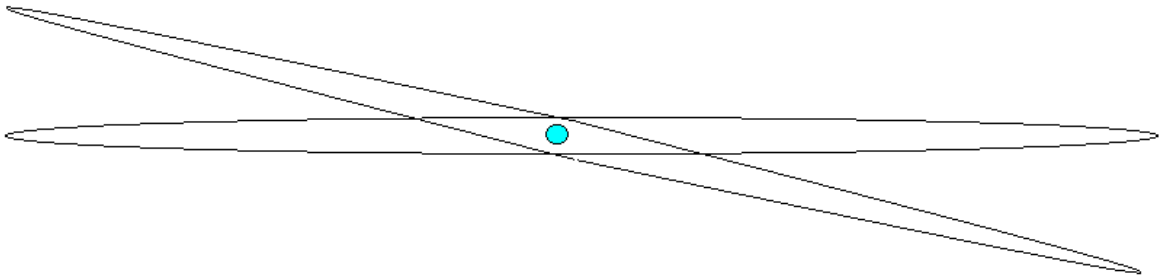

Fig. 2. Example of angular error on orbit metric error.

Measuring an object's location with real sensors, which have real error effectively implies determining a volume where the object is likely to be. This volume around the measured point is described by the covariance matrix (a 6 x 6 matrix in either the position-velocity space or in the orbital parameter space). Users tend to place far greater importance on the position variances (3 of the diagonal elements) than on other parts of the covariance matrix. One measurement's covariance matrix can be propagated to the time of another measurement so that the two measurements can be combined. The result is approximately the intersection of the two volumes.

Consider two optical measurements taken relatively closely in time and propagated to the same time as shown in Fig. 3 below.

Fig. 3. Combining covariances at a very acute angle.

Because the angle between the two is small, the reduction in error is relatively small. But consider two measurements taken at 90 degrees apart in the orbit. This looks more like the following, Fig. 4 (after propagating the first measurement 90 degrees along the orbit to the second measurement):

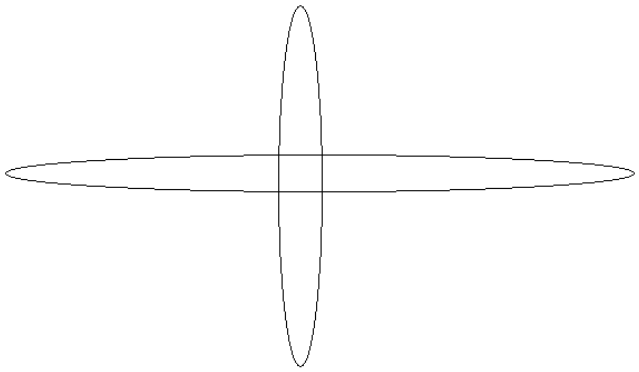

Fig. 4. Combining covariances from orthogonal measurements.

As shown above, taking successive optical measurements closer to 90 degrees apart and farther away from 0 or 180 degrees apart will lead to smaller intersecting volumes and therefore smaller combined covariances. The measurements do not need to be precisely 90 degrees apart in the orbit. 70% of the benefit is realized at 45 degrees.

Alternatively, if the first observation is optical, then almost any radar measurement which is not close to orthogonal to the optical measurement will be complementary (orthogonal in the sense that the line from the optical sensor to the sensed object is orthogonal to the line from the radar sensor to the sensed object). In particular, an immediate radar observation collocated with the optical sensor (or at least not orthogonal to it) will greatly reduce the combined covariance. The same reasoning applies if the first observation is radar and the second is optical. Any non-orthogonal geometry will produce a complementary measurement and greatly increase the orbit metric precision.

A typical radar used for tracking deep space objects has about 0.02 degrees of error in angle and anywhere from 26 to 160 meters in range error. For a geosynchronous satellite about 36,000 km from the radar, the angular error translates into about 13 km of error for both the along-track and cross-track measurements. So, radar is about two orders of magnitude more accurate in range than in angle, for objects at this distance. Meanwhile, an optical telescope, while giving no range information, is accurate to about 0.004 degrees in angle, which translates to about 2.5 km of error in angle for geosynchronous objects. Obviously, the most precise measurement would involve both an optical and a radar measurement in relatively quick succession; however, the deep-space sensors are currently over-taxed so this is not possible. Also while an optical measurement might take around 1 minute, a deep space radar measurement will take considerably longer. (Mitigating this somewhat is the fact that radar can operate at any time of day while the optical sensors only operate at night.)

4. SCHEDULING PROBLEM SETUP

3160 deep space and geosynchronous satellites were considered for the purpose of this study. A NORAD element set from Apr 13, 2006 was the basis for the satellites' orbital parameters. Nine sensors were scheduled for Deep Space and Geosynchronous object tasking: 5 optical sensors and 4 radars. Sensor data was obtained from *Fundamentals of Astrodynamics and Applications* (Vallado, 2001) [2].

In order to model the Earth orbits of every satellite in our catalog, we used NORAD's SGP4 propagation code (available from www.celestrak.com). Updated by Lt Col. David Vallado, then revised again in 2008 (Vallado and Crawford, AIAA-2008-6770)[3], this is the most up-to-date version of publicly-available propagation code that most closely resembles that which is actually used by NORAD to generate Two-Line Elements. The code was tested and matched with example orbit propagations from papers in 1980 and 2008. Since we do not have the capability to take actual observations and measurements, for the purposes of this simulation we assume that the orbits generated by SGP4 are 100% accurate and represent a true measurement.

Ideally, a sensor track at a given time would consist of several individual observations, which would then undergo differential correction to correct the TLE to reflect the new information provided by the track. For our purposes, since we were not actually updating TLEs, we just needed to calculate the covariance matrix for each measurement, which we did using particle techniques. An observation at time t consists of a range, azimuth, and elevation value for radar sensors, or right ascension and declination for optical sensors. The assumed true values of the observation, as calculated by the look angle calculations and SGP4, are perturbed by Gaussian error added in accordance with the published sensor errors from Fundamentals of Astrodynamics and Applications[2].

Once these error-added observations are calculated, the Herrick-Gibbs method of orbit determination is used to produce a set of position and velocity vectors at time t. The covariance of the sample is found through the standard covariance calculation for each element of position and velocity, producing a 6x6 matrix from our set of 6-D vectors:

$$\mathbf{Q_n} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})(x_i - \overline{x})^T.$$

A program was written that generates visibility windows and look angles for a given satellite. First, the satellite's assumed position is calculated over the course of the time window specified, at 1 minute time steps. This produces a long array of position vectors generated with SGP4; all in Earth-Centered Inertial (ECI) coordinates. Then, for each sensor at each time interval, the sensor's position is calculated in ECI coordinates from its latitude, longitude, altitude, and the time. Next, the satellite's position is transformed to topocentric-horizon coordinates, with the sensor as the origin (see http://celestrak.com/columns/v02n02/). From here, range is $\sqrt{[r_S^2 + r_E^2 + r_Z^2]}$, elevation is $\sin^{-1}(r_Z / range)$, and azimuth is $\tan^{-1}(-r_E / r_S)$.

5. PROTYPE SCHEDULER

We have initially concentrated on the covariance reduction issue, since this is particular to the space object tracking scheduling domain, and left other aspects of the scheduler unsophisticated. The prototype scheduler explicitly attempted to achieve approximately orthogonal measurements as its primary goal so it could be determined how much improvement in the covariances was possible.

The Scheduler uses the visibility windows to compute a day's schedule. Two main considerations currently exist that determine how satellites are scheduled. First, the prototype tries to schedule each satellite once per day (following the lead of reference 1), giving preference to satellites that have not been scheduled recently. Second, it attempts to capture different parts of the satellite's orbit each time a track is scheduled. It generates preferred optical scheduling times for satellites, each based on the last successful observation. The Scheduler is looking for a different part of the orbit: 90 or 270 degrees away from the last observed mean anomaly, an orbital parameter that corresponds to the satellite's position in its orbit. Due to a lack of optical resources relative to the demand, once they fill up, radar is used as an alternative.

Prototype Scheduler Algorithm:

* 24 hours is scheduled at a time, based on last successful sensor observation.
* Take in a set of visibility windows: satellite, sensor, time, duration, and last anomalies
* Goal: schedule different parts of the orbit while not starving any satellites
* The list of satellites to schedule is sorted by longest time since last observation
* Generate ideal optical window for each satellite
* Ideal optical window is at 90 or 270 degrees from last mean anomaly, in order to get observations at different parts of the orbit
* Attempt to schedule in order
* When there are conflicts, expand the window to 60 to 120 degrees or 240 to 300 degrees.
* If still are conflicted, try to schedule radar 24 hours from the last observation, +- 6 hours
* If still conflicted, try to schedule radar 24 hours from the last observation, +- 12 hours
* Once all satellites have attempted scheduling, write the schedule, go to the next day, read the schedule, repeat.

6. CALCULATING RESULTS

After each day, a schedule is written. The simulation was conducted over 3, 10, and 30 days with little difference noted in the results for 10 and 30 days, so most of the analysis was done with 10 days worth of data. The schedule file consists of a series of observation tuples: satellite, sensor, and time. We can generate ephemerides for each observation time, using the provided sensor errors in azimuth, elevation, range (radar) and right ascension, declination (optical) to provide a Gaussian distribution around the "true" values as determined by the SGP4 model. Turning these ephemerides into position and velocity vectors, via Herrick-Gibbs (FAA), we can generate a sample covariance matrix for each scheduled observation. We can then combine these observations with the covariance combination formula described by Fig. 5.

One specific issue was introduced by our simulation methodology, which treats the SGP4 model as 100% accurate for propagating orbits infinitely into the future. In reality, there are unmodeled effects and perturbations on the model and, for live satellites, there is potential for maneuvers to occur. Both of these factors increase with the length of time since the last measurement. So whenever a measurement covariance was propagated into the future (typically to combine it with another one), error proportional to the amount of time was added to the covariance matrix being propagated. Failure to add this error results in unrealistically small variances at the end of the simulated period, because every measurement only reduces the previously reduced covariance matrix. We chose to add an

amount of error per day that was approximately equivalent to a pair of good cross-fixed optical measurements. This was done by adjustments in variance of 1.5 km^2 in position and .002 (km/s)^2 in velocity along the diagonal per day. So when a new measurement was made, the old covariance matrix was propagated to the time of the new measurement, and the variance described above was added and then combined with the new covariance matrix according to the following formula:

C1: Adjusted Old Covariance
C2: New Measurement Covariance
CF: New Combined Covariance (i.e. fused covariance)
Cf = C1 * Inverse(C1 + C2) * C2

Chaining these covariances together, we reach a final 3, 10, or 30-day covariance measurement for each satellite.

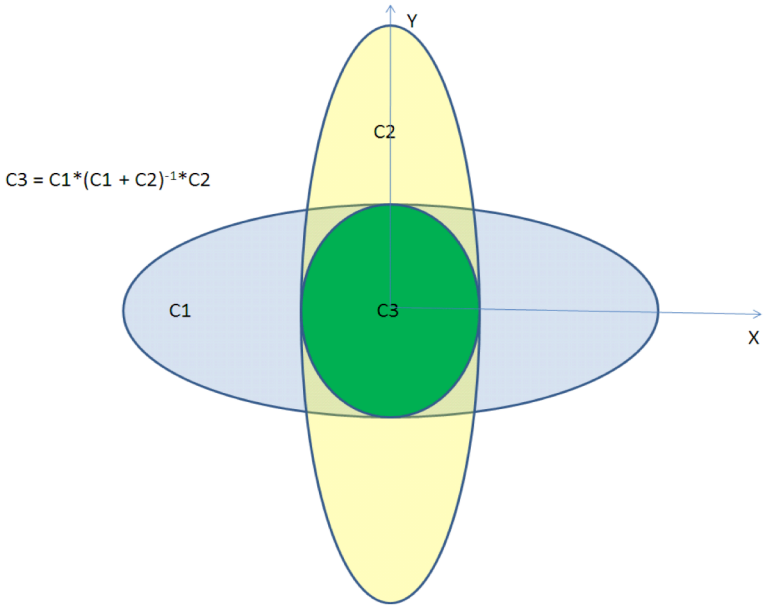

Fig. 5. Covariance combination formula and corresponding sketch.

7. RESULTS

3160 deep space objects from the catalog were tasked and scheduled by the prototype scheduler and compared to a simulation of the current method of scheduling for a 10-day period and the resulting covariances calculated in the position-velocity space at the end of the 10-day period. From these 6 x 6 matrices, the 6 variances were extracted and compared as described by Table 1 below.

Table 1. 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

The first three rows refer to the x, y, and z components of the position variances in kilometer- squared units and the last three rows refer to the x, y, and z components of the velocity variances in kilometer per second squared units. The first row is the average of the variances across all 3160 objects for the 10-day schedule of observations output by the prototype scheduler and the second column is the corresponding average for the schedule produced by the current scheduling method. The third column is the ratio of the second column divided by the first and is a measure of how much better the prototype-produced variances were. The last three columns are similar to the first three but refer to the standard deviations of the variances instead of the average.

Clearly, having the scheduler purposefully schedule complementary measurements does significantly reduce the resulting uncertainty of the objects' locations after a ten-day period, producing variances 3 to 4 times better. The ratio of the standard deviations of the variances is even greater, indicating that the current method variances have a far greater spread, which is very undesirable. These high standard deviations for the current method indicate that there were some very bad measurement errors for some objects. One hypothesis to explain this is that the geosynchronous objects, although a very important part of the deep space regime, might have significantly higher measurement variances than the average because current method would tend to sense these objects in the same parts of their orbits each time. To investigate this hypothesis we ran the same statistics on the 674 GEO objects, resulting in Table 2 below.

Table 2. 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

As hypothesized, the GEO objects had significantly higher average position variances for the current scheduler algorithms, especially in the X and Y dimensions. (The Z dimension is oriented "up" and the GEO objects tend to be near the equator, so large range optical errors from earthbound sensors tend to have a relatively small error in the Z-direction.) Some growth is expected, since angular measurements increase linearly with distance and GEO objects are significantly farther away than the average DS object. This can be seen by the slightly higher mean variances for the prototype. However, the X and Y components of the position variances for the current method double so that the ratio with the prototype is in the 5 to 6 range (or 4 to 5 if the Z component is included.) Even more significantly, the standard deviation of the current method variances is very high, indicating that there are some very large measurement errors, even after 10 days of observations. The prototype's standard deviations are 17 times better for the GEO objects. The GEO belt is an especially important deep space region and covariance- oriented scheduling greatly improves the accuracy of objects' positions and significantly reduces the variability of that accuracy (i.e. greatly reduces the worst-case situations).

These deep space results will not carry over to near-earth objects, where errors are reduced considerably due to their closer proximity and short orbit periods, which tend to give more variability in the position of the orbit where measurements are taken and the different angles between the sensor and object tend to give complementary angles.

8. FUTURE WORK

The main purpose of the prototype was to verify that significant covariance improvement was possible and to somewhat quantify that improvement. However, the scheduling algorithm used was fairly simple in many ways. A full-scale system will use explicit covariance calculations and covariance fusion calculations as part of the scheduling decision process. Specifically, when selecting between competing observation choices, the choice that

reduces the existing covariance the most should tend to be chosen, at least for the deep space objects where covariance is considered very important. Near earth objects may have other metrics considered important such as the number of observations per day or minimizing the time between successive observations. Especially important objects, which may need to be revisited frequently, may also need to focus on optimizing metrics different than covariance. Specific sensors, tailored to specific missions such as small debris sensing, will need to be freed from regular object tracking as much as possible to allow more time to the tasks that only they can perform. Finally, more sophisticated scheduling algorithms will allow more observations, and a more important (from each object's perspective) and better distribution of observations to be taken with the same resources.

Constraints on assigning resources to requests typically involve Line of Sight (LOS) (timing) issues involving orbital mechanics and also generally include a probability of detection calculations involving range, angles, cross section, and sensor sensitivity and precision. It is generally assumed that one set of sensors is used for LEO objects and a separate set is used for DS objects although this is not completely true. Highly Elliptical Orbit (HEO) objects may qualify as DS objects but pass near enough to the Earth at perigee that LEO sensors can track them then. Within each category (LEO, GEO, HEO, and DS) the timing constraints are similar. LEO objects orbit several times a day, have relatively narrow (10-15 minutes) time windows with LOS to specific locations on the ground, and are relatively near to those locations. The set of locations that a LEO object has LOS with is small and rapidly changing. GEO objects orbit once per day, meaning they are essentially fixed in longitude. This means that they have LOS to a large, relatively unchanging fraction of the earth all the time but at a long range. Non-GEO DS objects tend to have long time windows of LOS to specific locations and the set of those locations changes relatively slowly over time.

The schedule and full-scale scheduler must be flexible. Various high priority requests, including those related to in-space emergencies, may occur, which might require near-instantaneous changes to the schedule. Sensors may also break down, requiring real-time schedule changes. A very common occurrence is failure to track the requested object. In some cases this might require the scheduling of additional tracking attempts. So the schedule should have some built-in flexibility to handle emergencies and other real-time events. In addition, the scheduler must also be flexible by being able to reschedule based on these real-time events quickly. Being able to respond within the 24-hour execution cycle in order to be responsive implies that scheduling will be done in cycles. A new 24-hour schedule might be generated every few hours or it might be rescheduled immediately in response to high priority events. Either case however implies that during rescheduling, already scheduled tasks will exist and a decision be made to either change them or leave them as-is. The scheduler should also have the long-term flexibility to easily accommodate new resources and new types of resources as these are developed.

9. OVERVIEW OF A SOLUTION

A new generation SSN scheduler must actually consist of a family of interrelated schedulers because different parts of the SSN scheduling problem have different performance measures and goals and different sensors are more schedulable than others. Therefore, different resource and time selection criteria are appropriate for different kinds of observation requests. Scheduling algorithms to optimize LEO frequent revisit sensing should minimize or eliminate the use of sensors better utilized on small debris sensing as well as minimizing redundant observations from other sensors. Scheduling for DS frequent revisit optimization must utilize predictions of surveillance search results and efficiently schedule space-based optical and ground-based radar and optical sensors by considering what parts of the GEO belt are in sunlight and which sensors are in solar exclusion. Scheduling of the remaining LEO objects should attempt to optimize the number of tracks per day for each object. Scheduling the DS objects should optimize some measure of the accuracy of the known position of each object such as the current, combined covariance matrix. This will involve seeking complementary measurements. Scheduling SOI requests should maximize total (complementary) information collected for each object. The value of each type of observation may depend on the attributes of the type of object being identified.

The use of the Aurora intelligent scheduling framework facilitates this family of interrelated schedulers because it allows for mixing and matching of different scheduling algorithms. In particular, different types of requests, can use totally different resource and time window selection criteria and methods. An Aurora-based intelligent SSN scheduling application would have one ordered queue of observation requests (for both orbit metric and SOI) to schedule and one global set of resources to schedule to meet those requests. A preprocessor would initially utilize predictions of which untasked sensors would likely track which objects (so that this information can be factored in to how many and which observations each object actually needs). The preprocessor considers both these predicted

future observations and recent previous successful observations to determine a preferred (complementary) observation set for each object. Based on this, the preprocessor, using the bottleneck calculation described further below, calculates the resources and times of most contention and the requests that most need those resources at those times. At the same time, it also roughly calculates, required global resource utilization and the utilization for different types of resources and tasks. This gives an initial prediction of the fraction of requests that are satisfiable. (E.g., if the required utilization of sensor resources is 120%, around 20% of the requests can't be met). This information is used to reduce the number of requests for the types, which will definitely not be met. This will be done differently based on the type of object (DS/LEO, FR). When reducing the number of requests for an object, its priority, recent set of successful observations (and current SOI and metric covariance states), and specific resources it could utilize are considered. Obviously, there is no reason to reduce requests for resources that are not over contended for. Other considerations are to prevent starvation and preserve requests for higher priority objects and those with currently poor (e.g., old) information.

As mentioned previously, many SSN sensors can only be tasked (i.e., told a desired number of observations and priority for each object) and not scheduled (i.e., told to observe a specific object at a specific time) and do make the specific scheduling decisions themselves locally. Some degree of scheduling control can be executed over these sensors, especially for the LEO case, which is the great majority of them. Through the following scheme, the global scheduler can either dictate a single observation opportunity to the sensor or a single specific set of adjacent opportunities (e.g., the 3 opportunities in a row constituting the 4^{th}, 5^{th}, and 6^{th} opportunities of the day. The scheme is relatively simple. The sensor's 24-hour day is divided up into time periods corresponding to the shortest LEO orbital period, about 90 minutes. The first time period is designated for the objects sent to the sensor with top priority (which would not necessarily correspond to their actual priorities as input by the users). The second time period would be for all top priority objects requiring 2 observations and second priority objects. The third time period would be for all top priority objects requiring 3 observations, all second priority objects requiring 2 observations, and all third priority objects. The fourth and subsequent time periods and priorities are defined similarly. To schedule an object, for example, at the fifth time period, simple give it a priority of 5 and a number of observations of 1. Alternatively, to schedule that object in each of the consecutive opportunities, starting in the fifth time period, give it a number of observations of 3. For this scheme to work, the global scheduler does need to fill each time period with observations.

Since, like most NP complete problems, generating a schedule is much harder and more time-consuming than testing or grading one, it would be possible for the scheduler to include several different scheduling algorithms, all of which were independently generating schedules. Relatively simple and quickly executing grading software could then evaluate each schedule and select the best one. One algorithm might be the bottleneck avoidance algorithm (including swapping out earlier processed, lower priority tasks), described later. In situations where it is possible to service all tasks with proper consideration of resource contention, the best solution might be from the bottleneck avoidance algorithms. Nevertheless, there may also be some situations when all requests cannot be met in which another algorithm might perform better.

10. SYSTEM DESCRIPTION

The High Level Architecture for each of the specific schedulers is shown in Fig. 6 below. The modules can be assembled in an existing intelligent scheduling system architecture, which provides for customization of each decision point in the scheduling processing. For example, it handles resource usage profiles and visibility requirements, provides for pluggable resource/time window selection methods and satisfaction of temporal, spatial, and arbitrary constraints, and includes the notion of scheduling cycles, an evolving schedule, and the need for a separate preprocessing module.

An SSN application desiring scheduling provides an eXtensible Markup Language (XML) (or other agreed on format) description of the scheduling problem including a description of the tasks to be scheduled, the resources available and the constraints. One type of common constraint for SSN applications is visibility. Normally these are calculated and stored in a separate file that is merely referenced in the XML description of the scheduling problem. However, mere transmittal of the task to the scheduler does not ensure that the required resources can be found to execute it. Whether enough resources exist to schedule all requested tasks is based on the number of resources available (and the optimality of the scheduling algorithm as shown in the earlier simple scheduling example). Associated with each task are its resource requirements and temporal, spatial, and other constraints. A typical

temporal constraint, for example, is that one particular task must be complete before another can begin (such as sensor configuration, mode selection, slewing, and other setup occurring before the sensor event can execute). Another example is that a specific task must be completed by a certain time (e.g., a specific tracking task must be completed by midnight). Constraints may exist associated with tasks, resources or both. For example, a constraint may state that a particular task must use a specific sensor at a specific location.

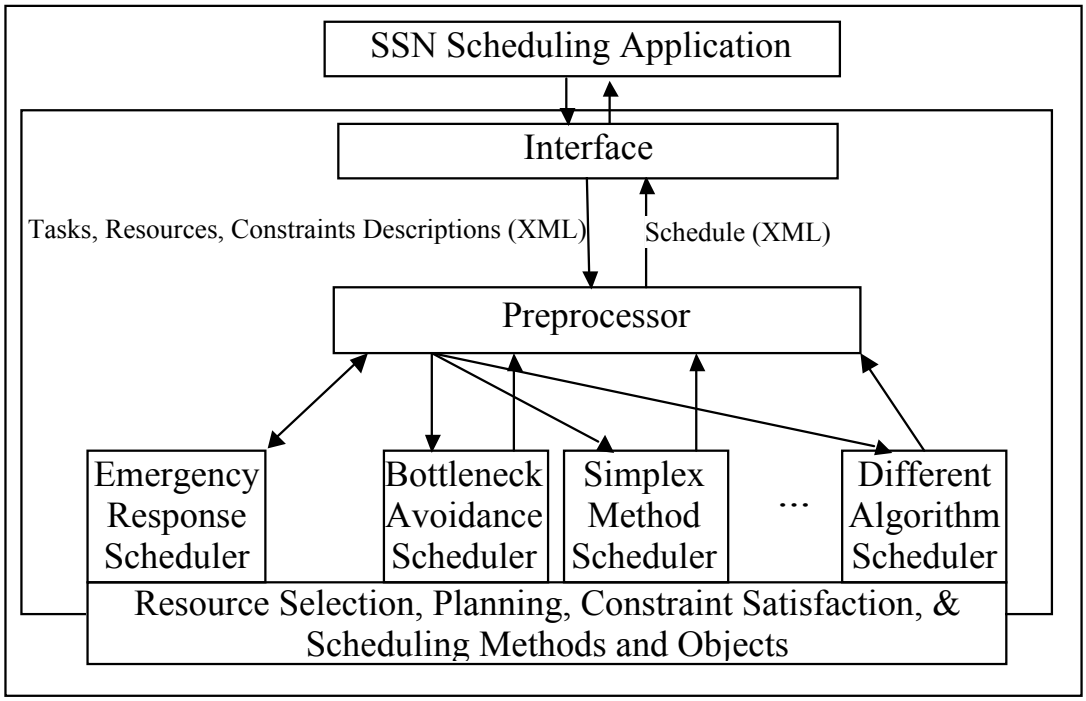

Fig. 6. High Level Scheduler Architecture.

The roles of the preprocessor in addition to those described above are to call each applicable scheduler, grade each returned schedule, and select the best schedule. Emergency requests requiring attention immediately would be passed to the Emergency Response Scheduler for immediate scheduling, so that the appropriate tasks can be passed back and executed immediately and the rest of the near-term, already-published schedule minimally disturbed. Otherwise, or for the remaining requests, the Preprocessor selects one or more appropriate Subschedulers. These Subschedulers will all be independent and therefore would easily fit into a distributed architecture with each Subscheduler having its own computing resources, if required.

The grading scheme for comparing two schedules can be more or less complex and does not have to be uniform across the space catalog objects. We anticipate that our metrics for measuring the quality of produced schedules will be different for different areas of space and different types of requests. For DS object metric requests, some measure of the accuracy of the known position of the object such as the current, combined covariance matrix would be used. Formulas or software for this calculation provided by the existing SSN scheduling and/or covariance calculation community could be used. For LEO objects, simply the number of tracks per day for each object could be used. There are empirical tables that link this simple metric to historically measured errors in tracked objects. For FR objects, the maximum and average time between observations would be used. SOI requests would have their own metrics appropriate for them.

Each Subscheduler conceptually takes as input the tasks, resources and constraints and produces, as output, the assignments to the tasks of resources and time windows that meet the constraints. Each task description includes what resources it requires and for how long and the constraints that it is involved in. In practice, most of the resources needed by tasks will have been predefined with only updates as to their availability being passed in (e.g., radar failure) at scheduling time. The Scheduler generally maintains its own record of the availability of each resource based on the tasks and the time windows currently assigned or tentatively assigned to them. The constraints may be hard (an absolute requirement) or soft (satisfaction is desired but not absolutely necessary) and may be temporal, resource-oriented, or of miscellaneous type.

Each Subscheduler has several decision points, each of which our generic intelligent scheduling architecture allows to be customized through selection of existing options or plugging in new software. One major set of decisions is what the scheduler should process and in what order to process. One common solution is to process tasks in due date or priority order. Another possibility is to process resources in the order in which they become available. Still another is to process constraints, the tightest ones first. We expect to process tracking requests, sorted so that the tasks that request the most constrained resources at the most constrained times are scheduled first. It is extremely important to note that the order in which processing will occur has a very large effect on the quality of the solution and that simple priority or time ordering schemes will often produce very poor results in any reasonably complex scheduling situation. Additional heuristics are also helpful to include in the ordering decision.

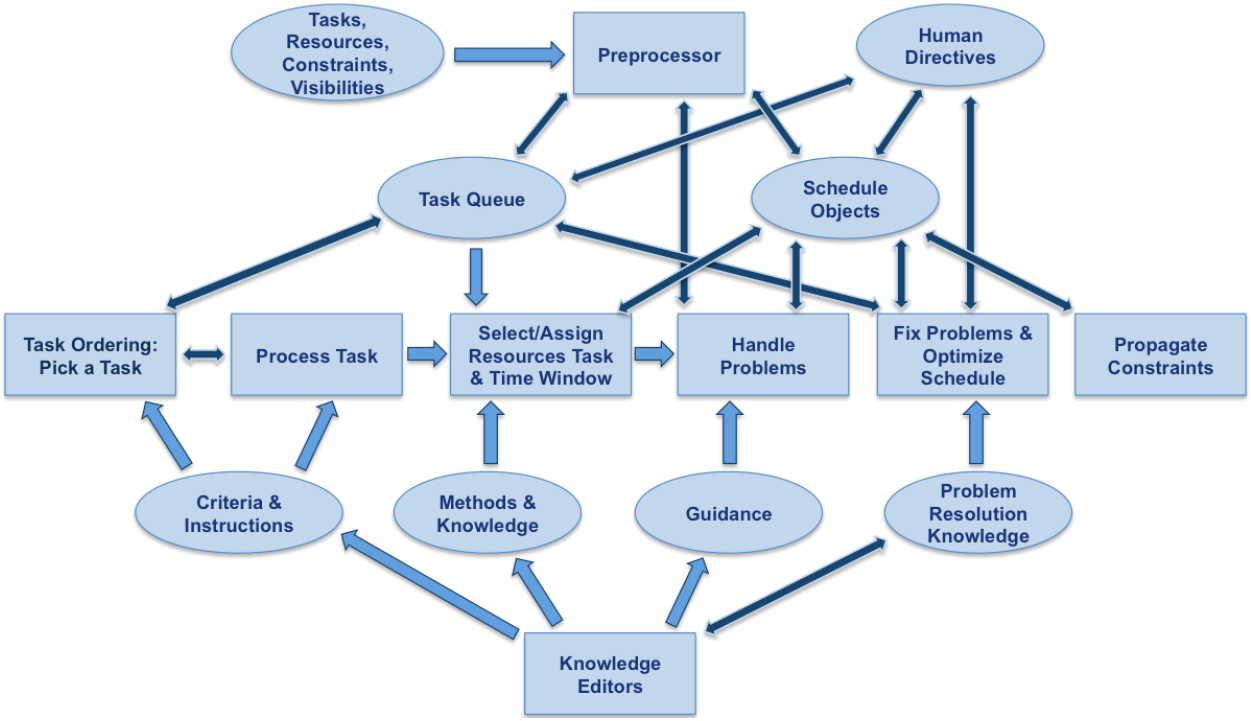

Fig. 7. Internal Scheduler Architecture.

Another major decision point is assigning to a task the needed resources and time windows, while satisfying the applicable constraints. Usually the time window is constrained by the need to get a task done by a certain time while having to wait for events or other tasks to finish before they can start. There may be an additional soft constraint to have the task finish as soon as possible or to start as late as possible. Usually there is at least some freedom in choosing the time window at the same time that the specific needed resources are selected. Choosing the set of resources that are available at the same time that best meet the constraints and that are best for the overall plan can be difficult. "Greedy" systems that choose what is best for the specific task or resource often produce poor results, overall. Some systems use a resource balancing approach whereby the least utilized resource is chosen, under the assumption that if a later-processed task needs a specific resource instance, all of them will still be available; but this can also run into problems, depending on the processing order, especially in cases like the SSN where resources are always tight. A better approach is to choose the resource that other tasks are least likely to need and at the time at which they are least likely to need them. This is the approach in the algorithm described below.

The processing queue is reordered dynamically during the scheduling process and is largely based on contention modified by several factors. Contention is calculated by first "probabilistically" allocating each request across the possible sensors and time windows (based on LOS calculations), weighted by the quality, complementariness, and success probability of the observation and then for each sensor, summing up all possible "probabilistic" allocations. As specific scheduling decisions are made these bottleneck calculations have to be updated for the object just scheduled. These updates have to take into account that 1) the specific request has now been 100% scheduled at a specific sensor and a specific time (and therefore is not "partially" scheduled anywhere else) and 2) the

complementariness of other possible observations are changed to reflect the data that will likely be gathered by this observation. The bottleneck calculation is not just used for the order of processing but is also used to select specific resources and times of observations to avoid the bottleneck regions (while still balancing the need for quality, complementary, probable observations).

11. CONCLUSION

Space object tracking is an important function for maintaining the safety of manned and unmanned spacecraft. The current method of scheduling observations of space objects is an enormous improvement over the way it had been done previously but is hampered by the fact that many of the existing sensors must perform their own scheduling without regard to what other observations other sensors are performing. This prevents the observation schedule from being globally optimized. However, the results of this paper show that infrastructure changes to allow for globally optimizing the schedule would have significant benefit in terms of greatly reducing the error in position with which a deep space object is known. This reduction is especially great in the very important geosynchronous belt.

12. REFERENCES

[1] Miller, J., "Covariance Analysis for Deep-Space Satellites with Radar and Optical Tracking Data,", *Proceedings of the 2005 AAS/AIAA Astrodynamics Specialists Conference*, AIAA, San Diego, CA, 2005.

[2] Vallado, D., *Fundamentals of Astrodynamics and Applications*, 2nd ed., Microcosm, Inc., Hawthorne, CA, 2001.

[3] Vallado, D., Crawford, P., "SPG4 Orbit Determination," *Proceedings of the 2008 AIAA/AAS Astrodynamics Specialists Conference*, AIAA, San Diego, CA, 2008.