NASA/TM—2018-219779

Improvements to Integrated Tradespace Analysis of Communications Architectures (ITACA) Network Loading Analysis Tool

Nathaniel Lee and Bryan W. Welch
Glenn Research Center, Cleveland, Ohio

March 2018

NASA/TM—2018-219779

Improvements to Integrated Tradespace Analysis of Communications Architectures (ITACA) Network Loading Analysis Tool

Nathaniel Lee and Bryan W. Welch
Glenn Research Center, Cleveland, Ohio

National Aeronautics and
Space Administration

Glenn Research Center
Cleveland, Ohio 44135

March 2018

Acknowledgments

The primary author would like to thank his internship mentor, Bryan Welch, for his contributions and support throughout the course of his internship. The primary author would also like to thank the SCaN Center for Engineering, Networking, Integration, and Communications (SCENIC) team and George Bussey for their support of the Integrated Tradespace Analysis of Communications Architectures (ITACA) development efforts.

Level of Review: This material has been technically reviewed by technical management.

Available from

NASA STI Program
Mail Stop 148
NASA Langley Research Center
Hampton, VA 23681-2199

National Technical Information Service
5285 Port Royal Road
Springfield, VA 22161
703-605-6000

This report is available in electronic form at http://www.sti.nasa.gov/ and http://ntrs.nasa.gov/

Improvements to Integrated Tradespace Analysis of Communications Architectures (ITACA) Network Loading Analysis Tool

Nathaniel Lee[*] and Bryan W. Welch
National Aeronautics and Space Administration
Glenn Research Center
Cleveland, Ohio 44135

Abstract

NASA's SCaN Center for Engineering, Networking, Integration, and Communications (SCENIC) project aims to simplify and reduce the cost of space mission planning by replicating the analysis capabilities of commercially licensed software which are integrated with relevant analysis parameters specific to NASA Space Communications and Navigation (SCaN) assets and SCaN supported user missions. SCENIC differs from current tools that perform similar analyses in that it (1) does not require any licensing fees, (2) will provide an all-in-one package for various analysis capabilities that normally requires add-ons or multiple tools to complete. As part of SCENIC's capabilities, the Integrated Tradespace Analysis of Communications Architectures (ITACA) network loading analysis tool will be responsible for assessing the loading on a given network architecture and generating a network service schedule. ITACA will allow users to evaluate the quality of service of a given network architecture and determine whether or not the architecture will satisfy the mission's requirements. ITACA is currently under development, and the following improvements were made during the fall of 2017: optimization of runtime, augmentation of network asset preservice configuration time, augmentation of Brent's method of root finding, augmentation of network asset field-of-view (FOV) restrictions, augmentation of mission lifetimes, and the integration of a SCaN link budget calculation tool. The improvements resulted in (a) 25 percent reduction in runtime, (b) more accurate contact window predictions when compared to Systems Tool Kit (STK®) (Analytical Graphics, Incorporated) contact window predictions, and (c) increased fidelity through the use of specific SCaN asset parameters.

Nomenclature

EIRP	Equivalent Isotropically Radiated Power
FOV	field-of-view
GRC	Glenn Research Center
GSFC	Goddard Space Flight Center
ITACA	Integrated Tradespace Analysis of Communications Architectures
LOS	line of sight
NASA	National Aeronautics and Space Administration
SCENIC	SCaN Center for Engineering, Networking, Integration, and Communications
SCaN	Space Communications and Navigation
STK	Systems Tool Kit
UI	user interface
UTC	Coordinated Universal Time

[*] Fall Intern in Lewis' Educational and Research Collaborative Internship Project (LeRCIP), affiliated with Carthage College.

Introduction

The goal of NASA's SCENIC project is to provide NASA SCaN space mission users with a freeweb-based user interface (UI) to perform on-the-spot mission analyses that are currently performed through expensive and cost-recurring software. SCENIC, in providing a comprehensive UI with advanced analysis capabilities, will save its users both time and money in the early stages of mission planning using NASA's SCaN networks. One of the capabilities SCENIC wants to provide to its users is the ability to evaluate the service of existing and custom SCaN network architectures (a given collection of network assets used to service a mission) for mission satisfaction. An important factor to consider when analyzing the capabilities of a network architecture is network loading. Network loading is a term that refers to the portion of a network's services being used, and the loading will vary with the number of missions being serviced and the requirements of those missions. Overall, network loading is important for creating a network service schedule, which is what the user will consult to determine if an architecture can satisfy the mission's requirements. The tool SCENIC will use to perform loading analysis and create the network service schedule is known as the Integrated Tradespace Analysis of Communications Architectures (ITACA).

The objective of the internship was to develop and implement improvements to the MATLAB®-(The MathWorks, Inc.) based ITACA network loading analysis tool. By factoring in mission requirements, network asset information, and the requirements of other active missions during the time frame of interest, ITACA will create an optimized schedule using real-world operational parameters that will provide users with information detailing which network assets are available to service their missions, and when those assets will be able to establish and end contact (contact window). Additionally, ITACA will provide users with assessment metrics—effectively a quantitative scorecard that allows users to assess the ability of a given SCaN network architecture to provide proper mission service and satisfy the mission requirements.

Throughout the course of the internship, various improvements were made to refine ITACA's capabilities and improve ITACA's efficiency. These improvements include the use of preallocation and other optimization techniques to reduce runtime, the augmentation of network asset preservice configuration time and field-of-view restrictions, and the replacement of the secant method of root finding with Brent's method of root finding for more accuracy in contact window determination. Additionally, a SCaN link budget tool was integrated into ITACA to support SCENIC's building block development approach.

Methods and Results

The following section discusses the details of the implementation of each improvement to ITACA.

Runtime Optimization

The first task of the internship called for a code merge between two versions of ITACA that were developed independently by two summer 2017 interns. One version, developed by an intern at NASA Goddard Space Flight Center (GSFC), added new functionality to the tool. The other version, developed by an intern at NASA Glenn Research Center (GRC), was an optimized version of the original tool. Both interns began the summer with identical versions of the tool; therefore, a merge was needed to apply the optimized runtime techniques to the added functionality. The result would be a more functional and efficient version of ITACA.

A significant portion of the runtime optimization came through the application of preallocation techniques. Preallocation reduces the runtime by programming the computer to avoid dynamically resizing memory allocation every time a new entry is added to an array. For smaller arrays and data structures, the runtime effects of preallocation are negligible. Other runtime optimization techniques that were used included the use of logically indexed variables and refining portions of the code to make processes more efficient. Leveraging logically indexed variables reduces the runtime by avoiding the use of search algorithms. Instead, MATLAB® has the functionality of being able to return a matrix of boolean (logical) variables that indicate whether or not a conditional statement is true. The refinement of the code consisted of tasks such as removing unnecessary sections and variables, and rewriting sections to decrease runtime.

Overall, the runtime was reduced by about 25 percent per analysis on average. Originally, the tool would take approximately 65 to 70 sec to run an analysis for one day in a given year. After the optimization techniques were implemented, the tool took under 50 sec to run an analysis for the same timeframe. These metrics were recorded using the same PC platform; however, the actual runtime of ITACA may vary depending on the capabilities of the machine.

Preservice Configuration Time Implementation

When determining the contact windows between a network asset and a spacecraft, it’s important to account for the amount of time it takes the asset to prepare for RF configuration and gimbal pointing. Because different missions require different service specifications (i.e., frequency bands, modulation schemes, coding schemes), a network asset will need time to reconfigure and/or reorient in order to establish contact with a spacecraft. This preservice configuration time, also known as rest length, must be considered because it affects the contact times and durations. Given any line-of-sight access window, if an asset is not configured properly to service a spacecraft, the preservice configuration time will deduct from the total length of the access window, thereby also affecting the actual contact start and end times.

Previously, ITACA used a single hard-coded value of 5 min for the preservice configuration time of all SCaN network assets. However, 5 min is not an accurate time for the majority of network assets. In order to improve the accuracy of ITACA’s calculations, a method was implemented to allow the use of the correct preservice configuration time corresponding to each network asset.

Currently, preservice configuration times range from about 10 sec to 1 hr across all SCaN network assets, and the individual preservice configuration times vary from asset to asset. As a result, the implementation matches a given network asset with its corresponding preservice configuration time. To implement this functionality, the preservice configuration time of each network asset was added to ITACA’s reference database. When performing an analysis, ITACA uses a search algorithm to extract a network asset and its corresponding preservice configuration time from the reference database. Following this extraction, the preservice configuration time is factored into an optimization algorithm that accounts for all of the available network assets for the given mission. Using the preservice configuration times and other factors, the algorithm optimizes the network service schedule.

Brent’s Method of Root Finding Implementation

The contact times between a network asset and a spacecraft are found through the use of an interpolating root finding algorithm. A root can be defined mathematically as the value of the independent variable when the value of the dependent variable is zero. For computational purposes, the dependent variable may not be exactly zero, but within some predefined tolerance of zero. The concept behind the use of a root finding algorithm is that given a discrete set of line of sight (LOS) angles between a mission and network asset, by offsetting the LOS angles according to the horizon of a network asset, the roots of the function represent the times where the spacecraft enters and exits the line-of-sight access window. Moreover, given that the elevation data is discrete, the roots must be interpolated, rather than extrapolated. ITACA, by deploying an interpolating root-finding algorithm, can determine the start and end times of a contact window to within seconds when the discrete time simulation steps are on the order of minutes.

Previously, ITACA deployed the secant method of root finding to determine the contact times. The secant method finds roots by iterating through a series of secant lines constructed through two points on the function with opposing signs. The method starts by defining an interval inclusively between two points on the function of opposite sign (Ref. 4). For the secant method to work properly, the two elected points must be reasonably close to the expected root; *reasonably close*, however, is not well-defined. If the start and end points of the selected interval are not reasonably close to the root, the algorithm may not converge. A secant line is drawn between the two points, and the function is evaluated at the root of the secant line (interpolation). This completes the first iteration of the secant method. For the second iteration, the newly interpolated point becomes the “next point,” and a secant line is drawn from the second point of the last iteration (in this case, the point at the end of the interval) to this next point. Once again, the root of the secant line is found and the function is evaluated at the root. This sets up the third iteration of the secant method. This iterative process continues until an interpolated point has a value within the predetermined tolerance of zero. In this fashion, the method takes advantage of the intermediate value theorem.

Two issues with the application of the secant method were discovered, however. The first issue is that for some orbits, specifically when the rotation rate of the Earth exceeds the orbital speed of a spacecraft, the secant method does not converge. The second issue is that for many cases, the secant method would converge incorrectly. In other words, for some cases the contact times output by the secant method were not accurate when compared to STK® contact times. On average, there were large deviations in one or two of the contact times for one out of every 10 missions, and the deviations would typically be on the order of minutes. STK® is the standard commercially licensed software used to perform orbit propagation and calculate access windows between spacecraft and/or ground stations. Therefore, STK® presents a viable verification tool against our final implementation.

An illustration of the secant method’s improper convergence can be seen in Figure 1, as it shows how the secant method iterates step-by-step from boxes 1 to 8. The interval of the function illustrated has two roots. The first root occurs on the left side of the interval and can be seen by the blue circle just below the red circle labeled y1 in box 1. The root that the method is attempting to find, however, is the root on the right side of the interval which should occur above the blue circle labeled y2 in box 1. Note that in Figure 1, the axis is represented by the green dashed line, the existing function is represented by the solid black lines, and the interpolated function is represented by the black dashed line. The next paragraph will walk through each box, explaining the iteration steps of the secant method.

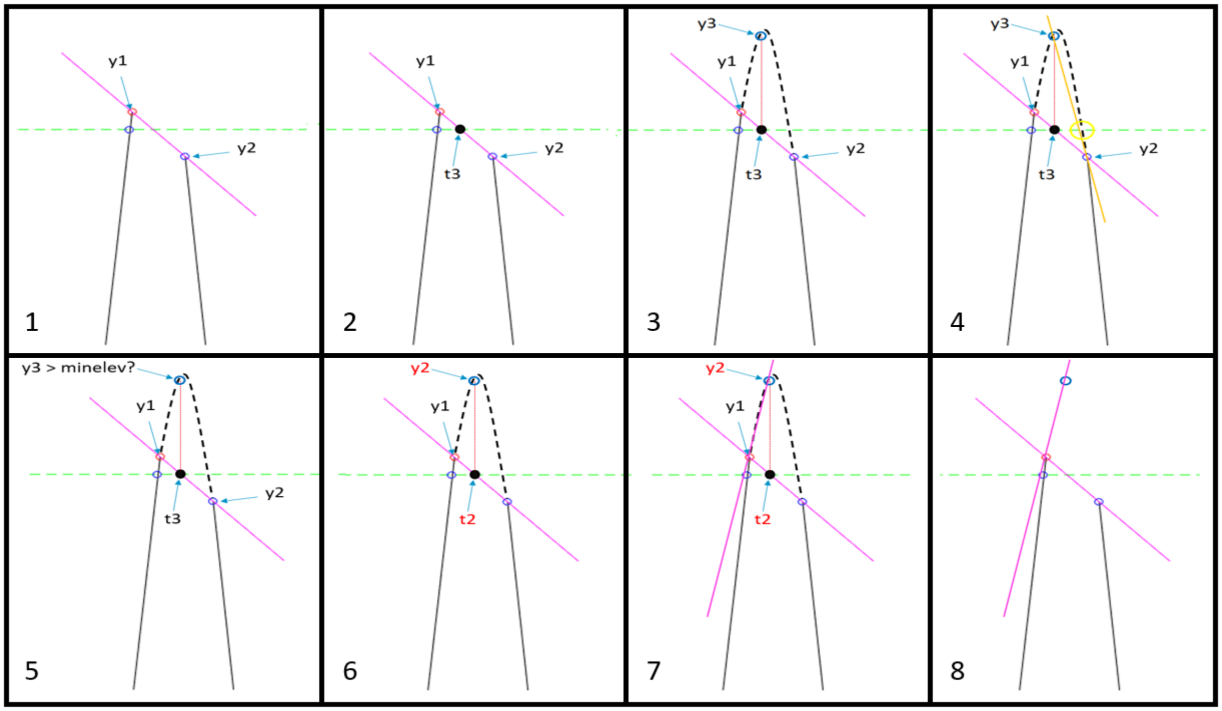

Figure 1.—An illustration of the first and second iterations of the secant method used in ITACA. The example is taken from a case used when the network asset is a ground station. The first iteration is represented in box 1, and the second iteration is represented in box 8, with boxes 2 to 7 illustrating the construction of the second iteration starting from the first iteration.

Figure 1 box 1 shows the secant line of the first iteration of the secant method for the root occurring on the right side of the interval. The points y1 and y2 are the two points defining the interval, so the secant line is drawn between them. In box 2, we can see where the root of the secant line occurs, which is labeled t3. The value of the function is then evaluated at t3 (interpolation), and that point is labeled y3 in box 3. Normally, the secant method would then make y2 the first point of the next iteration (the new y1), and y3 would become the second point of the next iteration (the new y2), so the next secant line would be the one represented in orange in box 4 and the method would converge properly. However, the secant method originally used in ITACA had a slight modification. As illustrated in box 5, the algorithm determines whether or not y3 is greater than the minimum elevation angle, which represents the field-of-view limit of a ground station. If y3 is greater than the minimum elevation, y3 still becomes the new y2, but the current y2 is ignored and the current y1 value remains the y1 value for the next iteration. As a result, the next secant line drawn is illustrated in boxes 7 and 8. As the secant method continues to iterate, it can be observed from Figure 2 that the root will converge incorrectly to the root occurring on the left side of the interval.

To avoid cases of divergence or improper convergence, it was proposed that the secant method be replaced with a more robust method that is guaranteed to converge: Brent's method. Brent's method is a hybrid method combining the method of inverse quadratic interpolation, the secant method, and the bisection method (Ref. 1). The advantage of Brent's method is that in combining three different methods, the strengths of each method can be leveraged during the root finding process. The main difference between the secant and inverse quadratic interpolation methods and the bisection method, is that the latter is considered a bracketed method, which means the interval of interpolation is constrained, guaranteeing convergence. The former two methods are considered open methods, meaning they have no interval boundaries for iteration. Consequently, both methods may stray away from the area of the root and either improperly converge or diverge. This is why the bounds of the initial interval for the secant method must be reasonably close to the root. The advantage of open methods, however, is that they converge more quickly than bracketed methods for most cases (Ref. 4). The speed of a method is determined by the number of iterations it takes for a method to find a root. Leveraging the advantages of both types of methods, Brent's method prefers to use either quadratic interpolation or the secant method for speed, but may defer to using the bisection method if necessary. The way Brent's method achieves this objective is by evaluating its performance at each iteration. Based on this evaluation, the method will decide whether to use inverse quadratic interpolation, the secant method, or the bisection method for the next iteration.

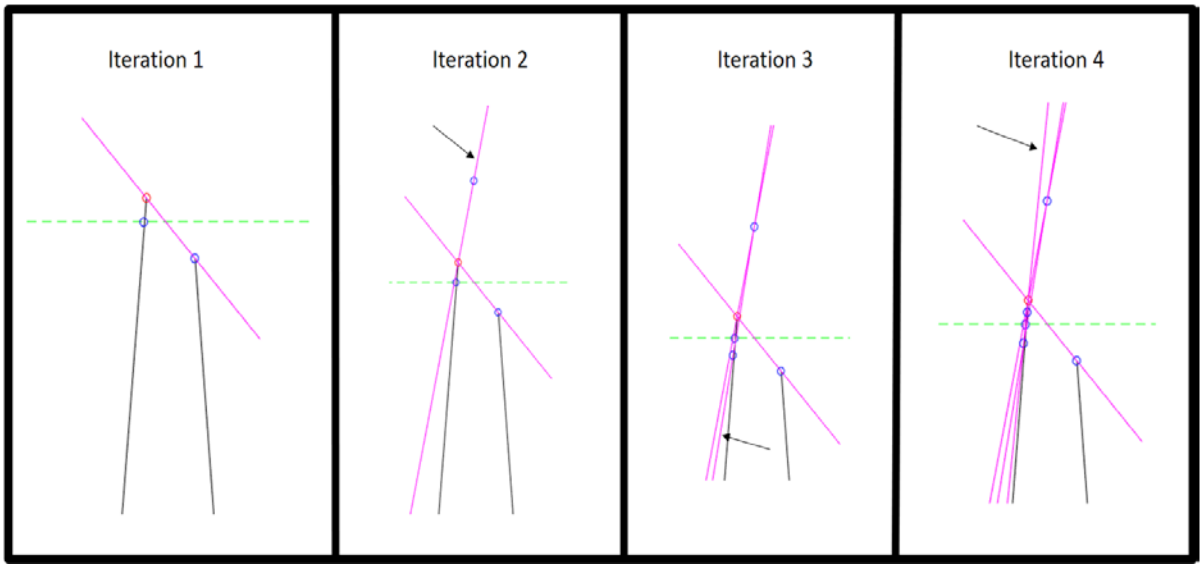

Figure 2.—An illustration of the first four iterations of the secant method. It can be seen that the method will converge to the wrong root.

The implementation of Brent's method in ITACA is an adaptation from the original method presented by R.P. Brent in 1970 (Ref. 1). After replacing the secant method with Brent's method, inaccuracies in the contact windows were resolved. Table 1 shows a collection of contact windows that were compared between STK®, Brent's method, and the secant method for the test case that was illustrated in Figure 1 and Figure 2. Table 2 also shows a comparison of contact windows between STK®, Brent's method, and the secant method, but for a different test case. The characterization of each orbit, timeframe, and ground station location can be found below:

Both test cases were run for the same timeframe and used the same ground station location:

- Timeframe: June 18, 2016 12:00:00 to June 19, 2016 12:00:00 Coordinated Universal Time (UTC)
- Ground station latitude: 1.132 rad
- Ground station longitude: –2.5806 rad
- Ground station altitude: 0.3311 km

The first test case (Table 1) had the following Keplerian orbital elements:

- Semi-major axis: 7166.5 km
- Eccentricity: 0
- Inclination: 1.7016 rad
- Argument of perigee: 4.4927 rad
- Right ascension of the ascending node: 4.8949 rad
- True anomaly: 2.8391 rad

The second test case (Table 2) had the following Keplerian orbital elements:

- Semi-major axis: 6914.3 km
- Eccentricity: 0
- Inclination: 1.7017 rad
- Argument of perigee: 2.7651 rad
- Right ascension of the ascending node: 1.7656 rad
- True anomaly: 1.6561 rad

TABLE 1.—A TABLE COMPARING CONTACT WINDOWS OUTPUT BY STK®, BRENT'S METHOD, AND THE SECANT METHOD FOR A GROUND STATION ACCESS TEST CASE

Ground station contact window comparison case 1					
STK®		ITACA—Brent's method		ITACA—Secant method	
Start time, UTC	Stop time, UTC	Start time, UTC	Stop time, UTC	Start time, UTC	Stop time, UTC
00:08:33	00:21:07	00:08:34	00:21:08	00:08:34	00:21:08
01:48:12	01:59:17	01:48:14	01:59:16	01:48:14	01:59:16
03:27:26	03:36:17	03:27:28	03:36:14	03:27:28	03:36:14
05:05:28	05:13:36	05:05:31	05:13:33	05:05:31	05:13:33
06:42:25	06:52:18	06:42:26	06:52:15	06:42:26	06:52:15
08:19:49	08:31:48	08:19:49	08:31:46	08:19:49	08:31:46
09:58:52	10:11:29	09:58:52	10:11:29	09:58:52	10:11:29
11:40:23	11:50:57	11:40:24	11:50:57	11:40:24	11:50:57
20:19:48	20:24:31	20:19:44	20:24:38	20:19:44	20:19:44
21:58:02	22:09:18	21:58:03	22:09:21	21:58:03	22:09:21
23:37:36	23:50:16	23:37:37	23:50:16	23:37:37	23:50:16

TABLE 2.—A TABLE COMPARING CONTACT WINDOWS OUTPUT BY STK®, BRENT'S METHOD, AND THE SECANT METHOD FOR A GROUND STATION ACCESS TEST CASE
[This test case differs from that shown in Table 1.]

Ground station contact window comparison case 2					
STK®		ITACA—Brent's method		ITACA—Secant method	
Start time, UTC	Stop time, UTC	Start time, UTC	Stop time, UTC	Start time, UTC	Stop time, UTC
08:54:45	08:55:41	08:54:53	08:55:35	08:54:53	08:54:53
10:26:41	10:35:28	10:26:42	10:35:28	10:26:42	10:35:28
12:01:03	12:10:46	12:01:03	12:10:46	12:01:03	12:10:46
13:35:43	13:43:57	13:35:43	13:43:57	13:35:43	13:43:57
15:10:25	15:15:40	15:10:23	15:15:39	15:10:23	15:15:39
16:44:21	16:47:13	16:44:18	16:47:14	16:44:18	16:44:18
18:15:58	18:21:06	18:15:55	18:21:06	18:15:55	18:21:06
19:47:38	19:55:47	19:47:36	19:55:47	19:47:36	19:55:47
21:20:46	21:30:28	21:20:44	21:30:27	21:20:44	21:30:27
22:55:59	23:04:50	22:55:58	23:04:48	22:55:58	23:04:48

Observing the contact times in red in Table 1, it can be seen that the secant method's contact window stop time deviates from the contact window stop times of both STK® and Brent's method. Moreover, notice that the secant method's stop time is identical to the start time. This correlates directly with what is illustrated in Figure 1 and Figure 2, where the secant method converges to the wrong root, which, in this case, is a root that was already found. Observing the case for Brent's method, it can be seen that all contact windows are almost identical to those of STK®. The version of STK® used referenced an *obsolete* Earth Orientation dataset, leading to the slight differences between the times.

The discrepancies of the secant method can also be seen in Table 2. For the corresponding times in the red highlighted boxes, the secant method outputs a time that deviates from what STK® and Brent's method output, which also happens to be identical to the start time for that contact window. From the test cases presented along with many others, it was concluded that the secant method was less reliable than Brent's method, prompting the replacement of the secant method with Brent's method.

Field-of-View (FOV) Implementation

Another improvement made to increase the capabilities of ITACA, more specifically the contact window time predictions, was the implementation of variable asset field-of-view restrictions. Previously, ground asset contact windows were calculated with a fixed value for FOV restriction, and space asset contact windows were calculated without any FOV restriction. Consequently, the FOV restriction implementation consisted of two modifications: augmenting variable FOV restrictions for ground assets, and adding FOV restriction functionality with variable FOV capability for space assets. The FOV restriction implementation was applied after replacing the secant method with Brent's method.

The FOV restrictions for each network asset were taken from SCaN asset databases and added into ITACA's reference database. Code was then written to allow ITACA to properly extract the FOV values from the reference database and match the values with the corresponding assets. The process of extracting and matching the FOV restriction with its corresponding network asset was similar to the process used for the preservices configuration times.

FOV is the common term to describe the concept of what is visible to a spacecraft or network asset. However, for a ground station, the FOV restriction is also known as the *minimum elevation*. The minimum elevation refers to the angle above an asset's horizon where the FOV range is defined. Figure 3 illustrates the FOV geometry for a ground asset. It can be seen from Figure 3 that the FOV lines are defined by the minimum elevation angle above the horizon. Figure 4 expands on Figure 3 by showing the area of no contact based on the FOV restriction. Note that the *No Visibility* area in Figure 4 extends over the horizon to the FOV lines. It is for this reason that the FOV is considered a restriction. A larger minimum elevation angle corresponds to a larger FOV restriction.

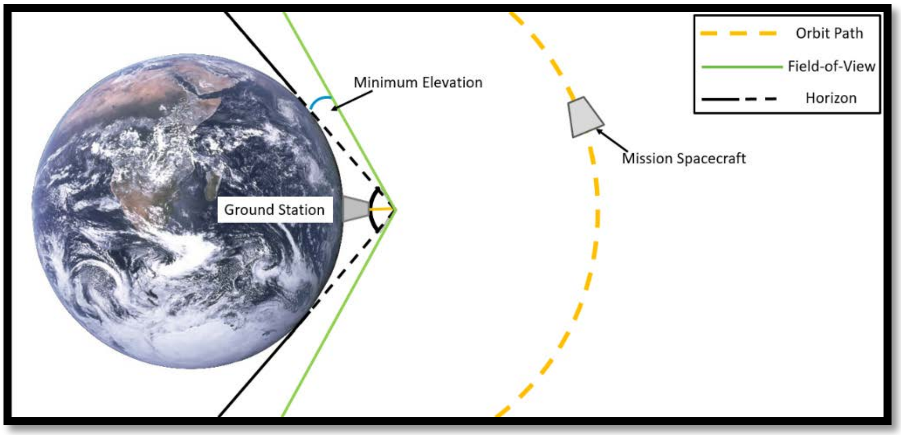

Figure 3.—Illustration of the FOV geometry for a ground asset.

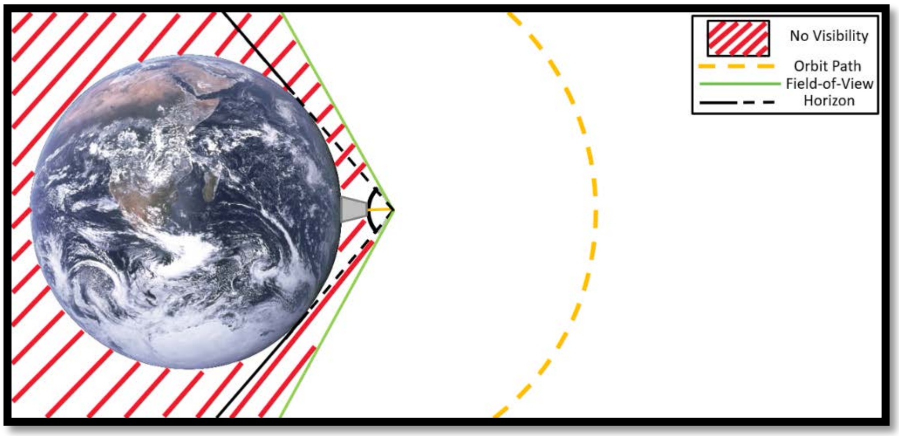

Figure 4.—FOV of a ground asset with area of nonvisibility shown.

Based on the illustration in Figure 4, the start and stop times of contact windows occur where the orbit of the spacecraft crosses the FOV lines of the ground asset (not pictured). In theory, the FOV lines extend infinitely, so that for any orbital altitude, if a spacecraft is above the minimum elevation angle of a ground asset, contact can be established.

For space assets, the FOV geometry is different. An illustration of the FOV geometry for a space asset is shown in Figure 5. Note that there are more angles involved in accurately representing the FOV restriction of a space asset as opposed to a ground asset. For a ground asset, the FOV value ITACA extracts from the reference database is the minimum elevation angle. For a space asset, the FOV value ITACA extracts is the angle denoted *Theta FOV* in Figure 5. Given *Theta FOV*, a minimum elevation angle can be determined in reference to an axis tangential to the orbit of the space asset. This angle is denoted *Minimum Elevation FOV* in Figure 5. In addition to the FOV restriction, the horizon of the asset must also be accounted for. The angles *Theta Horizon* and *Minimum Elevation Horizon*, which define the horizon of the asset are calculated within ITACA (Ref. 5). Together, the horizon and FOV restriction of a network asset are accounted for to determine the contact windows.

Because the FOV geometry for a space asset is more complex than that of a ground asset, there are three different scenarios that must be accounted for when considering how to determine the contact windows. All three scenarios are illustrated in Figure 6, Figure 7, and Figure 9 for multiple spacecraft orbit paths. Note that all three scenarios pertain to a situation where both the mission and the space asset (also known as a space relay) are orbiting the same central body. In this case, both the mission and relay are orbiting the Earth, and the relay in geosynchronous orbit. If the central body of the mission and the relay are different, the geometry will differ from what is pictured in Figure 5.

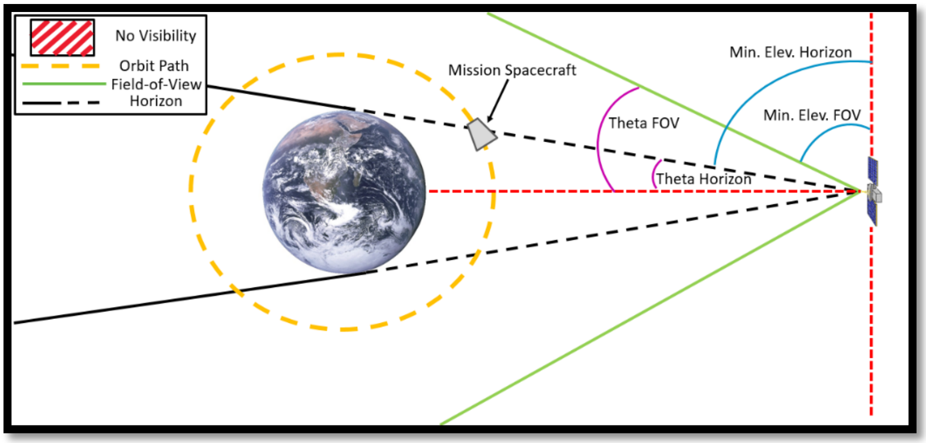

Figure 5.—FOV geometry for a space asset.

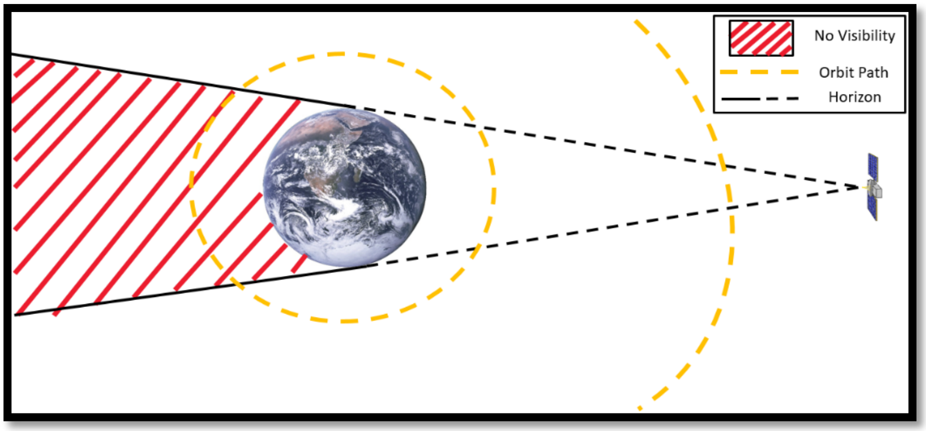

Figure 6.—Unrestricted space asset FOV scenario.

Figure 6 illustrates the scenario where the FOV is unrestricted for a space asset. Notice that for both orbit paths, contact can be established anywhere outside of the *No Visibility* zone (planetary occultation). This is the original scenario ITACA used to determine all contact windows for space assets.

Figure 7 illustrates the case where the FOV is less restricted than the horizon (*Theta FOV* > *Theta Horizon)*. Notice in Figure 7, notice that for the smaller orbit path, the contact windows are only determined by planetary occultation (also known as planetary blockage), meaning there is only one contact window per orbit. However, for the larger orbit path (which is not fully pictured), there are a total of three contact windows per orbit, rather than one. One contact window occurs when the spacecraft leaves the area of planetary occultation and is within the FOV of the asset. This window ends when the spacecraft exits the FOV of the asset. The next contact window begins when the spacecraft reenters the FOV of the asset, and promptly ends when it leaves the FOV of the asset. The last contact window begins when the spacecraft reenters the FOV of the asset and ends when the spacecraft crosses below the horizon back into the planetary occultation *No Visibility* zone. For an illustration of these contact windows, refer to Figure 8.

Figure 8 represents a zoomed out illustration of the scenario in Figure 7. The portions of the orbit paths highlighted in blue are the portions where the contact windows occur. As aforementioned, notice that for the smaller, circular orbit, there is only one contact window. On the other hand, for the larger, elliptical orbit, there are three total contact windows. Both orbit paths must factor in planetary occultation, but the visibility of the elliptical orbit path at a given time is also determined by where the spacecraft is relative to the FOV of the asset.

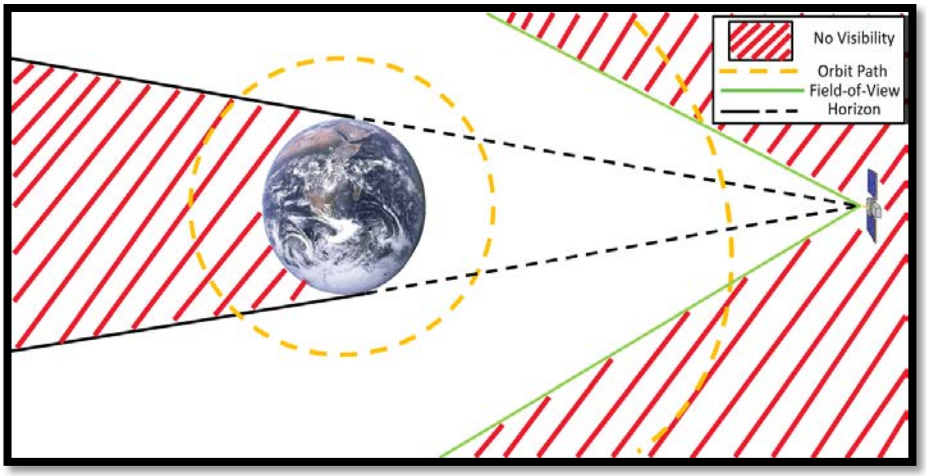

Figure 7.—Space asset FOV less restricted than horizon.

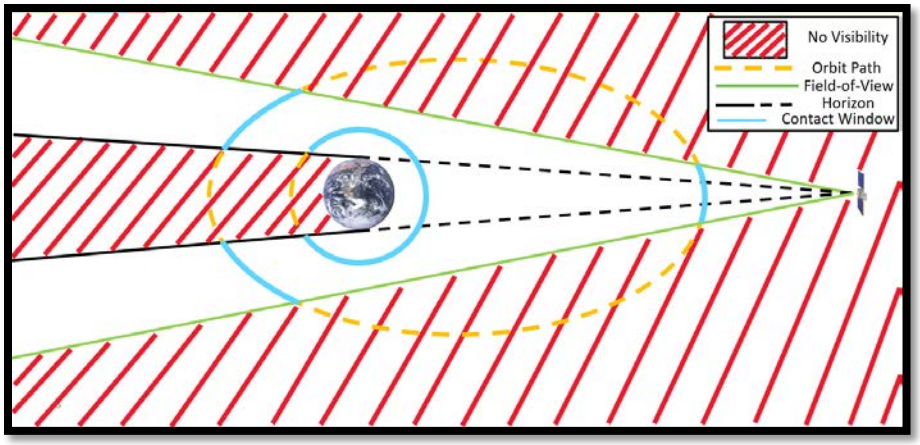

Figure 8.—Space asset FOV less restricted than horizon. The blue portions of the orbit represent contact windows for each orbit path. There is one contact window for the smaller, circular orbit path. There are three contact windows for the larger, elliptical orbit path.

Finally, Figure 9 illustrates the case where the FOV is more restricted than the horizon (*Theta FOV < Theta Horizon)*. In this case, the contact windows for both orbit paths can be determined the same way. Each orbit path will have one contact window per orbit, and the start and end of the window will occur when the spacecraft enters and exits the FOV of the asset, respectively.

In order to integrate all three scenarios into Brent’s method, flags were established to help determine which roots were necessary to accurately determine the contact windows. For the scenario illustrated in Figure 6, only the roots for the horizon correspond to contact window times. For the second scenario illustrated in Figure 7, both the roots of the horizon and the FOV correspond with contact window times. For the last scenario in Figure 9, only the roots of the FOV correspond with contact window times. One subtlety to note is that because the horizon and FOV boundaries extend infinitely outward from the asset, some roots need to be ignored because they do not correspond to contact windows. In the first scenario, the roots that occur on the horizon between the planet and the asset are ignored (roots that occur on the dashed portion of the horizon) because planetary occultation is not a factor in this area of the orbit. The case is likewise for scenario two. For scenario three, only the FOV roots between the planet and the asset correspond to contact window times. These roots are ignored. Figure 10 gives one example of when the roots are ignored to determine the contact windows.

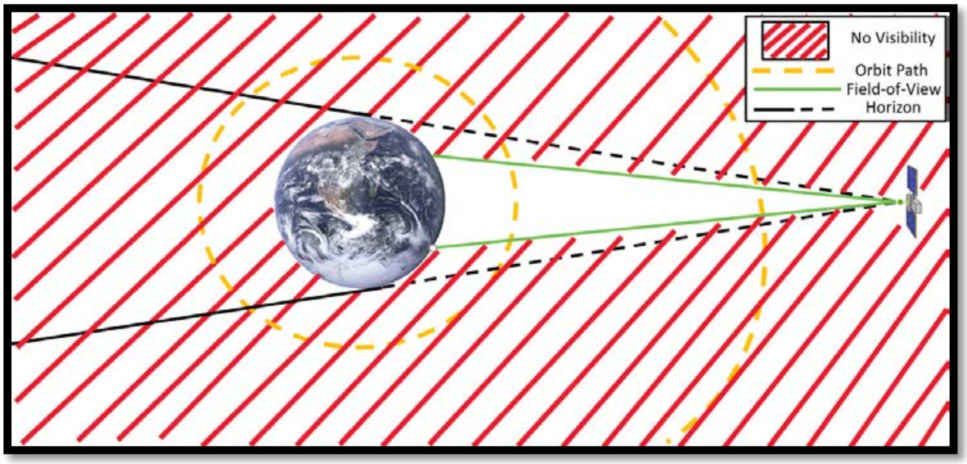

Figure 9.—Space asset FOV more restricted than horizon.

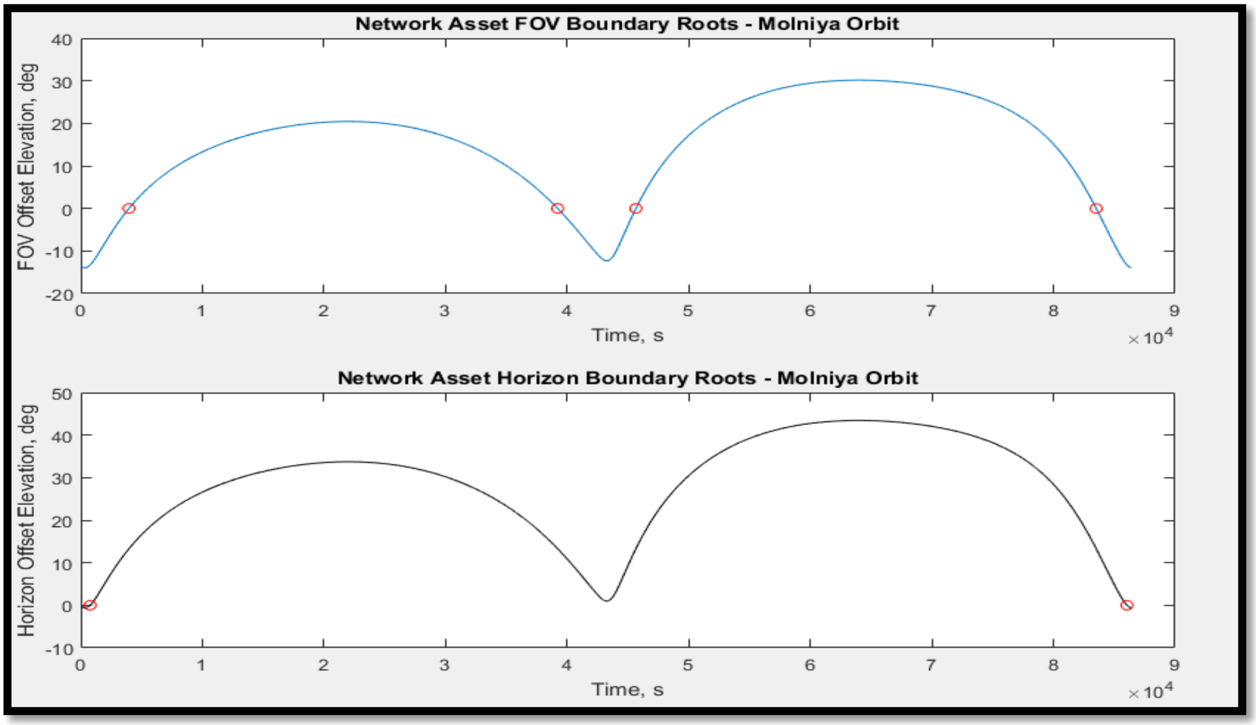

Figure 10.—Molniya orbit contact window root plots. The top plot illustrates the roots defined by the network asset's FOV. The bottom plot illustrates the roots defined by the network asset's horizon. All of the roots are plotted in red.

Figure 10 displays two subplots showing each contact window root found for a scenario where a spacecraft on a Molniya orbit is in contact with a space-based asset. The Molniya orbit is a highly elliptical Earth orbit named after several Soviet communications satellites launched beginning in the 1960s (Ref. 2). The characterization of the Molniya orbit used in Figure 9 is defined by the following Keplerian orbital elements:

- Semi-major axis: 26,554 km
- Eccentricity: 0.72
- Inclination: 1.1065 rad
- Argument of perigee: 4.7124 rad
- Right ascension of the ascending node: 1.9199 rad
- True anomaly: 0 rad

The timeframe and relay location for the scenario are as follows:

- Timeframe: June 18, 2016 12:00:00 to June 19, 2016 12:00:00 UTC
- Space relay latitude: 0 rad
- Space relay longitude: –0.802 rad
- Space relay altitude: 35,786 km

The top subplot contains the spacecraft elevation offset by the minimum elevation angle/horizon angle of the network asset’s FOV (*Minimum Elevation FOV* in Figure 5), while the bottom subplot contains the elevation data offset by the asset’s horizon (*Minimum Elevation Horizon* in Figure 5). Notice also that the time scale of each subplot matches so that the path of the asset can be followed as it corresponds to the case illustrated in Figure 7. Notice that between 40,000 to 50,000 sec, there are roots in the bottom subplot that are ignored. These roots correspond to those that would be found in the dashed portion of the horizon boundaries in Figure 5. As can be seen in Figure 5, the spacecraft is within view of the asset in this area of the orbit. Therefore, the roots can be ignored.

Following the implementation of the asset FOV methods into ITACA, the contact window predictions were compared to those output by STK® for the same scenario. Table 3 shows a comparison table for the contact windows of the Molniya orbit referenced in Figure 10.Table 4 shows another comparison table of contact windows from both ITACA and STK® for a test case with a circular Earth orbit that also exceeds the FOV and horizon boundaries.

Notice in both Table 3 and Table 4 that the contact windows between ITACA and STK® are similar to within seconds. Overall, the similarity between the contact windows from ITACA and STK® shown in Table 3 and Table 4 show that the field-of-view implementation works properly and as expected. More broadly, the comparable contact windows show that ITACA is able to replicate the analysis and scheduling capabilities of commonly used commercially licensed software, which is one of the fundamental objectives of the SCENIC project.

TABLE 3.—COMPARISON TABLE OF CONTACT WINDOWS OUTPUT BY STK® AND ITACA FOLLOWING THE IMPLEMENTATION OF ASSET FOV METHODS FOR THE MOLNIYA ORBIT REFERENCED IN FIGURE 10

Contact window comparison—Molniya orbit			
STK®		ITACA	
Start time, UTC	Stop time, UTC	Start time, UTC	Stop time, UTC
00:12:47	01:06:00	00:12:50	01:06:03
10:53:39	12:40:55	10:53:37	12:40:55
23:11:54	23:54:12	23:11:54	23:54:05

TABLE 4.—COMPARISON TABLE OF CONTACT WINDOWS OUTPUT BY STK® AND ITACA FOLLOWING THE IMPLEMENTATION OF ASSET FOV METHODS FOR A CIRCULAR ORBIT WITH A SEMI-MAJOR AXIS OF 20,000 km (OUTSIDE THE FOV OF THE ASSET)

Contact window comparison—Circular orbit			
STK®		ITACA	
Start time, UTC	Stop time, UTC	Start time, UTC	Stop time, UTC
00:00:00	03:42:18	00:00:00	03:42:18
06:18:57	08:04:38	06:18:51	08:04:35
10:25:14	12:28:11	10:25:12	12:28:09
12:57:26	15:17:35	12:57:40	15:17:35
17:58:15	19:42:14	17:58:22	19:42:18
22:02:23	23:55:53	22:02:27	23:55:30

Mission Lifetime Implementation

The implementation of mission lifetimes into ITACA was done to improve multiyear analysis accuracy. In order to properly create a network service schedule, a survey of active missions is necessary. Active missions compete for SCaN services during their lifetimes, so an accurate survey of active missions is vital to optimizing a service schedule. If ITACA accounts for the requirements of inactive missions when creating the schedule, the schedule may reflect less availability for certain networks or network assets. Ultimately, this would lead to a lower user satisfaction rating. Accounting for the approximate lifetime of a mission will help ITACA estimate the number of active missions per year, which will help give a more realistic user satisfaction rating. Ideally, it would be advantageous to account for each mission's lifetime individually. However, based on data limitations, mission extensions, and other factors that may affect the lifetime of a mission, the lifetime of a mission is estimated based on its *use case*.

Use cases are a collection of categorizations used to group missions together based on common attributes. In total, there are currently 29 use cases (Ref. 3). Missions categorized by the same use cases tend to be very similar in objectives and requirements. Therefore, the assumption that missions grouped by the same use case have similar lifetimes can be made.

To implement mission lifetimes into ITACA's calculations, an approximate lifetime value was assigned to each use case and added into ITACA's reference database. The approximate lifetime values were obtained by averaging the lifetimes of current missions within a given use case. The lifetimes are extracted within the tool and entered into a separate array known as a *life sheet*. This life sheet contains all of the mission lifetimes for each use case, and for every year that ITACA runs an analysis, it references the life sheet to assess whether the lifetime of the current mission has been reached. If the lifetime has been reached, it is removed from the calculation. The opposite is true if the lifetime of the mission has not been reached.

Integration of SCaN Link Nonrepeater Tool V5

Part of SCENIC's development plan is to implement a system of programming that incorporates designing and using different building blocks throughout the SCENIC UI. In other words, SCENIC is making an effort to develop tools and functions that can be used by other tools and functions within SCENIC. These sub-tools have been termed *building blocks*, and the philosophy adheres to the principle that code should be recycled when possible.

Using the building block system makes code revisions simpler because it allows for the simultaneous updates of several pieces of code. For example, there may be several tools within SCENIC that calculate link characteristics for different purposes. In this case, it would be best to use a common link calculator rather than independently implementing a link calculator in each of the tools. This is due to the fact that if updates are made to the link calculator, all of the tools that require the calculator will be updated simultaneously rather than individually.

This example of a link calculator applies directly to one of the tasks of the internship. One of the final improvements made to ITACA was the integration of the SCaN link nonrepeater tool. In a nonrepeater link, there is direct contact between the mission and the network asset. Other scenarios include the use of repeaters, where a repeater acts as an intermediate node in the data transmission process. The *SCaN link tool*, as it's more commonly known, is currently active on the SCENIC UI and can be used to perform link analyses. ITACA, as part of its scheduling capabilities, must factor in the minimum possible data rate for a link between an asset and a spacecraft. Therefore, some method must be used to calculate the minimum data rate, which is a link characteristic. Since the SCaN link tool is capable of calculating data rates, the integration of the SCaN link tool into ITACA was an effort that supported the building block philosophy.

The integration of the SCaN link tool required the restructuring of several portions of the code and the addition of more data into ITACA's reference database. Previously, a streamlined process was used to estimate the data rates for each link. This streamlined process included the use of several simplifying assumptions. Hence, many of the parameters necessary for the SCaN link tool calculations were not available within ITACA. As a result, mission data for Equivalent Isotropically Radiated Power (EIRP), frequency bands, networks, modulation schemes, coding schemes, and science types were all added to the reference database. Similar to the mission lifetimes, it is difficult to account for every mission's EIRP, frequency band, network, modulation scheme, coding scheme, and science type. Therefore, these parameters were grouped by use case.

The SCaN link tool integration had two major benefits. The first is that the data rate calculations within ITACA will be more accurate. Because the previous method used several simplifying assumptions, the data rate values were not always accurate to within an acceptable tolerance. The second advantage of the integration is that the expansion of the reference database led to a decrease in the amount of code it takes to calculate and store the data rates. The portion of code dealing with the data rate calculations was reduced from approximately 250 lines to just under 90 lines. The data rate portion of the code is now more condensed and easier to read.

The last portion of the SCaN link tool integration involved optimizing the integration. The optimization consisted of applying a method to reduce the amount of calculations performed by the SCaN link tool during an analysis. While the input parameters for the SCaN link tool are grouped by use case, it is common for use cases to share parameters. In this case, the runtime can be optimized by preventing the SCaN link tool from repeating calculations using the same parameters. To implement this prevention, code was added to store the parameters of every calculation the SCaN link tool performs. The next time the tool is slated to perform a calculation, the code will check the input parameters and compare them with the calculations already performed. If the parameters match those of a calculation already performed, the tool will assign the output of the original calculation, rather than repeating the calculation.

Overall, the SCaN link tool integration increased the fidelity of ITACA by providing more accurate data rate values, as well as contribute to the building block system supported by SCENIC.

Conclusion

The ITACA network loading analysis tool will be used by SCENIC to provide network architecture assessment capabilities to its users. ITACA creates both an optimized network service schedule and quantitative assessment metrics that allow the user to determine whether or not a certain architecture will satisfy the mission requirements. Both of these features will help users decide if the SCaN program has the necessary capabilities to service their mission(s) properly. Given the scope of ITACA in SCENIC, the improvement efforts of the internship will have a direct impact on the future plans and capabilities of SCENIC. Furthermore, some of the efforts of the internship will be applied to other tools and capabilities within SCENIC. Despite the success of improvements made to ITACA over the course of the internship, further improvements must be made to fully support future SCENIC capabilities.

One improvement that can be made is expanding the FOV implementation to include FOV restrictions of the mission spacecraft. The implementation discussed in this report assumes FOV restrictions of only the network assets, and does not account for any FOV restrictions of the mission. This is a simplifying assumption, as many missions contain antennas that do have an FOV restriction. Implementing this improvement will require the determination of the spacecraft's orientation to assess whether or not the FOV paths from both the asset and the mission coincide.

Another improvement that can be made to ITACA is expanding its capabilities to non-Earth-centric scenarios. For example, ITACA should be able to support potential space or ground assets on Mars or the Moon. These scenarios are important to include in SCENIC's capabilities, as future mission planning will involve missions that extend deeper into the solar system.

Furthermore, although a task of the internship was to optimize the runtime of ITACA, there are still sections of the code that can be refined to further reduce runtime. These refinements may involve restructuring parts of the tool that were not a focus of this internship.

ITACA is collaboratively developed by personnel at both NASA GRC and GSFC, and SCENIC will continue the development of ITACA with the hope that it will be ready for integration into the UI in late 2018.

References

1. Brent, R.P. (1971). An algorithm with guaranteed convergence for finding a zero of a function. Computer Journal, 422–425.
2. Chatters, E.P., Eberhardt, B., and Warner, M.S. (2009). Orbital Mechanics. In B.C. Tichenor, Air University, Space Primer (p. 92). Montgomery: Air University Press.
3. Horne, W. (2016). Future Mission Space Communication and Navigation Needs. National Aeronautics and Space Administration.
4. Secant Method, False Position Method, and Ridders' Method. (1992). In W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery, NUMERICAL RECIPES IN C: THE ART OF SCIENTIFIC COMPUTING (pp. 354–355). Cambridge: Cambridge University Press.
5. Wertz, J.R. (1999). Earth Geometry Viewed from Space. In W.J. Larson, and J.R. Wertz, Space Mission Analysis and Design (pp. 112–114). El Segundo: Microcosm, Inc.