Decentralized Telescope Management for Satellite Tracking

Feyza Merve Hafızoğlu
Tandy School of Computer Science
University of Tulsa
Tulsa, OK, USA
Email: feyza-hafizoglu@utulsa.edu

Roger Mailler
Tandy School of Computer Science
University of Tulsa
Tulsa, OK, USA
Email: mailler@utulsa.edu

***Abstract*—Currently, over 100 million objects orbit our planet in Low-Earth Orbit (LEO). To track these objects, the United States utilizes 29 sites that comprise the Space Surveillance Network (SSN). However, numerous countries and corporations continue to expand their utilization of space leaving in their wake a sea of space debris. The rate that the number of objects is increasing has now outpaced our ability to build additional sites and there is an increasing concern for the safety of our astronauts and other space-borne assets. Another solution is to use large numbers of low-cost optical sensors, which can be easily deployed at a fraction of the cost of a traditional tracking station. However, the introduction of these systems, which are subject to time and position constraints, creates a complex coordination problem that must be solved in a communication limited environment. In this paper, we describe the satellite tracking problem and compare a resource allocation and scheduling representation for the problem. As an initial solution, scheduling tasks to telescope-slot pairs by a central facility is adopted. Then, we compare central and distributed repair mechanisms in terms of tasks completed and the communication cost. Finally, different distributed repair mechanism are proposed and compared.**

I. Introduction

On October 4, 1957 a new chapter in human history began when the U.S.S.R launched Sputnik I, the first ever man-made satellite. Since that time, humankind has increasingly explored and utilized space to enhance our scientific knowledge and improve life on Earth. In the wake of this exploration, however, are remnants of our short history in space, in the form of space debris. Currently, it is estimated that over 100 million pieces of space junk exist in Low-Earth Orbit (LEO) with over 21,000 objects being larger than 10 cm.

Residing in this narrow band of space between 100 and 1,200 miles above sea level are the Hubble telescope, the International Space Station (ISS), and many of our communication satellites. In fact, every manned space flight, with the exception of the Apollo missions to the Moon, has occurred in LEO. To protect these important assets the U.S. Space Surveillance Network (SSN) monitors the trajectories of large orbiting objects using 29 sites located around the world [1]. However, as the number of countries and private organizations that gain access to space has increased, the job of tracking the artifacts they deposit has become more difficult.

One solution is to add additional assets to the existing network. Three additional sites are expected to come online by 2017 and will allow the SSN to track 200,000 objects that are larger than 1 centimeter [2]. This upgrade is expected to cost around $3.5 billion and by the time it is finished will still only allow the SSN to track a small fraction of the material that is considered lethal to a manned space mission. At the same time, there has been increasing interest in using large numbers of low-cost optical telescopes as a more cost-effective method for improving the SSN's capacity. Adding these new assets has the potential to improve both the number and the fidelity of tracks the network can produce. However, optical sensors, such as telescopes, create additional complications because they are only useful during particular times of the day; must be positioned in remote locations; and are subject to weather conditions and other unforeseen outages.

This work specifically aims to address the telescope to satellite tasking problem created by the addition of a large number of telescopes spread around the planet. It focuses on creating new distributed techniques to manage allocation repairs that may be required during the execution of a daily tracking schedule. This will improve the overall performance of the system, which currently disregards tasks that are missed.

This paper is organized as follows: Section II describes the telescope allocation problem and gives two alternative representations of the problem. Section III introduces the communication environment of telescopes and gives several solution approaches including both centralized and distributed mechanisms. Section IV evaluates the central and distributed approaches, and finally Section V concludes with a summary and addresses directions for future research.

II. Tracking the Satellites

Many of LEO satellites are bright enough to be seen by the naked eye as they streak across the night sky. For a telescope to observe a satellite, two constraints should be satisfied: (1) telescope should be in twilight time (the satellite must be illuminated by the sun, yet the background

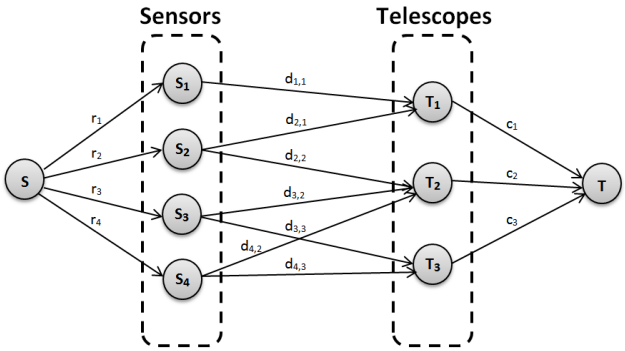

Figure 1. Max-flow in bipartite graph with telescopes

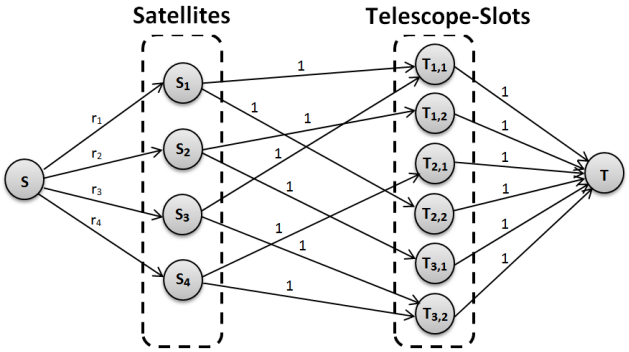

Figure 2. Max-flow in bipartite graph with telescope-slots

be dark) and (2) the satellite must be within the observation range of the telescope. We address all factors that prevent tracking, such as weather and hardware failures, as telescope failures.

It typically takes a few seconds to five minutes for a satellite to pass over a point. A telescope should complete its observations within this time. The time required to track a satellite varies based on the position of the satellite. In our model, we assume that one day is divided into slots, which have a fixed length, e.g. two minutes. The daily schedule of the telescope consists of tasks that are arranged one task per slot where the task is defined as tracking a satellite.

The complexity of telescope to satellite tasking problem arises from the huge numbers of satellites and telescopes and the combined effects of the rotation of the Earth with the movement of the satellites. Geocentric satellites usually complete fifteen orbits a day on average, i.e., their orbital period is approximately one hundred minutes, which is quite fast. If the Earth was not rotating, the set of telescopes that could track a satellite during a day would be fairly small and fixed. However, both the satellite and Earth have orbital motion, which causes a shift of the projection of the satellite on Earth in every orbit. This means during a single day, the satellite could be observed from a significant portion of the Earth's surface. This makes the set of the candidate telescopes for tracking the satellite a very large portion of the total number of telescopes, which certainly complicates the resource allocation problem.

Currently, tasking of the SSN is done on a daily basis by calculating the number of observations needed for each satellite in order to maintain an accurate model of its trajectory. The number of observations differs among satellites [3]. For instance, the satellites that experience more atmospheric drag need to be tracked more frequently than satellites that experience lower atmospheric drag and move as expected (defined by the orbital motion equations). Given the tracking requirements, a resource allocation is then centrally computed using a hill-climbing technique that selects assignments based on marginal returns [4]. A list of satellites along with their desired number of observations is sent to each of the telescopes in the SSN where a fine-grained schedule is generated that optimizes the tracking allocation.

A. Allocating Tasks to Telescopes

The tasking of telescopes to satellites can be formalized as a resource allocation problem where the problem is to assign a (usually limited) number of resources (telescopes) to a set of tasks (tracking satellites). Each of the tasks may have different resource requirements, and may have the potential for varying utility depending on which resources they utilize. The goal is to maximize the global utility of the assignment by choosing the right options for the tasks and assigning the correct resources to them.

More formally, SSN task allocation problem can be formulated as a set m telescopes $\{t_1, \ldots, t_m\}$, a set n satellites $\{s_1, \ldots, s_n\}$, and a set U of utility functions

$$U = \{U_1, \ldots U_n | U_j(x_{1j}, \ldots, x_{mj}) \longmapsto \mathbb{R}\},$$

where each U is associated with one satellite.

The goal of the problem is to come up with an allocation such that $\sum_{j=1}^{n} U_j(x_{1j}, \ldots, x_{mj})$, x_{ij} denotes the number of tracks allocated to t_i for s_j, is maximized subject to the following constraints:

$$\forall_{i \in m} \sum_{j=1}^{n} x_{ij} \leq c_i \tag{1}$$

$$\forall_{i \in m, j \in n} \;\; 0 \leq x_{ij} \leq d_{ij} \tag{2}$$

where c_i is the tracking capacity for t_i and d_{ij} is the number of opportunities for tracking s_j using t_i for a given day. The first constraint prevents the over-utilization of a telescope on a given day. The second constraint prevents an allocation where a telescope tracks a satellite more than it is able to be seen. Overall, the complexity of the generalized form of this problem is NP-hard and approaches for solving it using dynamic programming [5] and branch and bound [6] methods have been developed.

One of the simplest utility functions we can create for a given satellite is of the form [3]:

$$U_j(x_{1j}, \ldots, x_{mj}) = r_j \Big[\sum_{i=1}^{m} x_{ij}/r_j \Big]^a \quad (3)$$

where r_j is the required number of tracks and a is a tuning parameter for adjusting the shape of the utility function. This simplified utility function implies that it does not matter which telescope collects the data, only that enough measurements are made to meet the requirement. This allows the problem to be cast as a complete-compatibility SensorCSP [7] problem and can be solved using a max-flow algorithm on the bipartite graph formed by having the telescopes on one side and the satellites on the other connected by edges with capacity d_{ij} (see Figure 1).

This model has one major flaw however, because it completely ignores the underlying schedule within the telescope. For optical sensors, which only have limited time during sunrise and sunset where they can track satellites, this oversight can have a significant impact on the final solution quality because the likelihood of a scheduling conflict is quite high.

B. Scheduling Tasks to Telescope-Slots

One way to mitigate the effects of the problem in allocation is to explicitly manage time by representing the sensors as telescope-slots. A slot is discrete unit of time that sets boundaries for the execution of the task. This fairly straightforward expansion allows us to represent the following additional underlying constraint:

$$\forall_{i \in m,t} \sum_{j=1}^{n} x_{ij}^t \leq 1, \quad (4)$$

where a telescope-slot, denoted $x_{ij}^t \in \{0, 1\}$, can only be used to track a single satellite. In other words, x_{ij}^t has a value of 1 if telescope t_i can track satellite s_j at slot t, and 0 otherwise. The corresponding utility function becomes:

$$U_j(x_{1j}^1, \ldots, x_{mj}^k) = r_j \Big[\sum_{t=1}^{k} \sum_{i=1}^{m} x_{ij}^t/r_j \Big]^a, \quad (5)$$

where k is the number of slots in a day. Finally, the goal is to maximize

$$\sum_{j=1}^{n} U_j(x_{1j}^1, \ldots, x_{mj}^k). \quad (6)$$

The max-flow representation of scheduling tasks to telescope-slot pairs is given in Figure 2. Nodes on the right side represent the telescope-slot pairs. A telescope-slot node is included in the bipartite graph, when there exists at least one satellite that can be tracked by the corresponding telescope-slot pair ($\forall i \in m, t \sum_{j=1}^{n} x_{ij}^t \geq 1$). An edge is added between a satellite and a telescope-slot node, only if the satellite can be tracked by the telescope in this slot, i.e. $x_{ij}^t = 1$. The capacity of each telescope-slot pair obviously become one.

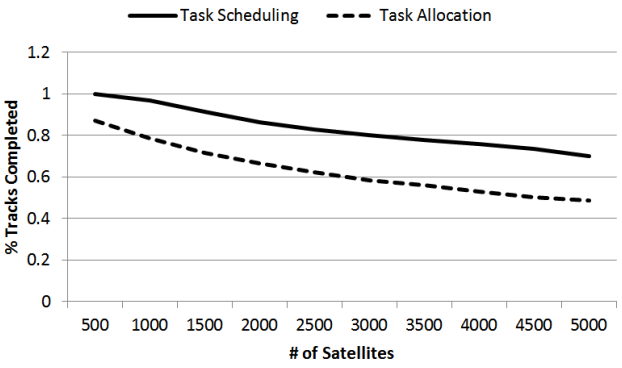

Figure 3. Results of Allocation vs. Scheduling

Figure 3 depicts the quality profiles for a problem with 200 telescopes and varying number of satellites. The graph clearly shows that as the resources become more constrained, the solution quality of the resource allocation that does not take into account scheduling collisions suffers considerably.

The separation of the centralized resource allocation from fine-grained scheduling has some advantages and disadvantages. First, telescope management requires real-time management of resources at the millisecond level, so it is advantageous given the communication bandwidth to perform local scheduling. Second, localized management allows the telescopes to conduct other missions, such as missile warning, where they can instantaneously redirect their resources to support these higher priority tasks. Third, it is computationally easier to create a rough allocation and then to allow the remote sites to fill in the details. However, one serious disadvantage is that tasks that cannot be locally scheduled or rescheduled are simply ignored. This scheme works fairly well when the ratio of telescopes to satellites is reasonable (see Figure 3). However, as the amount of space trash has increased, a significant number of the daily tasks have been left unsatisfied.

III. Communications Environment and Repair Mechanisms

Initially, we considered distributed constraint optimization protocols to derive an optimal tasking. The focus was on scalability and the potential speed up that is often associated with parallelizing the search. We quickly became frustrated with the poor performance of existing, provably optimal distributed techniques because they only operate on problems with a small number of variables that have small domain sizes within a system where they are loosely constrained. These difficulties, however, caused us to consider that maybe using distributed techniques to solve the initial tasking problem is a mistake. In fact, looking at the problem with fresh eyes, it became apparent that the initial daily planning

only needs to be done once per day giving it 24 hours to complete. Therefore, a perfectly sensible way to compute a solution is to do it in a centralized manner.

We assume that there exists a central facility making the initial tasking of telescopes at the beginning of each day. This central facility uses the fine-grained scheduling (see Section II-B). The algorithm starts by determining which telescope can track which satellite at what time based on the time and position constraints (see Section II). A max-flow solver is then used to produce a solution for the bipartite graph. Finally, the corresponding schedule is sent to each telescope.

Because of the size of the geographic area and the remote locations where many of the telescopes will be located, the central facility cannot directly communicate with all telescopes. We assume that both the central facility and the telescopes have a limited communication range and that they can only directly communicate with the entities within this range. So, the central facility can directly send the schedules to the telescopes that are located within its communication range. Otherwise, a multi-hop communication network model is adopted by using the shortest path to the particular telescope attached on the schedule message.

Once the allocation is made, however, many factors can affect the actual utility of the system. For example, intermittent cloud cover can cause tremendous variability in the number of observations that can be successfully produced by a given sensor. In order to produce the best possible results, the execution should be monitored and the allocation adapted based on the SSN's actual performance. This is complicated because the information needed to compute the value of x_{ij}^t is spread over a large geographic area and can only be gathered using explicit communication (see Section II). In addition, once the information is gathered, any changes to the allocation need to be redistributed back to the telescopes.

A. Centralized Repair Mechanism

The simplest method for monitoring and adjusting the allocation during execution is to have the telescope report failures to a centralized facility, which recalculates the tasking based on the current state. This certainly makes sense considering that it is very likely that the data being collected by the telescopes is already flowing to central facility for fusion and analysis.

In the case of a missed task, the central facility is informed about the failure. This will take time based on the distance between the central facility and the telescope, and the speed of the communication medium. The central facility reconstructs the bipartite graph by adapting to the current state of the day. The previously, successfully completed tracks are removed by decreasing the value of corresponding r_j variables. The telescope-slot nodes belong to the slots in the past are also eliminated. The new schedules for the rest of the day are computed and sent back to the telescopes.

There are two issues with this approach. First, recomputing a new solution after every failure can cause considerable solution instability. In particularly difficult instances, modifications can occur for a majority of the telescopes, which causes a considerable amount of communication over the multi-hop network and frequently interrupts the execution of the schedules. In additions, in environments where communication is intermittent and may require several steps to reach the facility, this paradigm begins to break down because information cannot be communicated fast enough to keep up with the changes. The combination of environmental dynamics and slow communications make a distributed approach appropriate for the execution time monitoring and management of the SSN.

B. Distributed Repair Mechanisms

Given the details of this problem, our hypothesis is that a distributed approach to the repair mechanism can potentially produce better quality solutions with lower communication costs. Our approach revolves around the idea that in case of a failure, a small group of telescopes can solve the problem and find a new home for the task, without disturbing the whole network. Once the initial scheduling is sent to the telescopes, the distributed repair mechanisms work without needing a central facility.

The simplest distributed repair mechanism is to allow telescopes to repair their schedules locally, i.e., the telescope tries to reschedule the failed task within its own schedule. If there is an available slot to track the same satellite during the day, this slot is reserved. This approach requires additional, up-front information because the telescopes must understand which satellites are observable by them at what times. Thankfully, this information is sent (attached to the initial schedule) by the central facility.

To understand the likely success of local rescheduling, the mechanics of the problem should be analyzed. Considering the time constraint, the potential slots are limited to twilight times. Remember that it takes a few seconds to five minutes for a satellite to pass over a location, which leaves the telescope with a very limited window of opportunity to reallocate the failed track. Once the satellite passes over the telescope it is unlikely that it will get another opportunity before twilight passes. If this occurs in the morning twilight, however, there is a fairly good chance for rescheduling because the evening twilight time has not passed yet. On the other hand, when the track failure occurs in the evening twilight, there is a lower chance to reschedule it because evening twilight is the last chance to make observations.

Certainly, one can imagine a more complex *coordinated* repair mechanism and, in the rest of this section, we introduce three distributed repair mechanisms: informed push, targeted push, and conservative broadcast.

1) Informed Push: Our overall approach to solving the execution time solution repair problem is inspired by dis-

tributed max-flow algorithms based on push relabeling [8], [9]. Like a push-relabeling, the protocol is quite simple, but works surprisingly well.

When a telescope recognizes that it is going to or has missed a track, it generates a push message. A push message consists of three parts. First, it has information about the missed task, which includes the name of the satellite and its expected trajectory. Next, it includes a list of telescopes that could potentially see the satellite in the future. This "candidate list" is generated by the central facility and included as part of the original tasking. Finally, the message includes the "message path", which is just a list of telescopes that have seen the message in the past to prevent cycles.

Once the message is created, the telescope follows a very simple algorithm, which is the same it uses when it receives a push message from another telescope. First, it checks for an empty slot in its schedule. If it has one, then the push message goes no further and the task has found a new home. Otherwise, it appends its name to the message path and sends the message to one of its neighbors that is not in the message path but preferably in the candidate list. If all neighbors are in the message path, the message dies. The process continues until either the track finds a new home, or it reaches a dead end. We refer to this method as the informed push protocol because, assuming a fairly connected communication network, the message is more likely to move along the future path of the satellite than to move to a location where the satellite will not travel. Due to its simplicity, low computational overhead, and need for strictly local communication, it is a perfect candidate solution for the SSN reallocation problem.

2) Targeted Push: The targeted push protocol is similar to informed push protocol, but it requires the additional information of observability conditions of the satellites by each telescope. In other words, complete information about which satellite can be tracked by which telescope at what time is sent to the telescopes by the central facility. It is not costly to obtain this information, since it is only sent once a day with the initial schedule. Furthermore, because it is a cooperative environment, there is not a concern for privacy in sharing this information with other telescopes. The telescopes are also aware of the network structure, which allows them to compute the shortest path. However, the telescopes do not know the schedules of other telescopes. It is far too costly to keep this information up to date because it requires broadcasting the schedule to the network whenever a change occurs.

In case of a failure, the telescope delegates the task to the closest telescope possible. If the telescope that is delegated the task is busy in the corresponding slot, the task is delegated to another telescope. The message format is same with the format introduced in informed push protocol.

3) Conservative Broadcast: Both the informed and targeted protocols perform their search in a serial manner: exploring one possible repair at a time. Often in a distributed setting it is better to explore multiple options simultaneously. A broadcast protocol requires only the knowledge of neighbor telescopes. In the case of a track failure, the telescope broadcasts a message to its neighbors, and the neighbors send this message to their neighbors. The message is broadcasted to neighbors recursively up to a certain level specified within the message.

The broadcast protocol needs an additional check to ensure that the task under considerations doesn't get adopted by too many telescopes. In the conservative version of this protocol, there are three types of messages: query, availability, and reservation messages. The query message is sent to ask the availability of the neighbors for the task. When the neighbors of the source telescope receive the message, the broadcast level of the message is decreased by one and sent to the neighbors except the neighbor that is the source of the message. When the level of the message reaches zero, it is not broadcasted anymore.

A receiver of the message that can perform the task sends an availability message to the telescope that is the initiator of the broadcast, but does not reserve the slot yet. The initiator telescope replies to the availability message that arrives earliest and can perform the task the soonest via a reservation message to schedule the slot. If the telescope that receives the reservation message has already reserved this slot for another task, it simply initiates a broadcast for this task to find an available slot.

IV. Evaluation

To test this protocol, we developed an abstract, but realistic satellite tracking environment, e.g., the dimensions of the Earth, length of the day and twilight times, the motion of satellites, the rotation of the Earth, and the capabilities of the optical telescopes.

We conducted experiments with 200 telescopes in a 360 X 180 cylindrical area that represents the surface of the Earth. The telescopes were randomly placed in this environment avoiding the areas in the upper and lower sixth of the map because it is unlikely that telescopes would be placed in these regions of the Earth due to their remoteness and harsh weather. Furthermore, satellites rarely pass over these regions and for significant portions of the year these areas have very short or no twilight hours at all.

We then add a fixed number of satellites (500, 1000, or 1500) that are given random initial trajectories and then follow an orbital path. Each satellite is required to be observed three times a day. We simulated the rotation of the earth by moving the day/night regions across the environment as the simulation unfolds. Since telescopes can only see satellites during sunrise and sunset, this restricts their activity to only 48 minutes over the course of a day.

Like the real world system, an initial tasking is generated at the start of the day (12 AM GMT) and is sent over a

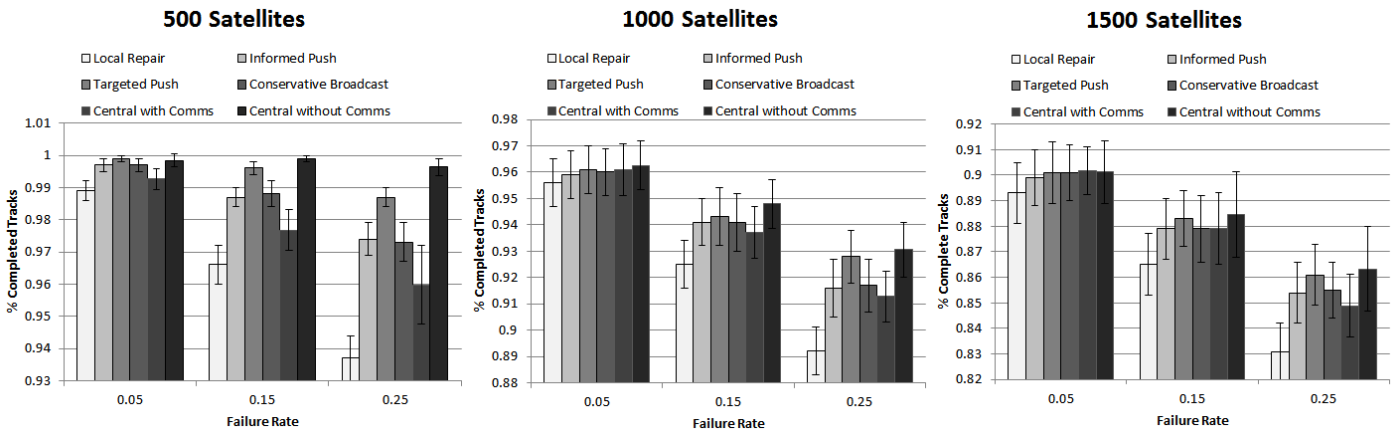

Figure 4. % Completed tracks by using 200 telescopes

multi-hop communications network. This network is formed by setting a communication radius for the telescopes to ensure that the resulting network is fully connected. For 200 telescopes, the maximum distance between two telescopes is 22 hops. Finally, to simulate environmental dynamics, each track has an associated failure rate, which varied from 5% to 50% depending on the test run.

In our domain, there are five repair mechanisms: centralized, local repair, informed push, targeted push, conservative broadcast (with a broadcast level of 3). For the communications network, there are two configurations: with cost and without cost. In the configuration with cost, the entities are connected only if they are within each other's communication range, so takes time for a message to arrive at its destination. For instance, when the central facility computes a new solution after a failure, it considers the communication costs by making sure that the earliest task in the new solution is no earlier than the time when the telescope, which is located furthest from the central facility, receives the message. Similarly, in the case of sending an availability message, the time taken to complete the protocol is considered in deciding availability. The second communication configuration (without cost) is simply a fully connected network and each entity is able to directly communicate with the rest of the network.

There are two sets of experiments: (1) the comparison of the distributed repair mechanisms with the centralized repair mechanism, and (2) the comparison of the distributed repair mechanisms by using higher number of satellites and failure rate. We conducted tests to measure the informed push, a central node that uses max-flow and experiences no communication delays, and a central node that uses max-flow and experiences communication delays based on the topology of the multi-hop network.

Each test run simulates a 24 hour period. We collected data from 100 runs for each configuration to measure average performance with the exception of centralized repair mechanism. For the centralized mechanism, we conducted 10 runs because of the amount of time needed to recompute entire solutions every time a failure occurred.

A. Centralized vs. Distributed Repair

Central and distributed repair mechanisms are evaluated based on the overall solution quality and the cost of the solution as it is desirable to obtain a high quality, low-cost mechanism. We defined two metrics: % completed tracks and cost of communication, to measure the quality and the cost, respectively. The quality metric is the percentage of the tracks that are completed successfully, as a result of either the initial scheduling or the delegation of the failed task. The total number of required tracks is equal to $3 * n$, where n is the number of satellites. The cost metric is the total number of messages. Delivering a message between any two nodes in a multi-hop network is counted as one message. Therefore, the messaging cost for communicating over distance is equal to the number of hops the message took to be delivered.

1) Quality: Figure 4 shows the averages and standard deviations of the percentage of completed tracks for varying number of satellites, the failure rates, and the repair mechanisms. To represent the optimal solution, central without communication cost mechanism is used. In this mechanism, when a failure occurs, the telescopes are immediately sent the new solution by the central facility. The graphs show that although not optimal, the distributed mechanisms work fairly well in repairing failed tracks when compared to both the "optimal" solution (max-flow without communication delay) and the central node with communication delays. None of the distributed mechanisms is worse then the centralized mechanism except two cases: (1) informed push mechanism results has slightly fewer completed tracks in case of 1000 satellites and a failure rate of 0.05, and similarly (2) informed and targeted push mechanisms have a lower completeness than the centralized repair mechanism does.

The performance of the mechanisms decreases as the failure rate and/or number of satellites increase. Local repair mechanism has severe drops in performance with increasing

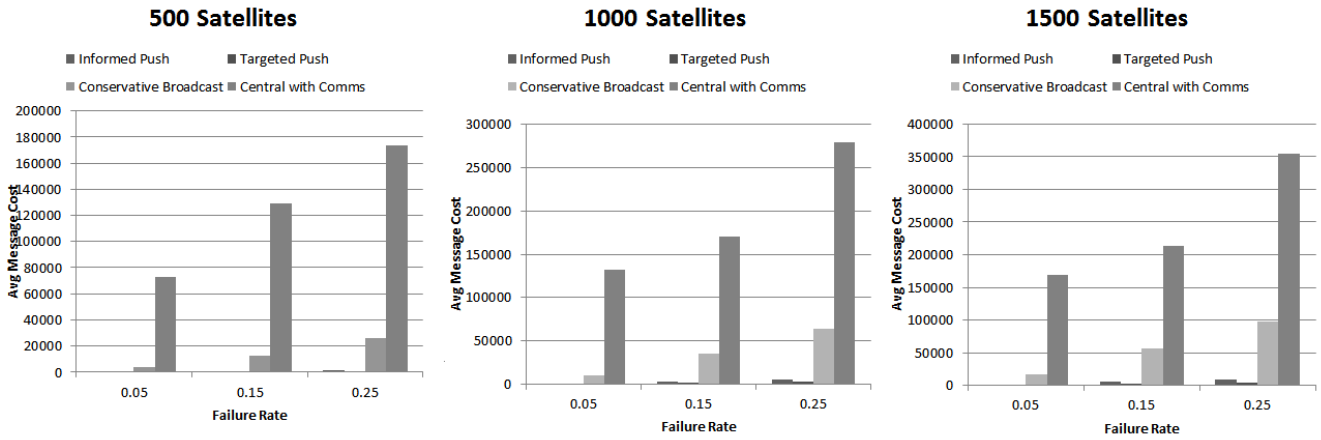

Figure 5. Message costs by using 200 telescopes

failure rate and/or number of satellites, while the performance of optimal solution slightly decreases. 200 telescopes are almost enough to track 500 satellites by the optimal solution. However, its success rate decreases in case of 1000 and 1500 satellites.

The targeted push mechanism outperforms the other non-optimal mechanisms in general. In fact, when the problem is not tightly constrained by the ratio of telescopes to satellites, e.g. in the case of 500 satellites, the informed push and conservative broadcast mechanisms outperform the centralized with delay method. In the case of 1000 satellites, the centralized with delay method barely catches the performance of these mechanisms for failure rate 0.05.

The results indicate that in terms of quality of the solution, the distributed repair mechanisms remarkably outperform central repair mechanism and achieve higher completeness rates close to the optimal repair.

2) Cost: Figure 5 depicts the average communication costs for the repair mechanisms. Looking at the overall network traffic, it can be seen that when communications are delayed, the network becomes considerably burdened. At first we thought this was caused by solution instability that occurs because the central node generates a solution from scratch every time a task is missed. However, as we later found out, the heavy network usage is more likely caused by information trickling into the centralized node, which causes it to generate new solutions nearly every minute. This causes the cost of the centralized approach to greatly exceed the cost of the distributed approaches. Conservative broadcast, informed push, and targeted push follows the centralized mechanism, respectively. For the distributed mechanisms, the costs varies around a thousand while the cost of the centralized mechanisms varies around hundreds of thousands in the case of 1500 satellites. Although these results are not surprising, it is important to address the difference between the communication costs.

As the results clearly state, all three distributed repair mechanisms achieve a better performance compared to central repair mechanism. Even though the central facility acquires more information than the telescopes do. However, this information does not help it to accomplish the mission because the communication delay prevents it from being able to react in a timely enough manner.

B. Comparing Distributed Repair Mechanisms

After comparing distributed and central approach, we increase the tightness of the problem to further investigate the performance of distributed repair mechanisms. The number of satellites and failure rate are increased to 2000 and 0.50, respectively. Although, the number of satellites can be increased further, we prefer to use a higher failure rate because of the computational cost of the initial solution produced by the central facility. Similarly, high computational costs of central repair mechanism and optimal solution prevents us from including these mechanisms in this experiment.

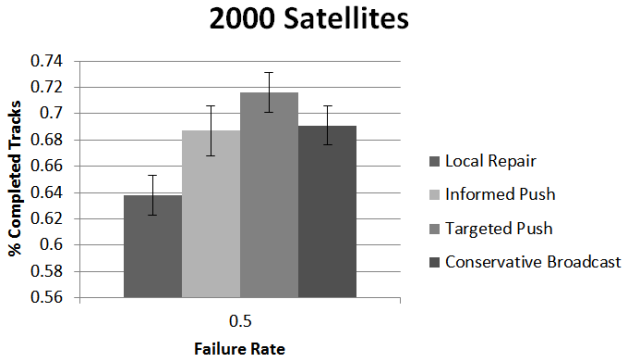

Figure 6. % Completed tracks by using 2000 satellites, and 200 telescopes with a failure rate of 0.5

Figure 6 shows the percentage of complete tracks using 200 telescopes. Of the various mechanism tried, the local repair mechanism obtains the worst performance. The graphs in Figure 4 does not reflect the contribution of repair mechanisms as well as this stressed environment does. The magnitudes of the differences in two settings are as follows: the increase in the percentage of complete tracks is at most

2% approximately in Figure 4 while it reaches a value of 8% in this tight environment.

Targeted push mechanism obtains the highest completeness rate, and is followed by informed push and conservative broadcasting mechanisms. Interesting, the performance of two former mechanisms are almost same (conservative broadcasting mechanism's completeness rate exceeds informed push's completeness rate by a value of 0.004).

The average communication costs for informed push, targeted push, and conservative broadcasting mechanisms are 37831, 18461, and 438468, respectively. The communication cost of conservative broadcasting mechanism is almost eleven times larger than the communication cost of informed push mechanism.

Among the three distributed mechanisms, targeted push dramatically outperforms the others in terms quality and cost. However, it has a drawback that it requires quite a bit of additional information to work. The cost of information transfer should be taken into account. Another aspect of sending that much information is that this requirement may open the doors for attacking the network.

Although, the communication cost of broadcasting method is much higher, in most cases it may be faster to find a home for a task because the query message is sent to a large number telescopes in k time steps, while task delegation message is received by only one telescope at a time. The speed of finding a home for a task may affect the overall performance of the system because telescopes that may have been able to track the satellite may lose the opportunity while the message is in transit.

V. Conclusion

As more countries and corporations gain access to space, the difficulty in managing the over-burdened resources used to track the debris they leave behind has reached a critical level. New strategies and methods must be developed to not only deal with the what is currently there, but what will be there in the next 20 years. The seriousness of this problem is starting to attract many researchers, but it is a challenging problem that pushes our capabilities to their very limits.

The goal of this study is to model the satellite tracking problem and analyze different approaches, both distributed and centralized, that are designed to provide solution repair when events cause tracking tasks to be missed. To fulfill this purpose, we developed an abstract environment that mimics the important characteristics of this real problem. The assignment of tasks to telescopes is modeled in two different ways: allocating tasks to telescopes and scheduling tasks to telescopes. The latter model is used to compute an initial solution by using a max-flow solver. The initial solution is computed centrally and sent to telescopes by a central facility. To mimic the dynamics of the environment, we propose failure rates in our model. This brings the repair mechanisms into play. We compare distributed and centralized repair mechanisms. The results show that distributed approach outperforms the central approach in terms of quality and cost when communications takes time to travel to and from the central facility.

This study is an initial step in the development of a solution to this problem. For future research, we will continue with a central approach for an initial solution and a distributed approach for handling the dynamics in the environment. For future work, we have three major plans: using a more realistic model in terms of geographical telescope distribution and failure distribution, developing more sophisticated decentralized coordination mechanisms, and deploying a real-world test system to validate the robustness and quality of our solutions.

VI. Acknowledgments

This material is based on research sponsored by the Air Force Research Laboratory, under agreement number FA8750-11-2-0066. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory or the U.S. Government.

References

[1] *USSTRATCOM Space Control and Space Surveillance Fact Sheet*, http://www.stratcom.mil, 2008.

[2] Lockheed Martin, "Lockheed martin space fence radar prototype tracking orbiting objects," 2012.

[3] J. G. Miller, "A new sensor allocation algorithm for the space surveillance network," *MILITARY OPERATIONS RESEARCH -ALEXANDRIA-*, vol. 12, no. 1, pp. 57–69, 2007.

[4] J. Miller, "A new sensor allocation algorithm for the space surveillance network," in *74th MORS Symposium*. Working Group 5, 2006.

[5] T. Ibaraki and N. Katoh, *Resource Allocation Problems*. Cambridge, MA: MIT Press, 1988.

[6] K. Mjelde, *Methods of the Allocation of Limited Resources*. Chichester, UK: John Wiley & Sons, 1983.

[7] e. a. Fernandez, C., *Communication and Computation in DisSCP Algorithms*. Ithaca, NY: -, 2002.

[8] J. Marberg and E. Gafni, "An o($n^2m^{1/2}$) distributed max-flow algorithm." University of California, Los Angeles, Tech. Rep., 1987.

[9] T. Pham, I. Lavallee, M. Bui, and S. H. Do, "A distributed algorithm for maximum flow problem," in *4th International Symposium on Parallel and Distributed Computing*, July 2005, pp. 131–138.