This is a peer-reviewed, author's accepted manuscript of the following conference output:
Cowlishaw, Robert ; Arulselvan, Ashwin ; Riccardi, Annalisa. / Multi-objective optimisation strategy for on-orbit fault-tolerant decision making.
Paper presented at 2024 IEEE World Congress on Computational Intelligence (WCCI), Yokohama, Japan.

Multi-Objective Optimisation strategy for On-Orbit Fault-Tolerant Decision Making

1st Robert Cowlishaw
Mechanical and Aerospace Engineering
University of Strathclyde
Glasgow, UK
robert.cowlishaw.2017@uni.strath.ac.uk

2nd Ashwin Arulselvan
Management Science
University of Strathclyde
Glasgow, UK
ashwin.arulselvan@strath.ac.uk

3rd Annalisa Riccardi
Mechanical and Aerospace Engineering
University of Strathclyde
Glasgow, UK
annalisa.riccardi@strath.ac.uk

***Abstract*—With an increasing number of satellites in orbit, consensus across a heterogeneous group of satellites can lead to a more neutral, unbiased, and accurate decisions. Fault tolerant consensus algorithms such as Practical Byzantine Fault Tolerance (pBFT) require communication with all other network members up to 4 times. In a network with thousands of satellites in space on different trajectories, this time can approach millennia. Therefore, identifying a subset of satellites that can form a sub-network able to converge to a consensus decision in a useful time window, while maximising the number of members to increase consensus accuracy and trustworthiness, can be formulated as a multi-objective combinatorial optimisation problem. The problem is explained and defined with the optimisation method and the consensus algorithm steps described. Metrics for measuring the output of the optimal pareto front are considered and applied to the front computed. The real satellite positions used generate a non-fixed topology and high latency scenario such as that of a real on-orbit decision being made. The trend shown over 100 days of satellite positions propagation with up to 82 International Charter: Space and Major Disasters satellites shows up to 22 satellites can be used in a subset with a near linear increase in consensus time along the optimal pareto front and exponential trend for the mean values computed over 100 runs of the NSGA-II algorithm. The minimum consensus time is found to be 47 minutes for a subset of 4 satellites for the given time frame.**

***Index Terms*—Consensus, decentralised network, satellites, combinatorial optimisation, multi-objective optimisation, practical byzantine fault tolerance**

I. Introduction

In the space domain, two primary trends are evident: a notable increase in the number of operational satellites and significant advancements in the capabilities of onboard sensors. These developments have led to an increase in data acquisition, which presents a critical challenge in terms of data processing. The decision regarding the location of this processing—either on the ground post-transmission or directly onboard the satellites—is crucial due to the associated costs and technical considerations.

Traditionally, satellite data processing has relied on manual work by specialists on the ground, involving the transmission of large raw datasets, which is both bandwidth-intensive and costly. Recently, there's been a shift to onboard Artificial Intelligence (AI) for processing, reducing the need for transmitting raw data. On-orbit processing, enabled by AI, means analyzing data directly on the satellite, sending only processed data or key insights back to Earth. This reduces data transmission needs and costs, streamlining the process and improving the speed and efficiency of data utilization, crucial for real-time applications like disaster response.

In the challenging environment of space, measurement inaccuracies are inherent due to several factors. These include limited precision of instruments, which is often a consequence of the vast distances involved in space operations, as well as environmental influences like radiation-induced bit flipping and atmospheric effects during data collection. Furthermore, the use of AI for processing data, while innovative, is not infallible and may potentially amplify these inaccuracies. In the context of collaborative international space initiatives, such as the International Charter: Space and Major Disasters (ICSMD) [1], cooperation across national boundaries is commonplace. However, in such multinational collaborations, issues of neutrality and the potential for geopolitical bias are often subjects of scrutiny. This is particularly critical given the high costs associated with satellite deployment and the significant value of the data they generate.

The necessity for reliable decision-making in on-orbit operations is paramount. For instance, in orbital monitoring for space environment management, the accuracy of Collision Detection Messages (CDMs) is crucial. These messages are vital for the safe operation of spacecraft, as maneuvering them is not only costly but crucial to avoid potentially catastrophic collisions. The trustworthiness of decisions made in orbit, therefore, is not just a matter of financial significance but also of operational safety and effectiveness.

Therefore, to improve the accuracy and trustworthiness of the output data from on-orbit cognitive cloud computing, redundant architecture can be used. This redundancy can be established within a single satellite system; however, expanding the approach to encompass multiple satellites offers the advantage of diverse perspectives and further reinforces redundancy. Relying solely on ground-based infrastructure located in a single country may inadvertently reintroduce biases. Therefore, a more robust approach involves aggregating and analyzing data through cognitive cloud computing across a network of communicating satellites. This methodology enables conclusions or decisions to be formulated entirely in orbit, leveraging the collective data and processing capabilities of multiple satellites. This on-orbit decision-making process not only mitigates the risk of geopolitical bias but also enhances the overall robustness and reliability of the data processing and decision-making framework in space-based operations.

To find consensus between these different communicating satellites different consensus algorithms already exist. Some examples include Practical Byzantine Fault Tolerance (pBFT) [2], Paxos/Raft [3] / [4] and Directed Acyclic Graphs (DAG) [5]. Some consensus mechanisms necessitate trusted parties, while others do not and are thus Byzantine Fault Tolerant (BFT). BFT mechanisms are structured to tolerate a certain proportion of the parties being inaccurate or malicious, whether intentionally or inadvertently. For instance, in Practical Byzantine Fault Tolerance (pBFT) [2], the system can function correctly even if up to one-third of the parties involved in the consensus are malicious. Consequently, by increasing the number of satellites involved in the consensus process, the system can accommodate a larger number of potentially malicious satellites while still arriving at the correct decision.

Inter-satellite communication required for such consensus mechanisms in this scenario is being developed with short and long distances between satellites as discussed in [6], [7] and [8]. However, longer-range communication tends to go through central points of possible failure, reducing the effectiveness of decentralised consensus mechanisms. For this work we consider short-distance communication to develop a more realistic decentralised scenario. [9] discusses multi-objective optimisation strategies in wireless sensor networks potentially applicable to this work. [10], [11] and [12] discuss similar optimisation strategies for signal routing in satellite networks showing the use of these strategies for satellite selection in tasks.

Grouping and clustering techniques have been used in low latency situations for resource allocation such as in [13] and for satellite networks in [14] and [15] however have only been used with centralised satellites in the cluster as well as only discussing fixed topology networks. Our work tests the methodology on current satellite orbits with high latency situations such as those with communication distance restrictions. [16] discusses fault-tolerant mechanisms in space however only applies these to single satellites. Our work applies similar mechanisms such as pBFT to an inter-satellite communicating network.

II. Problem Description

A. pBFT Algorithm

pBFT [2] is a replication process to find consensus while being able to tolerate Byzantine faults. Byzantine faults occur when a node (replicas) involved in a decision process is malicious, either on purpose or by error, and can cause the process to output erroneous results. By applying a consensus algorithm such as pBFT to the process, the process can handle a certain number of faulty replicas, however, the number of messages sent increases. In the case of pBFT, the process can handle a specific number of faulty replicas (f) for a total

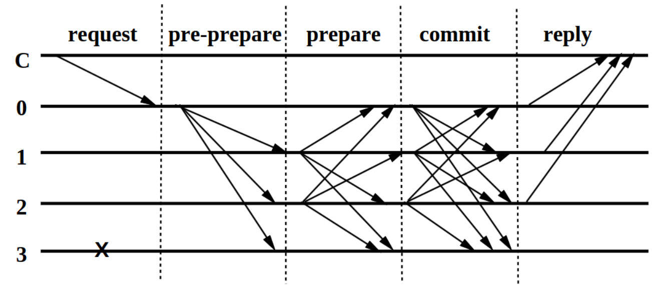

Fig. 1. pBFT algorithm scenario where replica 3 is faulty [2] (request is not required due to the assumption that the initial requester is the primary node)

number of replicas (r) involved in the process. The relationship between r and f in pBFT, while maintaining fault tolerances, is shown in equation 1 below.

$$r = 3f + 1 \tag{1}$$

The pBFT algorithm is made up of five stages: request, pre-prepare, prepare, commit and reply. The actual algorithm is more complex, but for this research, only the time taken to execute the algorithm is required, therefore, only the basic steps involved are considered as listed below and shown in figure 1. Here a scenario where replica number 3 is faulty is shown and the algorithm is simplified as the initial requester is considered to be the primary node. The list of steps are:

- Pre-Prepare - The primary replica sends the message to all other replicas.
- Prepare - All replicas, not including the primary, send the message to all other replicas, including the primary.
- Commit - All replicas, including the primary, send the message to all other replicas, including the primary.
- Reply - All replicas, not including the primary, send the message back to the primary.

B. Problem

To facilitate fault-tolerant decisions on-orbit, pBFT necessitates multiple communications among satellites in a subset. However, satellite communication is intermittent due to varying trajectories and pointing requirements. Although commercial inter-satellite communication networks, such as those developed by Viasat [17], are emerging for data forwarding in orbit, this mechanism would again depend on a centralized owner through which data must pass. Currently, inter-satellite communication technology has ranges of between 5km and 1000km [18] [19], and therefore satellites would need to come within this distance to communicate the consensus mechanism's messages.

In this study, we aim to identify the optimal criteria for selecting a subset of satellites. The primary objective is to minimize the time required for achieving consensus, taking into account the varying "in-range for communication" windows of the satellites within the subset, while maximizing the number of replicas in the subset. This can ensure a trade-off between efficient consensus time and robustness against faults within the network.

Two Line Element (TLE) data provides up-to-date satellite orbital characteristics, however, propagation of the satellites' trajectories becomes computationally expensive when propagating far into the future across many combinations of satellites.

There are currently 9195 active satellites in orbit according to CelesTrak [20] (data provided by the United States Space Force) giving 3×10^{415} possible subsets therefore being too many to compute. For a more relevant problem, it becomes necessary to decrease the value of n in the consensus. This reduction could stem from various reasons, such as the initiator of the consensus process lacking trust in or being restricted from accessing particular satellites. Additionally, the data from these satellites might not be directly applicable to the specific decision under consideration due to the lack of specific sensors.

C. Satellite-based Emergency Mapping Use Case

The International Charter: Space and Major Disasters (ICSMD) [1] is a collaboration across many space agencies around the world to produce satellite data maps for natural disaster coordination and response. For this use case, the satellites considered in the consensus algorithm are currently active satellites officially available to produce data for ICSMD [1]. These satellites also produce data for a similar cause and therefore there is the possibility that they would need to make a collective decision. This reduces the number of satellites available to 82, giving a total value of possible combinations of 1.98×10^{26}.

III. Problem Definition

To formalise mathematically the problem statement of finding the subset of satellites that minimises the time taken for consensus while maximising the number of replicas, we need to formulate the corresponding objectives and constraints function as in equation 2.

$$\begin{aligned} \min \quad & f_1(\mathbf{x}) \\ \max \quad & \sum_{i=1}^{n} x_i \\ \text{s.t} \quad & \\ & \mathbf{x} \in \{0,1\}^n \\ & \sum_{i=1}^{n} x_i \geq 4 \\ & c(\mathbf{x}) = 1 \end{aligned} \tag{2}$$

where $\mathbf{x}$ is the decision vector, $f_1(\mathbf{x})$ if the function that models consensus time, and $c(\mathbf{x})$ is the function that models completeness of the consensus mechanism.

The completeness function $c(\mathbf{x})$ is given in equation 3 is the number of messages that can be sent, through the consensus network, within a maximum time (t_{max}), divided by the total number of messages needed for completing the consensus mechanism, for the specific set of satellites encoded in $\mathbf{x}$.

$$c(\mathbf{x}) = \frac{\sum_{i=1}^{m_{max}} m_i}{m_{max}} \tag{3}$$

where $m_i \in \{0,1\}$, if $m_i = 1$ represents the successful communication of message through the i-th link in the consensus network. A maximum time t_{max} needs to be set to avoid calculating the "in-range for communication" times otherwise the computation would become infeasible. The maximum number of messages m_{max} communicated through the pBFT network is computed as in equation 4.

$$m_{max} = 2r^2 - r - 1 \tag{4}$$

$f_1(\mathbf{x})$ is defined by the time taken to reach completion of the consensus algorithm with the specific subset of satellites as encoded in $\mathbf{x}$. Therefore, if $c(\mathbf{x})$ does not equal 1, no value for $f_1(\mathbf{x})$ can exist as consensus has not been reached and the solution is considered infeasible.

The objective function $f_1(\mathbf{x})$ is defined through algorithm 1. First, the time and satellite variables are defined. This is proceeded by the Pre-Prepare Step, detailed in algorithm 2 which is as follows: for each satellite x_j that is not the primary, and for each time step t_k from t_0 to t_{max}, the position of the primary satellite and satellite x_j is obtained. Then, it is checked whether the distance between them is less than D, the maximum communication distance. If the distance is within this threshold, T_j—the time at which a satellite completes the consensus step—is set to the current time step t_k, and the loop iterating through time steps is terminated.

Algorithm 3 for the prepare step follows a similar method to the Pre-Prepare step. For each satellite sending communications x_i not including the primary and for each satellite receiving communication x_j, the prepare method follows a similar methodology as pre-prepare. The time step t_k updates the consensus step time ending for each x_j. Algorithm 4 is the same as the prepare step with x_i also including the primary satellite. Finally, the reply step, shown in algorithm 5 is the same as the pre-prepare step with the loops for the sender satellites and receiver satellites reversed. After all these consensus steps have been completed, the maximum value in $\mathbf{T}$ gives the final consensus time.

Algorithm 1 Satellite Consensus Time Algorithm

1: Select start time for consensus t_0 and maximum time window t_{max}.
2: Assign the primary p as the first satellite in $\mathbf{x}$
3: Initialise $\mathbf{T}$ as array of zeros with same length as $\mathbf{x}$
4: Set $T_{primary}$ to t_0
5: Run Pre-Prepare Step in algorithm 2
6: Run Prepare Step in algorithm 3
7: Run Commit Step in algorithm 4
8: Run Reply Step in algorithm 5
9: Output maximum value in $\mathbf{T}$ as final consensus time

Algorithm 2 Pre-Prepare Step

for each satellite $x_j \neq p$ in $\mathbf{x}$ **do**
  for each t_k from t_0 to t_{max} **do**
    Get position of p and x_j at t_k by propagating TLE data
    if $D \geq$ norm of position of $p - x_j$ **then**
      $T_j = t_k$
      break
    end if
  end for
end for

Algorithm 3 Prepare Step

for each satellite x_j in $\mathbf{x}$ **do**
  initialise $t_{receive}$ to zero
  for each satellite $x_i \neq p$ in $\mathbf{x}$ **do**
    if $x_i \neq x_j$ **then**
      for each t_k from T_i to t_{max} **do**
        Get position of p and x_i at t_k by propagating TLE data
        if $D \geq$ norm of position of $p - x_i$ **then**
          if $t_{receive} < t_k$ **then**
            $t_{receive} = t_k$
          end if
          break
        end if
      end for
    end if
  end for
  $T_j = t_{receive}$
end for

Algorithm 4 Commit Step

for each satellite x_j in $\mathbf{x}$ **do**
  initialise $t_{receive}$ to zero
  for each satellite x_i in $\mathbf{x}$ **do**
    if $x_i \neq x_j$ **then**
      for each t_k from T_i to t_{max} **do**
        Get position of p and x_i at t_k by propagating TLE data
        if $D \geq$ norm of position of $p - x_i$ **then**
          if $t_{receive} < t_k$ **then**
            $t_{receive} = t_k$
          end if
          break
        end if
      end for
    end if
  end for
  $T_j = t_{receive}$
end for

Algorithm 5 Reply Step

for each satellite $x_i \neq p$ in $\mathbf{x}$ **do**
  for each t_k from T_i to t_{max} **do**
    Get position of p and x_i at t_k by propagating TLE data
    if $D \geq$ norm of position of $p - x_i$ **then**
      $T_i = t_k$
      break
    end if
  end for
end for

IV. Problem Solution

The multi-objective algorithm NSGA-II [21] is used to minimise the problem stated in problem III. The NSGA-II implementation used to generate the results of this paper is the one from the Pymoo library [22]. The completeness and consensus time functions are coded in Golang to optimise runtime [1].

The trajectories for the 82 satellites involved in the ICSMD are calculated in advance and stored in memory. This pre-computation, carried out for the time interval between t_0 and t_{max}, includes the specific time points at which communication opportunities arise for each satellite pair. This approach is feasible due to the manageable number of satellites in the ICSMD network. By storing these trajectories and communication times, we eliminate the need for repeated trajectory propagation every time the fitness function is called, thereby enhancing computational efficiency. However, if this work was used for finding the optimal group for fault-tolerant decision-making, propagation of the current satellite positions would be required for real-time decision-making.

[1]The implementation can be found at github.com/strath-ace/smart-dao

A. Parameters of the NSGA-II Algorithm

A distance for D is chosen to be 500km a somewhat pessimistic inter-satellite communication compared to the upper bounds given in [18], however, still maintaining the ability to communicate throughout the Low Earth Orbit (LEO) altitudes of which the ICSMD satellites are in.

A timestep is chosen based on the distance D. As the 82 satellites in ICSMD are in LEO and in general follow low eccentricity orbits they have average velocities of approximately 7.5km/s. If both satellites are travelling in opposite directions, a timestep of 30 seconds for position propagation is required to register flybys at these velocities with $D = 500$km.

The initial time t_0 is set to an arbitrary value. This choice is made because the maximum time, t_{max}, is sufficiently distant into the future. As a result, this large time gap eliminates the need for averaging over multiple starting times, as the chosen t_0 ensures that the system has already reached a stable state by the time t_{max} occurs. For this reason, t_{max} is set to 100 days past t_0. This large time gap also provides a large pareto front while maintaining manageable computational requirements for satellite propagation.

The population size is set to 50. This is to limit the large amounts of satellite position data to be loaded into memory for each round of consensus. Also, the number of points on the Pareto front for the given t_{max} is predicted to be lower than 50.

The mutation properties are set to a low crossover probability of 0.1 as two groups of different satellite subsets are unlikely to crossover to produce better offspring and a low random mutation of 0.1 as small changes in the satellites in the subset will allow for finer optimisation when a large number of combinations need to be searched through.

The optimization runs consistently ended around the 500th generation. Consequently, we established the termination criteria for the NSGA-II algorithm to halt after 500 generations, with a tolerance of 0.0001 in the objective space.

For the problem presented, an analytical formulation of the optimal Pareto front does not exist and the true optimal front cannot be found by complete enumeration for computational reasons. Hence the non-dominated solutions computed from the fronts obtained from multiple runs of the optimisation algorithm are to be considered as the true optimal front for convergence calculations. This paper's implementation uses 100 runs to find this considered optimal front.

B. Pareto Fitness Metrics

To assess the convergence of each run the following metrics are used to compare each run against the true front.

- Generational Distance (GD) given by [23].
- Generational Distance Plus (GD+) given by [24].
- Inverted Generational Distance (IGD) given by [25].
- Inverted Generational Distance Plus (IGD+) given by [24].
- Hypervolume given by [26].

V. Results

From figure 2 it can be seen that fewer Pareto front values are found as the number of satellites in the subset increases. This is due to the threshold t_{max}, however, a consistent number of solutions could be found at all quantities of satellites in the subset at the expense of far more enumerations. As the number of satellites in the subset increases, the distribution of consensus times in the calculated fronts increases. The optimal Pareto front also follows a nearly linear trend and the mean Pareto front follows an exponential curve, however, larger gaps between t_0 and t_{max} values could show different trends. This exponential curve for the mean Pareto front shows that it is increasingly difficult to compute consensus time for increasing values of r. These results can also be seen in table 3. With small subsets of satellites/replicas, the lowest consensus time can be 47 minutes for a subset of 4, whereas in the limited t_0 to t_{max} period a subset of 22 satellites is the maximum found.

The results of the metrics defined in section IV-B are shown in figures 4, 5, 6, 7 and 8. The metrics show that the multi-objective optimisation is consistently close to the optimal front and follows the same or similar trend to the optimal front. From the hypervolume a value of 1 would show a run where the calculated front is the optimal front, however as seen in figure 8, this does not occur and therefore means that the optimal front is found across multiple runs. This is also seen in all the distance metrics, figures 4, 5, 6 and 7 as no distance metric is calculated to be 0.

These findings demonstrate that the algorithm outlined in section III is capable of generating a Pareto front for scenarios involving real satellite orbits with a non-fixed topology in a high-latency environment. This Pareto front can be generated in real-time when decisions need to be made across satellites, enabling a trade-off between fault-tolerance security and the time taken for consensus to be reached.

Additionally, the algorithm for $f_1(x)$ can be employed

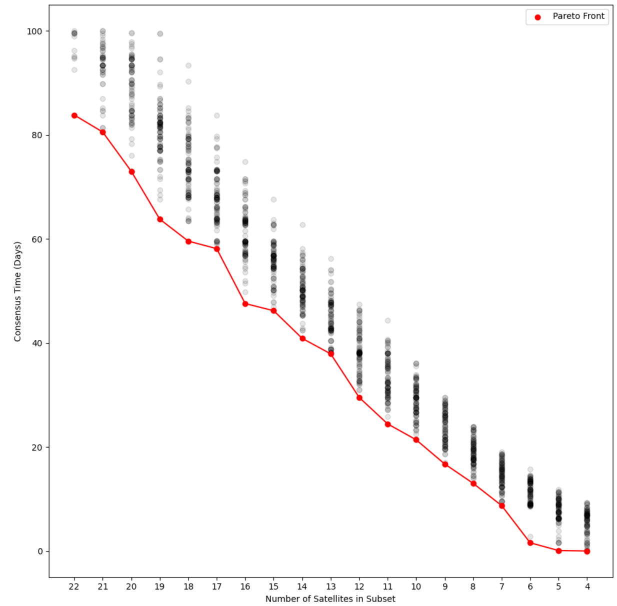

Fig. 2. Pareto front for all 100 runs/solutions over 100 days measured in consensus times

Satellite number	Pareto Front	Mean	Std	Number of solutions
4.0	0.03264	5.24038	2.63523	92.0
5.0	0.12014	7.26336	3.09587	100.0
6.0	2.78369	11.24996	2.15262	100.0
7.0	8.7667	14.60851	2.53878	100.0
8.0	14.02227	18.94334	2.46985	100.0
9.0	16.76534	23.89132	3.30766	99.0
10.0	21.40597	28.52884	3.22983	99.0
11.0	24.45425	32.9676	3.7929	99.0
12.0	29.54698	38.37755	4.21573	98.0
13.0	37.92339	44.25111	4.36737	99.0
14.0	42.46717	50.42043	3.88357	98.0
15.0	46.20989	56.11079	4.06631	96.0
16.0	47.58072	61.60903	4.69099	100.0
17.0	58.12867	67.61562	5.40676	98.0
18.0	59.5884	74.0521	6.31828	98.0
19.0	63.80717	80.82142	6.59454	96.0
20.0	73.04366	89.25224	6.27326	86.0
21.0	80.53986	93.31379	5.04003	51.0
22.0	85.98259	97.42879	3.46302	15.0

Fig. 3. Optimal Pareto Front Results for all 100 runs/solutions over 100 days measured in consensus times (days)

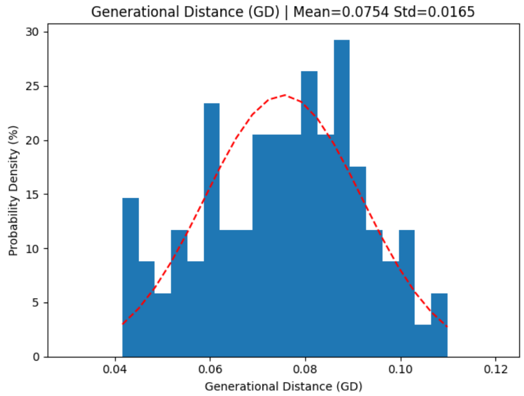

Fig. 4. Generational Distance of 100 runs against the optimal Pareto front

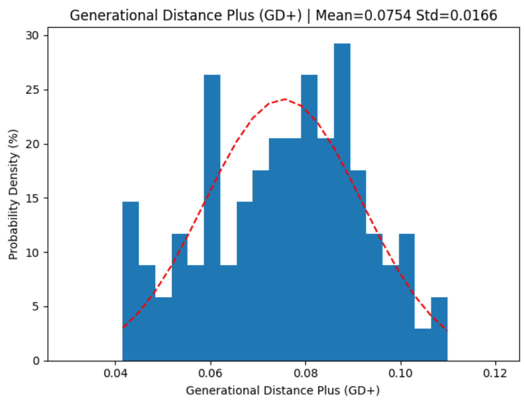

Fig. 5. Generational Distance Plus of 100 runs against the optimal Pareto front

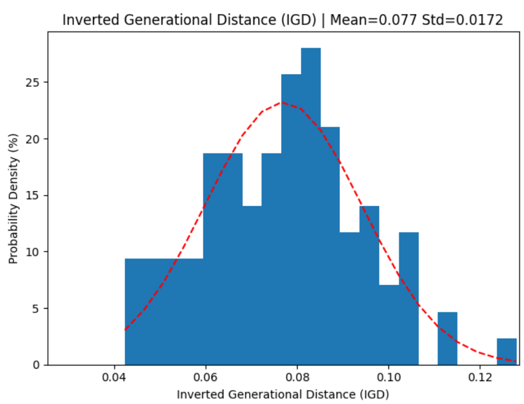

Fig. 6. Inverted Generational Distance of 100 runs against the optimal Pareto front

to identify a subset of satellites when the desired number of satellites is known, using a single-objective optimization approach. Alternatively, if consensus time is needed for a specific subset of satellites, the same algorithm can be utilized for this purpose.

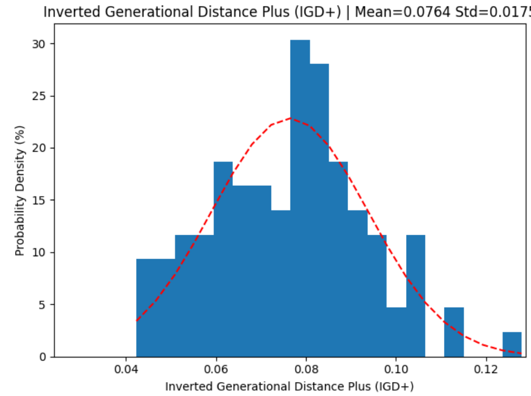

Fig. 7. Inverted Generational Distance Plus of 100 runs against the optimal Pareto front

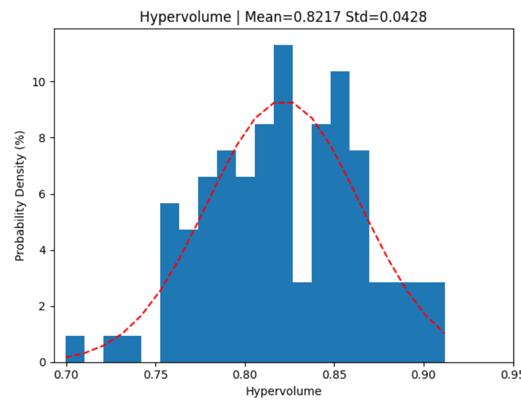

Fig. 8. Hypervolume of 100 runs divided by the hypervolume of the optimal Pareto front

VI. Conclusion

This paper shows a way of determining the time taken for pBFT consensus to occur in a subset of in-orbit satellites as a way of securing and increasing the accuracy of on-orbit decision-making. The NSGA-II algorithm is used to solve the optimal Pareto front of the problem defined that minimises the time taken for consensus while maximising the number of satellites/replicas in a subset. Five metrics are used to analyse the distribution of the 100 Pareto fronts generated to determine an accurate optimal Pareto front where the true Pareto front cannot be solved due to the large number of enumerations required to find the consensus times. The algorithm for finding these consensus times is defined. The algorithm propagates satellite positions from TLE data to compare the relative distance between the satellite subsets, giving a high latency and non-fixed topology scenario. The results show that within a 100-day period, the maximum number of satellites/replicas in the subset is 22. And 4 and 5 satellites/replicas in a subset can find consensus within 47 minutes and 173 minutes respectively. From the optimal Pareto front calculated a trade-off between accuracy due to the number of satellites/replicas in the subset and the consensus time can be made.

REFERENCES

[1] International charter: Space and major disasters. Accessed: 2024-01-05. [Online]. Available: https://disasterscharter.org

[2] M. Castro, B. Liskov *et al.*, "Practical byzantine fault tolerance," in *OsDI*, vol. 99, no. 1999, 1999, pp. 173–186.

[3] L. Lamport, "Generalized consensus and paxos," 2005.

[4] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *2014 USENIX annual technical conference (USENIX ATC 14)*, 2014, pp. 305–319.

[5] S. Popov, "The tangle," *White paper*, vol. 1, no. 3, p. 30, 2018.

[6] H. Guzmán-Miranda, J. Barrientos-Rojas, P. López-González, V. Baena-Lecuyer, and M. A. Aguirre, "On the structural robustness assessment of wireless communication systems for intra-satellite applications," *IEEE Transactions on Nuclear Science*, vol. 61, no. 6, pp. 3244–3249, 2014.

[7] C. F. Kwadrat, W. D. Horne, and B. L. Edwards, "Inter-satellite communications considerations and requirements for distributed spacecraft and formation flying systems," in *2002 Space Operations Conference*, 2002.

[8] R. Radhakrishnan, W. W. Edmonson, F. Afghah, R. M. Rodriguez-Osorio, F. Pinto, and S. C. Burleigh, "Survey of inter-satellite communication for small satellite systems: Physical layer to network layer view," *IEEE Communications Surveys Tutorials*, vol. 18, no. 4, pp. 2442–2473, 2016.

[9] Z. Fei, B. Li, S. Yang, C. Xing, H. Chen, and L. Hanzo, "A survey of multi-objective optimization in wireless sensor networks: Metrics, algorithms, and open problems," *IEEE Communications Surveys Tutorials*, vol. 19, no. 1, pp. 550–586, 2017.

[10] Y. Wu, G. Hu, F. Jin, and S. Tang, "Multi-objective optimisation in multi-qos routing strategy for software-defined satellite network," *Sensors*, vol. 21, no. 19, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/19/6356

[11] X. Wu, T. Vladimirova, and K. Sidibeh, "Signal routing in a satellite sensor network using optimisation algorithms," in *2008 IEEE Aerospace Conference*, 2008, pp. 1–9.

[12] R. Fdhila, T. M. Hamdani, and A. M. Alimi, "A multi objective particles swarm optimization algorithm for solving the routing pico-satellites problem," in *2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2012, pp. 1402–1407.

[13] X. Liu, J. Yu, J. Wang, and Y. Gao, "Resource allocation with edge computing in iot networks via machine learning," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3415–3426, 2020.

[14] S. Geng, S. Liu, and Z. Fang, "Resilient communication model for satellite networks using clustering technique," *Reliability Engineering System Safety*, vol. 215, p. 107850, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0951832021003690

[15] J. Liu, X. Zhang, R. Zhang, T. Huang, and F. R. Yu, "Reliable and low-overhead clustering in leo small satellite networks," *IEEE Internet of Things Journal*, vol. 9, no. 16, pp. 14 844–14 856, 2022.

[16] M. Fayyaz and T. Vladimirova, "Fault-tolerant distributed approach to satellite on-board computer design," in *2014 IEEE Aerospace Conference*, 2014, pp. 1–12.

[17] Viasat real-time space relay. Accessed: 2024-01-05. [Online]. Available: https://www.viasat.com/space-innovation/space-systems/intersatellite-communications/

[18] R. Radhakrishnan, W. W. Edmonson, F. Afghah, R. M. Rodriguez-Osorio, F. Pinto, and S. C. Burleigh, "Survey of inter-satellite communication for small satellite systems: Physical layer to network layer view," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2442–2473, 2016.

[19] C. F. Kwadrat, W. D. Horne, and B. L. Edwards, "Inter-satellite communications considerations and requirements for distributed spacecraft and formation flying systems," in *2002 Space Operations Conference*, 2002.

[20] Celestrak. Accessed: 2024-01-07. [Online]. Available: https://celestrak.org/

[21] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[22] pymoo: Multi-objective optimization in python. Accessed: 2023-12-14. [Online]. Available: https://pymoo.org/

[23] D. A. Van Veldhuizen, *Multiobjective evolutionary algorithms: classifications, analyses, and new innovations*. Air Force Institute of Technology, 1999.

[24] H. Ishibuchi, H. Masuda, Y. Tanigaki, and Y. Nojima, "Modified distance calculation in generational distance and inverted generational distance," in *Evolutionary Multi-Criterion Optimization: 8th International Conference, EMO 2015, Guimarães, Portugal, March 29–April 1, 2015. Proceedings, Part II 8*. Springer, 2015, pp. 110–125.

[25] C. A. Coello Coello and M. Reyes Sierra, "A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm," in *MICAI 2004: Advances in Artificial Intelligence: Third Mexican International Conference on Artificial Intelligence, Mexico City, Mexico, April 26-30, 2004. Proceedings 3*. Springer, 2004, pp. 688–697.

[26] C. Fonseca, L. Paquete, and M. Lopez-Ibanez, "An improved dimension-sweep algorithm for the hypervolume indicator," in *2006 IEEE International Conference on Evolutionary Computation*, 2006, pp. 1157–1163.