

## Toward Autonomous Cooperation in Heterogeneous Nanosatellite Constellations Using Dynamic Graph Neural Networks

G. Casadesus Vila<sup>a</sup>, J.A. Ruiz-de-Azua<sup>b</sup>, E. Alarcon<sup>c</sup>

<sup>a</sup>Stanford University, <sup>b</sup>i2CAT Foundation, <sup>c</sup>Universitat Politecnica de Catalunya (UPC)

### Abstract

The upcoming landscape of Earth Observation missions will be defined by networked heterogeneous nanosatellite constellations required to meet strict mission requirements, such as revisit times and spatial resolution. However, scheduling satellite communications in these satellite networks through efficiently creating a global satellite Contact Plan (CP) is a complex task, with current solutions requiring ground-based coordination or being limited by onboard computational resources. The paper proposes a novel approach to overcome these challenges by modeling the constellations and CP as dynamic networks and employing graph-based techniques. The proposed method utilizes a state-of-the-art dynamic graph neural network to evaluate the performance of a given CP and update it using a heuristic algorithm based on simulated annealing. The trained neural network can predict the network delay with a mean absolute error of 3.6 minutes. Simulation results show that the proposed method can successfully design a contact plan for large satellite networks, improving the delay by 29.1%, similar to a traditional approach, while performing the objective evaluations 20x faster.

**Keywords:** Satellite Communications, Contact Plan Design, Earth Observation Constellations, Dynamic Graph Neural Networks, Deep Learning, Simulated Annealing, Optimization

## 1. Introduction

### 1.1 Motivation

The current New Space era has been characterized by increased access to space, driven significantly by the advent of small satellites and reduced launch costs, with 3,500 Earth Observation (EO) satellites to be launched over the next decade [1, 2]. Space agencies, academic institutions, and private companies have adopted small satellite architectures for varied applications, ranging from scientific research and training to technology demonstrations and other space-based industrial applications. Now, the space sector stands on the cusp of another leap, transitioning towards assembling and operating large, network-enabled heterogeneous nanosatellite constellations that will enable new Earth observation capabilities.

Incorporating inter-satellite links (ISLs) in EO constellations brings the immense advantages of inter-satellite collaboration, allowing them to meet stringent mission requirements such as real-time high-resolution global mapping [3]. Seamless data-sharing capabilities across remote regions empower EO satellite constellations to create large observational networks. Beyond the data, satellites can leverage unused onboard resources from their peers, increasing the scientific return. This ecosystem of sharing and collaboration, enabled by the agile operation of a large nanosatellite constellation with inter-satellite communications,

allows for the maximization of resources, enhanced operational efficiency, and a collaborative approach to achieving mission objectives.

While collaboration and sharing offer numerous benefits, satellite constellations are subject to resource constraints and evolving network topologies, categorizing them as delay/disruption tolerant (DTN) networks. In contrast to traditional end-to-end networks, DTNs are characterized by intermittent connectivity, long and variable delays, and high error rates. In DTNs, routes between nodes need to be defined over long timespans, requiring new approaches to network management [4].

The complexity of managing large EO networks underscores the pressing need for autonomy [3, 5]. One of the critical problems in this context is how to efficiently schedule satellite communications given their communication capabilities, available onboard resources, downlink opportunities, and mission constraints. Generating this schedule is known as the contact plan design (CPD) problem. Traditional approaches, often centralized and reliant on human intervention, are becoming untenable, especially when dealing with constellations managed by diverse providers, as will be the case for the upcoming EO landscape. In light of these challenges, moving towards autonomous systems is not just beneficial but imperative. Autonomy can significantly enhance the constellation robustness, efficiency, and cost of operations, allowing satellites to make real-time decisions based on evolving mission needs.

## 1.2 Literature Review

Due to the increasing interest in inter-satellite connectivity, the CPD problem has received significant attention over the last few years [6]. Traditionally, the problem has been solved with three methodologies involving different levels of information and complexity: topology-based, traffic-based, and route-based.

Topology-based CPD methods are grounded on the deterministic nature of satellite orbital motion and only require the basic assumption of a known network topology. While they provide a simple and scalable solution, their performance is limited. On the other end of the spectrum, traffic-aware CPD methods consider the traffic demand between satellites and aim to optimize the network performance. However, these methods are computationally expensive and require a large amount of information, which is not always available. In this context, the authors in [7] formulate the CPD problem as a flow optimization problem and solve it using stochastic optimization. By further modeling the satellite resources such as buffer and battery capacity, the authors [8] propose a primal decomposition method and heuristic algorithm to solve the CPD problem while considering different mission requirements.

Route-aware CPD methods are a middle ground between the two previous approaches, requiring less information than traffic-aware methods while providing better performance than topology-aware methods. Several works [9, 10] proposed using Contact Graph Routing (CGR). This routing algorithm uses a contact graph to represent the contact plan and computes the best route between two nodes using a modified Dijkstra search, together with metaheuristic algorithms such as Simulated Annealing (SA) and Genetic Algorithms (GA) to maximize all-to-all route delay, provide fairness, and minimize undelivered traffic. The authors in [11] also resort to SA and include link constraints to preserve diversity in navigation satellite constellations. Under strong assumptions regarding the contact topology, recent work [12] focuses on multi-layer satellite networks. These works, however, consider single missions and small homogeneous constellations, often in the context of ground-based centralized coordination.

While current studies cover different CPD methodologies, available information, and performance requirements, they fall short of addressing the complexity of large heterogeneous constellations. In particular, they do not sufficiently study how the CP can be optimized onboard for large constellations managed by different entities, targeting computational performance and scalability. Furthermore, they do not consider the dynamic nature of the network, limiting their scope to homogeneous constellations of reduced size.

We focus on learning-based methods to address this gap and target a good balance between performance and scalability. Satellite and, more generally, communication networks comprise many fundamental components naturally represented in graph structures, such as the network topology or routing. In fields where data is represented as dynamic graphs, dynamic graph neural networks (DGNN) have shown outstanding results [13]. The success of such applications, including delay and traffic prediction in wireless networks [14], motivates their use for the CPD problem.

## 1.3 Paper Objective

This paper presents a method to model communication networks with dynamics topologies using dynamic graphs and employs DGNNs for network-level metrics computation. We present a DGNN architecture for dynamic network modeling that integrates the spatial processing of Graph Neural Networks (GNN) with Recurrent Neural Networks (RNN) temporal processing. This DGNN is trained to assess latency in satellite constellations represented as dynamic graphs and optimize contact plans using a simulated annealing-based metaheuristic algorithm. Though our primary focus is on the CPD problem, we see this approach as a foundation for utilizing dynamic graph-based learning in managing heterogeneous satellite constellations.

## 1.4 Overview

The remainder of the paper is organized as follows: Section 2 formulates the CPD problem with the implications introduced by heterogeneous EO constellations, Section 3 describes the proposed CPD method with the use of DGNN for the objective function learning, Section 4 shows the results of applying the method to different use cases. Finally, Section 5 remarks the conclusions of this work and possible future research directions.

## 2. Problem Statement

In this section, we formulate the CPD problem as a constrained optimization problem, similar to previous works [9, 11]. Two key differences are that we do not reduce the constellation to a finite set of periodic states due to the scale and heterogeneity of our target constellations and that we include source and destination traffic information for different types of nodes, namely satellites and ground stations.

### 2.1 System model

We consider a non-gostationary orbit constellation with a set of satellites,  $\mathcal{S} = \{1, 2, \dots, N_s\}$ , that can communicate with each other and a set of ground stations,  $\mathcal{G} = \{1, 2, \dots, N_g\}$ , at discrete time steps  $t \in \mathcal{T} = \{1, 2, \dots, N_t\}$ . We assume that each satel-

lite  $i$  needs to communicate with a subset of satellites and ground station  $\mathcal{C}_i \subset \mathcal{SUG}$ , which is known a priori.

We model the communication opportunities between satellites and ground stations, which depend on their communication capabilities and operational constraints, as a sequence of visibility matrices  $V = (V_1, \dots, V_{N_t})$ , where  $V_t \in \mathbb{R}^{(N_s+N_g) \times (N_s+N_g)}$ , and  $V_t(i, j) = 1$  if communication between node  $i$  and  $j$  is feasible at time  $t$  and  $V_t(i, j) = 0$  otherwise. Therefore, a visibility matrix represents the communication feasibility between satellites and ground stations at different time instances, capturing their communication constraints, e.g., the inter-satellite range for satellites and minimum elevation angle for ground stations, all assumed to be known a priori.

We define the contact plan to be generated as a sequence of matrices  $U = (U_1, \dots, U_{N_t})$ , where  $U_t \in \mathbb{R}^{(N_s+N_g) \times (N_s+N_g)}$  is the contact plan matrix at time  $t$ . The contact plan matrix  $U_t$  represents the scheduled communication links between nodes at time  $t$ , where  $U_t(i, j) = 1$  if there is a link to be established between nodes  $i$  and  $j$  at time  $t$  and  $U_t(i, j) = 0$  otherwise.

We assume that satellites and ground stations establish bidirectional links, i.e.,  $U_t(i, j) = U_t(j, i)$ , and that the maximum number of links that a satellite can establish with other satellites and ground stations is  $M_s$  and  $M_g$ , respectively. These constraints are motivated by mission requirements and limit the number of links a satellite can establish during the period for which the contact plan needs to be generated.

For a given contact plan, the best delivery time (BDT) between two nodes  $i$  and  $j$  at time  $t$ ,  $d(i, j, t)$ , is the time that it takes for a message sent a time  $t$  from satellite  $i$  to reach satellite  $j$ , accounting only for the changing topology of the network. That is, without taking into account other traffic or link capacity. The objective of the CPD problem is to find the contact plan that minimizes the BDT between pairs of nodes.

$$\text{minimize } F = \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{C}_i} d(i, j, t) \quad (1)$$

$$\text{subject to } U_t(i, j) = U_t(j, i), \quad \forall i, j, t \quad (2)$$

$$U_t(i, j) \leq A_t(i, j), \quad \forall i, j, t \quad (3)$$

$$\sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{S}} U_t(i, j) \leq M_s, \quad \forall i \quad (4)$$

$$\sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{G}} U_t(i, j) \leq M_g, \quad \forall i \quad (5)$$

where  $U_t(i, j)$  is the optimization variable encoding the contact plan. Constraint (2) enforces bidirectional links, (3) ensures that the contact plan is feasible under

the satellite topology, and (4) and (5) limit the number of links that a satellite can establish with other satellites and ground stations, respectively.

### 3. Methodology

In this section, we present the proposed method for the CPD problem, which consists of two main components: a heuristic algorithm based on simulated annealing for contact plan design and a dynamic graph neural network for evaluating the objective function.

#### 3.1 Simulated Annealing

Considering that the contact plan problem defined in the previous section would rapidly become computationally intractable, we use a heuristic algorithm based on simulated annealing (SA), a stochastic optimization method that mimics the slow cooling process in metallurgy to gradually reduce the search space and decrease the likelihood of accepting suboptimal solutions [15]. This approach has been demonstrated to be effective in topology design problems [6, 11], enabling the acquisition of sub-optimal solutions. Algorithm 1 (Appendix 5) presents the contact plan design algorithm based on SA.

We begin by creating an initial contact plan, from which we obtain a new contact plan while ensuring constraint satisfaction by adding or removing links. Then, we calculate and compare the objective functions of the new and the current contact plan. If the objective function of the new contact plan is improved, we accept the new contact plan and reduce the temperature  $T$ . Otherwise, we accept the new contact plan with a probability  $P = \exp(-(F_{new} - F_{curr})/T)$ , where  $F_{new}$  and  $F_{curr}$  are the objective functions of the new and the current contact plans, respectively. If the new contact plan is accepted, we reduce the temperature  $T$  by a cooling rate  $r$ . The algorithm stops when a set number of iterations  $N_{it}$  is reached.

The best delivery time between two nodes  $d(i, j, t)$  is typically calculated using algorithms such as Contact Graph Dijkstra Search (Algorithm 2, Appendix 5), a modified Dijkstra search that finds the best route through a contact graph [16], which is a graph representation of the contact plan that allows computing routes between nodes. However, since the above method can be computationally expensive for large networks, the following sections present a learning-based approach based on DGNNs.

#### 3.2 Dynamic Graph Neural Networks

The objective function is evaluated by inputting a contact plan encoded by a sequence of matrices  $U = (U_1, \dots, U_{N_t})$ . This sequence can be interpreted as a dynamic graph, where each matrix  $U_t$  represents

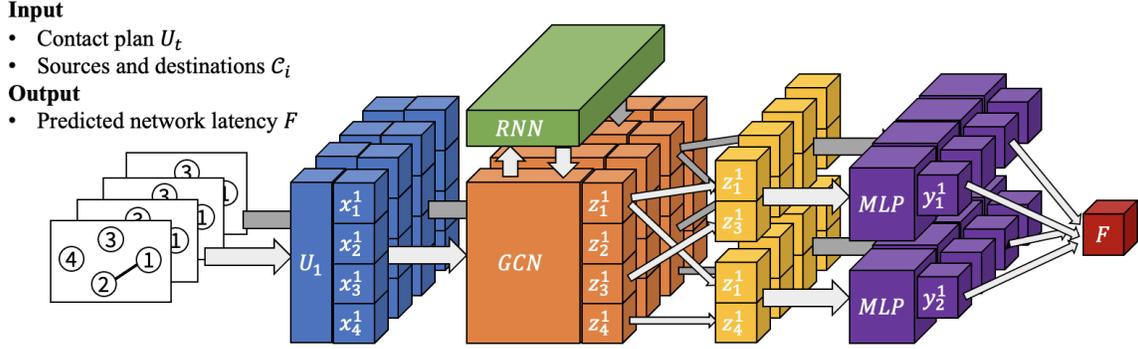


Fig. 1: Dynamic graph neural network architecture for predicting latency in satellite constellation networks.

the adjacency matrix at a given time step. Therefore, we propose using a dynamic graph neural network (DGNN) to learn the objective function and use it to evaluate the contact plan performance. More specifically, we represent the contact plan  $U$  as a sequence of graphs  $G = (G_1, \dots, G_{N_t})$ , with  $G_t = (\mathcal{V}, \mathcal{E}_t)$ , where  $\mathcal{V} = \mathcal{S} \cup \mathcal{G}$  is the set of nodes and  $\mathcal{E}_t$  is the set of edges at time  $t$ . Each edge  $e_{ij} \in \mathcal{E}_t$  represents a link between nodes  $i$  and  $j$  at time  $t$ , as specified by the contact plan matrix at that time instance  $U_t(i, j)$ .

The selected DGNN model to learn the objective function  $F$  is EvolveGCN [17]. One of its main features is handling the addition and removal of nodes after training, overcoming the limitation of other methods in learning these irregular behaviors. The authors propose using a Recurrent Neural Network (RNN) to regulate a Graph Convolutional Network (GCN) model (i.e., network parameters) at different time steps. This approach effectively performs what is known as model adaptation by focusing on the model itself rather than the node embeddings. Therefore, the change of nodes poses no restriction, making the model sensitive to new nodes without historical information.

As in the original paper’s nomenclature [17], we will use subscript  $t$  to denote the time index, superscript  $l$  to denote the GCN layer index,  $n$  for the number of nodes—in our case  $n = N_s + N_g$ . At a time step  $t$ , the input data to the model consists of the pair  $(A_t \in \mathbb{R}^{n \times n}, X_t \in \mathbb{R}^{n \times d})$ , where the first element is the graph adjacency matrix—in our case, obtained from the contact plan  $U_t$ —and the second is the matrix of input node features. Specifically, each row of  $X_t$  is a vector of node  $d$  features.

The GCN consists of  $L$  layers of graph convolution, which includes a neighborhood aggregation that combines information from neighboring nodes. At time  $t$ , the  $l$ -th layer takes as input the adjacency matrix  $A_t$  and the learnable node embedding matrix  $H_t^{(l)}$ , and

uses the learnable weight matrix  $W_t^{(l)}$  to update the node embedding matrix to  $H_t^{(l+1)}$

$$H_t^{(l+1)} = \text{GCONV}(A_t, H_t^{(l)}, W_t^{(l)}) \quad (6)$$

$$:= \sigma(\hat{A}_t H_t^{(l)} W_t^{(l)}) \quad (7)$$

where  $\hat{A}_t$  is the normalized adjacency matrix and  $\sigma$  is the activation function. The initial embedding matrix is obtained from the node features,  $H_t^{(0)} = X_t$ .

The central component of the model is the update of the weight function  $W_t^{(l)}$  at each time step. The weights are updated by an rnnRNN that takes as input the node embedding matrix  $H_t^{(l)}$  and the previous weight matrix  $W_{t-1}^{(l)}$  and outputs the updated weight matrix  $W_t^{(l)}$  as

$$W_t^{(l)} = \text{RNN}(H_t^{(l)}, W_{t-1}^{(l)}) \quad (8)$$

Combining the graph convolution with the recurrent architecture, the authors define the evolving graph convolution unit, which is the basic building block of the EvolveGCN model.

Since we assume the satellites and ground stations that each satellite communicates with,  $\mathcal{C}_i$ , as known, for each time step, we create link embeddings by concatenating the embeddings of the source and destination nodes. That is, we concatenate the embeddings of the final layer at different time steps,  $H_1^{(L)}, \dots, H_{N_t}^{(L)}$ , to compute the objective function  $F$  using a fully connected neural network with a single output and average pooling.

#### 4. Results

In this section, we first present the experimental setup, including the parameters of the DGNN model and training. Then, we present the results of the contact plan design based on simulated annealing using a DGNN for evaluating the objective.

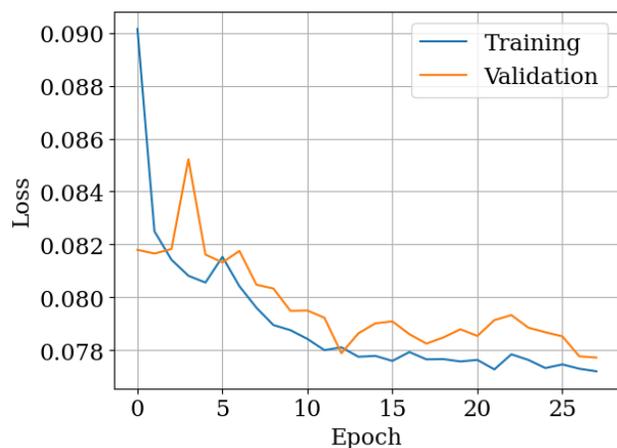


Fig. 2: Training and evaluation loss. The model is trained for 16 hours using synthetic data corresponding to 30 satellites and 20 ground stations. Hyperparameters are selected using a grid search, including different activation and loss functions, as well as the number of layers and sizes.

#### 4.1 DGNN Model

Due to the time required to compute the objective function for large constellations and generate training data, we trained the final model for 16 hours using synthetic data corresponding to 30 satellites and 20 ground stations. We performed hyperparameter tuning, in which we considered different activation functions and loss functions, namely  $L_1$  and  $L_2$ , as well as the number of layers and size for the graph neural network, the recurrent neural network, and the fully connected neural network. We also considered different architectures, such as a standalone GCN, and different ways of processing the node embeddings to predict the objective function.

The best-performing model is EvolveGCN-O, consisting of 2 layers of graph convolution with 64 hidden units, a fully connected neural network with 2 layers of 64 hidden units, and a single output. The model is trained using the Adam optimizer with a learning rate of 0.001 and the  $L_1$  loss function. The graph convolution layer has Randomized Leaky Rectified Linear Units (RReLU) as the activation function, and the fully connected neural network has Rectified Linear Units (ReLU) as the activation function. Figure 2 shows the training and validation loss for the selected model. The implementation uses the Deep Graph Library [18] and PyTorch [19].

Figure 3 shows the predicted and true normalized BDT for 100 different contact plans. The model successfully identifies contact plans with worse objective

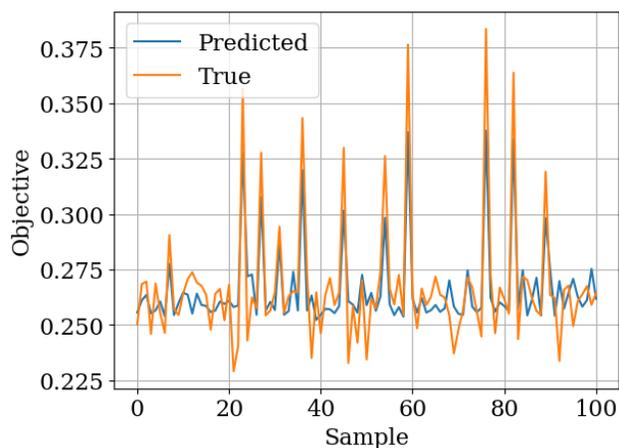


Fig. 3: Predicted and true normalized BDT for 100 different contact plans. The model successfully identifies contact plans with worse objective values and achieves lower accuracies on contact plans with lower objectives. It predicts the BDT of a contact plan with a mean absolute error of 3.6 minutes.

values and achieves lower accuracies on contact plans with lower objectives. This indicates that the model can learn the objective function and can be used to evaluate the performance of a given contact plan, which is the main objective of this work. Moreover, we expect better performance at the beginning of the optimization, when the contact plan has higher objective values, and worse performance at the end when the contact plan has lower objective values.

Fig. 4 shows the different routes from satellite 9 to satellite 28. The best route is colored in blue, passing through satellites 9, 0, 23, 10, 14, and 28, with a Best Delivery Time (BDT) of 8 minutes. The proposed DGNN can predict the average BDT of the network with a mean absolute error of 3.6 minutes. The different paths involving multiple satellites highlight the complexity of the objective function, which is not only dependent on the distance between the source and destination nodes but also on the dynamic topology of the network.

#### 4.2 Contact Plan Design Results

We assume that each satellite is only equipped with a single ISL and that inter-satellite communication or downlink cannot occur simultaneously. Regarding the optimization algorithm based on SA, we generate the initial random contact plan by solving a maximum matching problem, a well-known combinatorial optimization problem consisting of finding a maximum cardinality matching in a graph. This readily enforces the symmetry and visibility constraints. The constraints

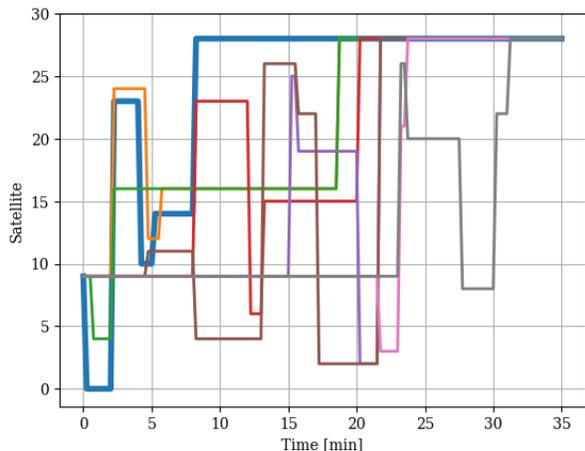


Fig. 4: Possible routes from satellite 9 to satellite 28. The best route is colored in blue, passing through satellites 9, 0, 23, 10, 14, and 28, with a Best Delivery Time (BDT) of 8 minutes. The proposed DGNN can predict the average BDT of the network with a mean absolute error of 3.6 minutes.

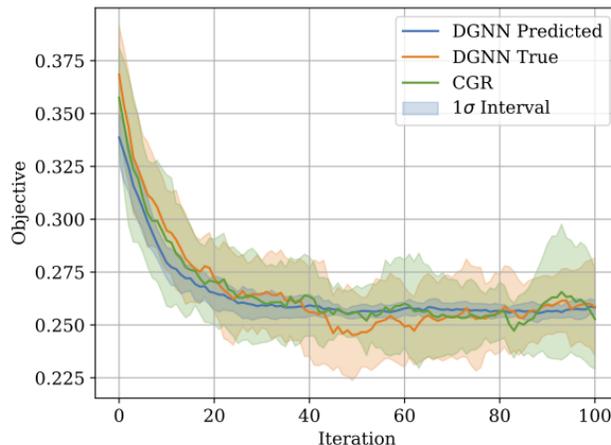


Fig. 5: Objective improvement using the DGNN to evaluate the objective function. The predicted objective values using the trained model are shown in blue, and the values computed a posteriori using the CGR algorithm are shown in orange. The shaded area represents the standard deviation over 10 runs.

Table 1: Normalized average best delivery time (BDT) of the initial and optimized plans and computing times when using CGR and the proposed DGNN to compute the objective function. The results show the mean and standard deviation over 10 runs.

Objective Function	CGR	DGNN
Initial normalized BDT	$0.36 \pm 0.02$	$0.37 \pm 0.02$
Final normalized BDT	$0.25 \pm 0.02$	$0.26 \pm 0.02$
Improvement	$29.6 \pm 7.3\%$	$29.1 \pm 7.34\%$
Computing time (100 it.)	$19.1 \pm 0.8$ min	$0.9 \pm 0.2$ min

related to the maximum number of contacts per satellite are enforced by recursively deactivating links.

We implemented the contact plan design based on simulated annealing in Algorithm 1 (Appendix 5) in Python. To generate a new plan, we randomly activate and deactivate new links while always ensuring constraint satisfaction. The initial temperature is set to  $T = 10$ , the cooling rate is set to  $r = 0.95$ , and the number of iterations is set to  $N_{it} = 100$ . It is essential to mention that when computing the BDT, we consider the delay from the average distance between satellite pairs.

Fig. 5 shows the decrease of the function value of the contact plan throughout the optimization using the trained model to predict the objective function. We include both the values predicted by the model during optimization as well as the true values computed a posteriori. The algorithm tends to converge after approxi-

mately 100 iterations, as better plans are not found and the temperature is reduced. We also observe how the predicted values are close to the true values during the early phases of the optimization when plans have more significant delays. However, the model’s performance decreases as the optimization progresses, and the plans have lower delays, indicating that the model cannot generalize well to the entire range of delays. This is shown by the large difference in standard deviation between the blue and the orange lines, representing the true values computed a posteriori. The model tends to predict a mean value close to the true value but does not capture the variance of the objective function.

Nonetheless, the proposed method is able to design a contact plan for the simulation use case successfully. Thanks to the initialization method based on constraint satisfaction and the implemented algorithm based on SA, we ensure that the contact plan is always feasible, allowing us to use the DGNN to evaluate the objective function.

Table 1 shows the normalized average BDT, i.e., the objective function value of the initial and optimized plans, and the computing times when using CGR and the proposed DGNN to compute the objective function. We observe that the optimized contact plan using the DGNN is able to improve the BDT by 29.1% compared to the initial contact plan, achieving similar results to optimizing using the CGR. However, using the DGNN, we can perform the objective evaluations 20x faster than when using CGR, which significantly

improves computational performance. Although, as shown in Fig. 5, the DGNN is not able to generalize well to the entire range of delays, it can provide a good approximation of the objective function and allow the optimization algorithm to find a good solution.

## 5. Conclusions

Designing a contact plan for large heterogeneous satellite networks necessitates consideration of the trade-off between performance and scalability. In this paper, we initially present a problem formulation for the contact plan design, taking into account traffic source and destination nodes without making any assumptions about the periodicity of the evolving satellite network topology.

To reduce the computational burden of computing the objective function for large satellite networks and tackle the need to update the contact plan periodically, we propose modeling the objective function, i.e., the average best delivery time of the network, using a dynamic graph neural network. This allows resorting to a metaheuristic algorithm based on simulated annealing.

The presented architecture captures the spatial and temporal information of the satellite network using a hybrid model that involves graph convolution and a recurrent neural network, being able to predict the average BDT of the network with a mean absolute error of 3.6 minutes. Simulation results show that the proposed method is able to design a contact plan for a large satellite network successfully, improving the BDT by 29.1%, achieving similar results to optimizing using CGR but performing the objective evaluations 20x faster.

Future work will focus on analyzing the proposed solution in scenarios with contact plan distribution, extending the model to predict network metrics with traffic information of the satellites, and extracting network and topology information from the learned model. While this work focuses on a specific instance of the contact plan design problem, we envision that the proposed learning-based method involving DGNNs opens the door to new solutions for autonomy in large, heterogeneous satellite networks.

## Acknowledgment

This work was supported by the “laCaixa” Foundation fellowship (ID 100010434, code LCF/BQ/EU21/11890112). This work was co-funded by the Government of Catalonia in the scope of the NewSpace Strategy for Catalonia and by the Spanish Ministry of Economic Affairs and Digital Transformation and the European Union–NextGeneration EU, in the framework of the Recovery Plan, Transformation and Resilience (PRTR) (Call UNICO I+D 5G 2021, ref. number TSI-063000-2021-5-6GSatNet-SS).

## A: Algorithms

This appendix presents the algorithms used in this work, namely the contact plan design based on simulated annealing (Algorithm 1) and the contact graph Dijkstra search (Algorithm 2).

---

**Algorithm 1:** Contact plan design based on simulated annealing. [15]

---

**Data:** adjacency matrices  $A_t$ , temperature  $T$ , cooling rate  $r$ , number of iterations  $N_{it}$

**Result:** contact plan  $U$

$U \leftarrow \text{InitialContactPlan}(A)$  ;

$F_{best} \leftarrow \text{ObjectiveFunction}(U)$  ;

**for**  $i \leftarrow 1$  **to**  $N_{it}$  **do**

$U' \leftarrow \text{GetNeighbor}(U)$  ;

$F' \leftarrow \text{ObjectiveFunction}(U')$  ;

**if**  $F' \leq F_{best}$  **then**

$U_{best} \leftarrow U \leftarrow U'$  ;

$F_{best} \leftarrow F'$  ;

**else if**  $\exp((F_{curr} - F_{new})/T) \leq \text{rand}(0, 1)$  **then**

$U \leftarrow U'$  ;

$T \leftarrow r \cdot T$  ;

**else**

$U \leftarrow U_{best}$  ;

---



---

**Algorithm 2:** Contact Graph Dijkstra Search. [16]

---

**Data:** Root contact  $C_{root}$ , source  $S$ , destination  $D$ , contact plan  $U$

**Result:** Route  $R_S^D$ , and best delivery time  $BDT$

$R_S^D \leftarrow \{\}$  ;

$C_{fin} \leftarrow \{\}$  ; // final contact

$BDT = \infty$  ; // final arrival time

$C_{curr} = C_{root}$  ; // current contact is root

**/\* contact plan exploration loop \*/**

**while**  $True$  **do**

**/\* contact review procedure \*/**

$C_{fin}, BDT = \text{CRP}(U, C_{curr}, C_{fin}, BDT)$  ;

**/\* contact selection procedure \*/**

$C_{next} = \text{CSP}(U, C_{curr}, BDT)$  ;

**if**  $C_{next}$  **then**

$C_{curr} \leftarrow C_{next}$

**else**

**break**

**if**  $C_{fin} \neq \{\}$  **then**

$C = C_{fin}$  ;

**while**  $C \neq C_{S,S}^{0,\infty}$  **do**

$R_S^D.hops \leftarrow \{C\}$  ;

$C = C.pred$  ; // previous contact

        Compute  $(R_S^D.tx\_win, R_S^D.volume)$  ;

---

## References

- [1] O. Kodheli, E. Lagunas, N. Maturo, S. K. Sharma, B. Shankar, J. F. M. Montoya, J. C. M. Duncan, D. Spano, S. Chatzinotas, S. Kisseleff, J. Querol, L. Lei, T. X. Vu, and G. Goussetis, "Satellite Communications in the New Space Era: A Survey and Future Challenges," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 1, pp. 70–109, 2021.
- [2] Euroconsult, "Earth observation data & services market: A 360° view of the demand and supply related to the earth observation data and services market," December 2022. 15th Edition.
- [3] G. Curzi, D. Modenini, and P. Tortora, "Large Constellations of Small Satellites: A Survey of Near Future Challenges and Missions," *Aerospace*, vol. 7, p. 133, Sept. 2020.
- [4] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '03, (New York, NY, USA), pp. 27–34, Association for Computing Machinery, Aug. 2003.
- [5] C. Araguz, E. Bou-Balust, and E. Alarcón, "Applying autonomy to distributed satellite systems: Trends, challenges, and future prospects," *Systems Engineering*, vol. 21, pp. 401–416, Sept. 2018.
- [6] J. A. Fraire and J. M. Finochietto, "Design challenges in contact plans for disruption-tolerant satellite networks," *IEEE Communications Magazine*, vol. 53, pp. 163–169, May 2015.
- [7] Y. Wang, M. Sheng, J. Li, X. Wang, R. Liu, and D. Zhou, "Dynamic Contact Plan Design in Broadband Satellite Networks With Varying Contact Capacity," *IEEE Communications Letters*, vol. 20, pp. 2410–2413, Dec. 2016.
- [8] D. Zhou, M. Sheng, X. Wang, C. Xu, R. Liu, and J. Li, "Mission Aware Contact Plan Design in Resource-Limited Small Satellite Networks," *IEEE Transactions on Communications*, vol. 65, pp. 2451–2466, June 2017.
- [9] J. Fraire and J. M. Finochietto, "Routing-aware fair contact plan design for predictable delay tolerant networks," *Ad Hoc Networks*, vol. 25, pp. 303–313, Feb. 2015.
- [10] J. A. Fraire, P. G. Madoery, J. M. Finochietto, and G. Leguizamón, "An evolutionary approach towards contact plan design for disruption-tolerant satellite networks," *Applied Soft Computing*, vol. 52, pp. 446–456, Mar. 2017.
- [11] H. Yan, Q. Zhang, Y. Sun, and J. Guo, "Contact plan design for navigation satellite network based on simulated annealing," in *2015 IEEE International Conference on Communication Software and Networks (ICCSN)*, pp. 12–16, June 2015.
- [12] W. Shi, D. Gao, H. Zhou, B. Feng, H. Li, G. Li, and W. Quan, "Distributed contact plan design for multi-layer satellite-terrestrial network," *China Communications*, vol. 15, pp. 23–34, Jan. 2018.
- [13] J. Skarding, B. Gabrys, and K. Musial, "Foundations and modelling of dynamic networks using Dynamic Graph Neural Networks: A survey," *arXiv:2005.07496 [cs, stat]*, May 2020.
- [14] J. Suarez-Varela, P. Almasan, M. Ferriol-Galmes, K. Rusek, F. Geyer, X. Cheng, X. Shi, S. Xiao, F. Scarselli, A. Cabellos-Aparicio, and P. Barlet-Ros, "Graph Neural Networks for Communication Networks: Context, Use Cases and Opportunities," *IEEE Network*, pp. 1–8, 2022.
- [15] D. Bertsimas and J. Tsitsiklis, "Simulated annealing," *Statistical science*, vol. 8, no. 1, pp. 10–15, 1993.
- [16] J. A. Fraire, O. De Jonckère, and S. C. Burleigh, "Routing in the Space Internet: A contact graph routing tutorial," *Journal of Network and Computer Applications*, vol. 174, p. 102884, Jan. 2021.
- [17] A. Pareja, G. Domeniconi, J. Chen, T. Ma, T. Suzumura, H. Kanezashi, T. Kaler, T. B. Schardl, and C. E. Leiserson, "EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs," *arXiv:1902.10191 [cs, stat]*, Nov. 2019.
- [18] M. Wang, D. Zheng, Z. Ye, Q. Gan, M. Li, X. Song, J. Zhou, C. Ma, L. Yu, Y. Gai, *et al.*, "Deep graph library: A graph-centric, highly-performant package for graph neural networks," *arXiv preprint arXiv:1909.01315*, 2019.
- [19] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.