Population Based Routing in LEO Satellite Networks

Anders Fongen
Norwegian Defence University College (FHS)
Lillehammer, Norway
email: anders@fongen.no

Lars Landmark
Norwegian Defence Research Establishment (FFI)
Kjeller, Norway
email: lars.landmark@ffi.no

***Abstract*—Packet switching in a Low Earth Orbit (LEO) satellite network may calculate the best traffic route through the less populated areas of the planet, in order to avoid relaying through the busiest satellites. A number of ideas for route calculation in a LEO system have been evaluated and the performance results are presented. This paper reports from ongoing research on Space Information Networks (SIN) with the purpose to offer application services in a LEO satellite network to mobile users. The conclusion is that route calculation based on population density gives a moderate, but significant improvement in resource utilization.**

Keywords*—*LEO satellites; space information networks; population density; mobile computing.

I. INTRODUCTION

The term SIN describes an information system located in space [1]. The concept is an evolution of satellite networks as they are known from the 1960s to present day, where satellites evolve from "radio mirrors" with plain wideband transponders, towards networks of interconnected satellites providing connectivity services based on store-and-forwarding of data packets. This evolution represents an *increasing system complexity* in the spacecrafts. It is a reasonable prediction that future LEO systems will offer application services and even an *Application as a Service* (AaaS) platform.

Which advantages can be achieved through the deployment of a SIN? Two main characteristics of the services distinguish a SIN from ordinary Internet services:

1) Global coverage for mobile clients,
2) Very low latency.

The round-trip time through a satellite at 500 km altitude can be as low as 3.3 milliseconds. Low latency will drive new time sensitive cooperative applications like remote surgery, autonomous aircrafts, etc., and is also one key property of 5G mobile networks.

The general design framework and a presentation of related problems and research questions have been previously presented in [2]. Beside the planned information services presented in that paper, a packet forwarding service will always be necessary as a baseline service to applications which cannot be designed along the guidelines for a SIN information service.

The chosen approach to the investigation of routing algorithms has included the earth's population distribution as a parameter. Satellites flying over densely populated areas of the earth are assumed to be busy serving requests from surface clients, and should not be burdened by additional packet relay tasks. The position, and thus the population density within the satellite's footprint can be calculated by anyone at any time and taken into account for the route calculations. Links between satellites can be predicted based on their relative positions so a link discovery protocol and subsequent link state distribution is unnecessary. The effect of this approach has been evaluated through simulation in a software model.

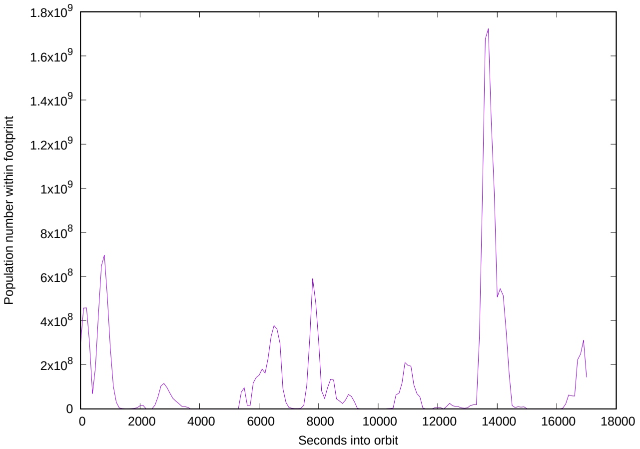

Figure 1. The population number inside the footprint of a satellite during three subsequent orbits.

The contributions of this paper are the results from a novel approach to routing optimization in LEO satellite networks, where the population density serves as a factor in link weight calculations.

The remainder of this paper is organized as follows: In Section II, the design rationale for the use of population data in the route calculations is presented. Section III briefly presents related research, and Sections IV and V describe the software based simulation platform and the satellite constellation chosen for the experiments. Sections VI and VIII present the evaluation of the three first routing methods and the delay-tolerant routing method. Section IX presents conclusions and topics for future research.

II. DESIGN RATIONALE

An important choice in our SIN studies is to include the earth's population density into the traffic analysis and resource planning. In particular for lower altitudes, the satellites will spend large fractions of their time over inhabited areas, mixed with shorter intervals of extremely high density. Figure 1 shows the population number inside the footprint of a satellite during three subsequent orbits (283 minutes). The orbits cover different great circles of the planet and, therefore, show different results. It is likely that the rate of incoming requests will follow a similar pattern. An appealing idea is to allow idle satellites to offload busy ones, since neighbouring satellites in the network can communicate through high speed inter-satellite links. This approach to resource planning is extending the traditional design of LEO satellite systems.

A comprehensive model of a LEO system has been modeled in software and has allowed a wide range of design ideas to be evaluated for efficiency and resource consumption.

III. RELATED RESEARCH

The term SIN has been used to describe networks of satellites and high altitude aircrafts (drones, balloons) with different service levels. Existing satellite networks like Iridium and Starlink [3] offer only communication services, the latter on a very large scale and with high bandwidth. A number of authors have proposed "Cloud Computing in Space" through the addition of larger satellites with sufficient energy and computing resources for taking on these tasks [4][5].

The concept of "Cloud Computing in Space " has strong relations to "Edge computing" (EC), and there are several studies on how to solve resource management problems when EC is applied to a LEO system. Li et al. [6] analyse some of these problems as well as provide a survey on similar studies. European Telecommunications Standards Institute (ETSI) engages in Mobile EC as a service oriented strategy for mobile network, but, like most other efforts, does not deal with the problems of frequent hand-over operations of services with a large state space. Also, none of the mentioned efforts propose the application of population density data as a parameter for their resource planning.

The results presented in this paper will not deal with technical details in the communication technology, but rather view the SIN as a distributed system. The authors are not aware of other efforts to investigate routing mechanisms based on population density.

IV. SOFTWARE MODEL

Based on the design rationale presented in Section II, a software model, written in Java, is in continuous development for the purpose of testing different hypotheses related to the operation of a SIN. With this model, we have studied link formation, request traffic formation and distribution, cache performance, routing efficiency, state transfer during handover etc.

The software model is equipped with population density data [7] which is used to calculate the population number within reach of the satellite at a given altitude. The colorized backdrop in Figure 2 illustrates the population distribution as seen from 500 km altitude.

Furthermore, the software allows any number of satellites to have common or individual orbital elements, although a simplification has been introduced in the current version to assume that all orbits are circular, not elliptic.

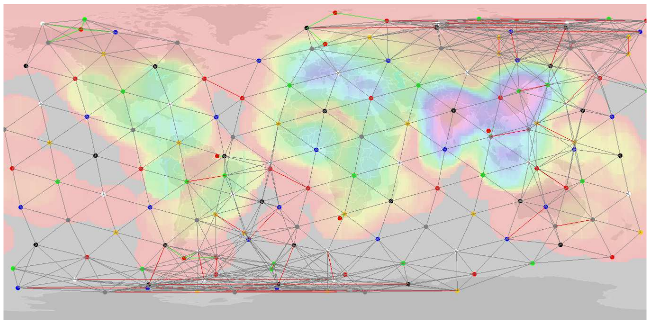

Figure 2. Screenshot from the satellite constellation model

During experimentation with the software model, it has been possible to establish a constellation with a reasonable number of satellites, which is able to form a completely connected grid yet with an altitude low enough to allow for very low latency and inexpensive and lightweight ground terminals.

V. A CANDIDATE CONSTELLATION FOR STUDY

Satellite networks servicing civilian mobile clients using handheld equipment tend to operate in LEO. E.g., the orbit altitude of Iridium satellites is 781 km, which allows for lightweight ground terminals without the need for antenna deployment. The inclination of the orbit can be made so steep that the polar regions are fully covered, or given a lower angle to spend more time over the densely populated latitudes closer to the equator.

The choice of orbit altitude determines the diameter of the *footprint*, e.g., the circular region of the earth's surface with potential connectivity, and also the maximum communication distance for inter-satellite links and uninterrupted service for ground terminals. Simply stated, a lower orbit altitude reduces the design constraints on the ground terminals and provides higher communication capacity, but increases the cost due to the higher required number of spacecrafts.

Figure 2 shows a screenshot from the software model, containing 150 satellites with an orbit inclination of 75 degrees and an altitude of 500 km. The colors on the backdrop indicate the population density inside the footprint of a satellite in that position (contrary to the local density at that exact position). The population density is considered to be a parameter for the estimation of the request rate received from ground surface clients. Other possible parameters, like regional Internet penetration, time of day and day of week may be taken into account at a later time. Changes in the demographics are assumed to be slow and are not taken into account.

VI. ROUTING DECISIONS IN A LEO NETWORK

The purpose of this paper is to evaluate different routing methods. Even though some LEO networks (e.g., Starlink) route traffic through terrestrial networks, the focus will be kept on routing through inter-satellite links. The availability of these links is predictable, as stated in Section I.

Four different algorithms for route calculation have been evaluated:

1) Hot potato: Establish the geographical direction (bearing) to the receiver and choose the link with the direction nearest to this bearing.
2) Unweighted Dijkstra: Use Dijkstra's shortest path algorithm with all links given the same cost.
3) Weighted Dijkstra: Same as above, but with links given weight according to the population density within the receiver's footprint.
4) Delayed weighted Dijkstra: Same as above, but with an introduction of a delay to wait for a better route to occur.

The evaluation of the different algorithms was based on the calculation of the total population number within the footprints of the satellites along the route. The assumption is that a smaller value of this number indicates that mostly idle satellites are involved in the packet forwarding, and consequently an improved resource utilization.

Although the familiar Dijkstra's algorithm is used, this is still a source routing approach where the full route is calculated by the sender and attached to the message.

All four algorithms were tested on the same route going between ground stations in Lillehammer, Norway and Bankok, Thailand.

VII. EVALUATION OF THE ROUTE ALGORITHMS

This section will present experimental details and results of the four chosen routing algorithms listed in Section VI.

A. *Hot potato*

One simple routing algorithm is to simply "throw" the packet towards the destination. This is accomplished by selecting a link pointing in the best direction. Since the location of the target and the directly linked neighbors are known, this is just a matter of computing bearings and comparing angles between the resulting vectors.

As indicated on the example shown in Figure 3, an efficient route is selected with a nearly optimum number of hops, but is also prone to looping and the introduction of a time-to-live element in the message was needed to remove them from endless cycles. The packet loss ratio was, therefore, observed to be higher than the other alternatives.

B. *Unweighted Dijkstra*

The traditional Dijkstra's Shortest Path algorithm was applied to the link collection. Every link was given the same cost, which optimizes the number of hops only, without regard to the population number below.

In contrast to the hot potato method, a link is not selected unless it is a part of the path to the target, so looping does not occur and potential packet loss is detected during path calculation. Less resources are thus wasted on packet loss occurring after several hops.

One resulting path is shown in Figure 4, with the same number of hops as the hot potato method. The focus on the number of hops rather than bearing allows this method to choose other paths still effective, e.g., over the polar regions.

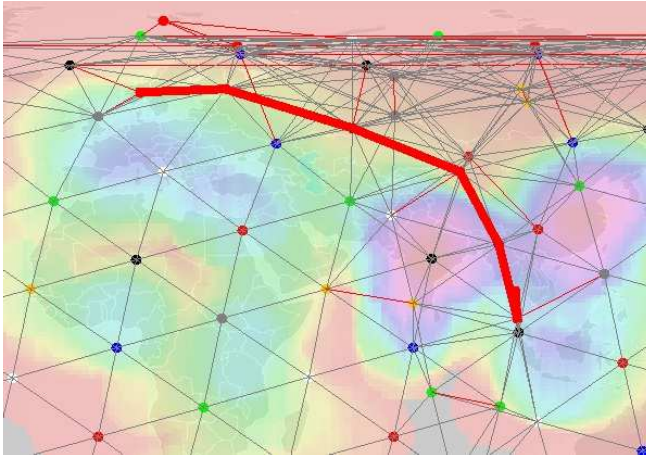

Figure 3. Example result from the hot potato routing algorithm.

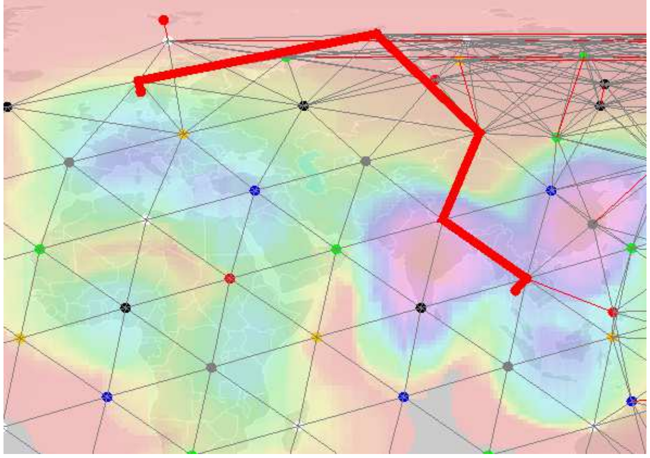

Figure 4. Example result from the unweighted Dijkstra's routing algorithm.

C. *Weighted Dijkstra method*

The weighted Dijkstra method is similar to the method applied in Section VII-B, but with link costs applied according to the population number inside the footprint of the satellite at the other end of the link. Links with zero population below were given a minimum cost to avoid ridiculously long routes over "free" links.

This method demonstrates the routing principle proposed in this paper; the two former methods are presented as baseline methods for the estimation of the effect of population-based routing calculations. Figure 5 shows one example of a route between Lillehammer and Bankok calculated using this method. It is clearly shown how the route now avoids the densely populated areas, but has a higher number of hops.

D. *Comparing the three methods*

The efficiency of the three aforementioned routing methods will now be compared. Since the satellite grid is in constant movement, the calculated cost for any route will change over time. For this reason, each of the three routes were calculated repeatedly over 2500 seconds, and the calculated

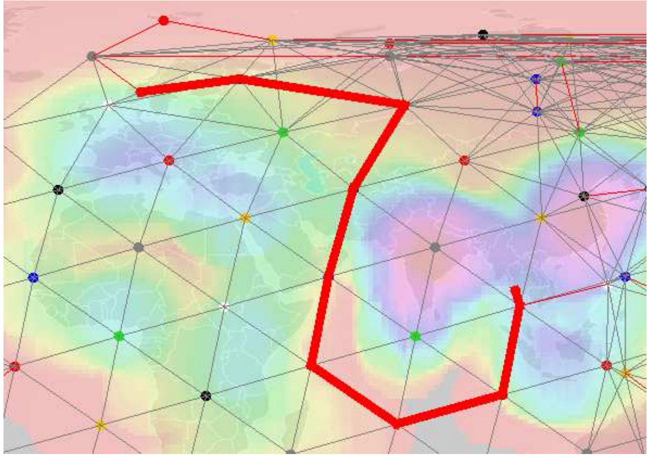

Figure 5. Example result from the weighted Dijkstra's routing algorithm.

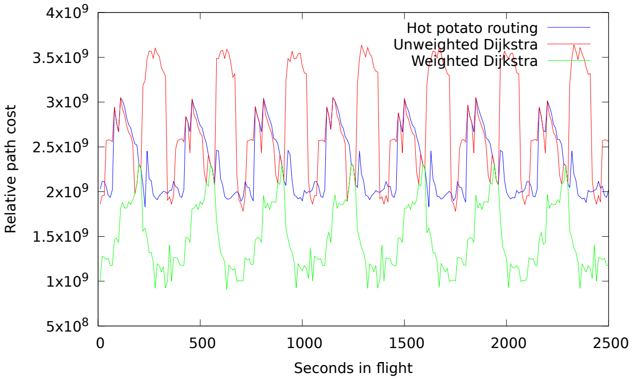

Figure 6. Routing cost for three different routing methods, plotted over time. The numbers on the y-axis should be regarded as relative, since the population dataset does not sum up to the actual earth's population.

costs plotted as shown in Figure 6. The plot offers some interesting observations:

1) The ratio between highest and lowest cost is no more than 3.5
2) The cost variations over time are smaller for the hot potato method than the two methods based on Dijkstra's algorithm
3) All three methods show a strong cyclic behaviour with a period of approximately 350-400 seconds.

This use of Dijkstra's algorithm is a source routing technique which allows many aspects to be taken into account for the routing decisions. Beside regional Internet penetration etc., policy decisions based on the sender's preferences on commercial, security and technical issues may have an impact on the chosen route.

VIII. A PROPOSED DELAY-TOLERANT ROUTING METHOD

TABLE I
RESULTING QUEUING DELAY AND ROUTING COST WHEN DIFFERENT PERCENTILES ARE USED AS SENDING THRESHOLDS

Percentile	Queuing delay (secs)	Routing cost ($\cdot 10^7$)
10	855.92	99.35
20	544.46	105.08
30	91.38	111.11
40	53.79	116.63
50	29.14	119.41
60	12.98	123.83
70	5.78	126.79
80	2.83	127.98
90	0.65	130.05

The strong cyclic properties of all the three routing methods evaluated in Section VII suggest the design of a routing calculation reserved for delay-tolerant traffic. The average routing cost can be reduced if we allow the packet to wait for an opportunity to send when the cost is relatively low. If successful, and acceptable for the communicating service, this method may even out the communication load of the satellites and in general improve the resource utilization of the system.

In order to evaluate this method, the routing cost between the endpoints was sampled every 10 seconds and the different percentiles of the cost distribution were calculated. Messages sent from the transport layer are queued until the calculated routing cost is lower than the required percentile. The average routing cost and queuing delay are measured and presented in Table I.

As expected, a lower percentile will cause a longer queuing delay, which is also apparent in the table. A little more surprising is the relatively small improvement (25%) gained through a much longer queuing delay. The results indicate that this method is not worth the efforts of implementation.

IX. CONCLUSION

Effective routing through a grid of LEO satellites has been the topic of this paper, where the population density on the surface below a satellite determines the link cost. The rationale for doing so was to employ communication resources in idle satellites flying over inhabited areas of the planet. The experiments conducted on a software based simulation platform show clearly the effect of route calculations based on these criteria. Future experiments will combine these experiments during actual service production to study the effect on their response times.

REFERENCES

[1] L. Bai, T. de Cola, Q. Yu, and W. Zhang, "Space information networks," *IEEE Wireless Communications*, vol. 26, no. 2, pp. 8–9, 2019.
[2] A. Fongen, "Application services in space information networks," in *CYBER 2021*. Barcelona, Spain: IARIA, Oct 2021, pp. 113–117.
[3] "Starlink web site," https://www.starlink.com/, [Online; retrieved 03-May-2022].
[4] S. Briatore, N. Garzaniti, and A. Golkar, "Towards the internet for space: Bringing cloud computing to space systems," in *36th International Communications Satellite Systems Conference (ICSSC 2018)*, 2018, pp. 1–5.
[5] S. Cao *et al.*, "Space-based cloud-fog computing architecture and its applications," in *2019 IEEE World Congress on Services (SERVICES)*, vol. 2642-939X, 2019, pp. 166–171.
[6] C. Li, Y. Zhang, R. Xie, X. Hao, and T. Huang, "Integrating edge computing into low earth orbit satellite networks: Architecture and prototype," *IEEE Access*, vol. 9, pp. 39 126–39 137, 2021.
[7] "Gridded population of the world v.4.11," [Online; retrieved 03-May-2022]. [Online]. Available: https://sedac.ciesin.columbia.edu/data/collection/gpw-v4/sets/browse