# Network verification on periodically changed topology

Fangdan Ye[1,2] Yahui Li[3,2] Zhiliang Wang[1,2] Jiangyuan Yao[4]
Yazheng Chen[1,2] Tianming Lan[1,2] Hewu Li[1,2] Xingang Shi[1,2] Xia Yin[3,2]
[1] Institute for Network Sciences and Cyberspace, Tsinghua University, China
[2] Beijing National Research Center for Information Science and Technology, China
[3] Department of Computer Science and Technology, Tsinghua University, China
[4] School of Computer Science and Cyberspace Security, Hainan University, China

## ABSTRACT

Network verification allows network operators to find existing or potential errors in network by verifying network properties. However, current network verification tools can only verify network with specific topology. In this paper, we propose a use case that needs to verify the network whose topology changes periodically, e.g. the space-terrestrial integrated network, and a methodology to verify this kind of network with existing network verification tools.

## CCS CONCEPTS

• **Computer systems organization → Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

## KEYWORDS

Network verification

**ACM Reference Format:**
Fangdan Ye[1,2] Yahui Li[3,2] Zhiliang Wang[1,2] Jiangyuan Yao[4] and Yazheng Chen[1,2] Tianming Lan[1,2] Hewu Li[1,2] Xingang Shi[1,2] Xia Yin[3,2]. 2019. Network verification on periodically changed topology. In *APNet 2019: Asia-Pacific Workshop on Networking, August 17-18 2019, Beijing, China.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/1122445.1122456

## 1 INTRODUCTION

As network structure and topology for users such as ISP or enterprise are growing more and more complex, network operators need much more time and effort to guarantee stability and reliability of network. Network verification can help operators to verify if network properties satisfy their expectation. Many network verification tools such as NOD[4], SymNet[5], Batfish[2] and Minesweeper[1] can take the forwarding information base or configurations as input and give the result of whether the network property we query such as reachability, loop-free or way-point is always satisfied.

However, existing verification tools verify network properties on one specific topology. When the topology gets changed, the verification result of network properties may get different.

*APNet 2019, August 17-18 2019, Beijing, China*
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

The space-terrestrial integrated network[3] has received widespread attention in recent years. Because the space network has great advantages in coverage and access, integrating the space network and the terrestrial network can bring convenience and flexibility to network users. Significantly, people design the space network as a multi-level satellite network. Satellites in the space network rotate around the earth in different cycles, and each satellite has its own periodic time.

If we hope to verify the space-terrestrial integrated network, obviously we cannot use existing network verification tools directly, because these tools don't support the input of a topology that changes periodically.

In this paper, we further abstract the use case into a general problem. Then we propose a methodology to convert a periodically changed topology into several static topologies. Finally we give an evaluation that verifies a periodically changed topology with Batfish.

## 2 DESIGN

### 2.1 Problem abstraction

Firstly, the topology of the satellite network can be regarded as a graph. Satellites are nodes, and the wireless links between them are edges.

We define the least common cycle $M$. When a common cycle ends, all satellites just return to their original position, and $M$ is the least one.

Then we define the following variables:

$T = [a, b]\ \ (0 \le a, b \le M)$ as a time interval in the least common cycle $M$.

$C = \{T_1, T_2, .., T_u\}$ as a set of time intervals.

$P_i = e_i \rightarrow C_i$ as a constraint of edge $e_i$. If and only if the current time is contained by at least one time interval in $C_i$ , $e_i$ gets connected.

$Q = \{P_1, P_2, .., P_n\}$ as the set of all constraints of $n$ edges in the topology.

Finally, the input of the topology is $G < V, E, Q >$, while $V$ is the set of all nodes, $E$ is the set of all edges, and $Q$ is defined as above. We should verify if network properties are always satisfied in the network with this topology.

### 2.2 Methodology

We notice that existing network verification tools just lack the ability to handle changing topologies. Therefore, we can convert the periodically changed topology into several static topologies, so that existing tools are able to verify these static topologies. Then we can analyse results of all static topologies and get the final answer.

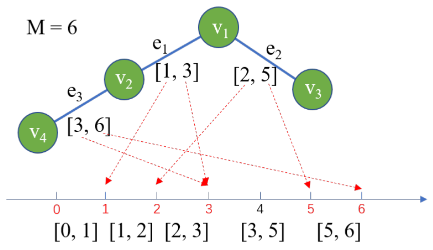

**Figure 1: An example to generate all time intervals. It's a periodically changed topology with the least common cycle is 6.**

Generally, we can take out all $T_i = [a_i, b_i]$ in $Q$, then put $a_i, b_i$ into a set $S$. Then we put 0 and $M$ into $S$. So we have

$$S = \{0, a_1, b_1, a_2, b_2, .., a_m, b_m, M\}$$

Next, after sorted and deduplicated:

$$S' = \{t_1, t_2, .., t_r\}\ ,\ 0 \le t_i < t_{i+1} \le M\ ,\ t_i \in S\ \ (i = 1, 2, .., r)$$

Then we get all time intervals:

$$[t_1, t_2], [t_2, t_3], .., [t_{r-1}, t_r]$$

Figure 1 shows the example of how to get time intervals. The time intervals are [1, 3], [2, 5], [3, 6], so $S = \{0, 1, 3, 2, 5, 3, 6\}$, $S' = \{0, 1, 2, 3, 5, 6\}$. Finally we get time intervals [0, 1],[1, 2],[2, 3],[3, 5],[5, 6].

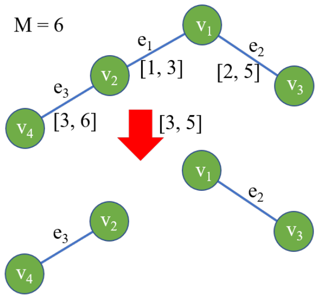

**Figure 2: An example to generate static topology in time interval** [3, 5]**.**

With each time interval $[t_i, t_{i+1}]$, we can generate a static topology $G_i < V, E_i >$. In the topology $G < V, E, Q >$, we can traverse every edge $e_k$ and get the constraint $P_k = e_k \longrightarrow C_k$ corresponding to the edge $e_k$. We put $e_k$ into $E_i$ if and only if $[t_i, t_{i+1}]$ is contained by at least one time interval in $C_k$. Then we get a static topology $G_i < V, E_i >$. Finally, we verify each static topology and confirm if network properties are satisfied in all topologies $G_1, G_2, .., G_{r-1}$.

Figure 2 shows the example of how we generate static topologies. As for time interval [3, 5], obviously $[3, 5] \notin [1, 3], [3, 5] \in [2, 5], [3, 5] \in [3, 6]$, so $e_2, e_3$. are up while $e_1$ is down, and the static topology in [3, 5] only contains edges $e_2, e_3$.

## 3 EVALUATION

We build a network with periodically changed topology as shown in figure 3. The three routers are connected via BGP and static routes. Then Batfish is used as our verification tool. In table 1, we give our verification results. We convert given topology into 6 static topologies in different time intervals. For each topology we use Batfish to verify the reachability of $r_1 \longrightarrow r_2$. Finally, we analyse these result together and find out the unreachable time interval [1, 4] for $r_1 \longrightarrow r_2$.

We find this methodology can help us split the time interval easily and generate the static topology that can be directly inputted into Batfish. And these topologies only contain all possible topologies. So we can verify network properties of the changing topology by verifying properties on all static topologies.

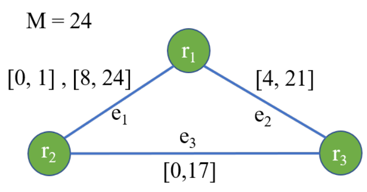

**Figure 3: The topology for evaluation**

**Table 1: Verification results**

| Time interval | Available edges | Reachability $r_1 \longrightarrow r_2$ |
|---|---|---|
| [0, 1] | $e_1, e_3$ | Verified |
| [1, 4] | $e_3$ | No route |
| [4, 8] | $e_2, e_3$ | Verified |
| [8, 17] | $e_1, e_2, e_3$ | Verified |
| [17, 21] | $e_1, e_2$ | Verified |
| [21, 24] | $e_1$ | Verified |
| total | | [1, 4] No route |

## 4 CONCLUSION AND FUTURE WORK

In this paper, we point out that existing network verification tools cannot directly verify network with periodically changed topology such as the space-terrestrial integrated network. Then we abstract the problem, and propose a methodology that converts changed topology into several static topologies.

Our future work will focus on more efficient algorithm to verify networks with the periodically changed topology.

## REFERENCES

[1] Ryan Beckett, Aarti Gupta, Ratul Mahajan, and David Walker. 2017. A general approach to network configuration verification. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. ACM, 155–168.
[2] Ari Fogel, Stanley Fung, Luis Pedrosa, Meg Walraed-Sullivan, Ramesh Govindan, Ratul Mahajan, and Todd Millstein. 2015. A general approach to network configuration analysis. In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. 469–483.
[3] Xiangyue Huang, Zhifeng Zhao, Xiangjun Meng, and Honggang Zhang. 2016. Architecture and application of SDN/NFV-enabled space-terrestrial integrated network. In *International Conference on Space Information Network*. Springer, 244–255.
[4] Nuno P Lopes, Nikolaj Bjørner, Patrice Godefroid, Karthick Jayaraman, and George Varghese. 2015. Checking beliefs in dynamic networks. In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. 499–512.
[5] Radu Stoenescu, Matei Popovici, Lorina Negreanu, and Costin Raiciu. 2016. SymNet: Scalable symbolic execution for modern networks. In *Proceedings of the 2016 ACM SIGCOMM Conference*. ACM, 314–327.