Poster: Automating Network Configuration with Natural Language Intents

Wenlong Ding[1], Jianqiang Li[1], Zhixiong Niu[2], Huangxun Chen[3], Hong Xu[1]
[1]CUHK [2]Microsoft Research [3]HKUST (Guangzhou)

CCS CONCEPTS

• **Networks** → **Network management**;

KEYWORDS

Network configuration automation, Large language models

ACM Reference Format:
Wenlong Ding, Jianqiang Li, Zhixiong Niu, Huangxun Chen, Hong Xu. 2024. Poster: Automating Network Configuration with Natural Language Intents. In *ACM SIGCOMM 2024 Conference (ACM SIGCOMM '24), August 4–8, 2024, Sydney, NSW, Australia.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3672202.3673721

1 INTRODUCTION

Network configuration is crucial for computer network design and operation. Network Operations (NetOps) teams typically use vendor-specific configuration languages to create and deploy scripts that achieve specific network intents, such as reachability between two networks [7, 18, 19]. This process necessitates a thorough understanding of network specifics and protocols. Also, NetOps teams may need multiple attempts to achieve correct configuration due to protocol complexity and interactions. They often use network verification tools [6, 9, 14, 16, 17, 19] to check if the intents are met and adjust configurations based on failure messages.

This manual configuration method is obviously labor-intensive and difficult to scale with growing network size and configuration complexity. Our proposed system, Etna, automates the process by enabling NetOps teams to specify high-level intents in Natural Language (NL) and directly generate deployable scripts for all devices across the network. As shown in Figure 1, Etna performs the following three tasks to achieve this goal.

- *Intent Understanding:* Etna interprets NL inputs to identify necessary network specifics and policies for configurations. For *reachability* intent as example, Etna identifies the source and destination networks with their prefixes and comprehends the policies to be taken, such as denying the source-destination pair (i.e. a flow) shown in Figure 1.
- *Intent Implementation:* Etna solves parameter settings and implements them in the network with proper protocols. In Figure 1, when implementing a *waypoint* intent with OSPF [2] that requires to route a flow traversing specific waypoint routers, Etna tunes OSPF link weights to make the desired path the shortest among all candidate paths of the flow.
- *Scripts Generation:* Etna generates configuration scripts by filling in vendor-specific templates with corresponding network specifics and protocol setting solutions, which are ready to be deployed and verified in devices across the network. For example in Figrue 1, Etna fills in Cisco's ACL and OSPF templates to achieve the desired intents.

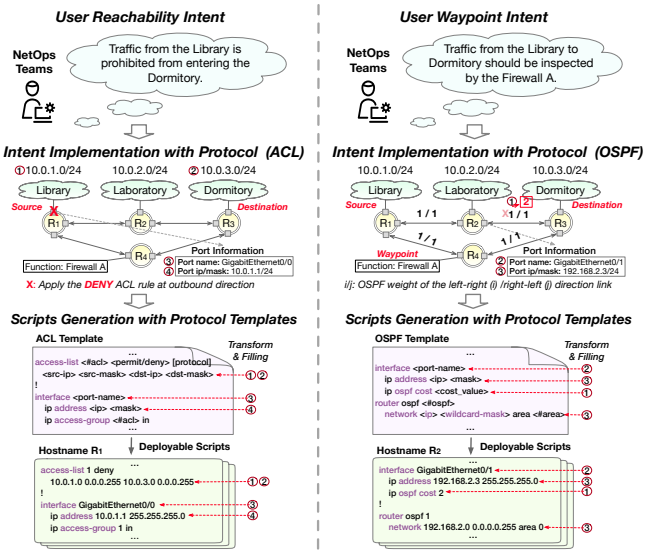

Figure 1: Examples of Etna's network configuration workflow.

Scripts Generation has a rich literature on mapping protocol settings and network specifics to templates [4, 5, 18], managing template databases [21], and translating configurations across vendors [7]. So Etna can adopt similar template-filling-based method to automate it. However, automating another two tasks poses unsolved challenges. *Intent Understanding* must accurately interpret NL intents with network specifics, despite NL expression variations and implicit configuration items. Conflicts between new intents themselves and existing settings complicate *Intent Implementation.*

Etna contributes to solving the three tasks with their challenges above to automate configuration generation from NL intents.

2 CHALLENGE AND MOTIVATION

We detail challenges and design motivations in Etna.

Intent Understanding. Intuitively, this task can be viewed as the famous Named-Entity Recognition (NER) problem [8, 10–12], where models are trained to identify entities (e.g. source endpoint) and implicit knowledge (e.g. intent policy) from expressions. However, traditional language models like BERT [20] in these works require

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
ACM SIGCOMM '24, August 4–8, 2024, Sydney, NSW, Australia
© 2024 Copyright held by the owner/author(s). Publication rights licensed to the Association for Computing Machinery.
ACM ISBN 979-8-4007-0717-9/24/08.
https://doi.org/10.1145/3672202.3673721

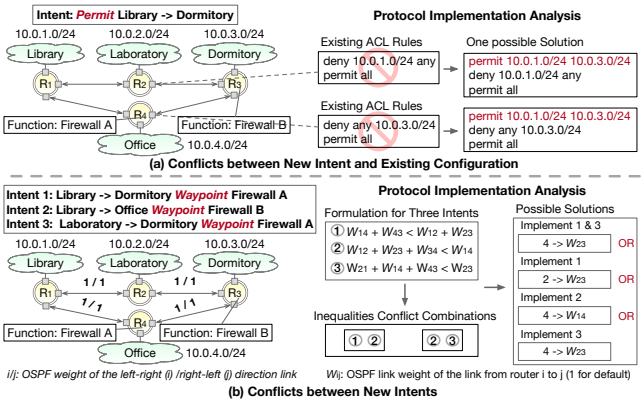

Figure 2: Simple examples of challenges in *Intent Implementation*.

repeated re-training for expression synonyms such as `Lib`, `Library` and `10.0.1.0/24` in Figure 1 and different network corpora when introducing new networks and devices. The recent advancement of Large Language Models (LLMs) with strong text processing abilities through prompting [1, 3, 13, 15] is more suitable for solving NL expression variations above without repeated retraining.

However, network-specific information necessary for configurations like prefixes and routers may not be explicitly provided in NL and differs between networks. Therefore, LLMs cannot independently identify such information. To address this, Etna provides LLMs with a domain-specific database containing this implicit information about currently operated networks, e.g. mapping `Library` to prefix `10.0.1.0/24` and router `R1` in Figure 1. Additionally, Etna should detail key identification elements for LLMs to understand, such as `source prefixes` or `policy` to allow or deny flows.

Intent Implementation. This involves non-textual tasks including topology analysis for ACL configuration ports and OSPF weight formulation for desired flow paths in Figure 1 and 2, which are inefficient for LLMs to solve [22]. Traditional topology-based methods are used here, but various configuration conflicts pose challenges.

New intent conflicts with existing configurations. For instance, when implementing *reachability* intent in Figure 2(a), conflicts arise with ACL rules on routing paths as they manage overlapping flows with opposite policies to our new intent. The common practice of configuring ACL at source router's port in Figure 1 obviously cannot resolve this conflict. One possible solution is to prioritize new intent over existing configurations, as the solution in Figure 2(a), but this method should be carefully considered by NetOps teams as they may not want to change certain old network settings.

Conflicts can also occur between multiple new intents. For instance, in Figure 2(b), three *waypoint* intents will be implemented with OSPF. We solve OSPF link weights with the formulation that makes the target path traversing specified waypoints the shortest compared to other paths of the flow for each intent. However, some intents' inequalities have conflicts, resulting in only partial intent achievement, such as intents 1 and 3, or each intent alone, as the solutions shown in Figure 2(b). These conflicts are difficult to recognize in NL without formulation. Additionally, different intent types can also conflict, such as when a *reachability* intent denies traffic of a flow, making the corresponding *waypoint* intents unachievable.

Etna's design aims to reconcile various conflict types mentioned above according to the specific needs of NetOps teams.

Dataset	Dataset Information		Evaluation Results				
	# Intents	# Chars Per Intent	Metric	Llama 2	Mistral	GPT-3.5	GPT-4
Hand-crafted	150	81.3	Accuracy	91.3%	86.7%	98.7%	100%
			Time (s)	2.25	2.59	1.38	3.98
AI-generated	300	81.9	Accuracy	88.0%	88.7%	98.3%	100%
			Time (s)	2.89	2.33	1.34	4.13

Table 1: Evaluated dataset and preliminary results of *Intent Understanding*.

3 INITIAL DESIGN

LLM Prompts for *Intent Understanding*. Prompts help LLMs specify and achieve understanding goals. We first encode key identification elements such as associated routers, prefixes and NL names of source, destination and waypoints, as well as the intent policy for examples in Figure 1. Prompts include description for each element such as `policy: <permit or deny intent flow>` for intent policy. Etna is also provided with implicit network specifics such as routers and prefixes by mapping endpoint NL names to such information with prompts like `Library -> [R1], [10.0.1.0/24]`. Note that synonyms like `Library` and `Lib` can be handled by LLMs efficiently instead of encoding both NL names in prompts. We also guide LLMs on how to identify key elements with NL intents and provided implicit information, such as extracting endpoint names from NL intents and utilizing implicit knowledge to identify routers and prefixes. LLMs return results of all identified elements with each element such as `source prefixes: [10.0.1.0/24]` for source prefixes. We finally include a few examples of input intents and result elements to improve LLMs' comprehension of the prompts.

Priority-Based Framework to Reconcile Conflicts. Etna assigns a priority to each intent and existing setting and maximizes priority sum across the network. This design satisfies intents or settings with higher priorities when conflicts happen. For example, if three intents in Figure 2(b) are equally important with the same priority, Etna will achieve intents 1 and 3. However, if intent 2 is crucial, we can assign it a higher priority than the sum of other intents for Etna to achieve it alone.

4 PRELIMINARY RESULTS

We present preliminary results for *Intent Understanding* with various famous LLMs in Table 1. Accuracy is measured by comparing LLMs' results with human-labeled ground truth. The entire result is accurate only if all identified elements in the result are correct. Inference time is the duration between entering NL intents and receiving entire results. LLMs are evaluated with provided APIs [1, 3]. We use two datasets in evaluation: a handcrafted dataset resembling intents in Figure 1 and an AI-generated dataset created by GPT-4 [3] prompting with intents in our handcrafted dataset.

Results demonstrate that state-of-the-art LLMs like GPTs achieve over 95% accuracy for both datasets, with GPT-4 even reaching 100%. Lightweight LLMs like Llama2 and Mistral can also maintain accuracy above 85%. Inference times range from a few seconds, with GPT-4 being the slowest at around 4 seconds, but compensating with its high accuracy. Both datasets yield similar accuracy and inference time results.

5 ACKNOWLEDGEMENT

This work is supported in part by funding from the Research Grants Council of Hong Kong (CRF C7004-22G) and from CUHK (4055199).

REFERENCES

[1] APIs of Our Evaluated LLMs. https://replicate.com/pricing.

[2] Cisco OSPF. https://www.cisco.com/c/en/us/support/docs/ip/open-shortest-path-first-ospf/7039-1.html.

[3] GPT. https://chat.openai.com/.

[4] Graph-Based Live Queries in AOS. https://apstra.com/products/.

[5] OpenConfig. http://openconfig.net/.

[6] Ryan Beckett, Aarti Gupta, Ratul Mahajan, and David Walker. A General Approach to Network Configuration Verification. In *Proc. ACM SIGCOMM*, 2017.

[7] Huangxun Chen, Yukai Miao, Li Chen, Haifeng Sun, Hong Xu, Libin Liu, Gong Zhang, and Wei Wang. Software-Defined Network Assimilation: Bridging the Last Mile Towards Centralized Network Configuration Management with NAssim. In *Proc. ACM SIGCOMM*, 2022.

[8] Maud Ehrmann, Ahmed Hamdi, Elvys Linhares Pontes, Matteo Romanello, and Antoine Doucet. Named Entity Recognition and Classification in Historical Documents: A Survey. 56(2):1–47, Sep. 2023.

[9] Aaron Gember-Jacobson, Raajay Viswanathan, Aditya Akella, and Ratul Mahajan. Fast Control Plane Analysis Using an Abstract Representation. In *Proc. ACM SIGCOMM*, 2016.

[10] Arthur S. Jacobs, Ricardo J. Pfitscher, Rafael H. Ribeiro, Ronaldo A. Ferreira, Lisandro Z. Granville, Walter Willinger, and Sanjay G. Rao. Hey, Lumi! Using Natural Language for Intent-Based Network Management. In *Proc. USENIX ATC*, 2021.

[11] Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. A Survey on Deep Learning for Named Entity Recognition. 34(1):50–70, Mar. 2020.

[12] Chen Liang, Yue Yu, Haoming Jiang, Siawpeng Er, Ruijia Wang, Tuo Zhao, and Chao Zhang. Bond: Bert-Assisted Open-Domain Named Entity Recognition with Distant Supervision. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020.

[13] Laria Reynolds and Kyle McDonell. Prompt Programming for Large Language Models: Beyond the Few-shot Paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, 2021.

[14] Steffen Smolka, Praveen Kumar, Nate Foster, Dexter Kozen, and Alexandra Silva. Cantor Meets Scott: Semantic Foundations for Probabilistic Networks. In *Proceedings of ACM SIGPLAN Symposium on Principles of Programming Languages*, 2017.

[15] Taylor Sorensen, Joshua Robinson, Christopher Michael Rytting, Alexander Glenn Shaw, Kyle Jeffrey Rogers, Alexia Pauline Delorey, Mahmoud Khalil, Nancy Fulda, and David Wingate. An Information-Theoretic Approach to Prompt Engineering without Ground Truth Labels. *arXiv Preprint arXiv:2203.11364*, 2022.

[16] Samuel Steffen, Timon Gehr, Petar Tsankov, Laurent Vanbever, and Martin Vechev. Probabilistic Verification of Network Configurations. In *Proc. ACM SIGCOMM*, 2020.

[17] Kausik Subramanian, Anubhavnidhi Abhashkumar, Loris D'Antoni, and Aditya Akella. Detecting Network Load Violations for Distributed Control Planes. In *Proceedings of ACM SIGPLAN Conference on Programming Language Design and Implementation*, 2020.

[18] Yu-Wei Eric Sung, Xiaozheng Tie, Starsky HY Wong, and Hongyi Zeng. Robotron: Top-Down Network Management at Facebook Scale. In *Proc. ACM SIGCOMM*, 2016.

[19] Bingchuan Tian, Xinyi Zhang, Ennan Zhai, Hongqiang Harry Liu, Qiaobo Ye, Chunsheng Wang, Xin Wu, Zhiming Ji, Yihong Sang, Ming Zhang, Da Yu, Chen Tian, Haitao Zheng, and Ben Y. Zhao. Safely and Automatically Updating In-Network ACL Configurations with Intent Language. In *Proc. ACM SIGCOMM*, 2019.

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, 2017.

[21] Wenlong Ding, Libin Liu, Li Chen and Hong Xu. ABC: Automatic Bottom-Up Construction of Configuration Knowledge Base for Multi-Vendor Networks. In *2023 IEEE 5th International Conference on Cognitive Machine Intelligence (CogMI)*, 2023.

[22] Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. *arXiv preprint arXiv:2309.03409*, 2023.