Red Hat

NetConfEval: Can LLMs Facilitate Network Configuration?

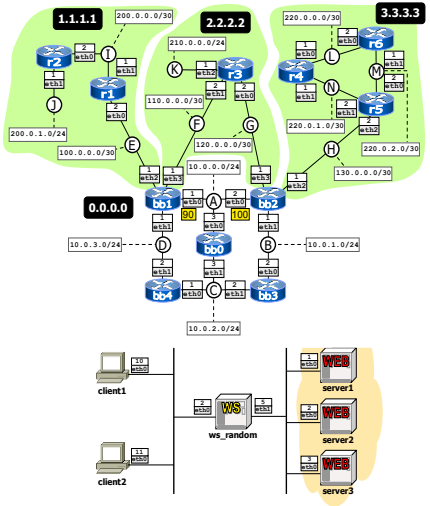
Changjie Wang[1], Mariano Scazzariello[1 2], Alireza Farshin[2], Simone Ferlin[3], Dejan Kostic[1 2], Marco Chiesa[1]

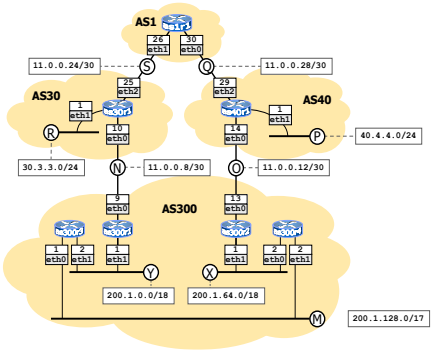
KTH Royal Institute of Technology [1] - RISE Research Institutes of Sweden [2] - Red Hat [3]

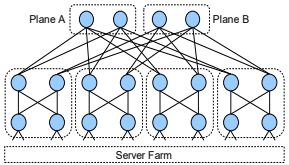
Orchestrating networks is a cumbersome task

network operator

configure



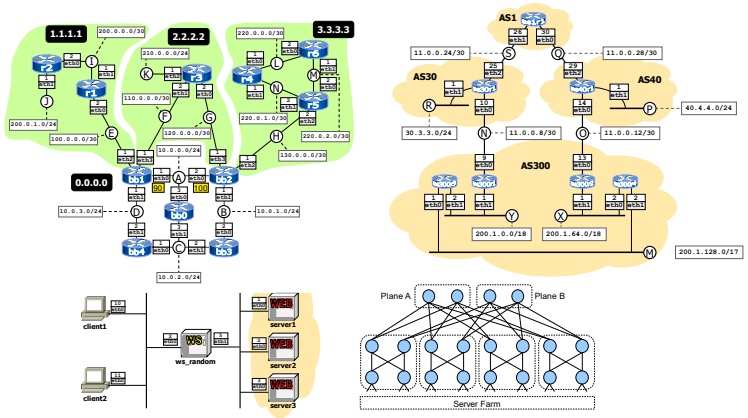




Orchestrating networks is a cumbersome task

network operator

configure



large software documentation

complex network/application failures

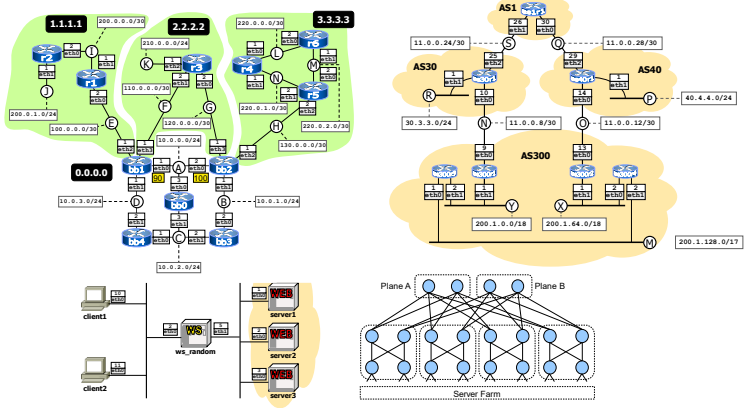
cyber-attacks or vulnerabilties

misunderstanding within the team

Orchestrating networks is a cumbersome task

network operator

configure



Manual configuration is costly, difficult, and susceptible to human error

minor error

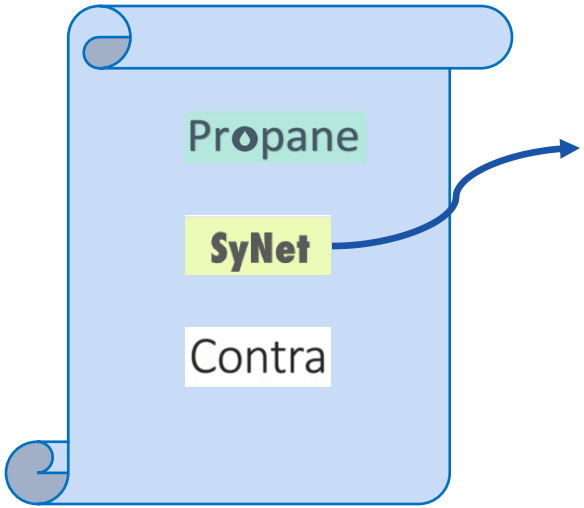
amadeus **Flight disruption**

Meta **Outage**

Network Synthesis Tools Help but ...

Network Synthesizer → Generate low-level configurations from intents

But employ a self-defined specification or language to describe the network intents



$$
\begin{array}{llllll}
(Program) & P ::= \overline{r} & (Literal) & l ::= a \mid \neg a & (Variables) & X, Y \in Vars \\
(Rule) & r ::= a \leftarrow \overline{l} & (Predicates) & p, q \in Preds & (Values) & v \in Vals \\
(Atom) & a ::= p(\overline{t}) & (Term) & t ::= X \mid v & &
\end{array}
$$

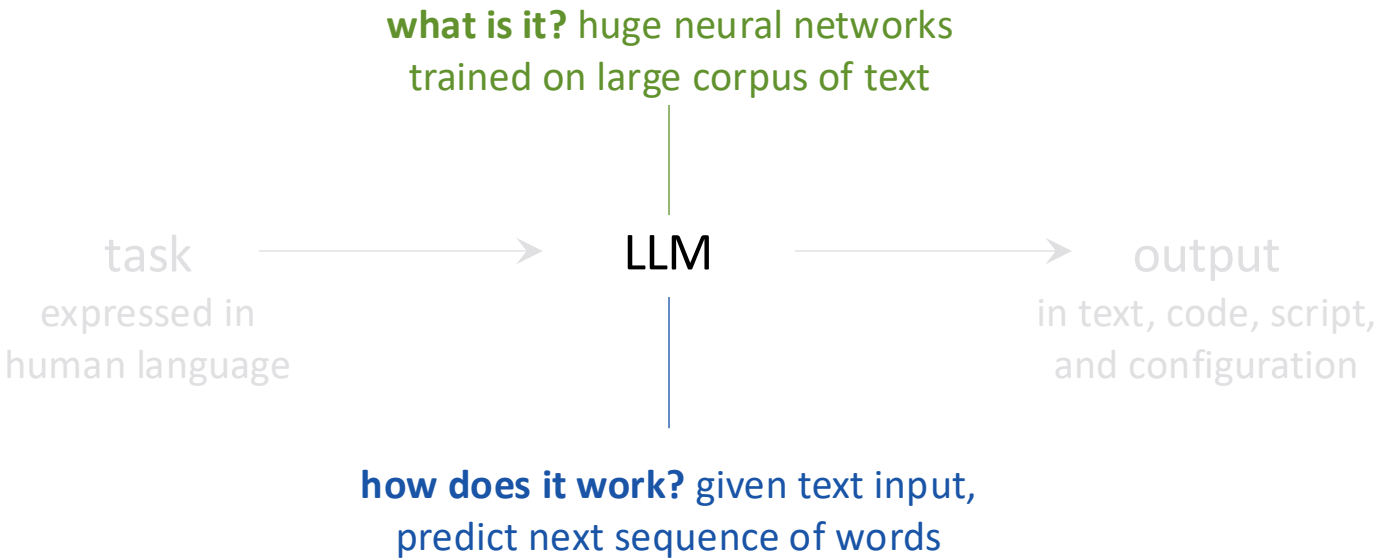
Syntax for Stratified Datalog

It's hard to learn and extend!

network operator

Can Large Language Models (LLMs) help?

Can Large Language Models (LLMs) help?



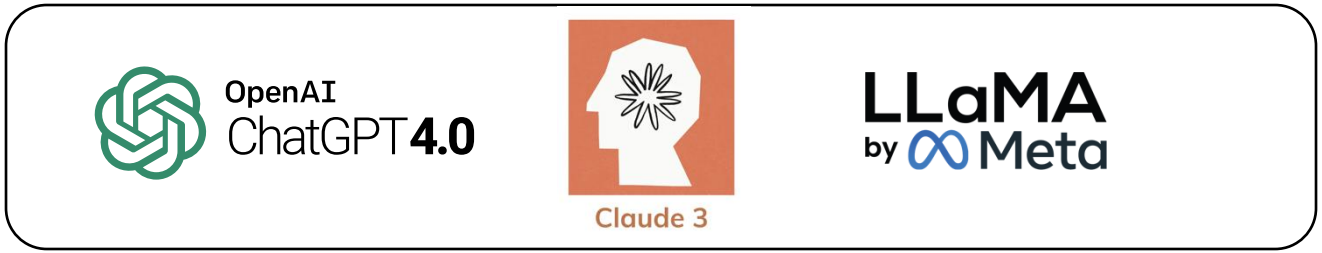
Can Large Language Models (LLMs) help?

Can Large Language Models (LLMs) help?

task → LLM → output

expressed in human language

in text, code, script, and configuration



Example of LLMs

Can Large Language Models (LLMs) help?

task
expressed in human language

→ LLM →

output
in text, code, script, and configuration

"LLMs can generate coherent, contextually relevant text based on prompts." →

knowledgeable

fast learner

creative

Can Large Language Models (LLMs) help?

task
expressed in
human language

output
in text, code, script,
and configuration

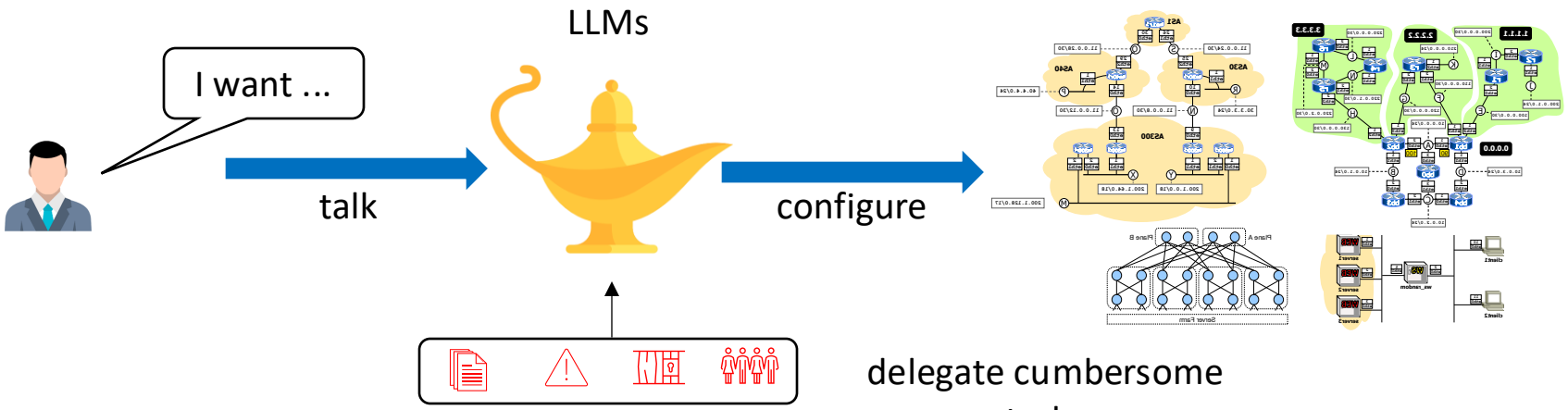
Can LLMs help in network orchestration?

"LLMs can generate coherent, contextually relevant text based on prompts."

fast learner

creative

Can LLMs help in network orchestration?



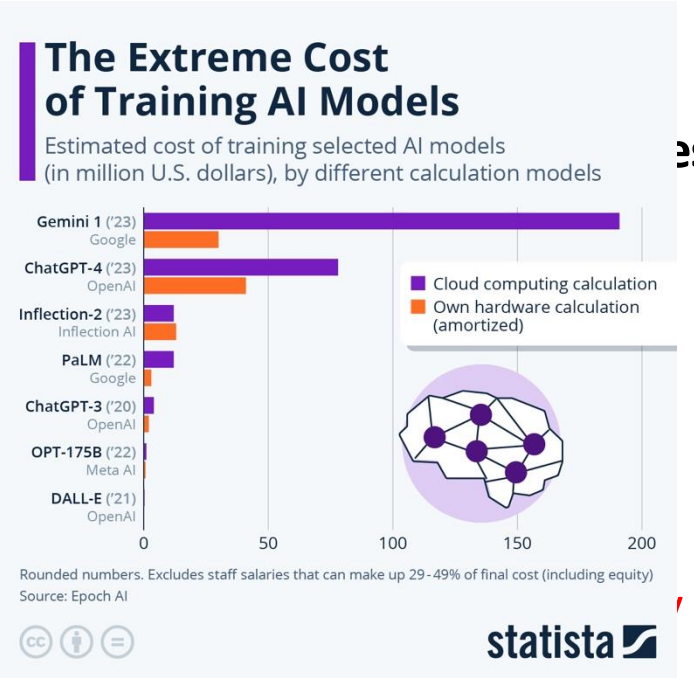
Can LLMs help in network orchestration?

Opportunities come with challenges

unreliability

costs

Can LLMs help in network orchestration?

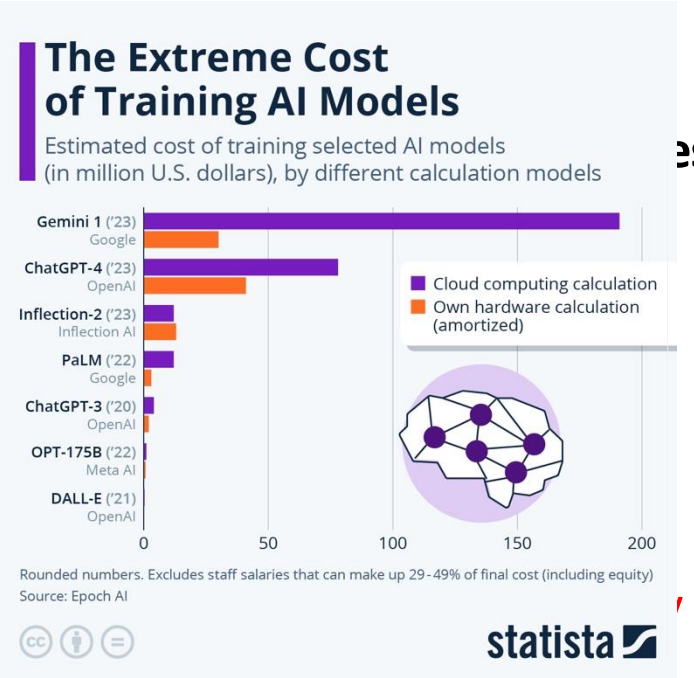


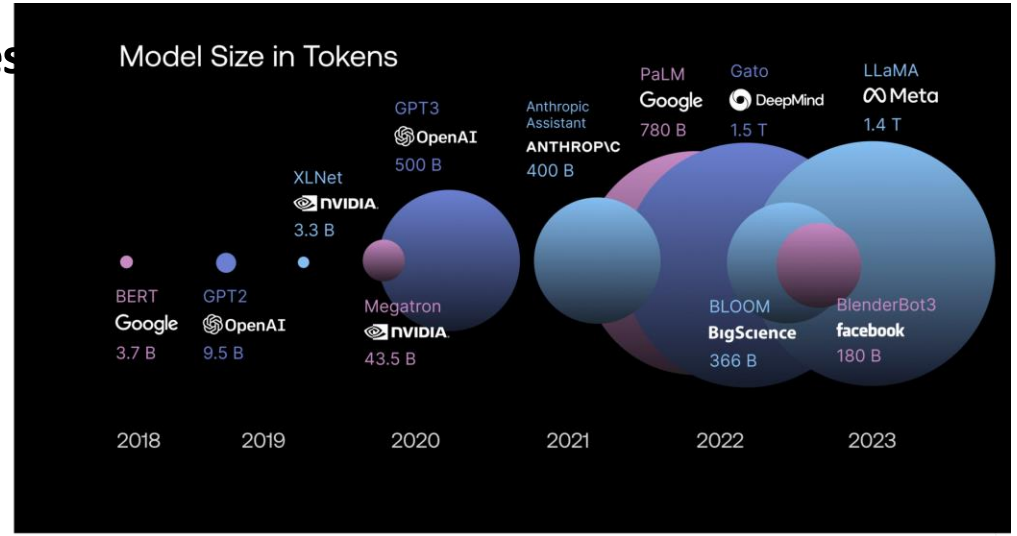
es come with challenges

costs

Source: https://www.statista.com/chart/33114/estimated-cost-of-training-selected-ai-models/

Can LLMs help in network orchestration?





Source: https://scale.com/guides/large-language-models#model-size-and-performance

Assess LLMs in today's networking tasks

1. Design a set of **benchmarks** (NetConfEval) to evaluate LLMs for networking

2. Formulate **takeaways** based on our benchmarking experiment

3. Present **prototypes** for LLM-based networking systems

We'll focus on three tasks in orchestrating networks

1. Translating high-level requirements to a formal specification format

2. Adapting code to new requirements

3. Generating low-level configurations

We'll focus on three tasks in orchestrating networks

1. **Translating high-level requirements to a formal specification format**

2. Adapting code to new requirements

3. Generating low-level configurations

Translating high-level requirements to a formal specification format

*”traffic from **Rome** to **Milan** must **traverse** a **firewall**”*

network operator

LLM

dictionary

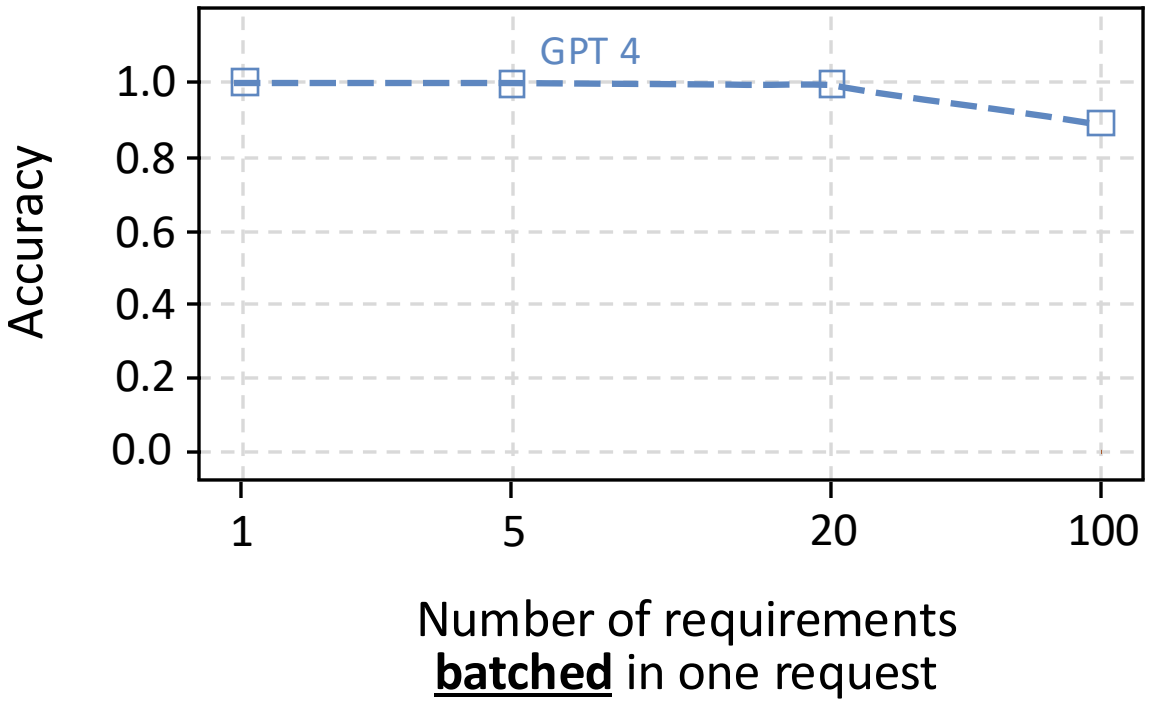
```
{"reachability": {
    ”rome": [”milan"]},
 "waypoint": {
    [”rome", ”milan"]:
       [”fw1”,”fw2”]},
 "avoidance": {}
}
```

Translating high-level requirements to a formal specification format

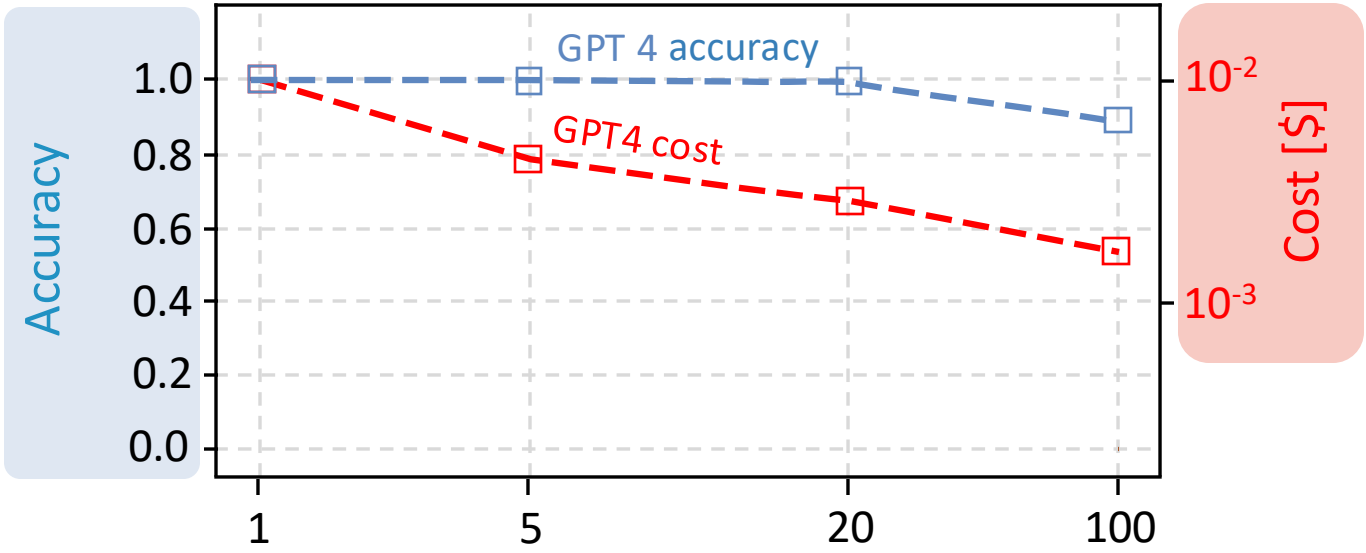
1. Generate 3200 network requirements focusing on reachability, waypoint, and load-balancing using Config2Spec[1]
2. Pick a certain number of requirements and sliced them with various batch sizes
3. Transform them to natural language based on predefined templates
4. Evaluate the efficiency of different LLMs in translation

[1] “Mining network specifications from network configurations”, NSDI 20 by Birkner, R., Drachsler-Cohen, D., Vanbever, L., & Vechev, M

GPT4 translates accurately requirements



GPT4 translates accurately requirements at a cost



Issues with very large language models

nsdi'24

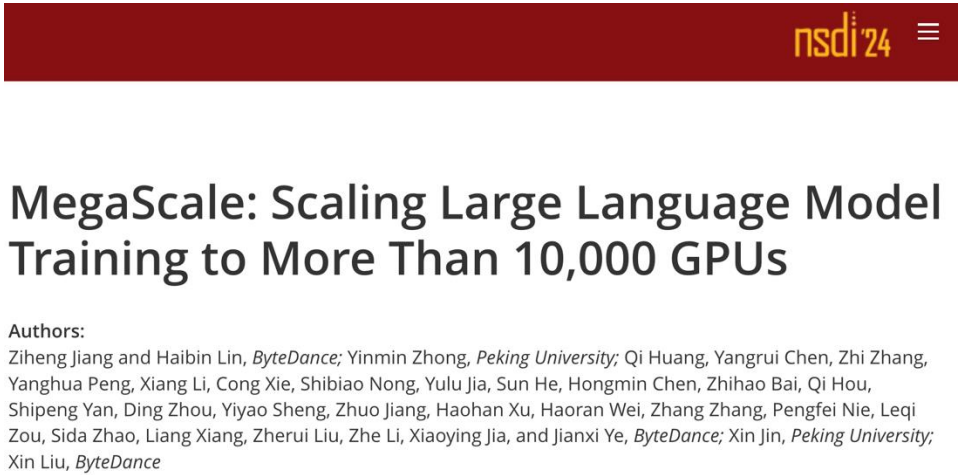
MegaScale: Scaling Large Language Model Training to More Than 10,000 GPUs

Authors:
Ziheng Jiang and Haibin Lin, *ByteDance;* Yinmin Zhong, *Peking University;* Qi Huang, Yangrui Chen, Zhi Zhang, Yanghua Peng, Xiang Li, Cong Xie, Shibiao Nong, Yulu Jia, Sun He, Hongmin Chen, Zhihao Bai, Qi Hou, Shipeng Yan, Ding Zhou, Yiyao Sheng, Zhuo Jiang, Haohan Xu, Haoran Wei, Zhang Zhang, Pengfei Nie, Leqi Zou, Sida Zhao, Liang Xiang, Zherui Liu, Zhe Li, Xiaoying Jia, and Jianxi Ye, *ByteDance;* Xin Jin, *Peking University;* Xin Liu, *ByteDance*

Problems with very large language models

Large language models:

- 1000B parameters
- slow inferences
- resource intensive
- hard to deploy



The quest towards smaller (cheaper!) models

Large language models:

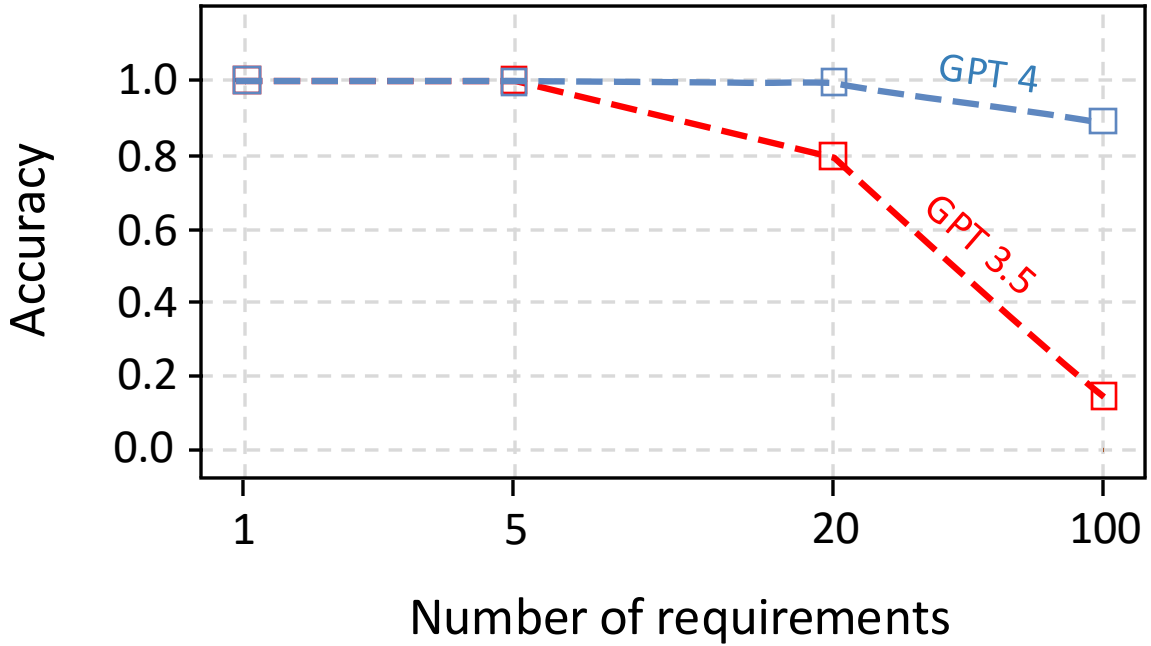
- 1000B parameters
- slow inferences
- resource intensive
- hard to deploy

Small/Medium language models:

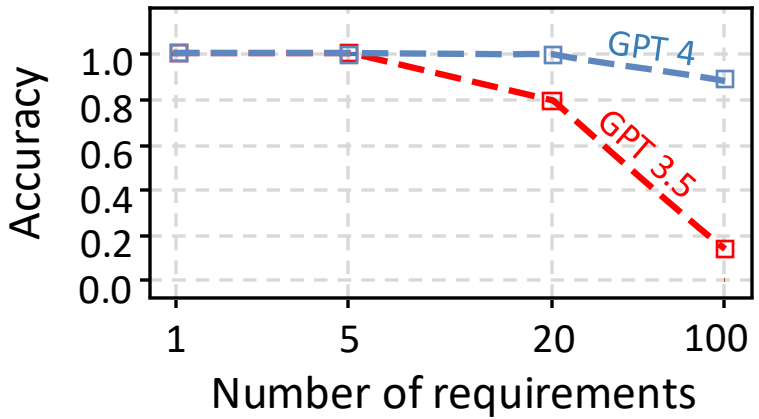
- 1B-100B parameters
- faster inferences
- deployable on a few GPUs
  - sometimes even on a laptop

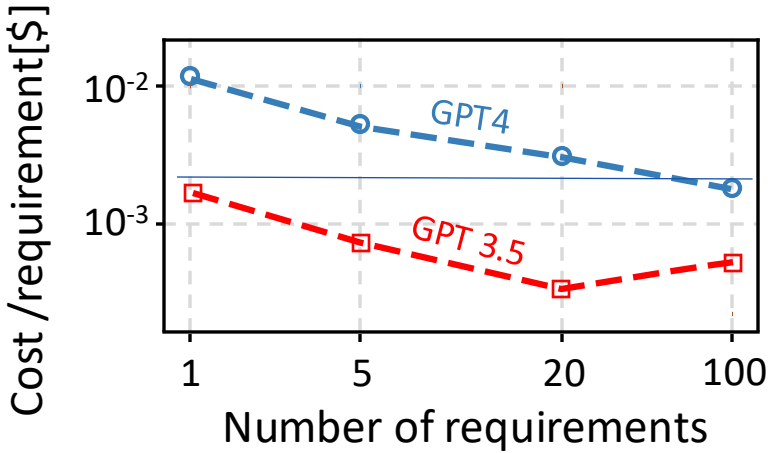
How do smaller models perform?

Sadly, smaller models perform worse



Sadly, smaller models perform worse, yet cost less





Can one specialized language models for one task?

General-purpose models:

- trained on any text
- know **everything** (almost)
- but may fail in **something**

how many r's are in the phrase "network orchestration"?

The phrase "network orchestration" contains one 'r'.

Can one specialized language models for one task?

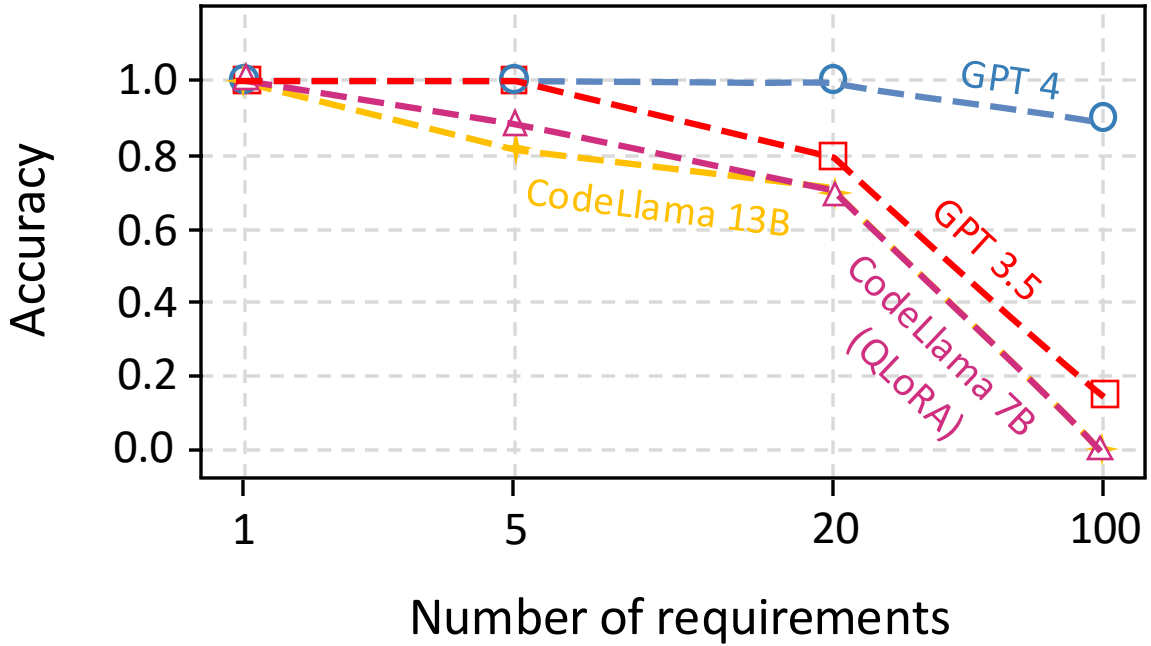
General-purpose models:

- trained on any text
- know **everything** (almost)
- but may fail in **something**

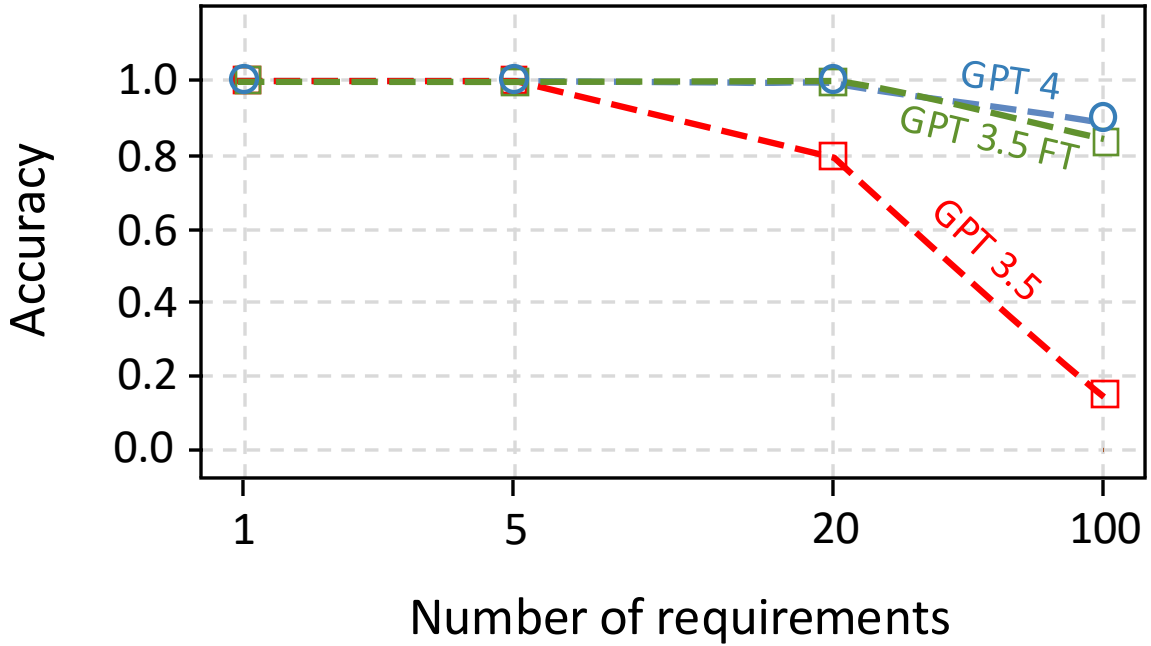
Specialized models:

- pre-trained on specific tasks, or
- fine-tuned from general-purpose

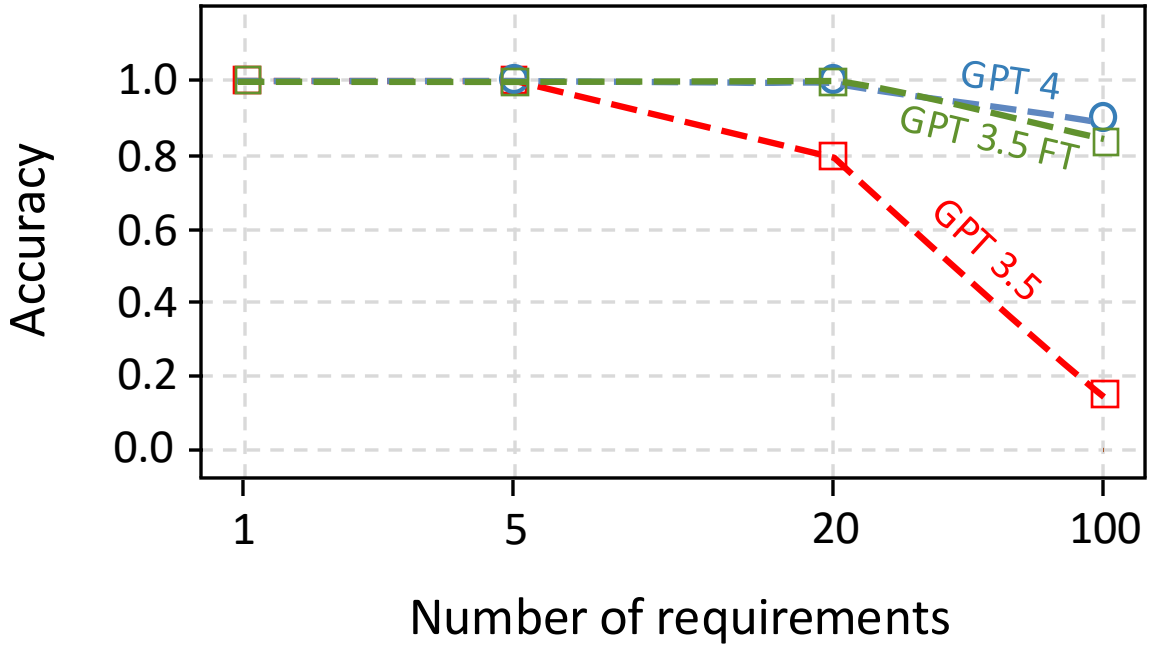
Small Specialized models perform poorly, but better



Larger Specialized models perform better



Larger Specialized models perform better



Can LLMs call API functions?

June 2023 (one function call) and November 2023 (parallel function calls)

Translating high-level requirements to a formal specification format

*”traffic from **Rome** to **Milan** must **traverse** a **firewall**”*

network operator

LLM

dictionary

```
{"reachability": {
    ”rome": [”milan"]},
 "waypoint": {
    [”rome", ”milan"]:
       [”fw1”,”fw2”]},
 "avoidance": {}
}
```

function calling

```
add_reachability(”rome”,”milan”);
add_waypoint(”rome”,”milan”,[”fw1”,”fw2”]);
```

Which one would be best?

dictionary

```
{"reachability": {
    ”rome": [”milan"]},
 "waypoint": {
    [”rome", ”milan"]:
       [”fw1”,”fw2”]},
 "avoidance": {}
}
```

+ compact

- rearranging items

function calling

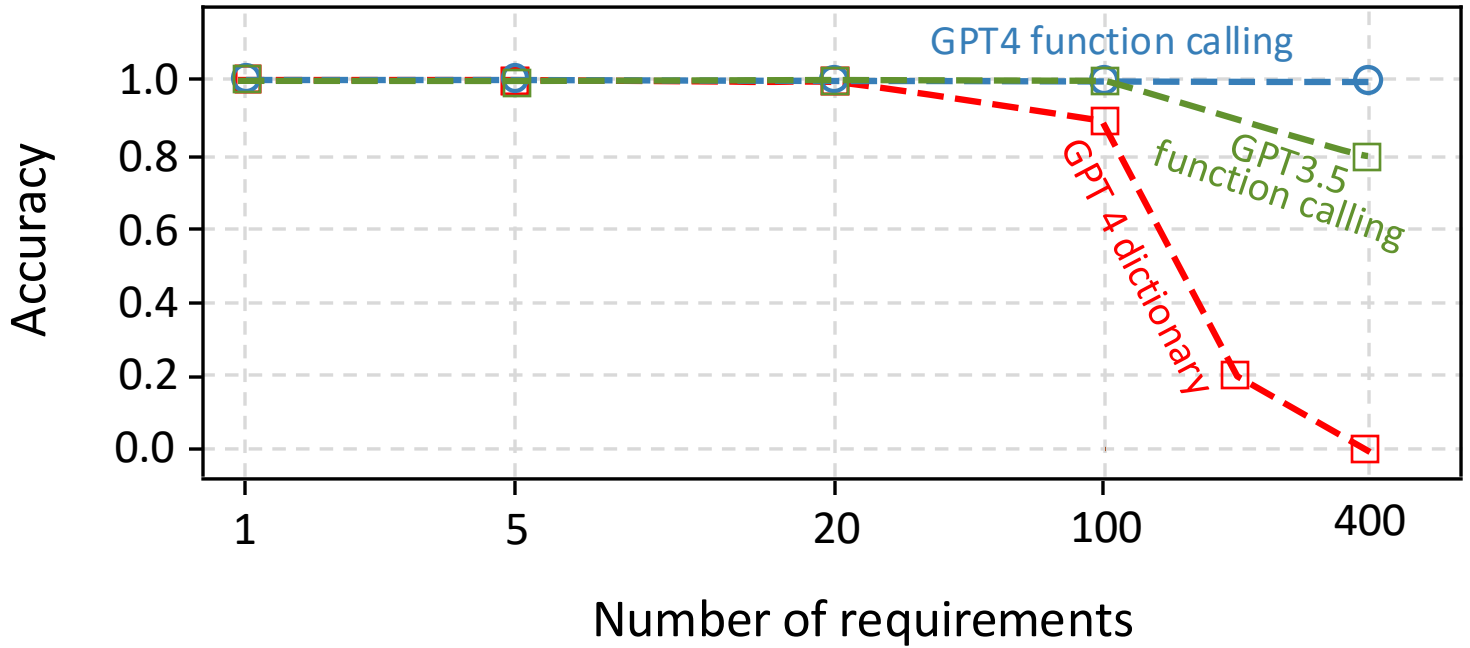
```
add_reachability(”rome”,”milan”);
add_waypoint(”rome”,”milan,[”fw1”,”fw2”];
```

- less compact

+ no re-ordering

Function calling versus dictionary data structure

LLMs are good at 1:1 translations



We'll focus on three tasks in orchestrating networks

1. Translating high-level requirements to a formal specification format

2. **Adapting code to new requirements**

3. Generating low-level configurations

Adapting code to new requirements. Why?

Developing modern software is **difficult**

- fast-paced due to **rapid** technological changes
- higher performance, resilience, and security guarantees

Developing modern software is **expensive**

- hire developers with a deep understanding of **numerous** systems, protocols, etc.
- development process becomes **time-consuming**, error-prone, and cumbersome

Adapting code to new requirements

*”Create a **function** that takes as input […] and produces **waypoint** paths as output”*

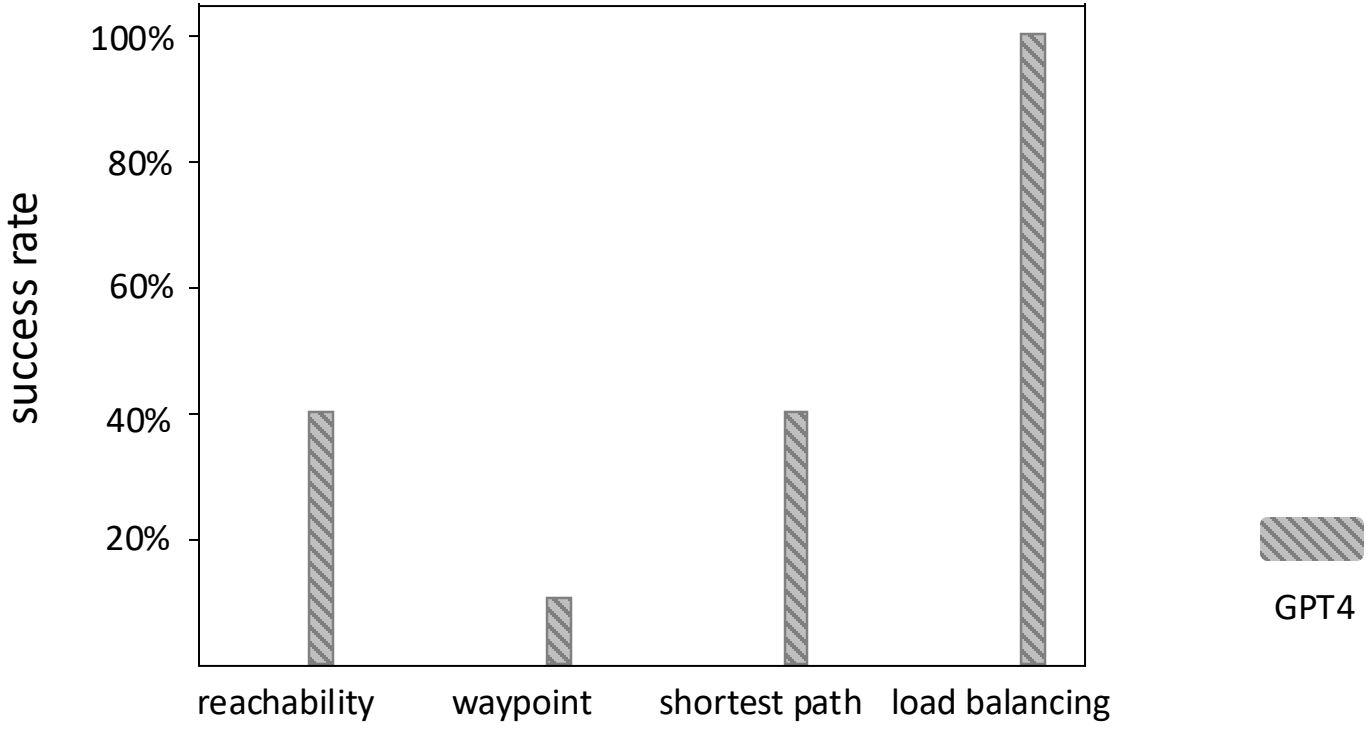
network operator

LLM

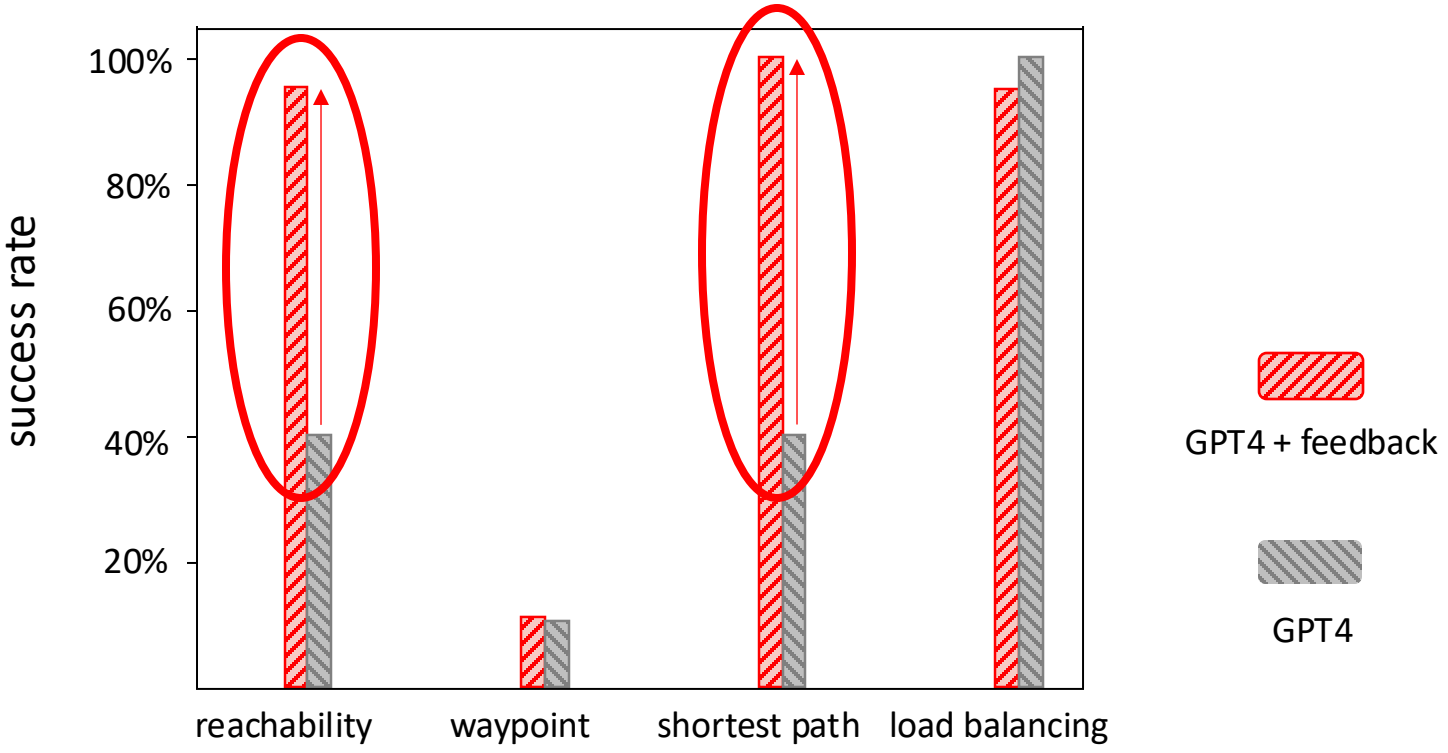
code generation

```
void Dijkstra(int source, const vector<vector<pair<int, in
    int n = graph.size();
    dist.assign(n, INF);
    set<pair<int, int>> active_vertices;
    dist[source] = 0;
    active_vertices.insert({0, source});

    while (!active_vertices.empty()) {
        int vertex = active_vertices.begin()->second;
        active_vertices.erase(active_vertices.begin());

        for (auto edge : graph[vertex]) {
            int neighbor = edge.first;
            int weight = edge.second;
```

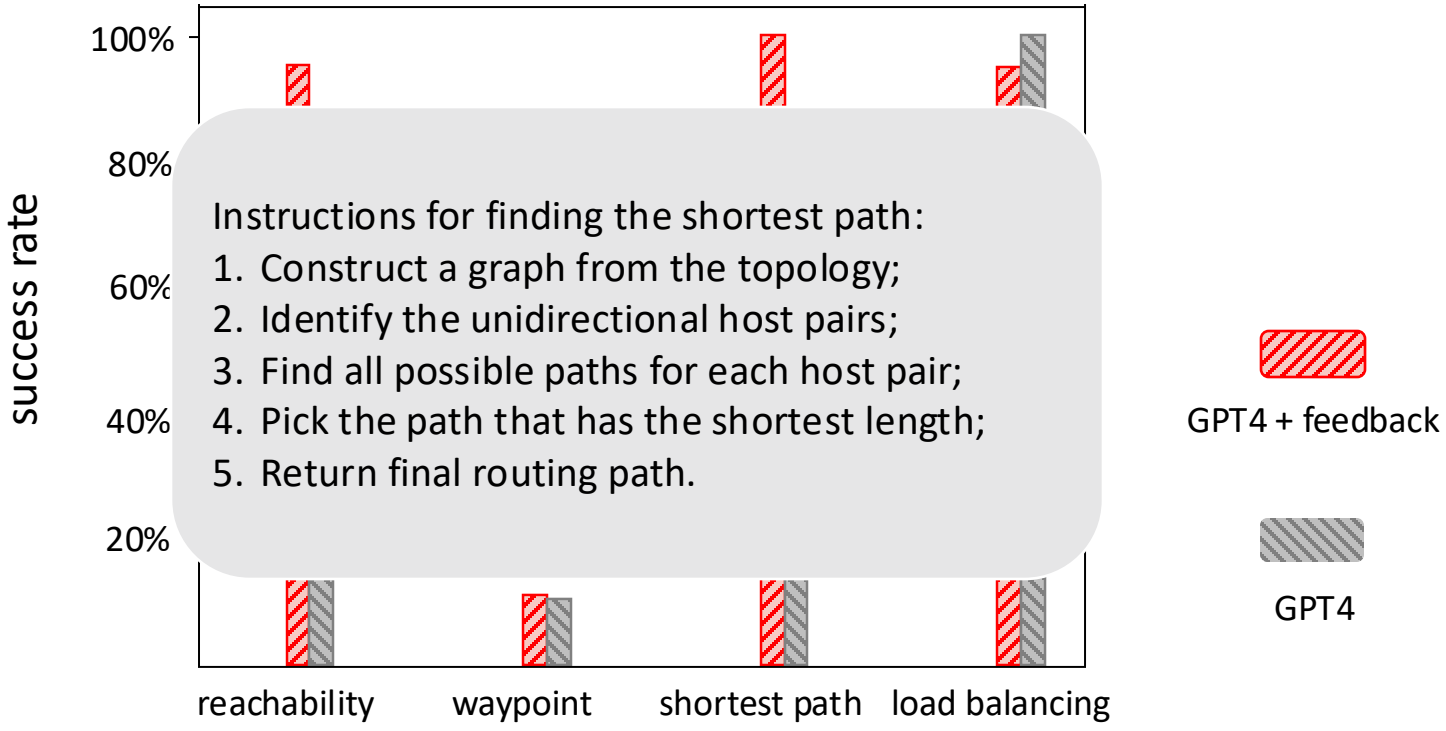
Poor performance even for simple tasks



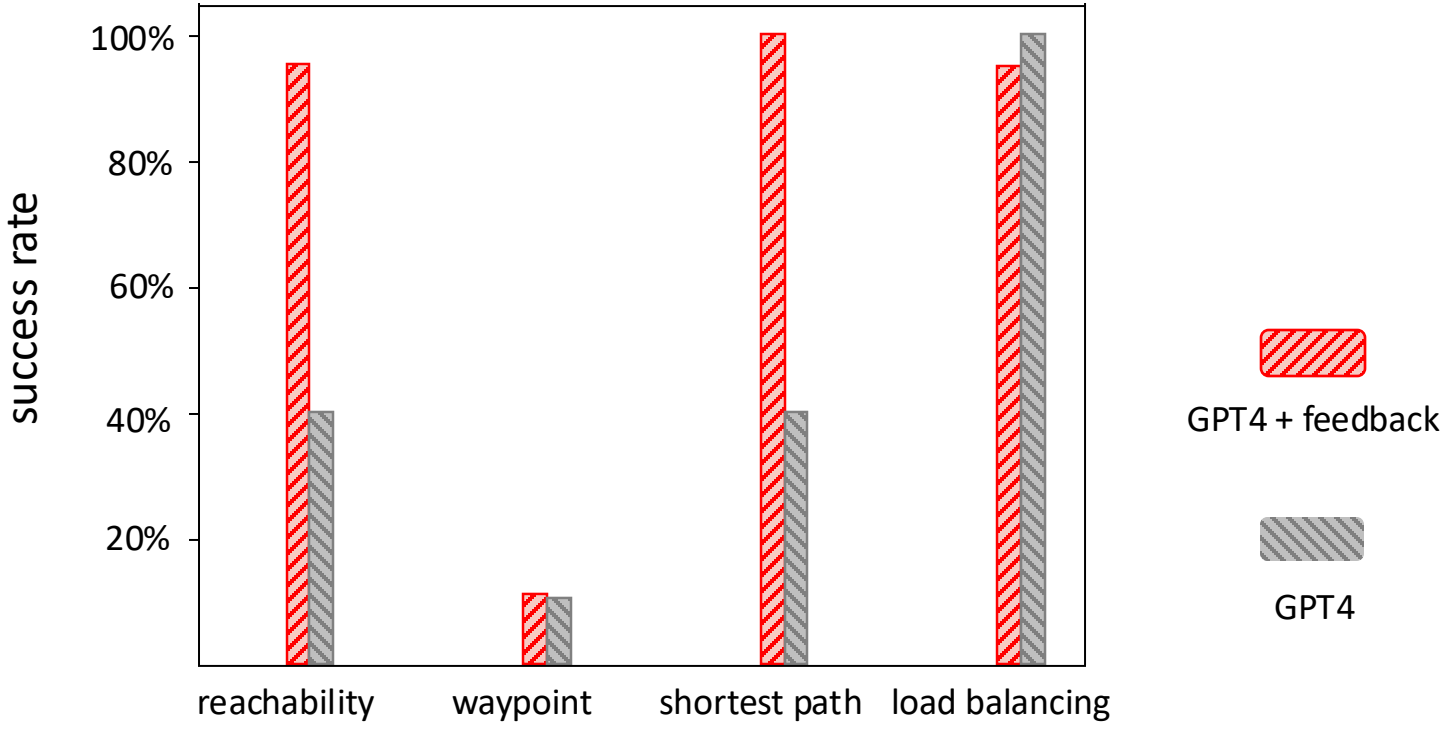
What if we give feedback?



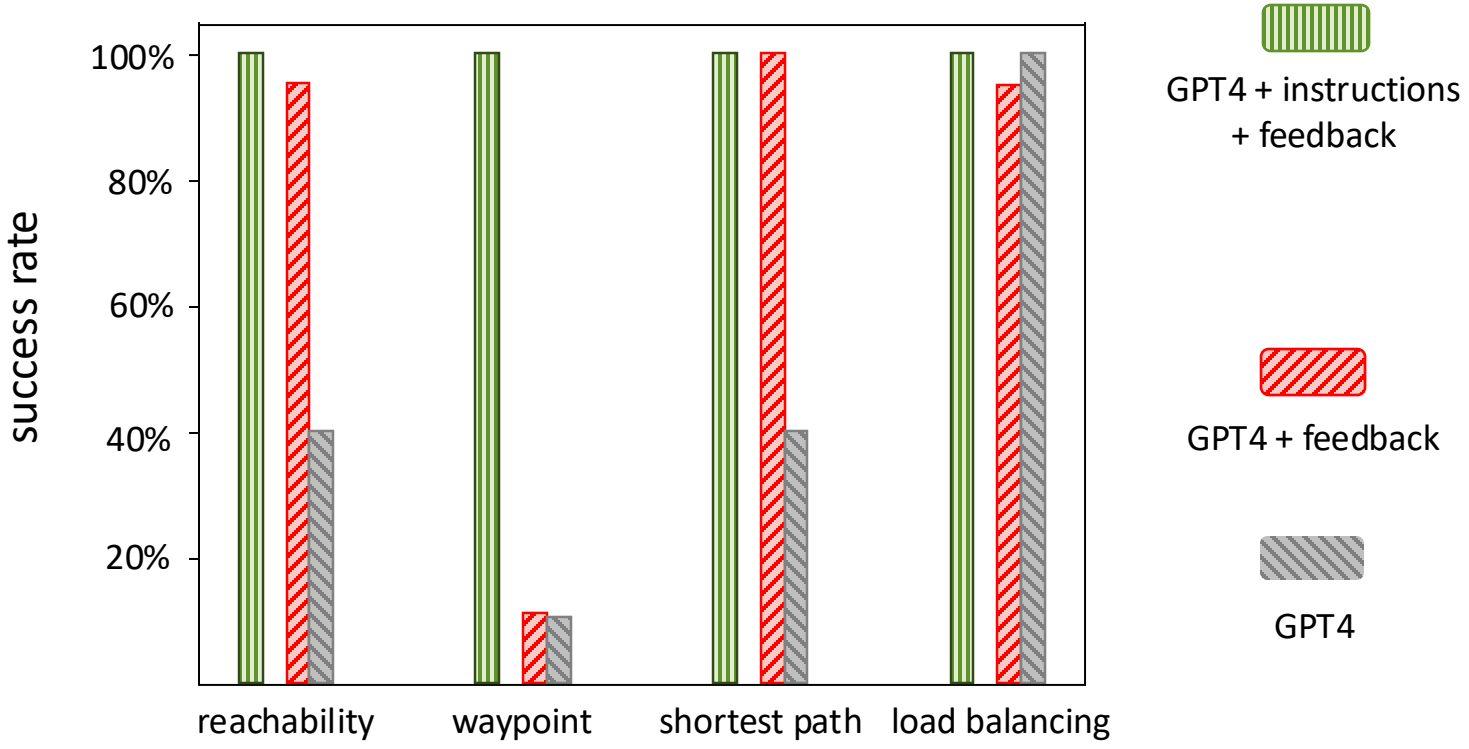
What if we also provide some algorithmic help?



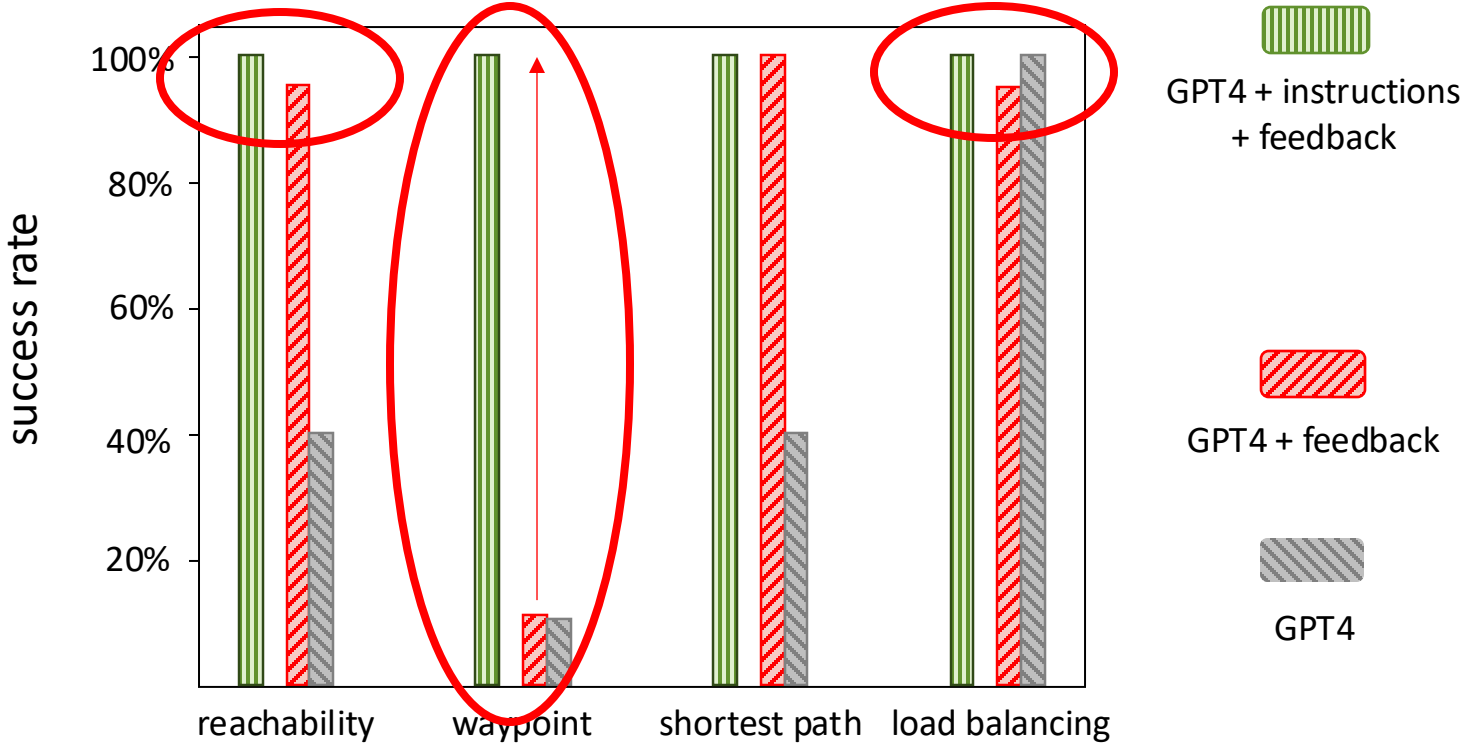
What if we also provide some algorithmic help?



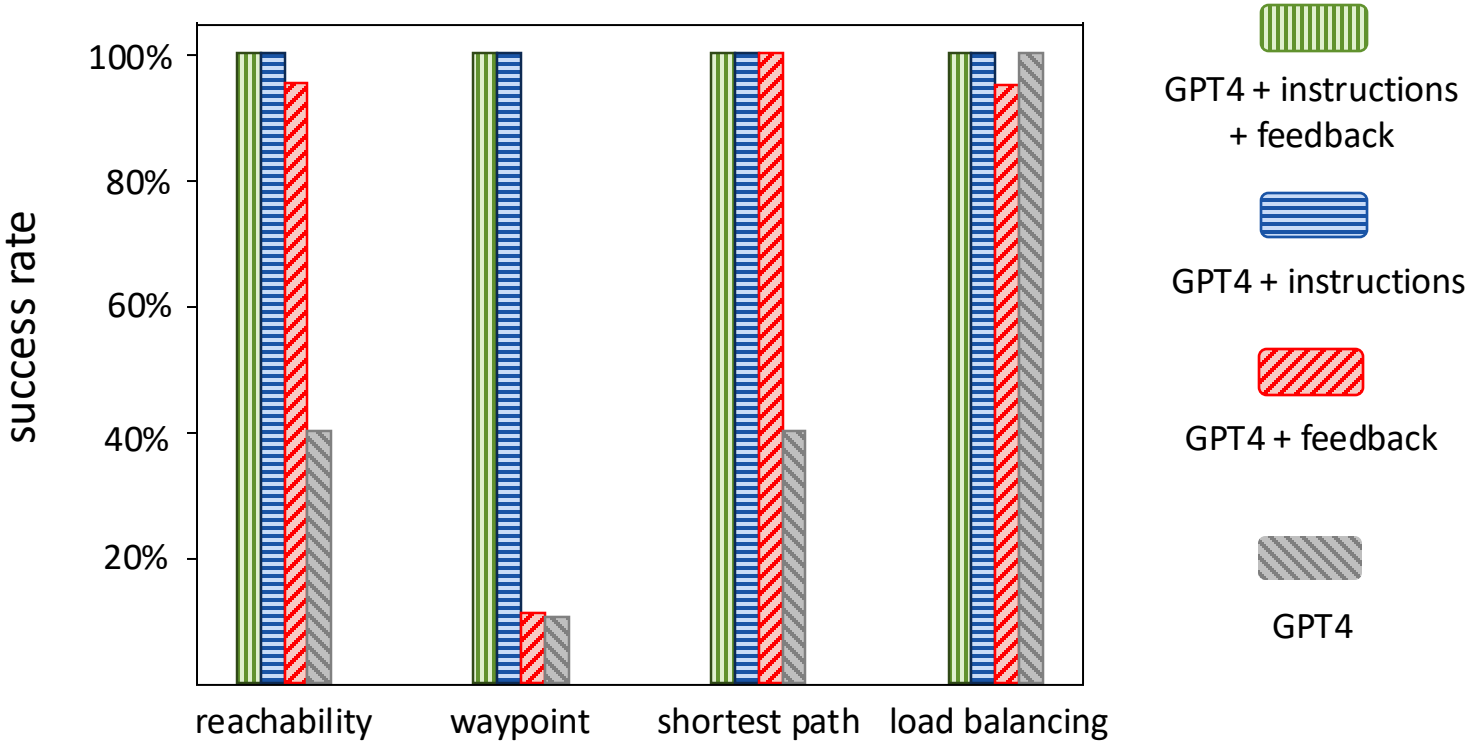
What if we also provide some algorithmic help?



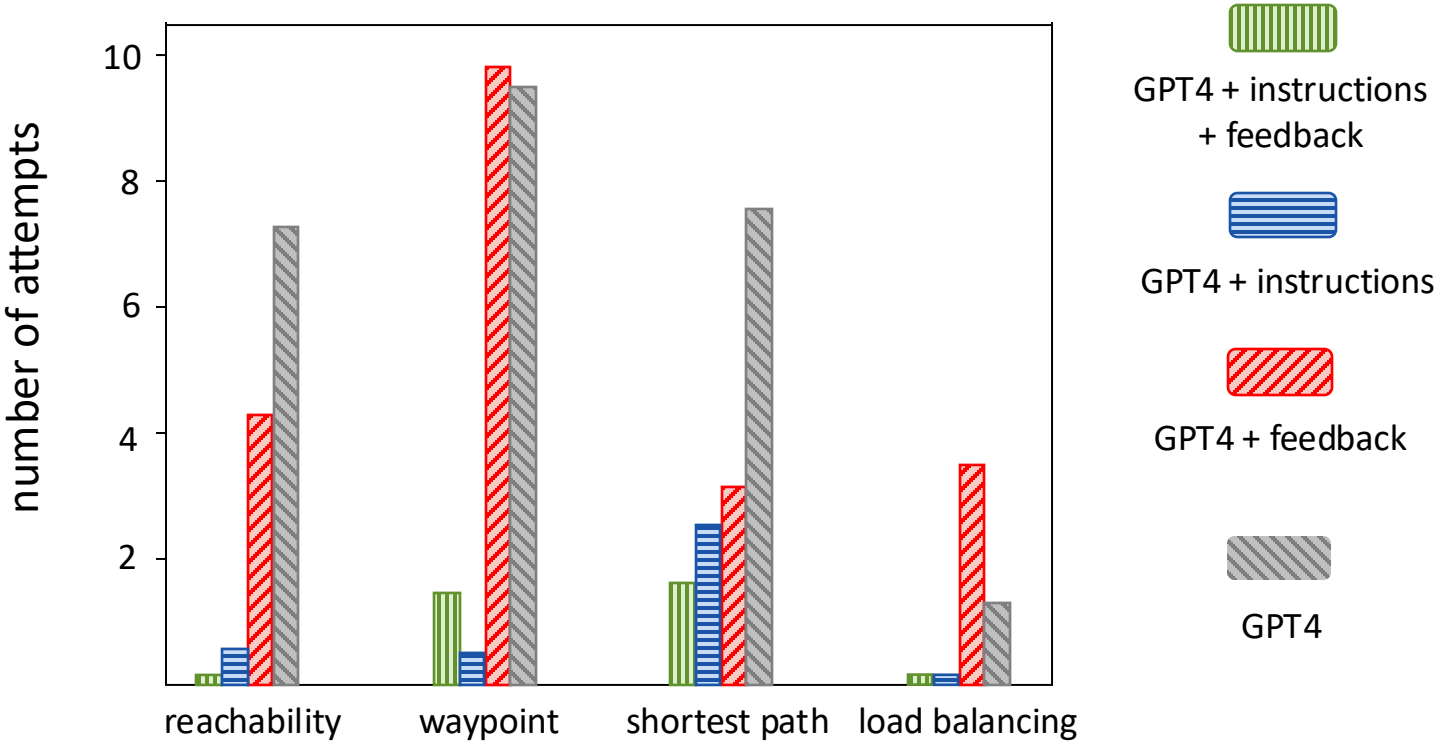
What if we also provide some algorithmic help?



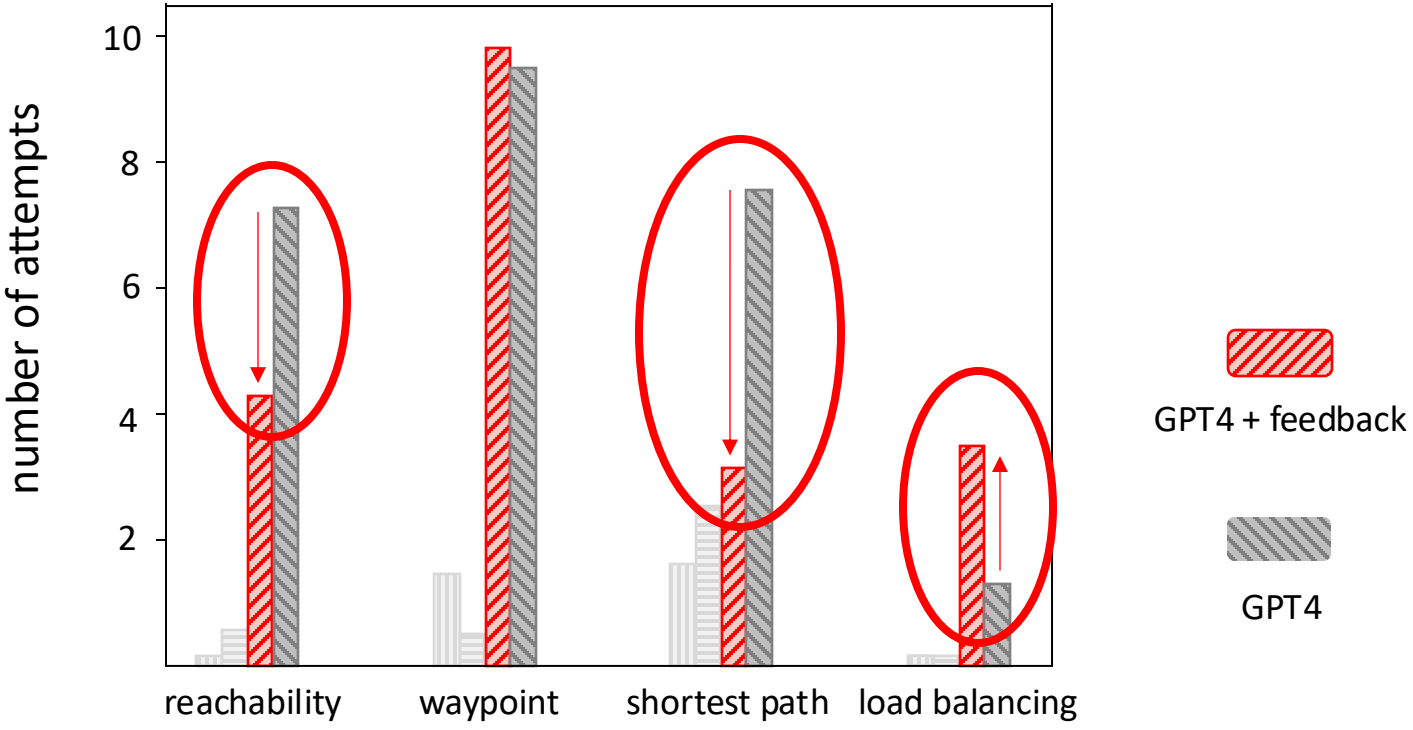
What if we provide algorithmic help without feedback?



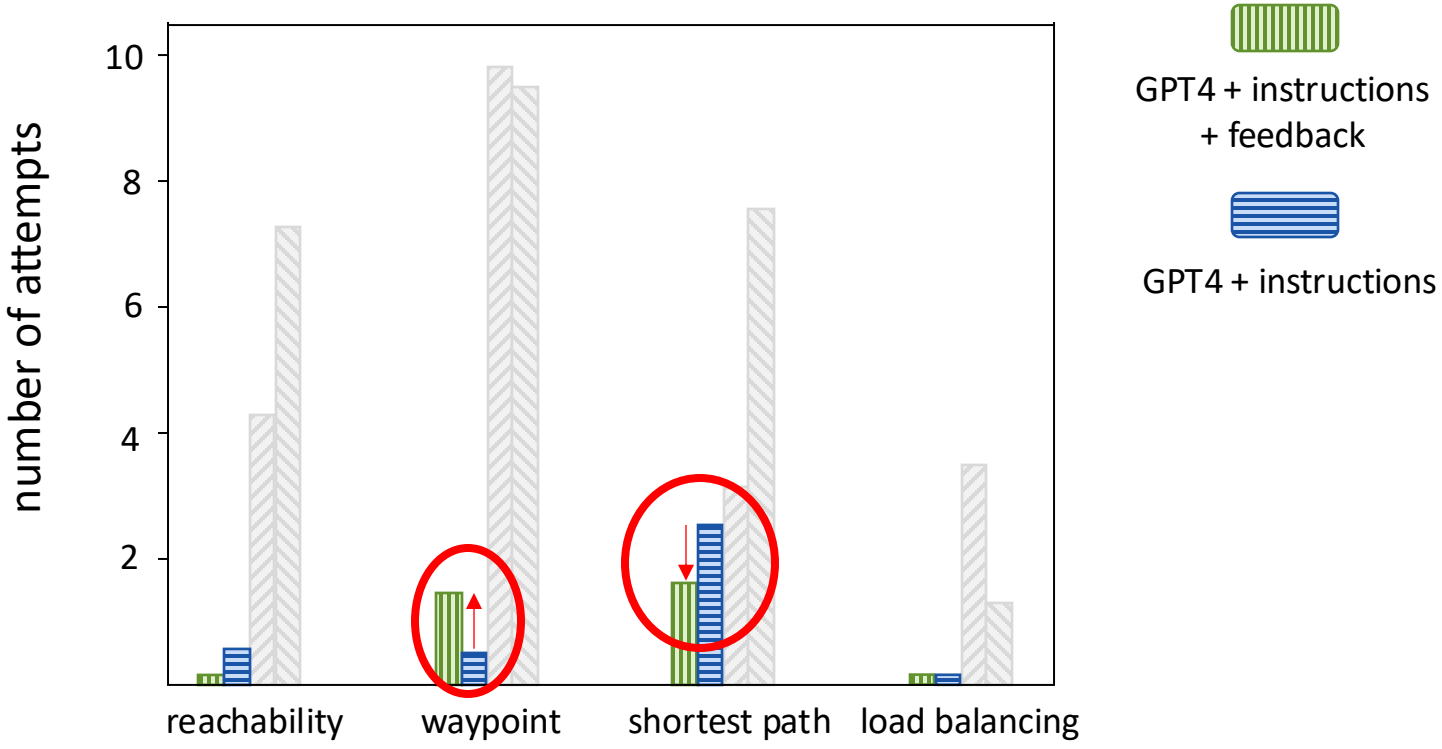
Does providing precise feedback always help?



Does providing precise feedback always help?



Does providing precise feedback always help?



Smaller models could not produce meaningful code

We tested a few additional models:

- **phy** (specialized in Python)
- **mistral**
- **codellama** 7B, 13B, 34B (with 4-bit quantization)
- **GPT 3.5**

None of these models generated correct code

- from basic syntax errors to wrong semantic of data structures, logic, …

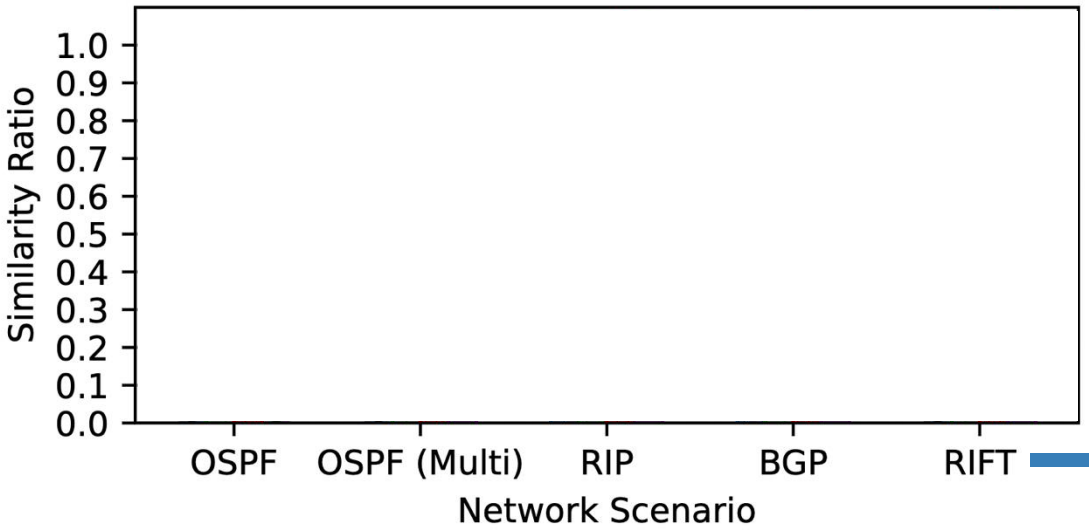
We'll focus on three tasks in orchestrating networks

1. Translating high-level requirements to a formal specification format

2. Adapting code to new requirements

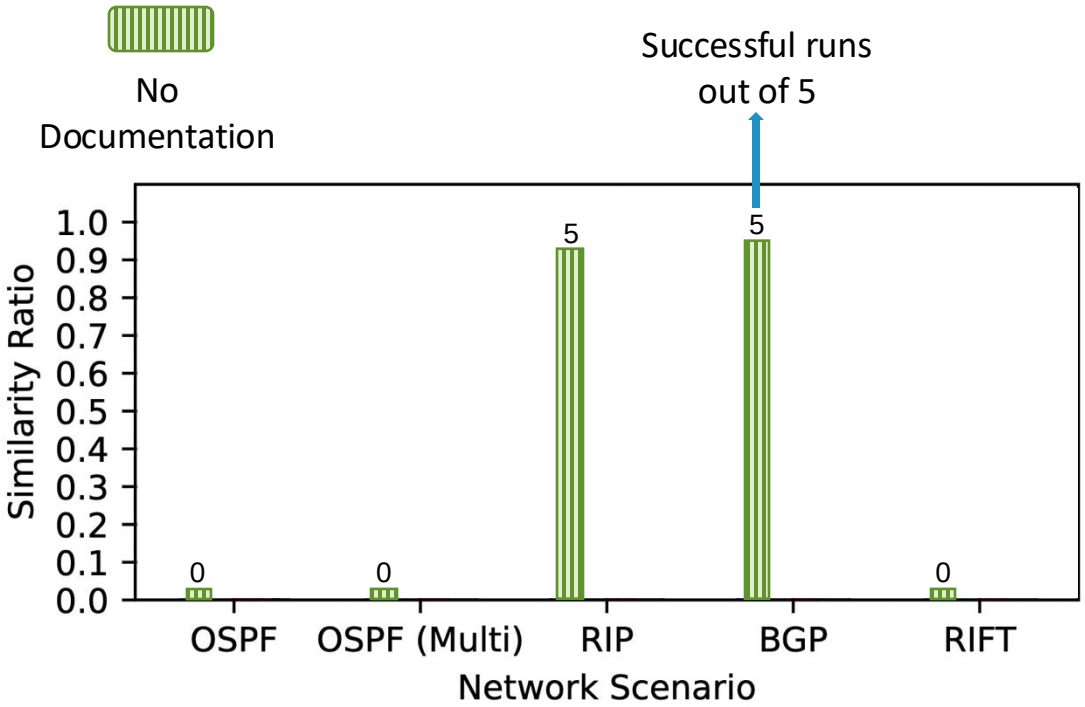
3. **Generating low-level configurations**

Generating low-level configurations



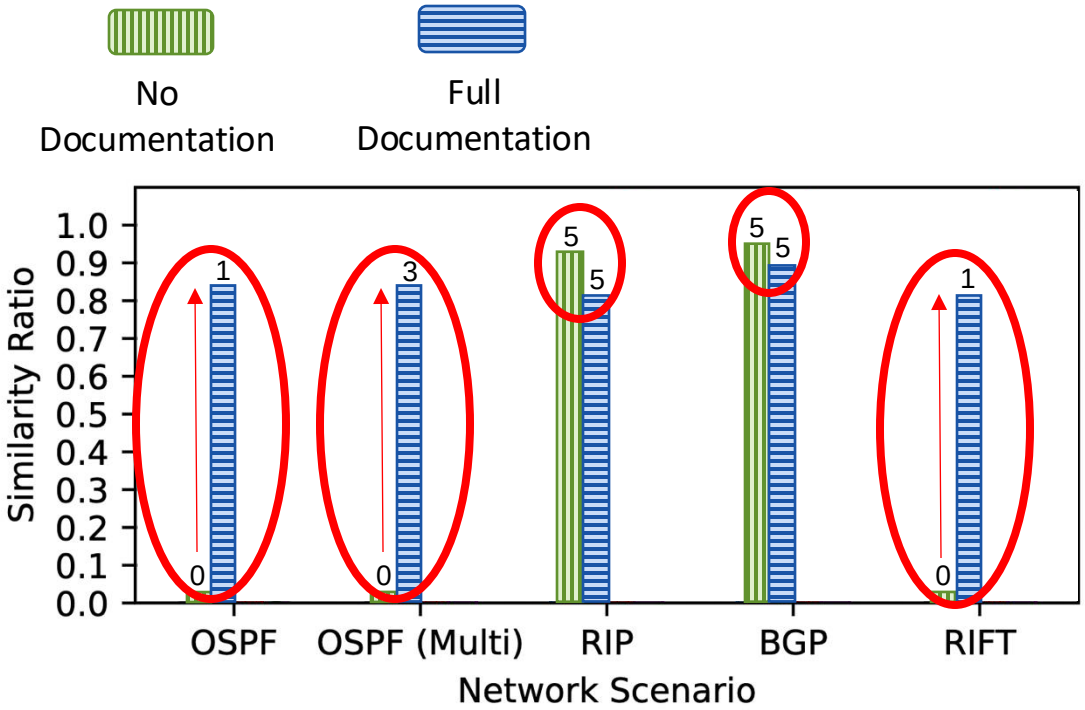
we wrote the documentation ourselves

Generating low-level configurations



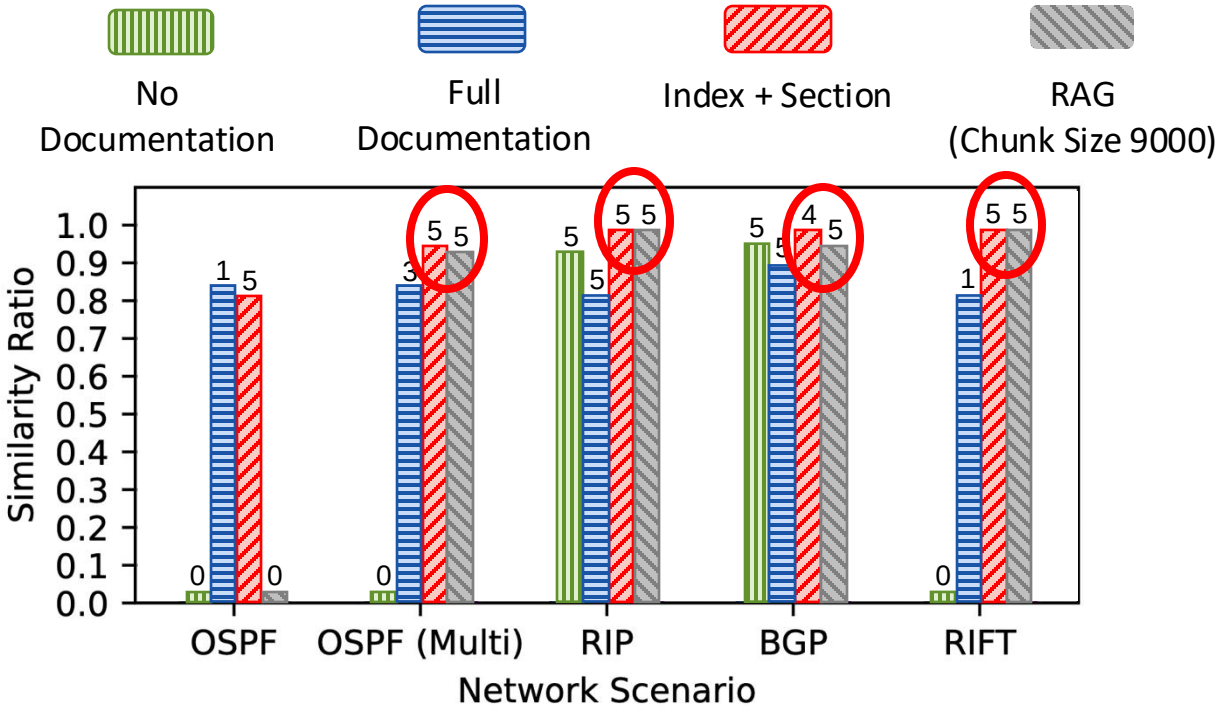
LLMs know something already!

Generating low-level configurations



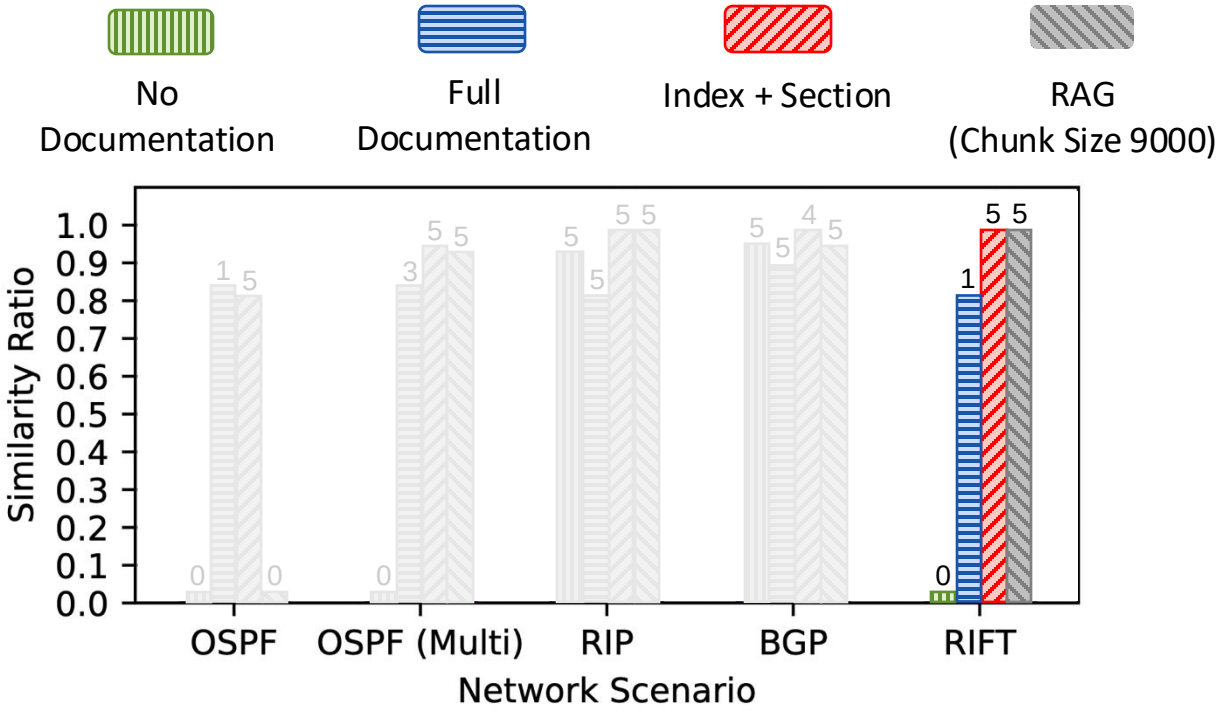
LLMs learn on-the-fly! **But Context window matters!**

Generating low-level configurations



Index + Section and RAG reduce the context size.

Generating low-level configurations



FRROUTING

LLMs can take advantage of knowledge without fine-tuning.

Building LLM-based system for networks

1. Split complex tasks into smaller subtasks

2. Support task-specific verifiers

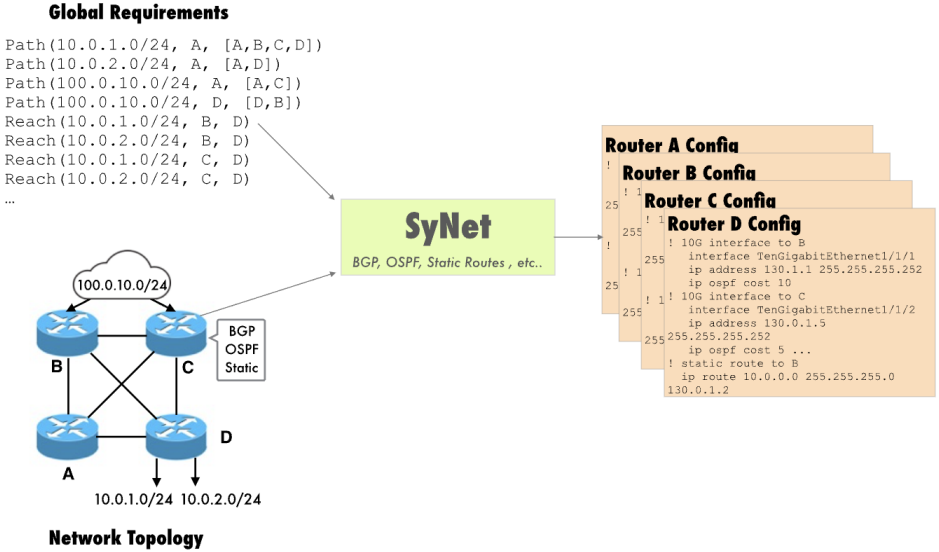
3. Keep humans still in the loop

Prototypes

1. LLMs in action with network synthesizers

2. LLMs from intents to low-level configuration

LLMs in action with network synthesizers



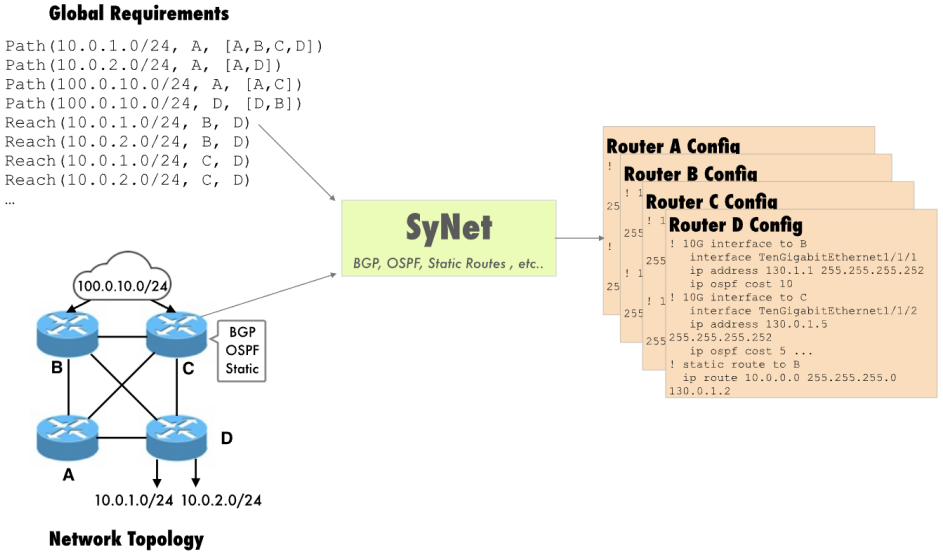
LLMs in action with network synthesizers

network operator

*"Traffic classified as **media** should be routed **from Rome to Milan** exclusively using the **OSPF** protocol"*

LLM

Does GPT know how to write SyNet code?
Not at all!



LLMs in action with network synthesizers

network operator

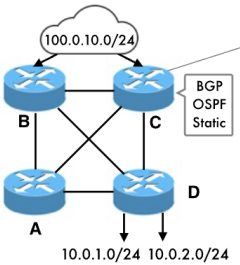
*”Traffic classified as **media** should be routed **from Rome to Milan** exclusively using the **OSPF** protocol”*

LLM

SyNET paper

Global Requirements

Fwd(media, rome, milan, ospf)



Network Topology

SyNet

BGP, OSPF, Static Routes , etc..

Router A Confia

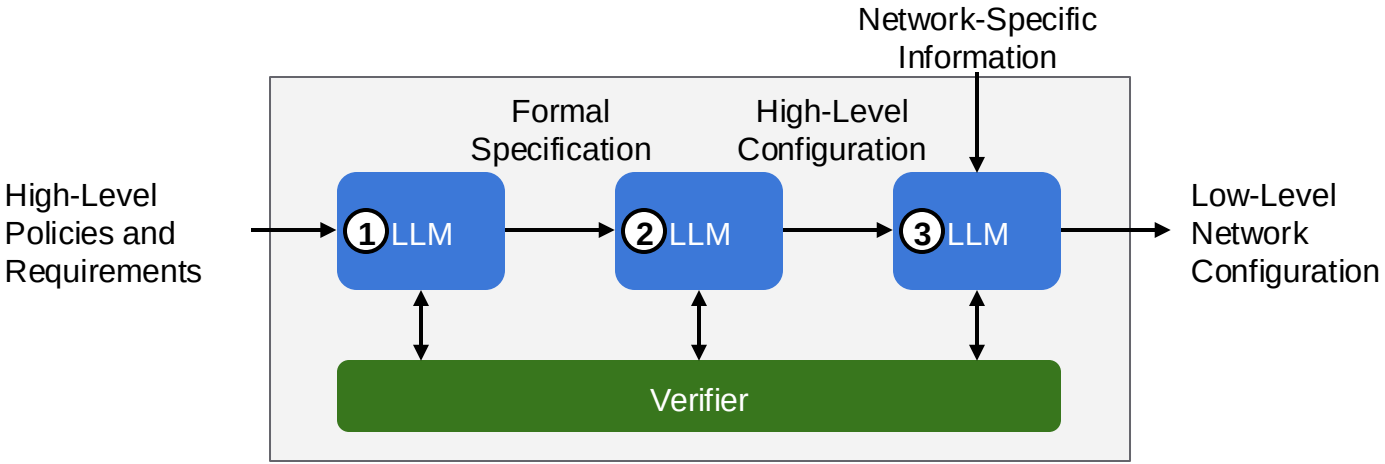
Router B Confia

Router C Confia

Router D Config

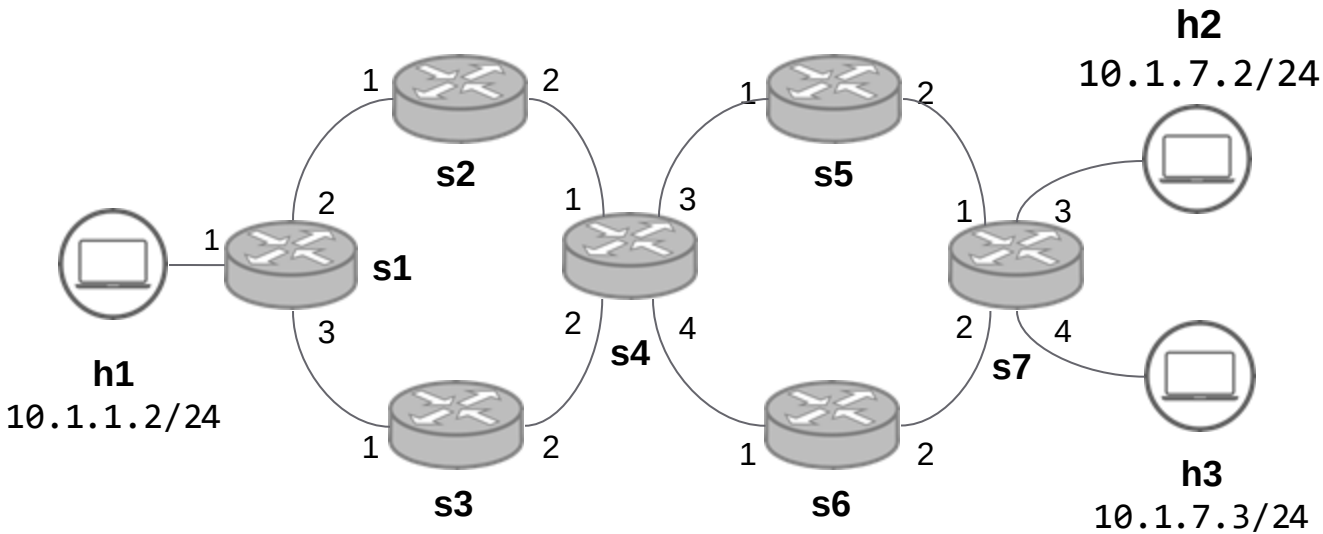
```
! 10G interface to B
  interface TenGigabitEthernet1/1/1
  ip address 130.1.1 255.255.255.252
  ip ospf cost 10
! 10G interface to C
  interface TenGigabitEthernet1/1/2
  ip address 130.0.1.5
255.255.255.252
  ip ospf cost 5 ...
! static route to B
  ip route 10.0.0.0 255.255.255.0
130.0.1.2
```

Netbuddy: LLMs from intents to configuration



Evaluated Topology

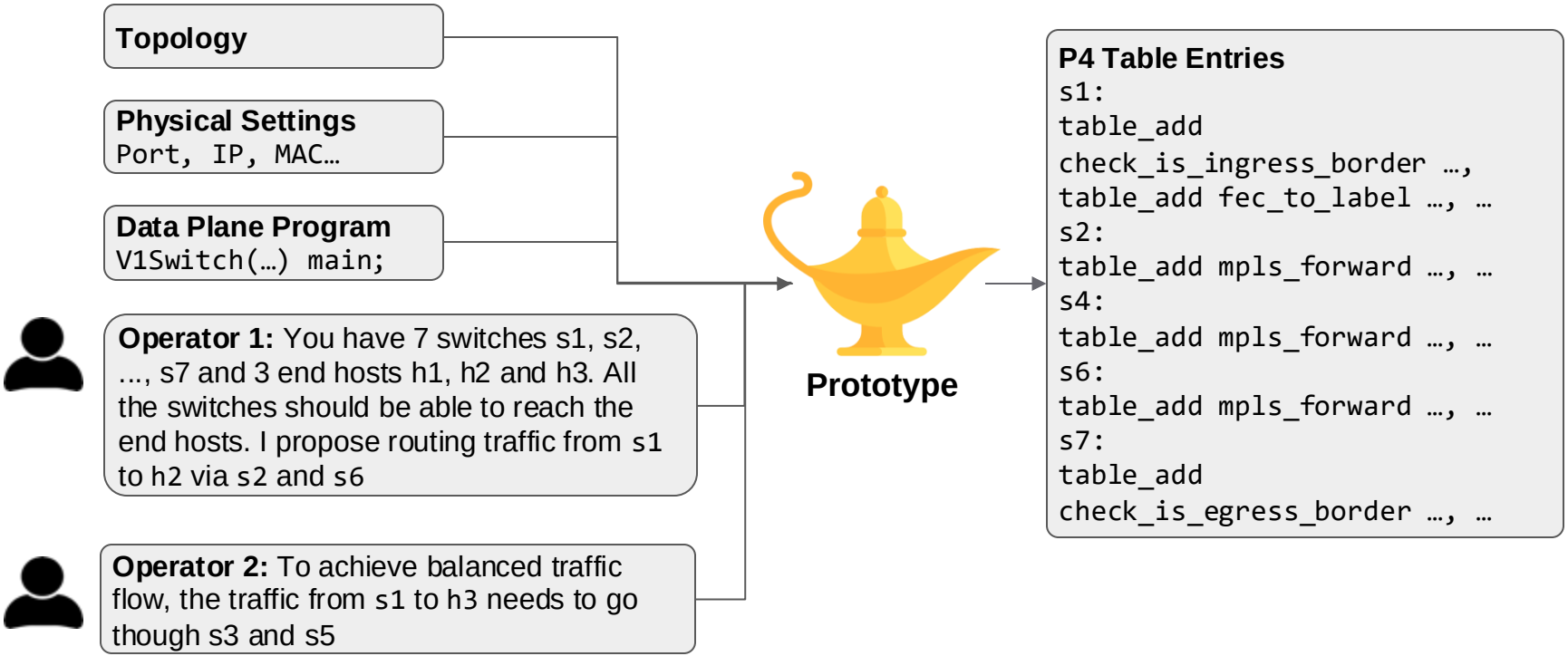
prototype video



Emulated using Kathará

From requirements to P4 code

prototype video

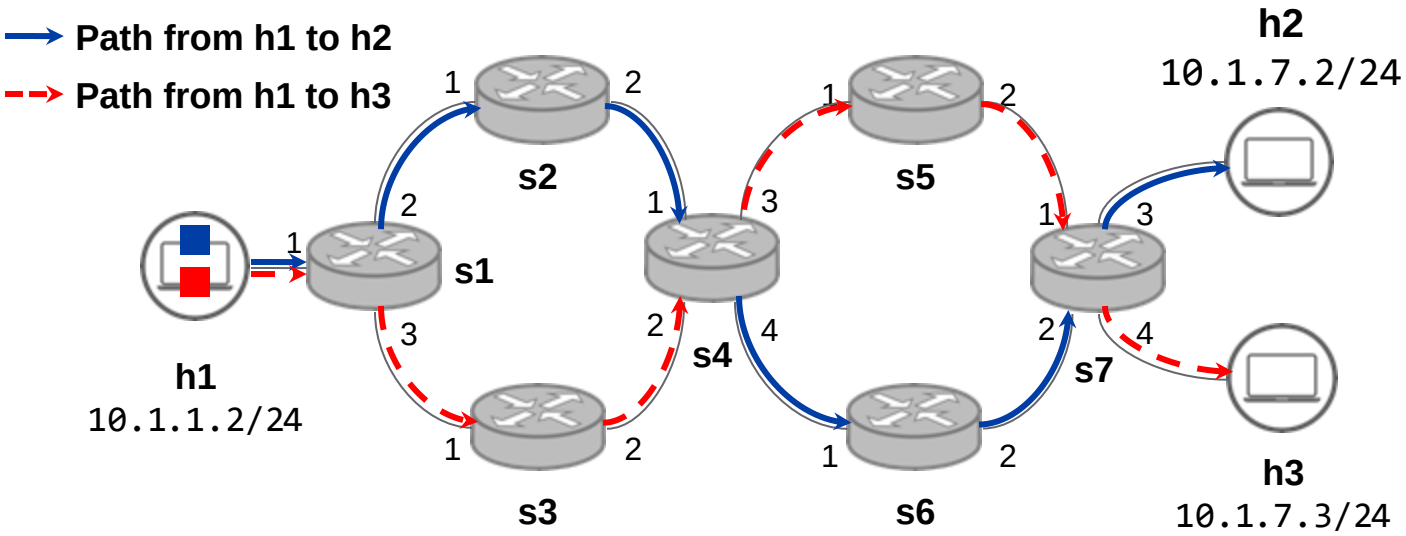


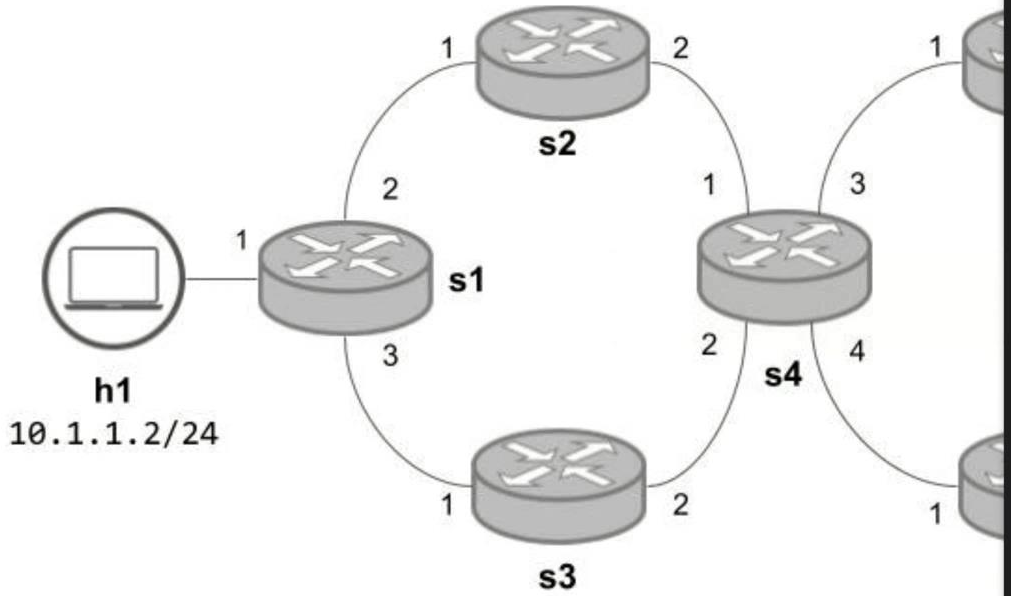
Controlling the network

prototype video

Operator 1: You have 7 switches s1, s2, ..., s7 and 3 end hosts h1, h2 and h3. All the switches should be able to reach the end hosts. I propose routing traffic from `s1` to `h2` via `s2` and `s6`

Operator 2: To achieve balanced traffic flow, the traffic from `s1` to `h3` needs to go though s3 and s5




1
2
s2
2
1
3
1
s1
s4
3
2
4
h1
10.1.1.2/24
1
2
s3
1

More application of LLMs

Applications	Related work
Network Configuration and Management	LLMs to Synthesize Router Configurations? (HotNets ’23) Network Management Using Code Generated by LLMs (HotNets ‘23)
Protocol Specification Extraction	Extracting Protocol Specifications Using LLMs (HotNets ‘23)
More Specific Tasks	NetLLM: Adapting LLMs for Networking (SIGCOMM ‘24)
Network Research	Reproducing Network Research Results Using LLMs (HotNets ‘23)
Network for LLMs	vLLM (SIGOPS ’23), Orca (OSDI ’22), FlexGen (ICML ’23), …

The work has been funded by the Swedish Research Council, Digital Futures, the Swedish Innovation Agency, the Leonardo super computer

Reliable LLM applications

Write a Python function to calculate the maximum number in a set.

Reliability

Test units

```python
def find_max_in_set(numbers_set):
    """
    Find the maximum number in a set of numbers.

    Parameters:
    numbers_set (set): A set of numbers.

    Returns:
    int or float: The maximum number in the set.
    """
    # Check if the set is empty
    if not numbers_set:
        raise ValueError("The set is empty, cannot determine the maximum.")

    # Initialize the maximum number
    max_number = float('-inf')

    # Iterate through each number in the set
    for number in numbers_set:
        # Update max_number if the current number is greater
        if number > max_number:
            max_number = number

    return max_number

# Example usage
example_set = {1, 3, 5, 7, 9}
print("The maximum number in the set is:", find_max_in_set(example_set))
```

Can we trust?

Reliable LLM applications

Write a Python function to calculate the maximum number in a set.

Reliability

vulnerability

```python
def find_max_in_set(numbers_set):
    """
    Find the maximum number in a set of numbers.

    Parameters:
    numbers_set (set): A set of numbers.

    Returns:
    int or float: The maximum number in the set.
    """
    # Check if the set is empty
    if not numbers_set:
        raise ValueError("The set is empty, cannot determine the maximum.")

    # Initialize the maximum number
    max_number = float('-inf')

    # Iterate through each number in the set
    for number in numbers_set:
        # Update max_number if the current number is greater
        if number > max_number:
            max_number = number

    return max_number

# Example usage
example_set = 1
print("The maximum number in the set is:", find_max_in_set(example_set))
```

Can we trust?

Reliable LLM applications by formal verification

Write me a Dafny code to calculate.

```
method findMax(arr: array<int>) returns (max: int)
  requires arr.Length > 0  // The array must not be empty
  ensures forall k :: 0 <= k < arr.Length ==> arr[k] <= max  // Ensure the result is the maximum
{
  max := arr[0];  // Initialize max with the first element of the array

  // Loop through the array to find the maximum
  var i := 1;
  while i < arr.Length
    invariant 0 < i <= arr.Length  // Loop invariant to help Dafny verify the loop
    invariant forall j :: 0 <= j < i ==> arr[j] <= max  // All elements before i are less than or equal to max
  {
    if arr[i] > max {
      max := arr[i];
    }
    i := i + 1;
  }
}
```

Reliability

We can trust the code, as long as

- the **verified property** is **correct**, and
- **verification passes**

Conclusions

We

- present the first **benchmark** for network orchestration and share experience
- automate common workflows with LLM-powered **prototypes**

Opportunities: LLMs can dramatically simplify and automate complex network orchestration tasks.

Challenges: unreliability, cost, …

SCAN ME

Thank you!

The work has been funded by the Swedish Research Council, Digital Futures, the Swedish Innovation Agency, the Leonardo super computer