Gigaflow: Pipeline-Aware Sub-Traversal Caching for Modern SmartNICs

Annus Zulfiqar

Ali Imran, Venkat Kunaparaju, Ben Pfaff,

Gianni Antichi, Muhammad Shahbaz

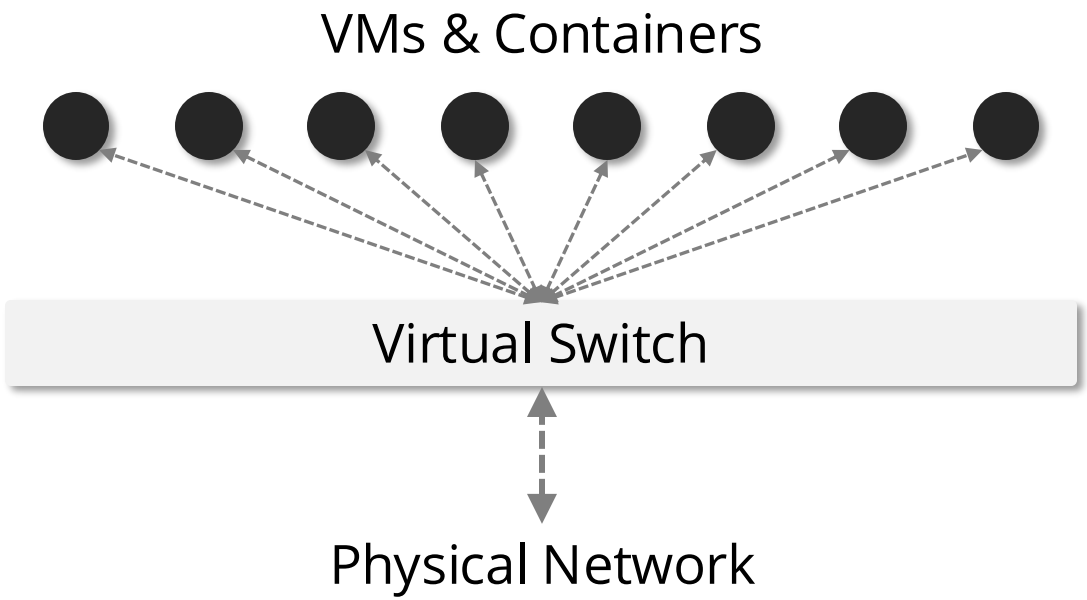
UNIVERSITY OF MICHIGAN

PURDUE UNIVERSITY

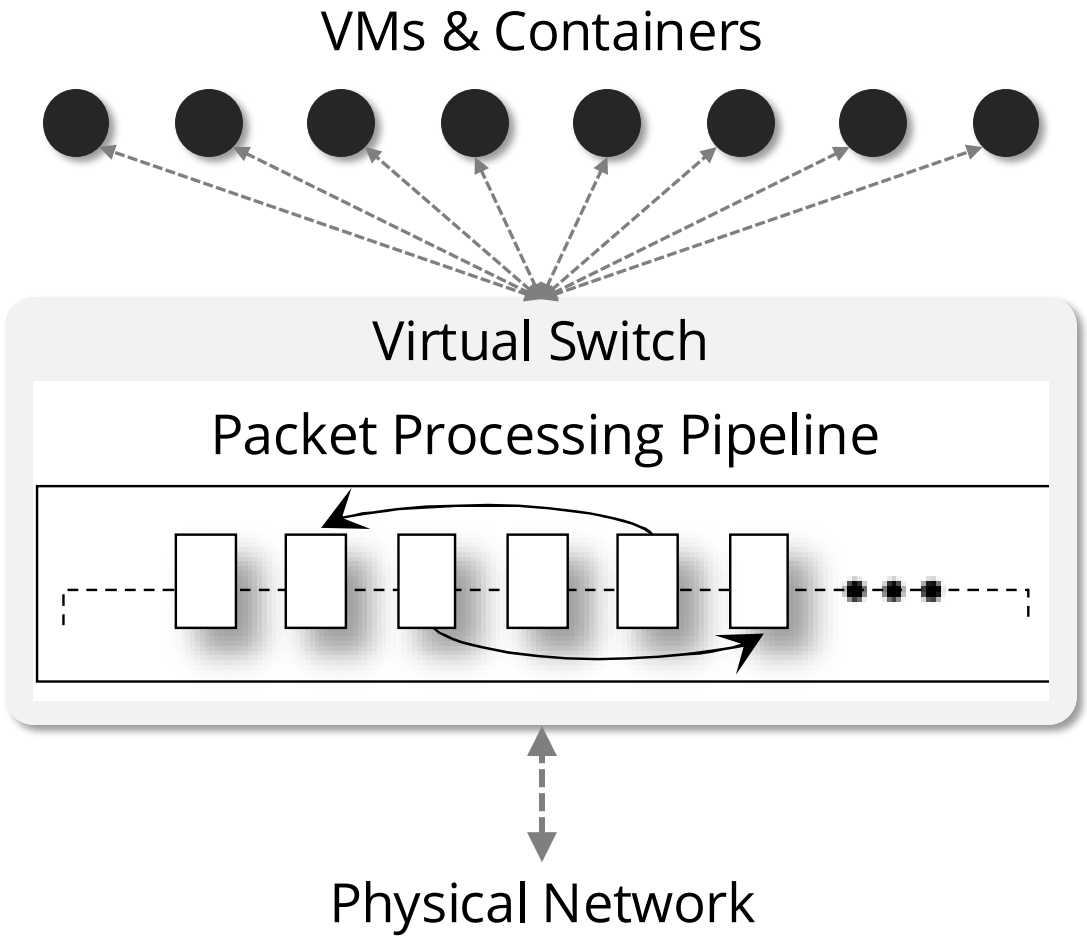
feldera

POLITECNICO MILANO 1863

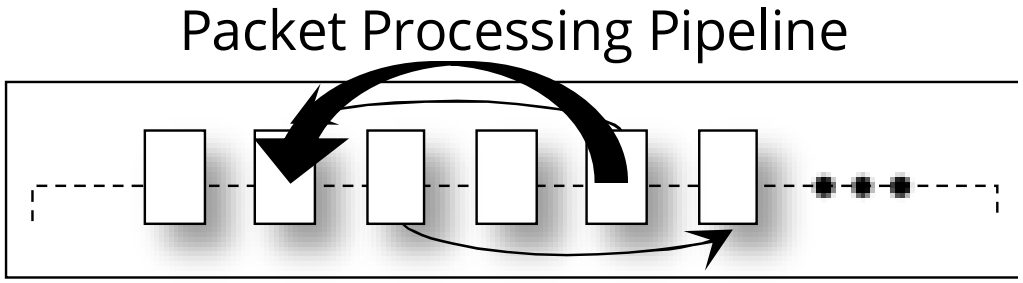
Virtual Switches in the Data Center



Virtual Switches in the Data Center

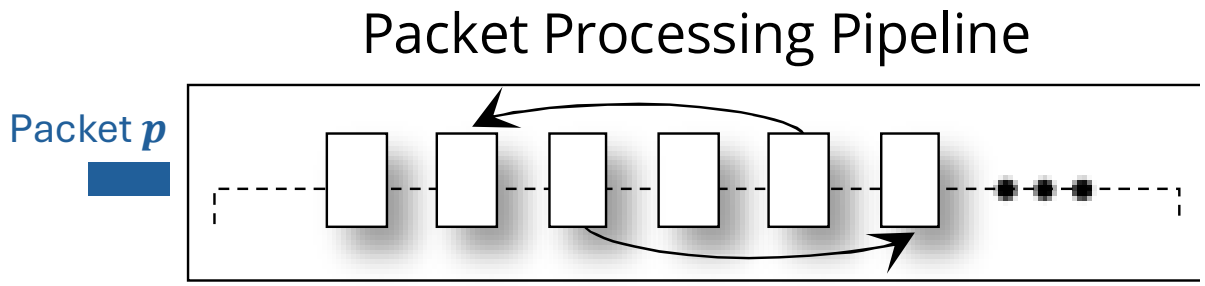


Virtual Switches in the Data Center

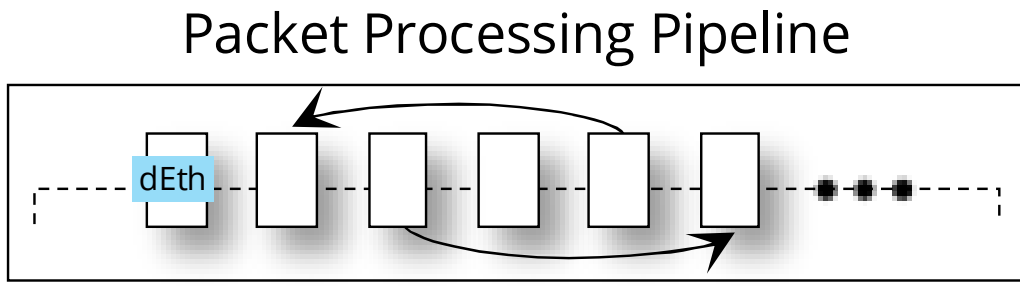


Some networking policies require **recursive application of rules**, e.g., encapsulation and decapsulation actions

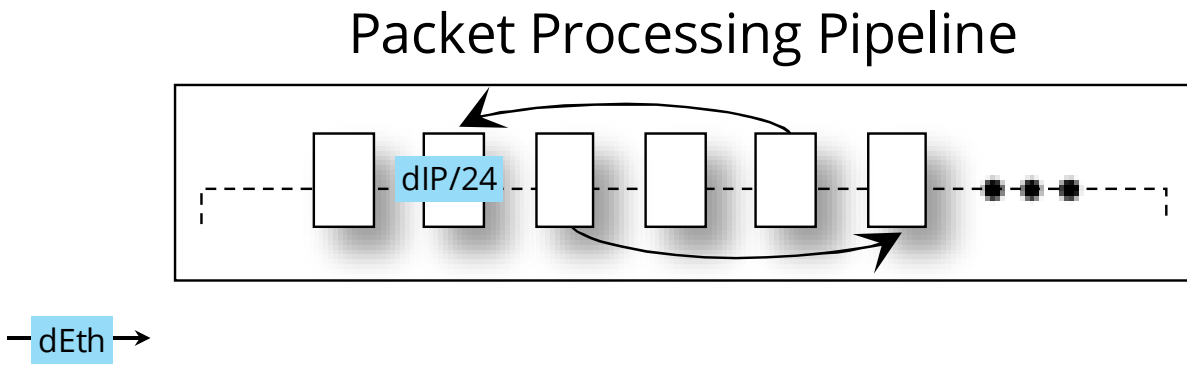
Virtual Switches in the Data Center



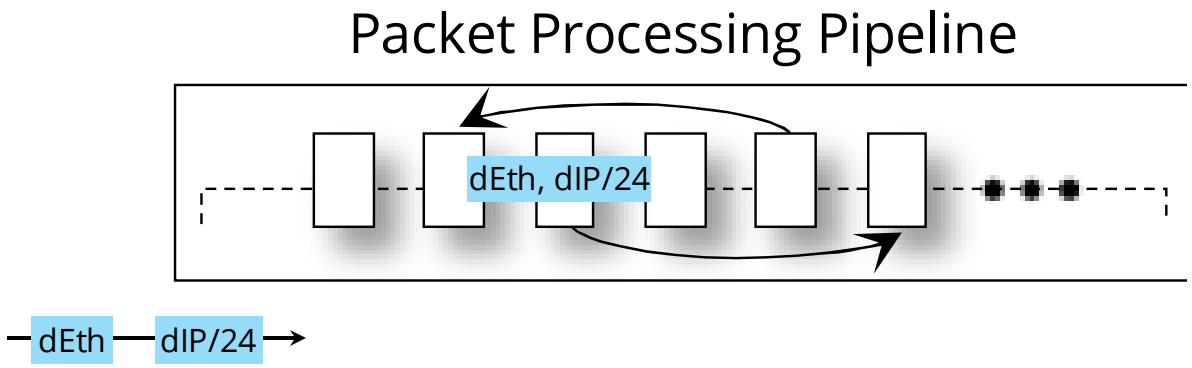
Virtual Switches in the Data Center



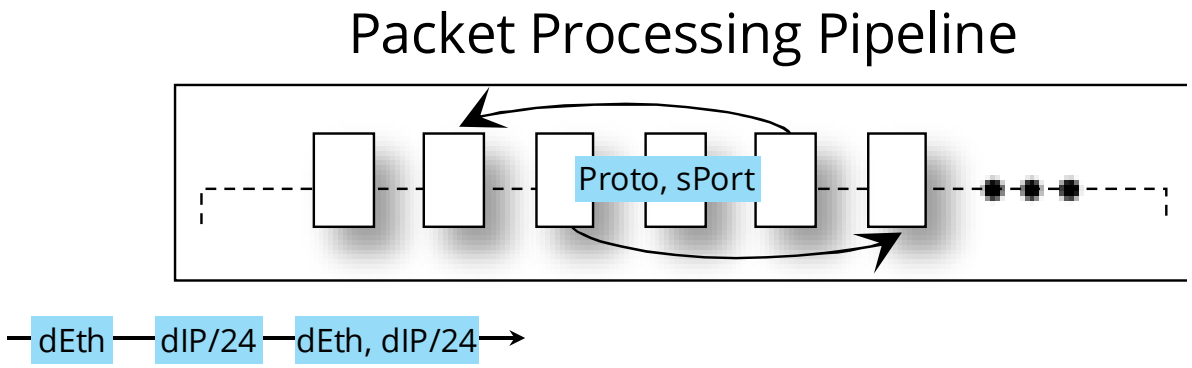
Virtual Switches in the Data Center



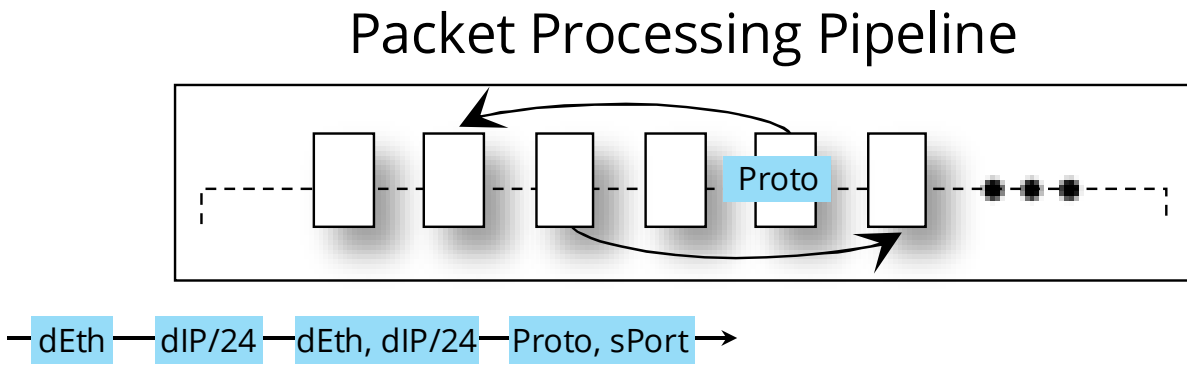
Virtual Switches in the Data Center



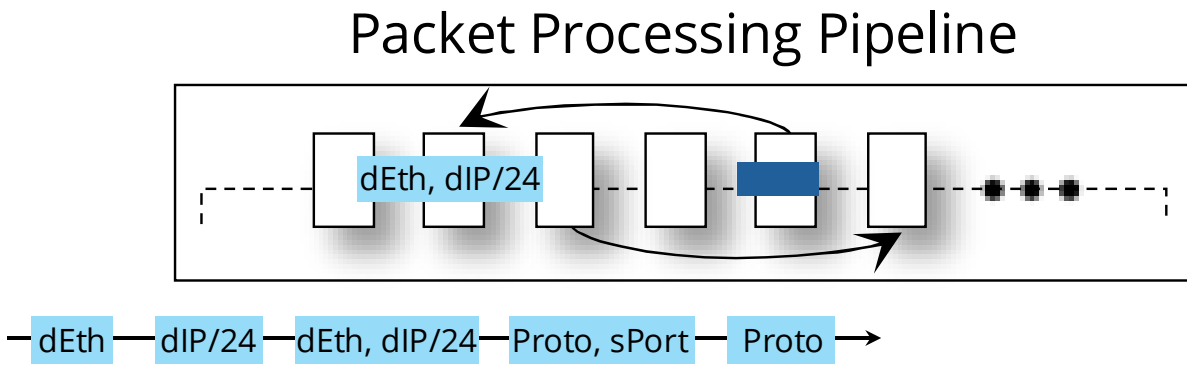
Virtual Switches in the Data Center



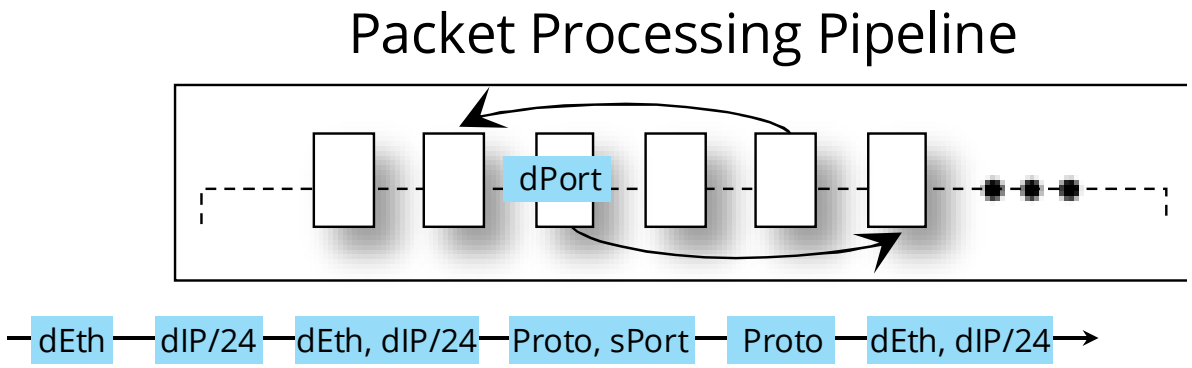
Virtual Switches in the Data Center



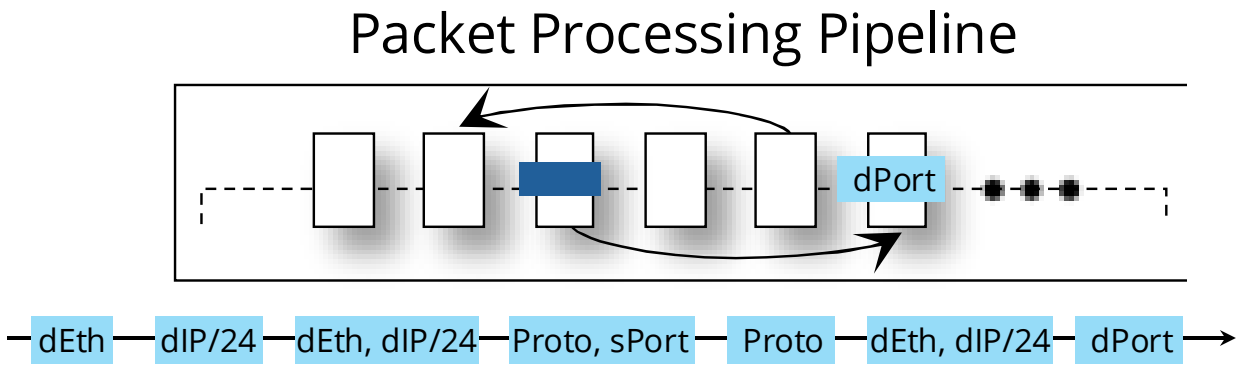
Virtual Switches in the Data Center



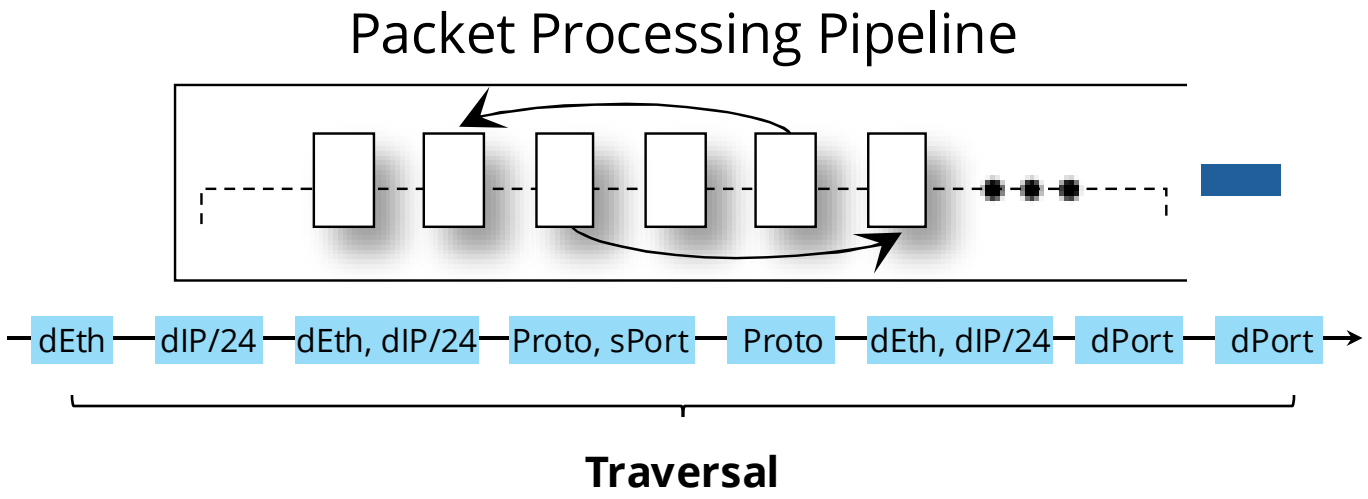
Virtual Switches in the Data Center



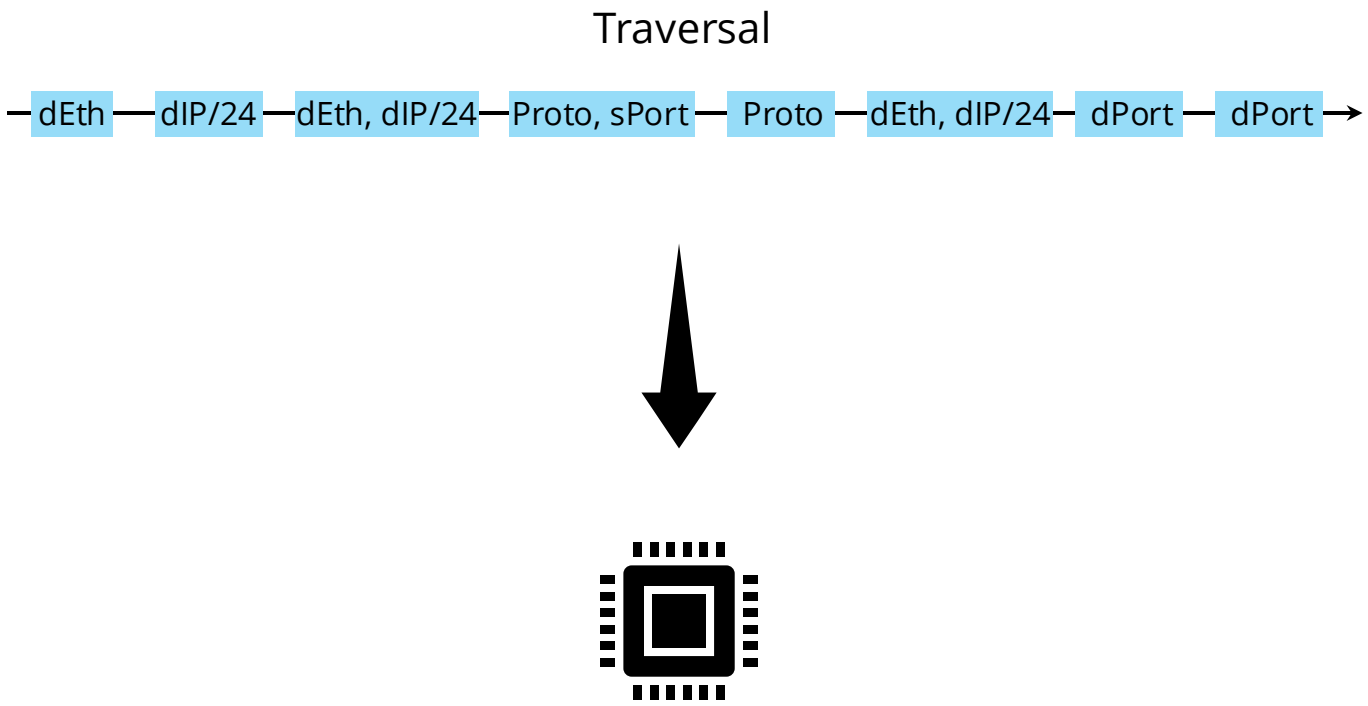
Virtual Switches in the Data Center



Virtual Switches in the Data Center

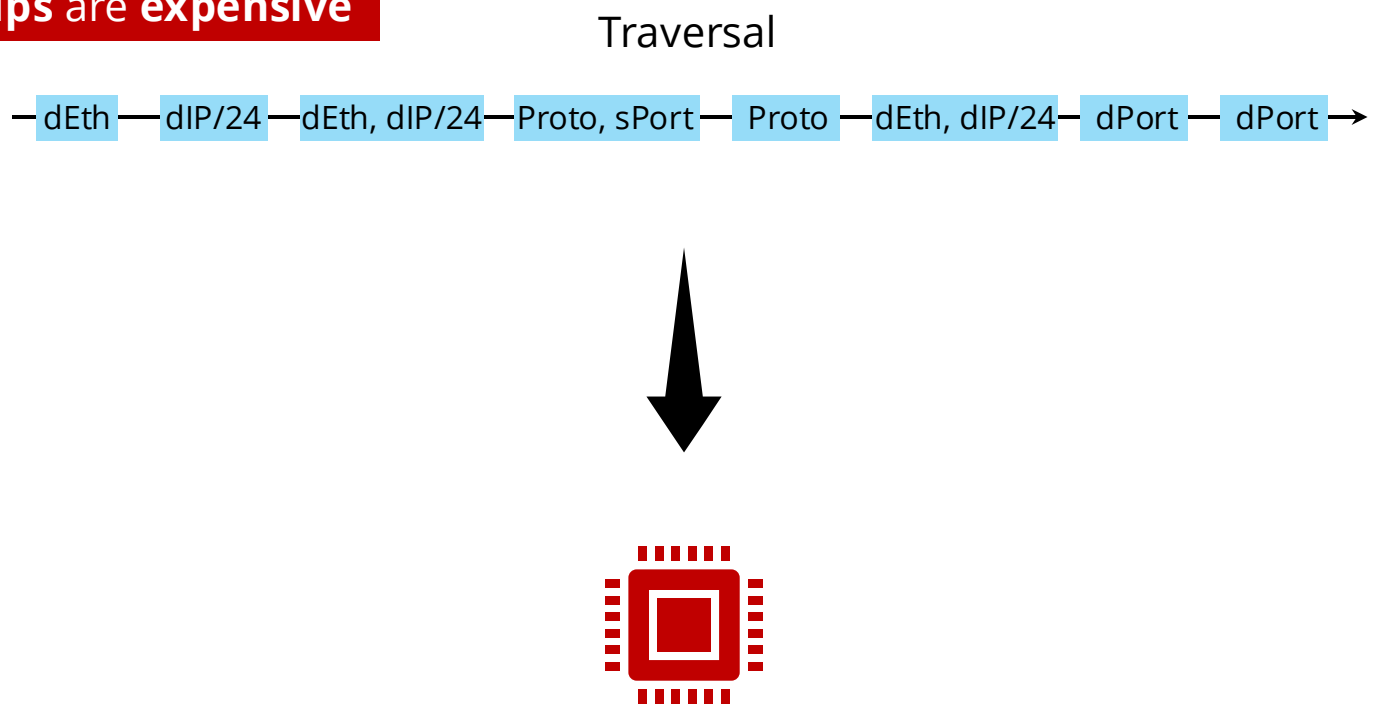


Virtual Switches in the Data Center



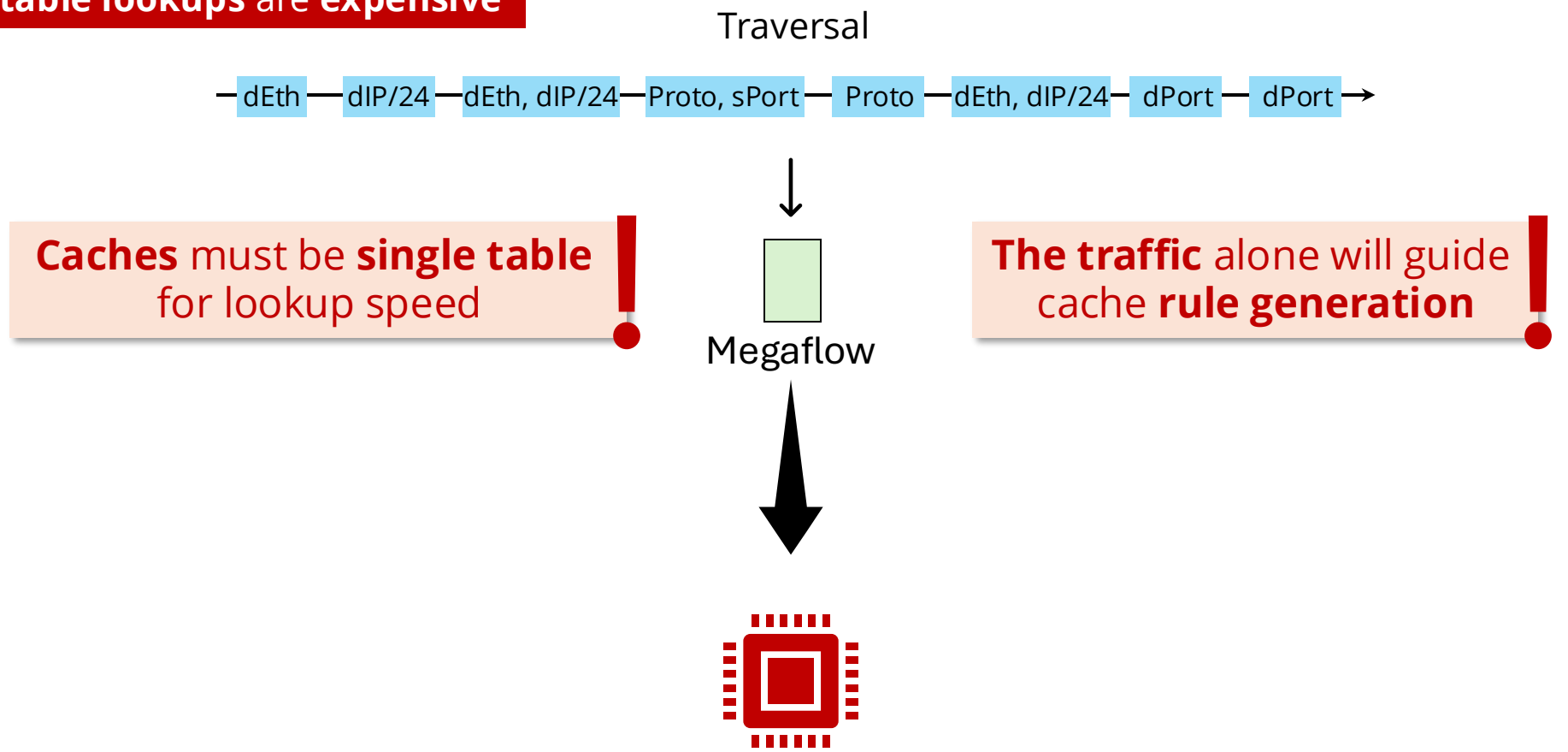
Virtual Switches in the Data Center

Multi-table lookups are **expensive**

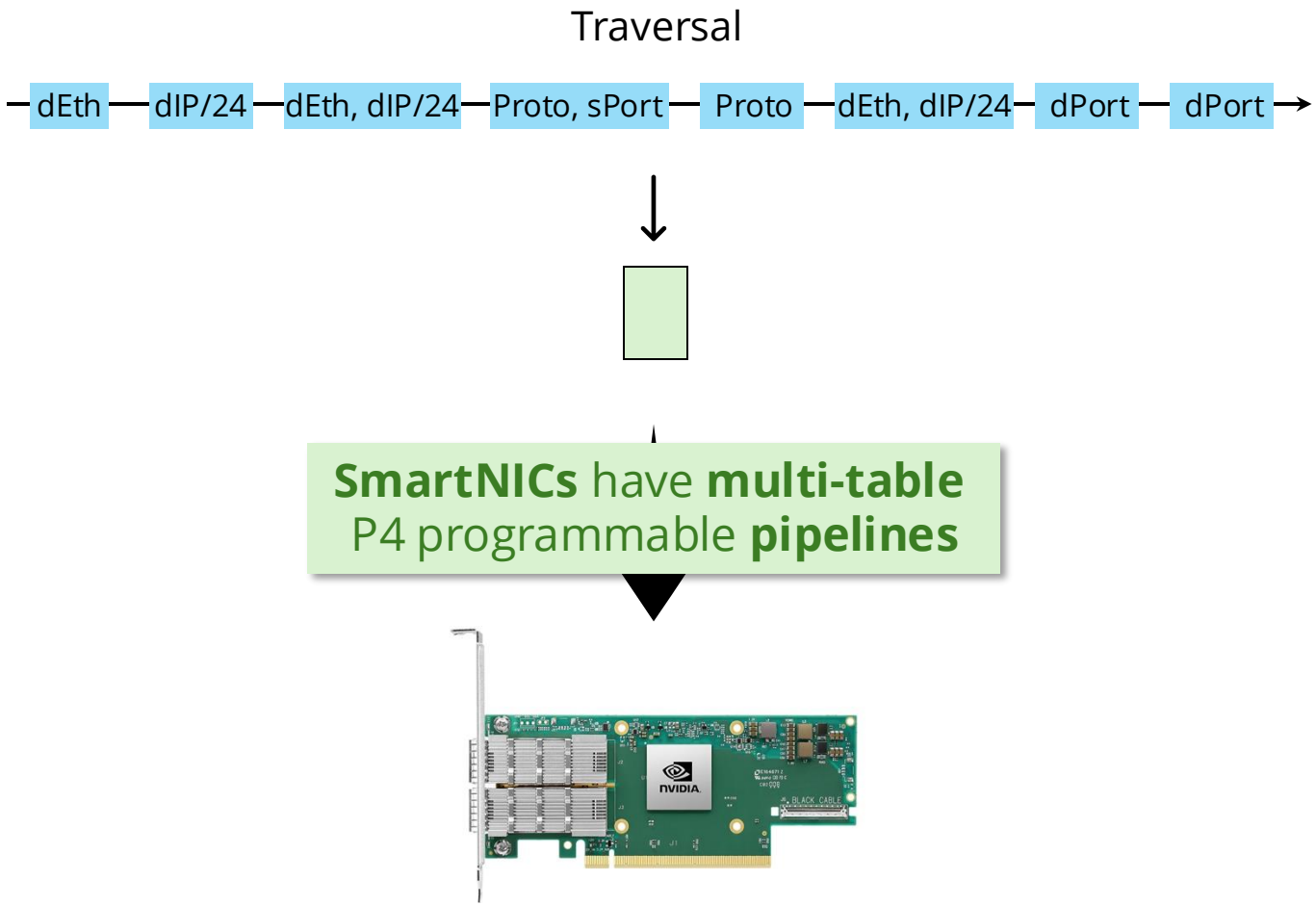


Virtual Switches in the Data Center

Multi-table lookups are **expensive**



The Target Architecture has Changed

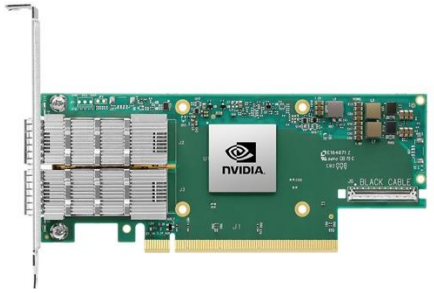


The Target Architecture has Changed

Traversal

dEth — dIP/24 — dEth, dIP/24 — Proto, sPort — Proto — dEth, dIP/24 — dPort — dPort →

It’s time for a **clean slate** virtual switch design

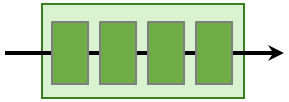


Offload the vSwitch to SmartNIC Tables?

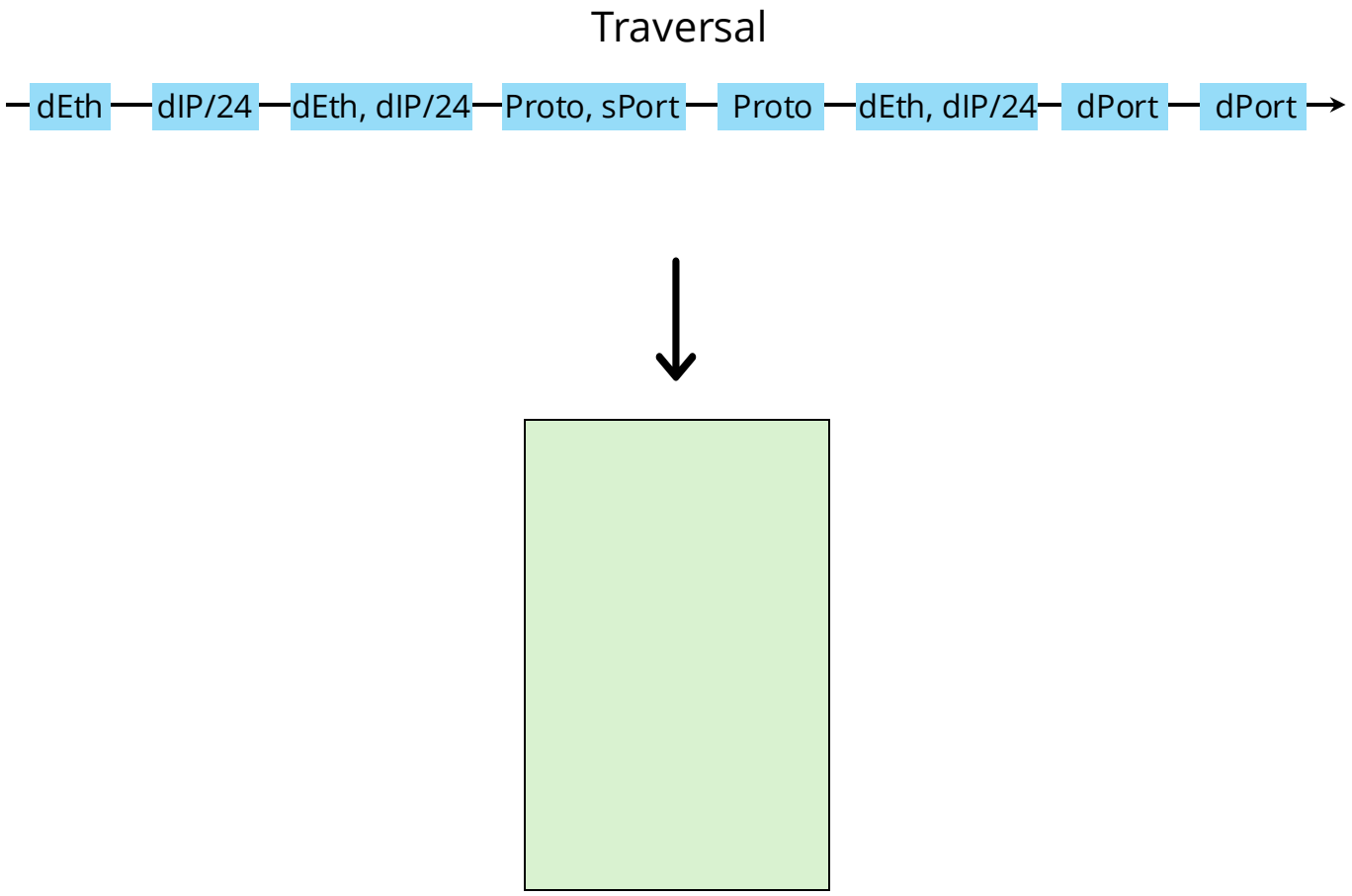
vSwitches provide an **infinite resource abstraction** to the controller

We **can't limit** the **vSwitch** packet pipeline's **flexibility**!!

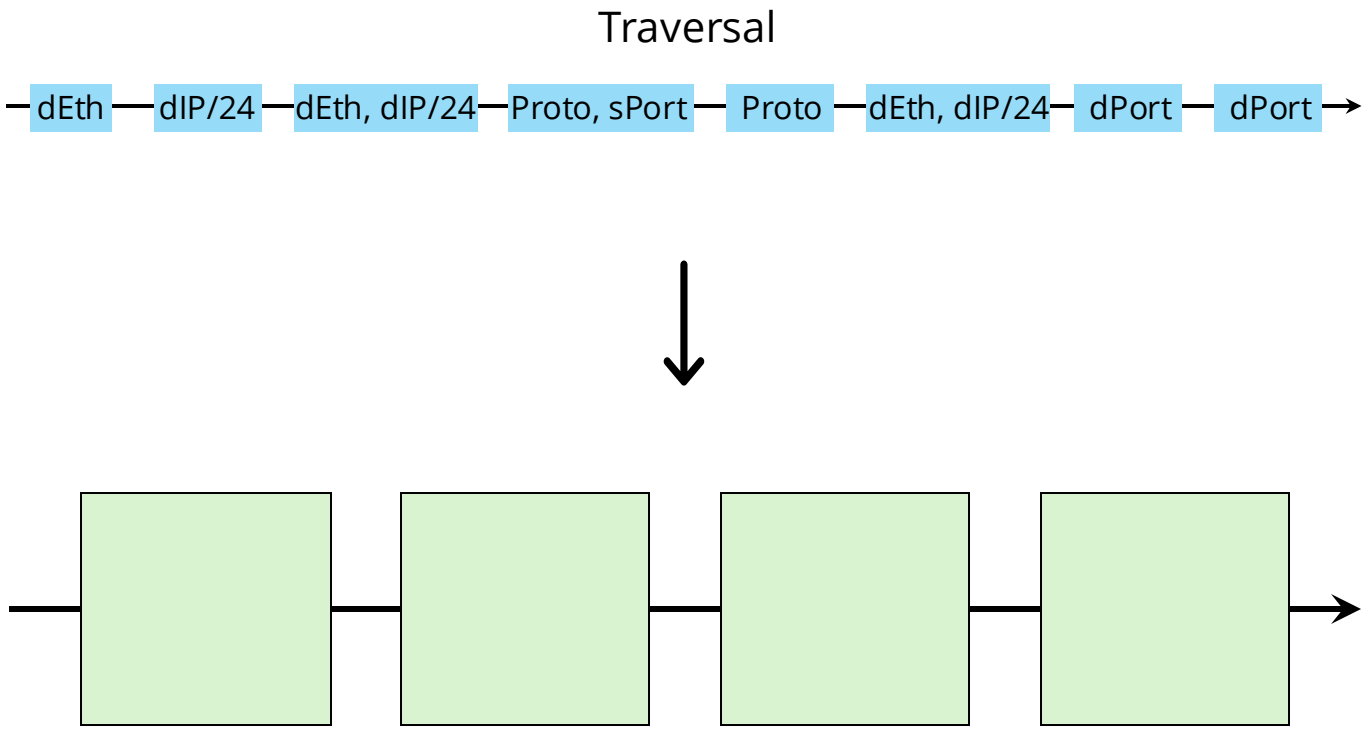
SmartNIC pipelines have a **handful of tables** and offer **limited control flow**



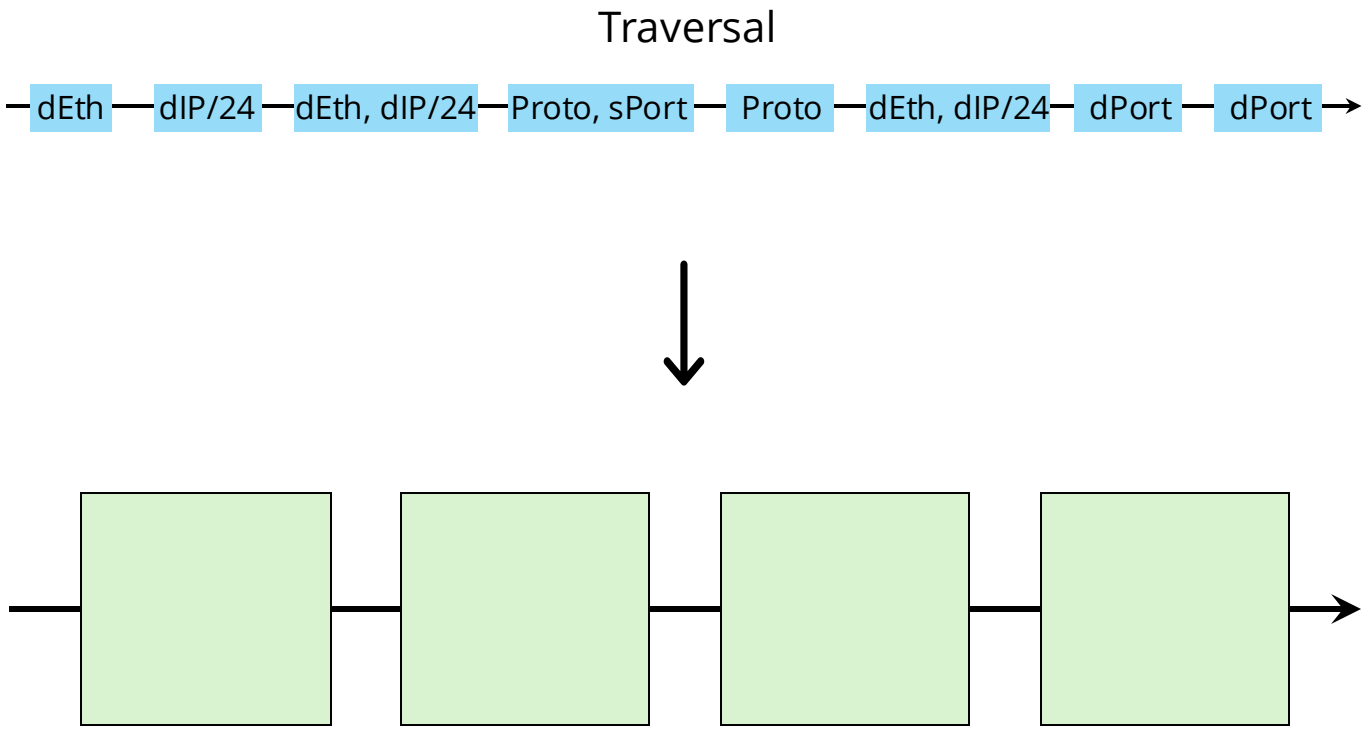
Cache Localities with Multiple Tables!



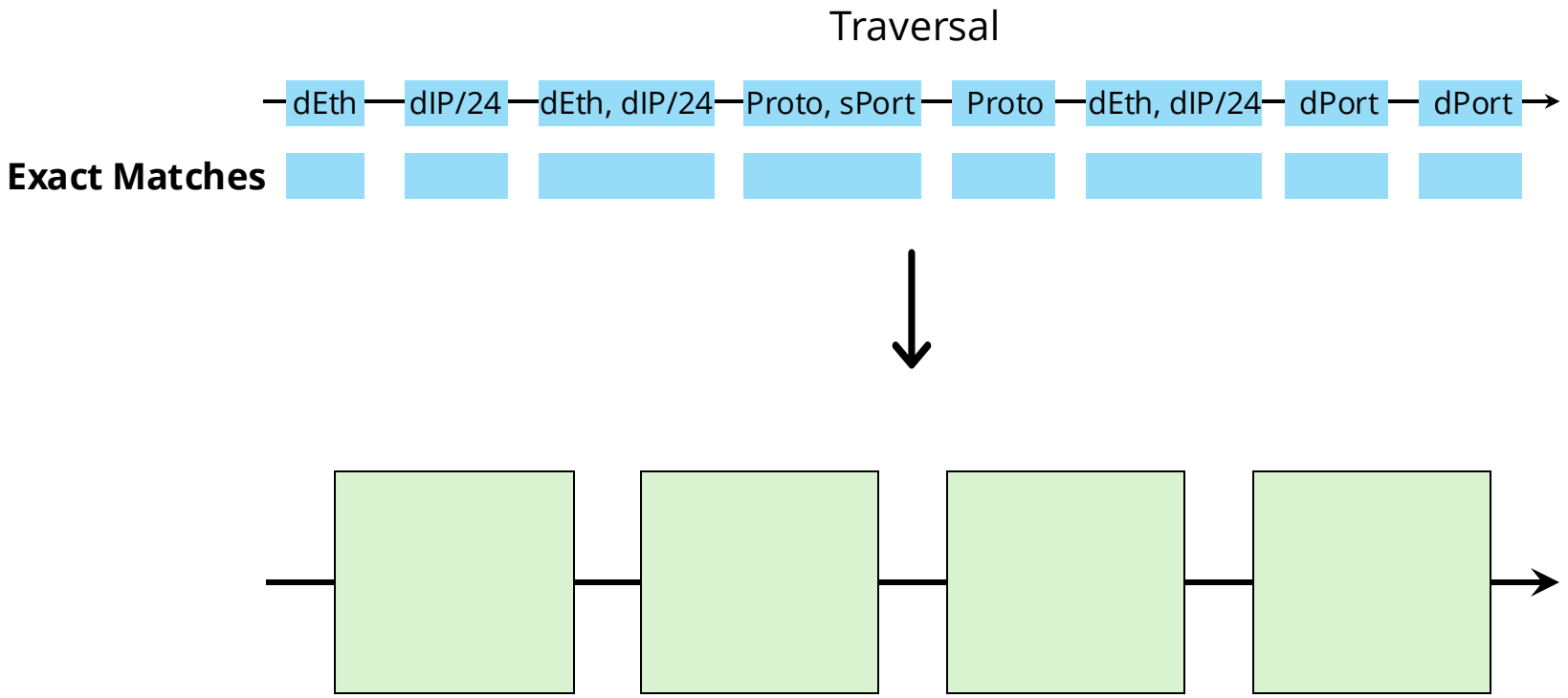
Cache Localities with Multiple Tables!



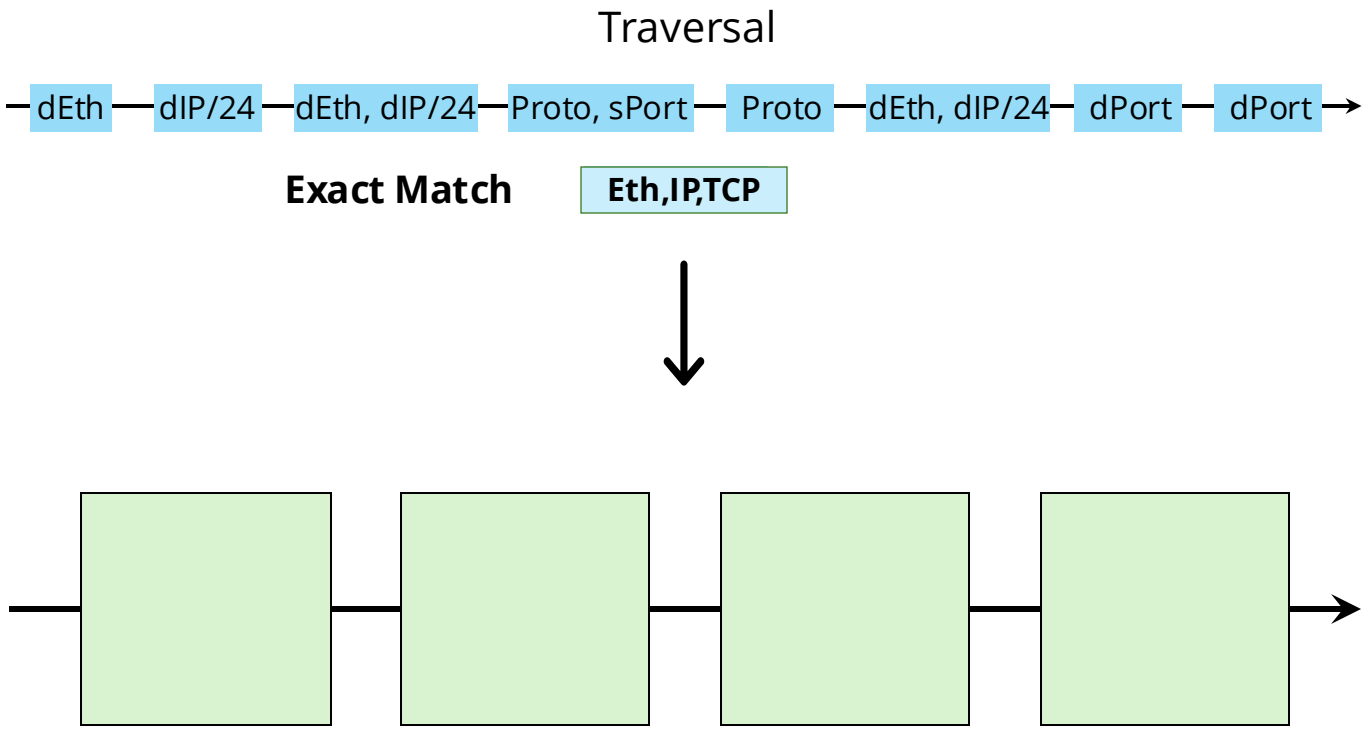
Caching *Temporal Locality* in the SmartNIC



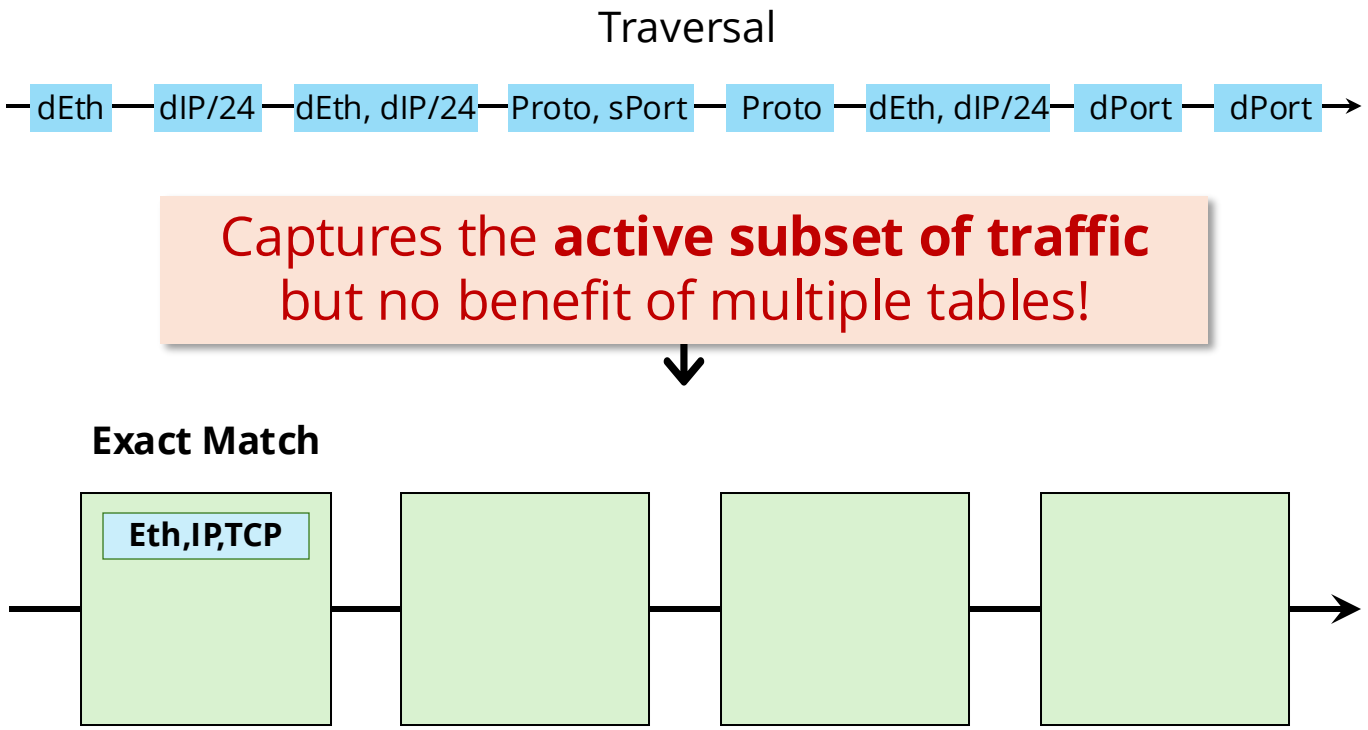
Caching *Temporal Locality* in the SmartNIC



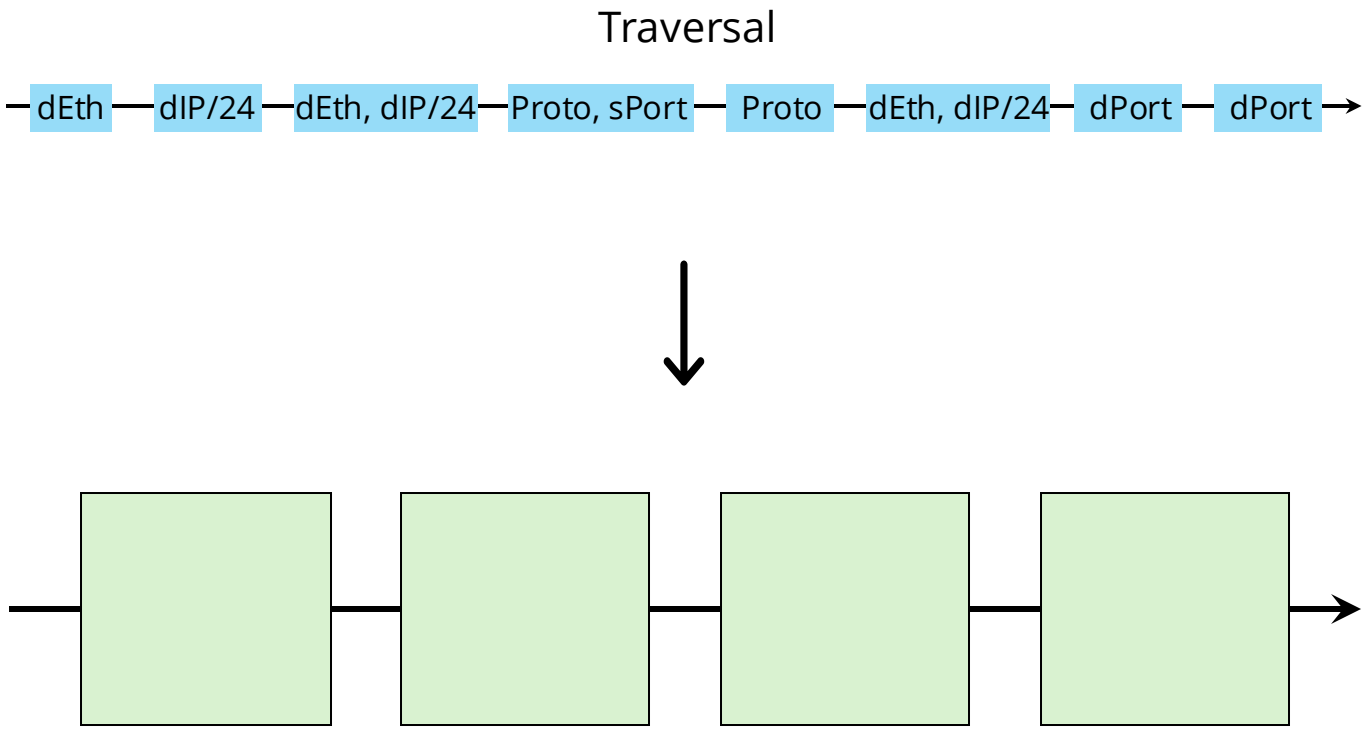
Caching *Temporal Locality* in the SmartNIC



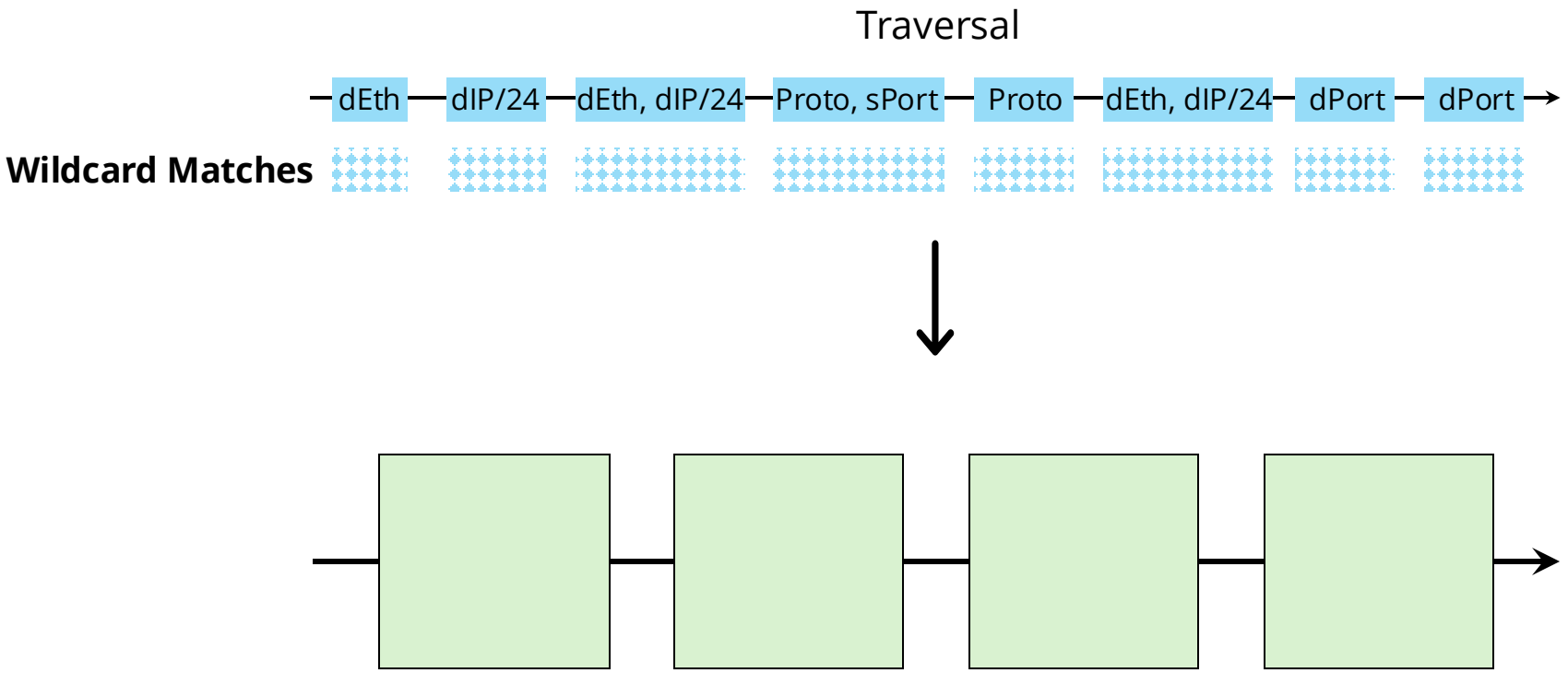
Caching *Temporal Locality* in the SmartNIC



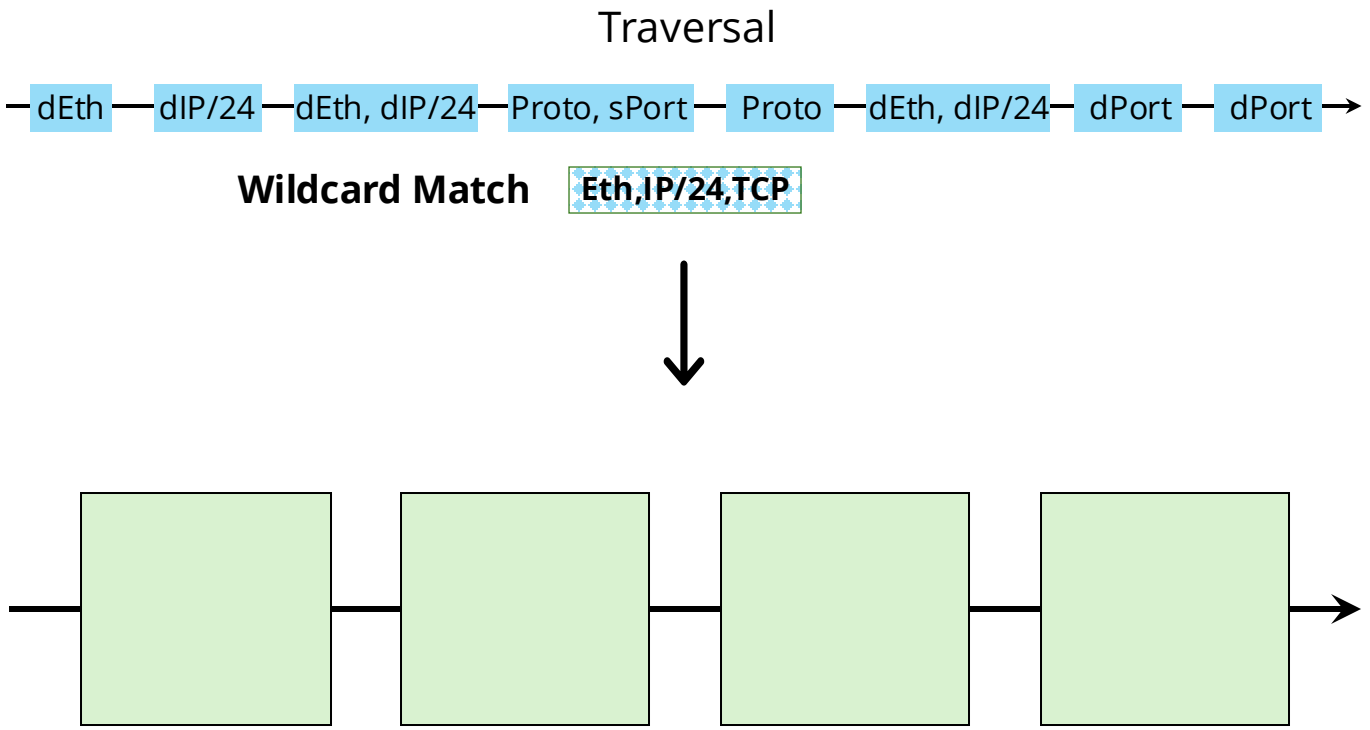
Caching *Spatial Locality* in the SmartNIC



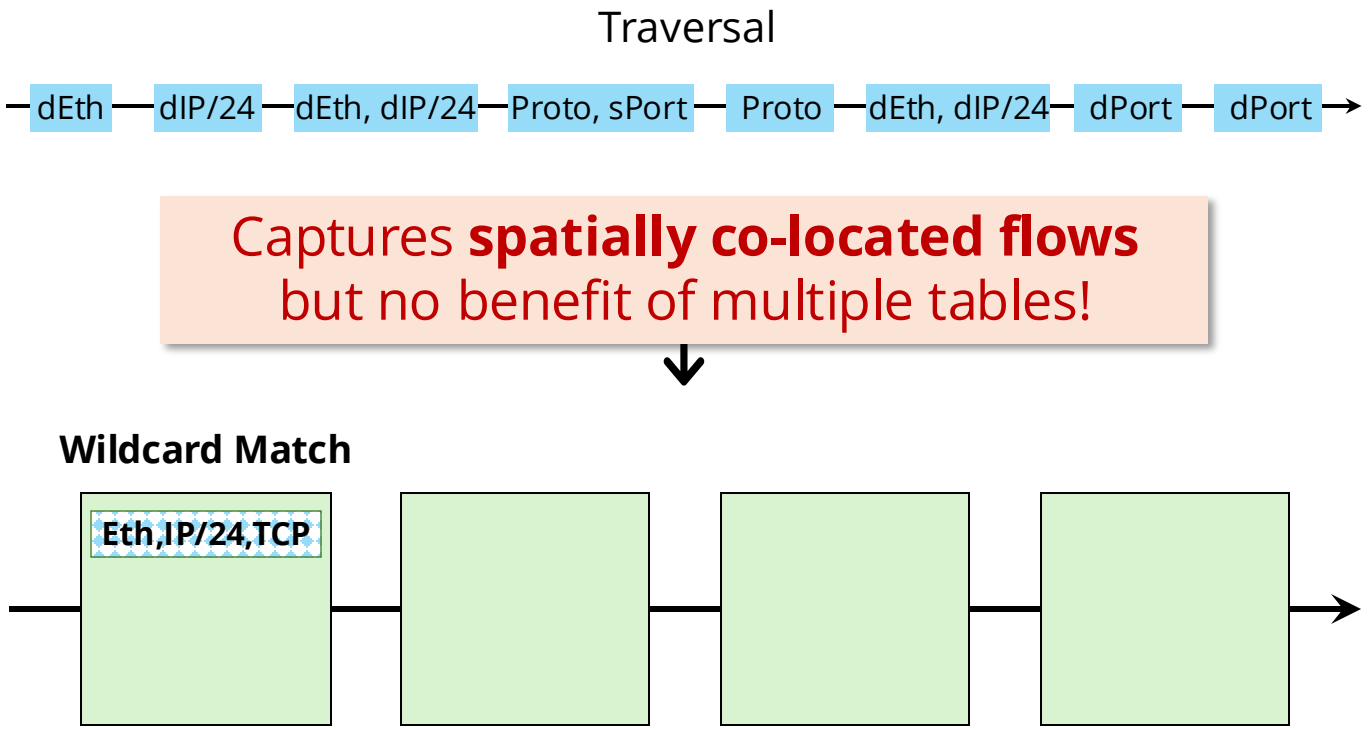
Caching *Spatial Locality* in the SmartNIC



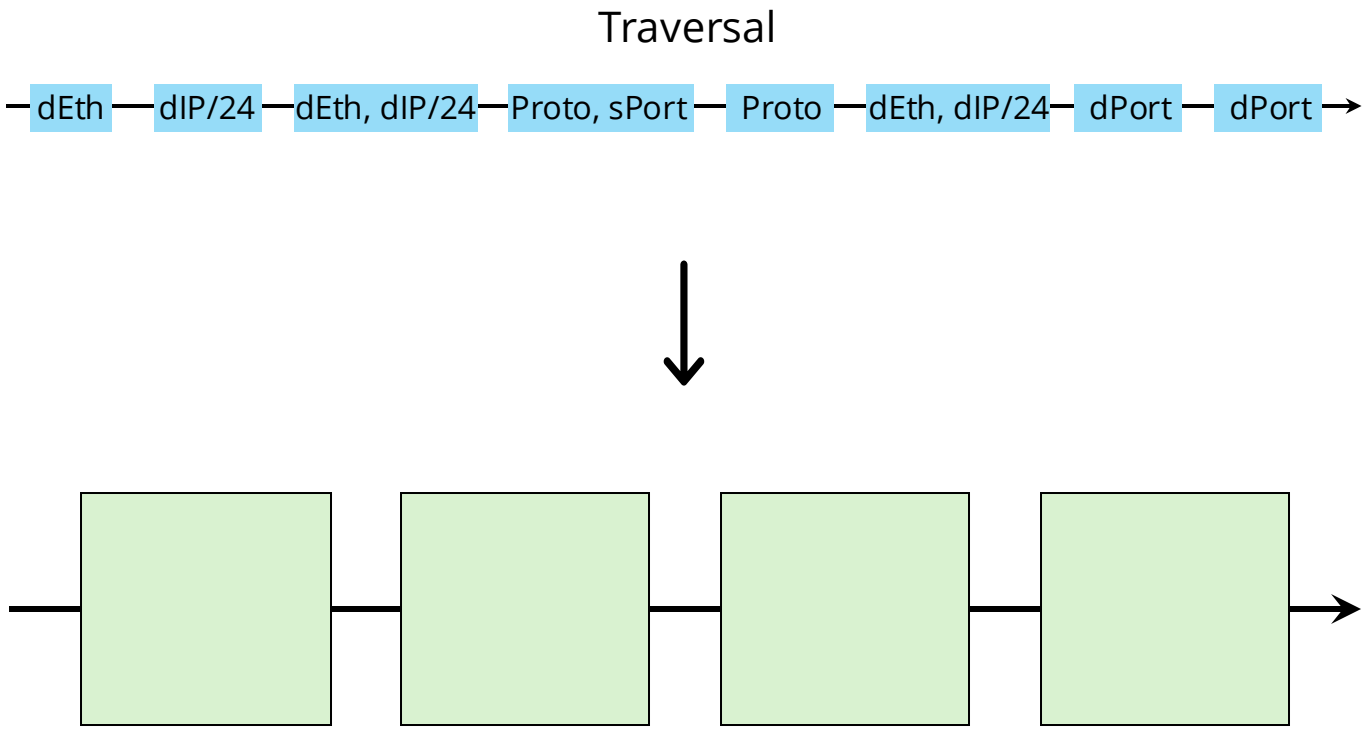
Caching *Spatial Locality* in the SmartNIC



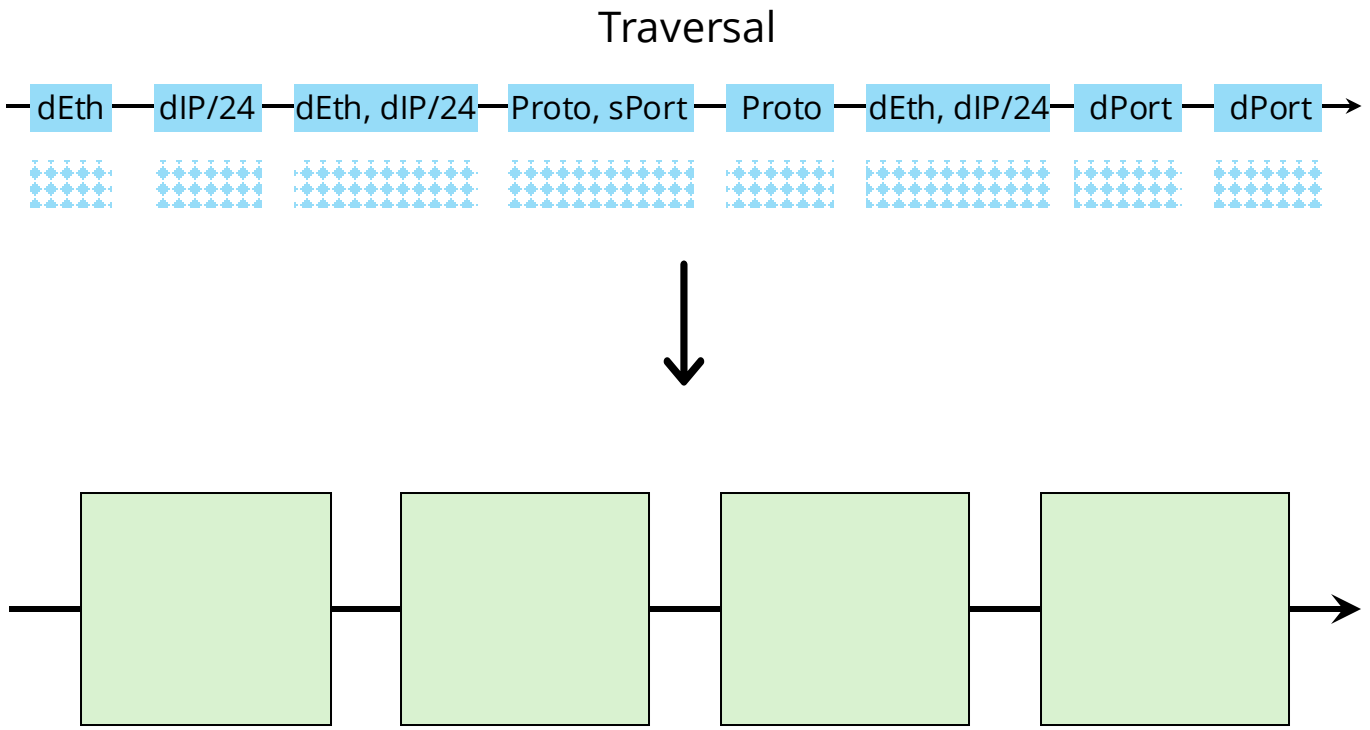
Caching *Spatial Locality* in the SmartNIC



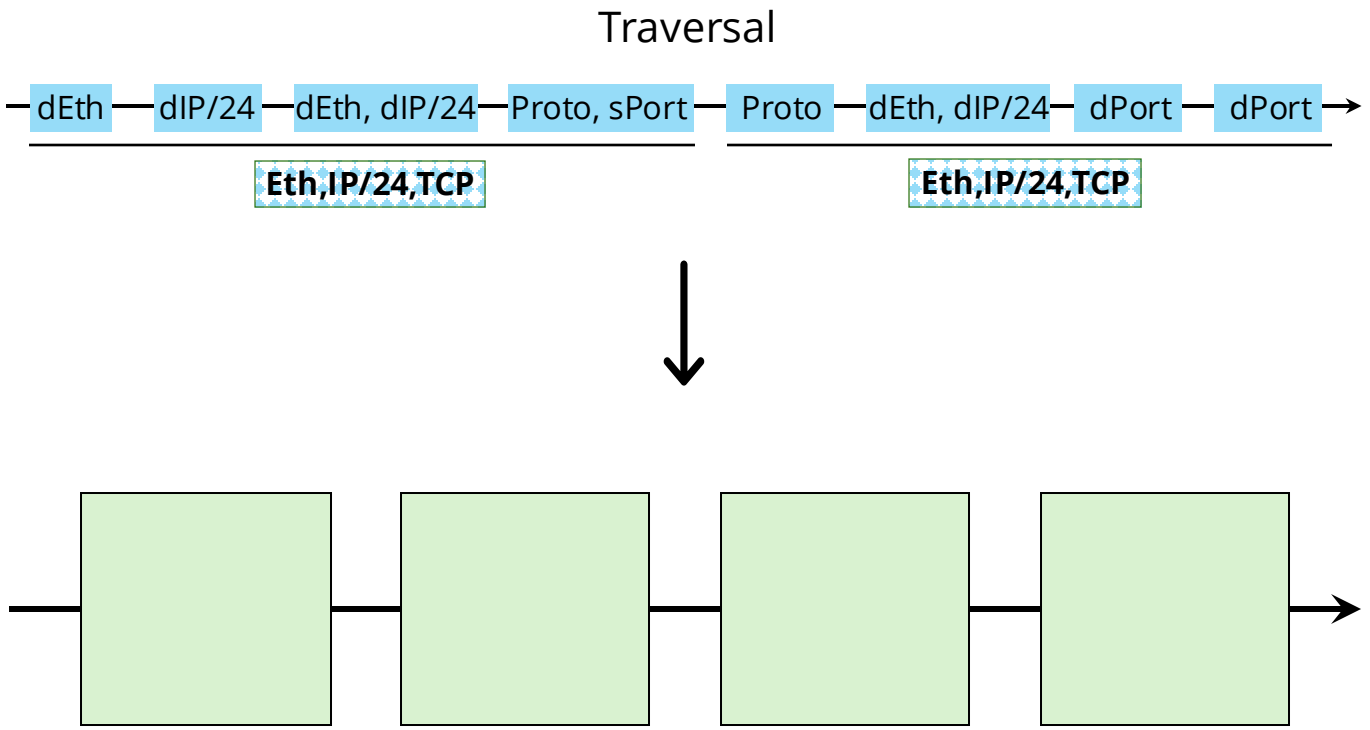
How about *Partitioning* the Traversal?



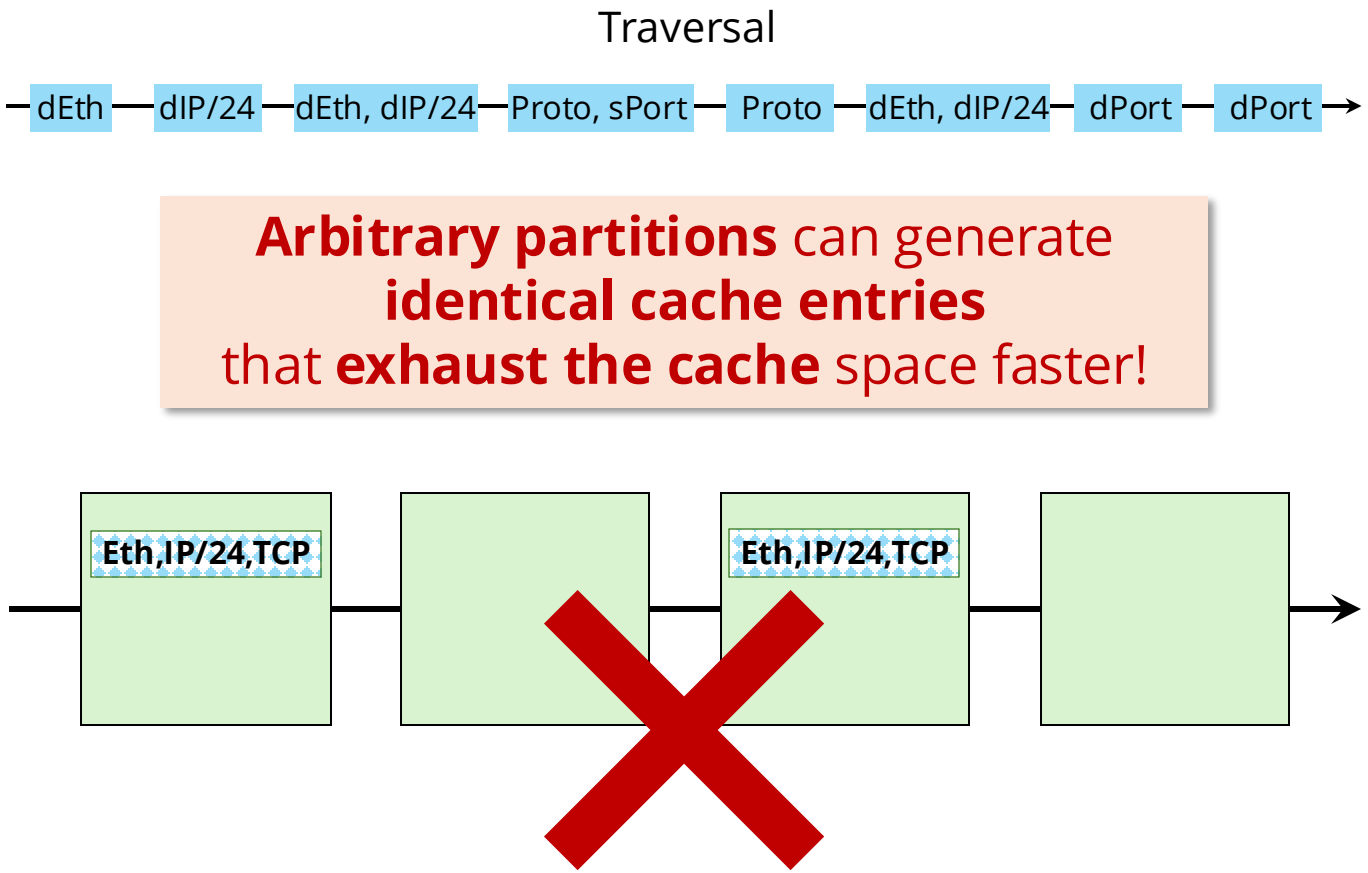
How about *Partitioning* the Traversal?



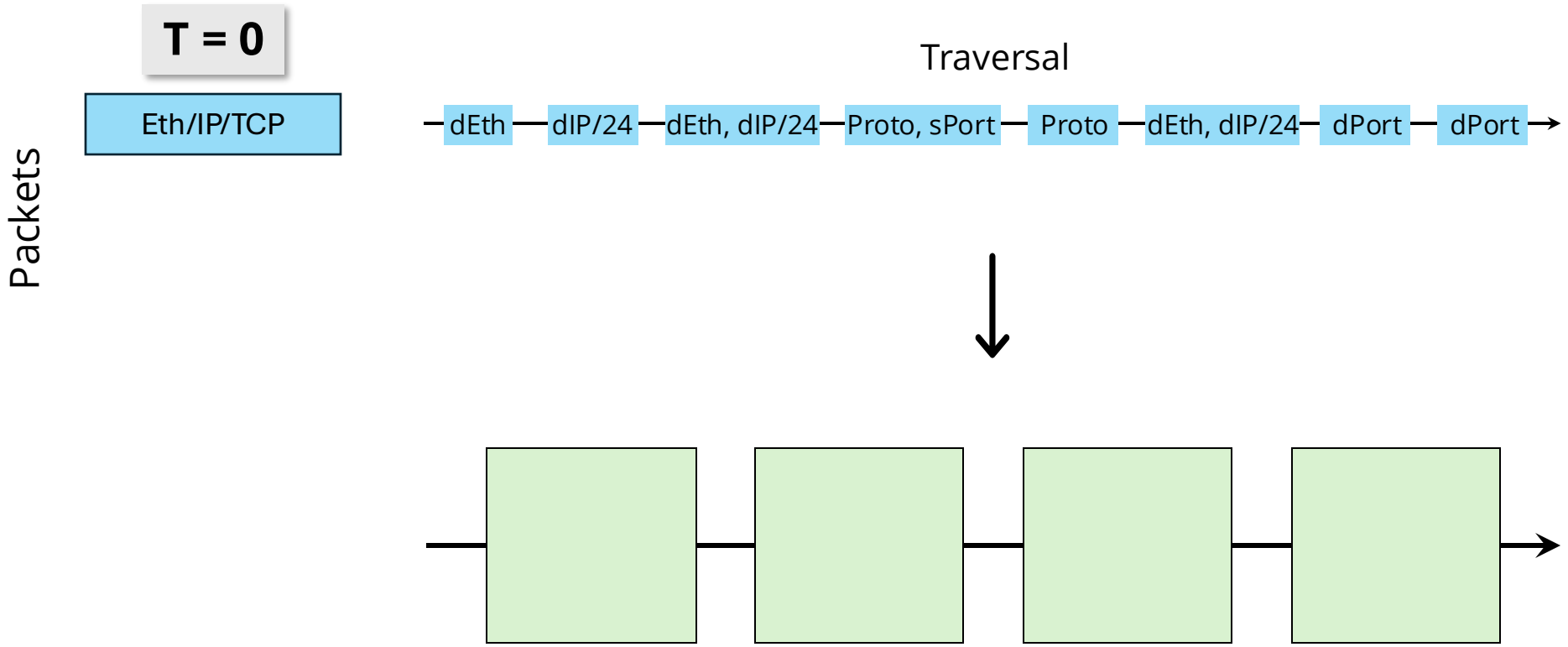
How about *Partitioning* the Traversal?



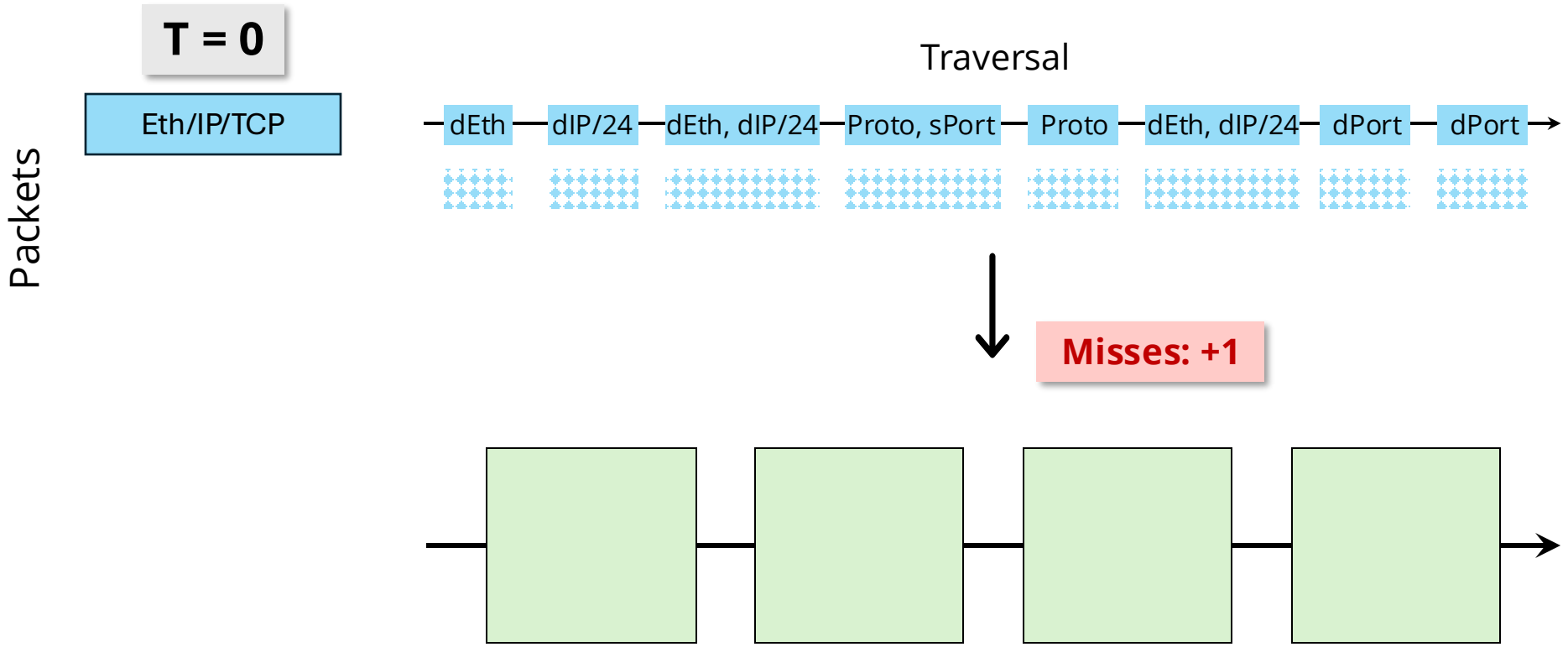
How about *Partitioning* the Traversal?



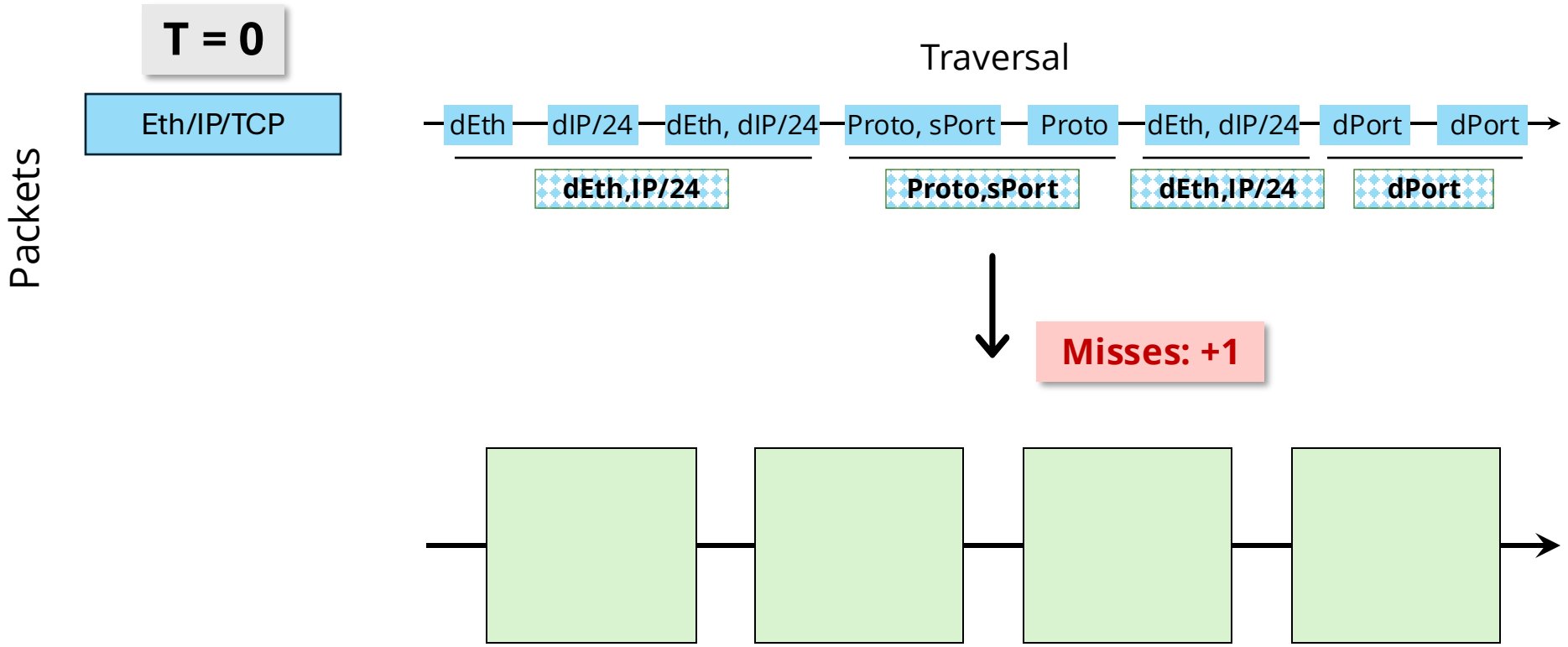
How about Smartly *Partitioning* the Traversal?



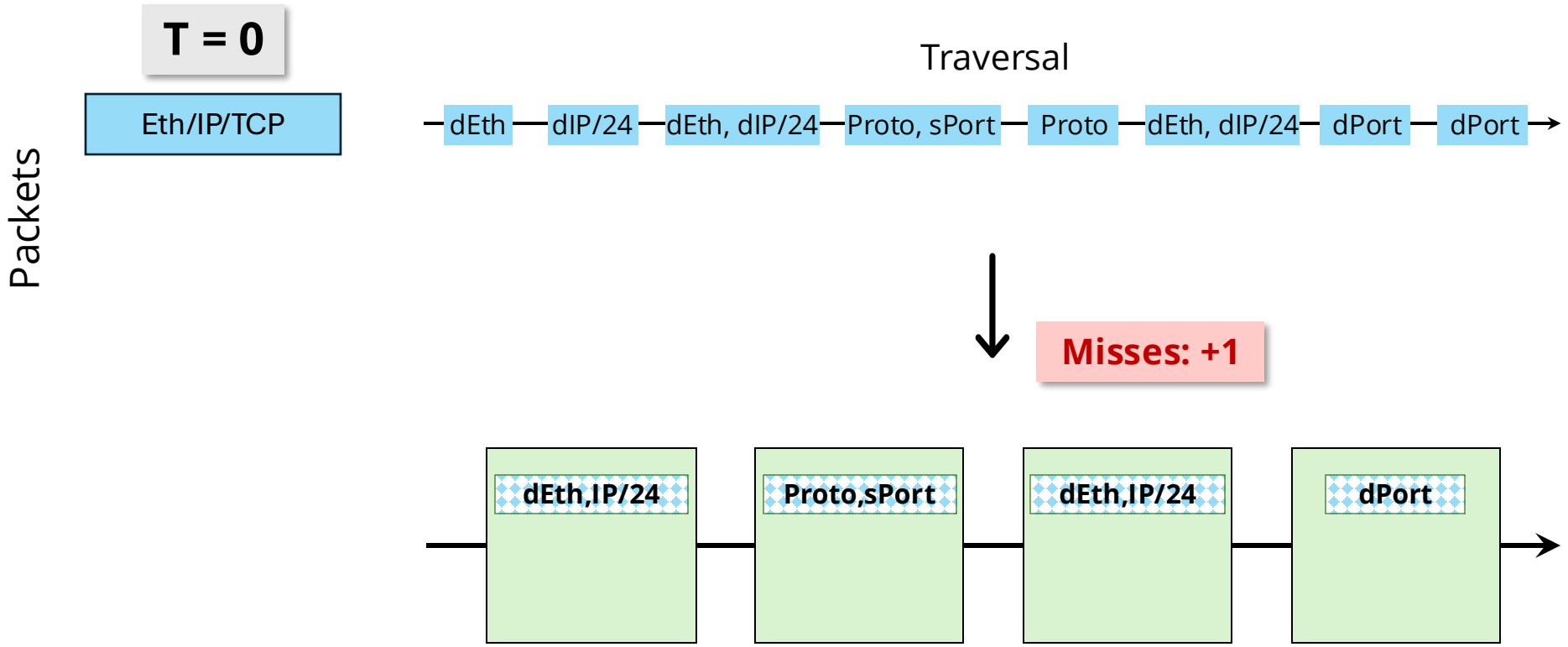
How about Smartly *Partitioning* the Traversal?



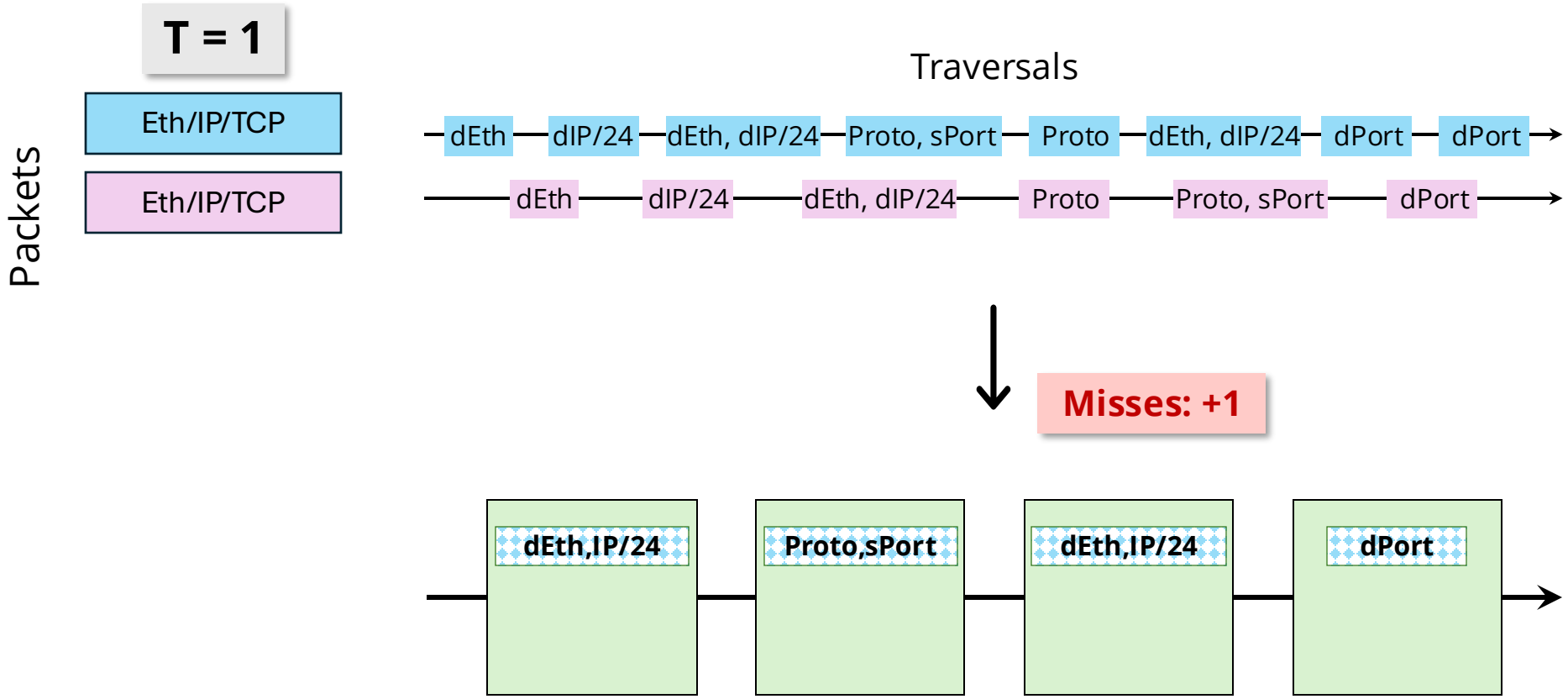
How about Smartly *Partitioning* the Traversal?



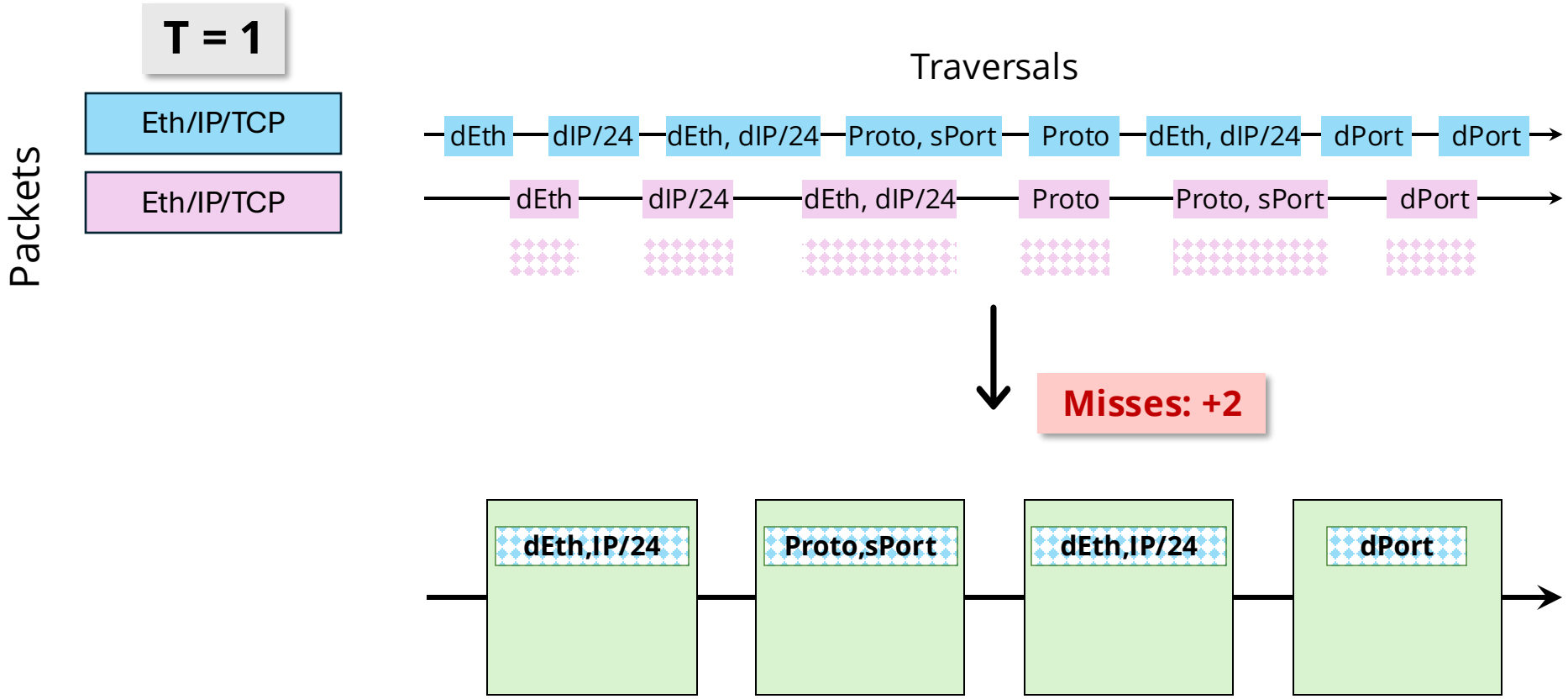
How about Smartly *Partitioning* the Traversal?



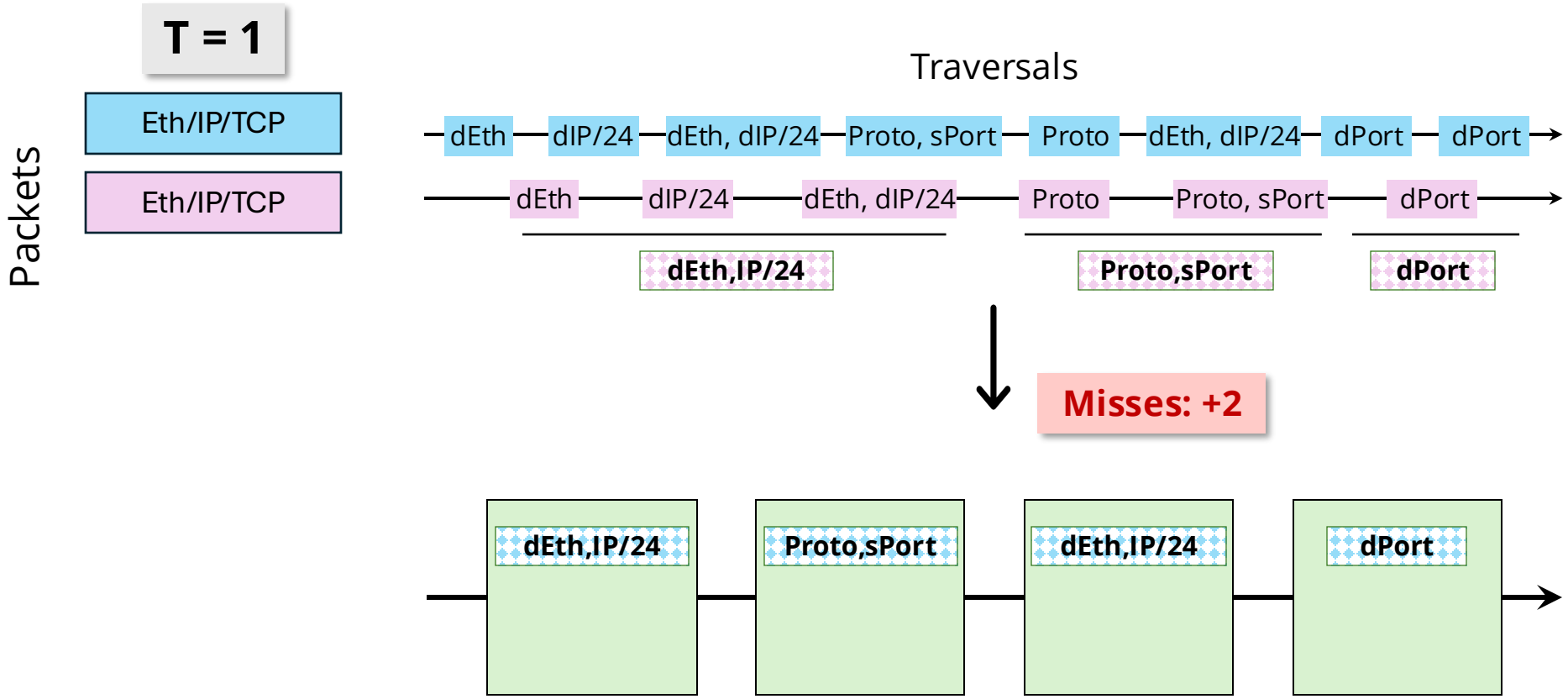
How about Smartly *Partitioning* the Traversal?



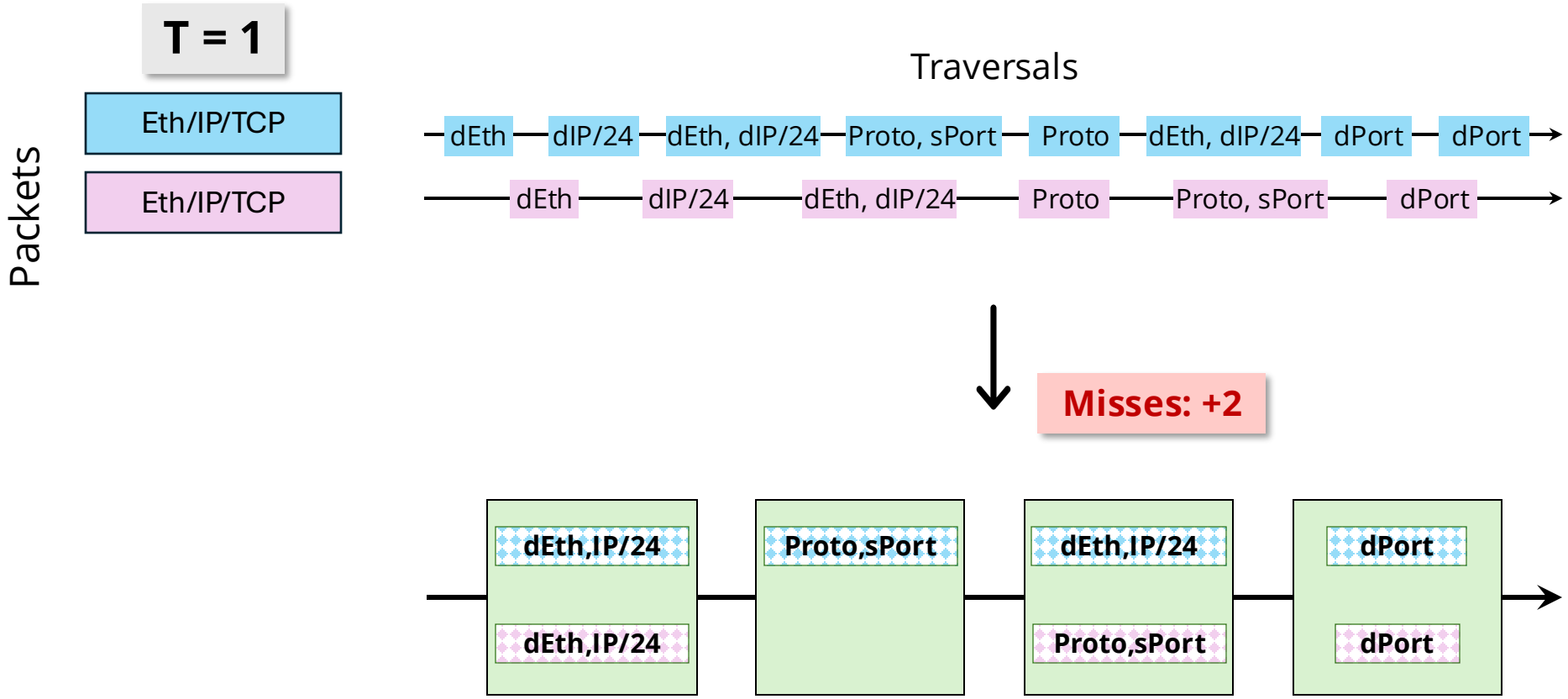
How about Smartly *Partitioning* the Traversal?



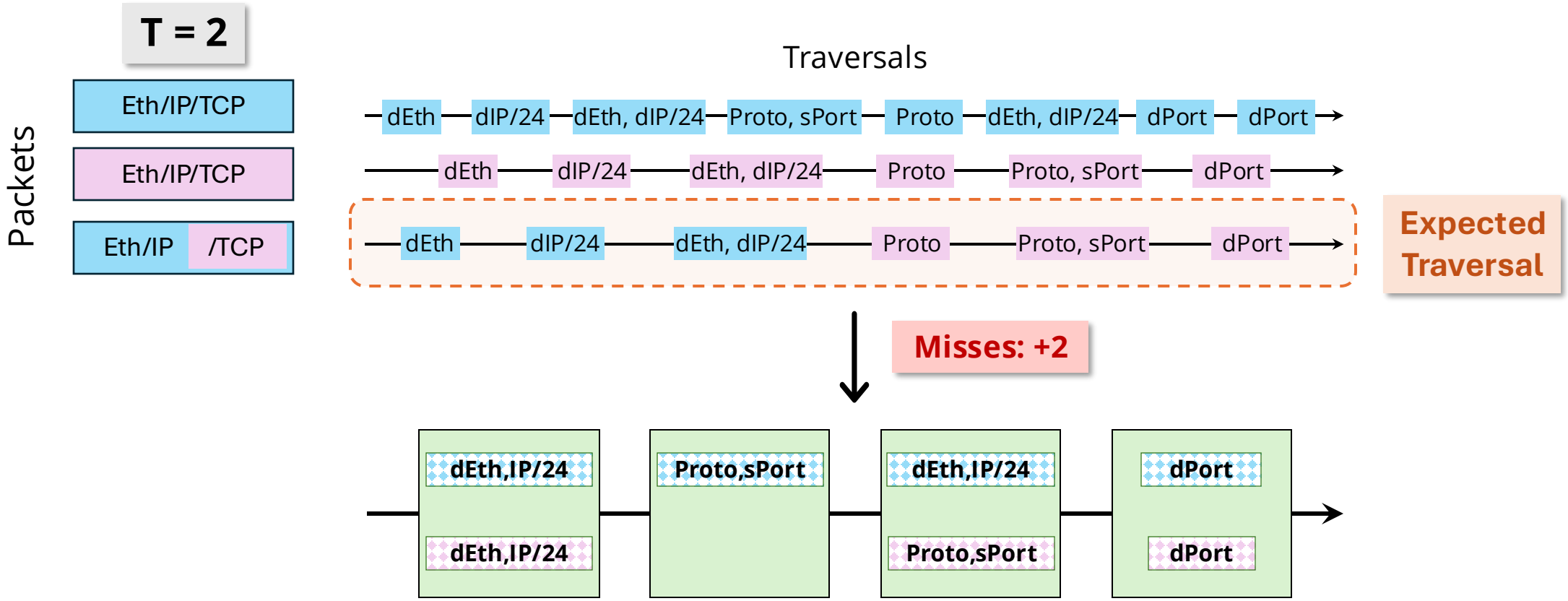
How about Smartly *Partitioning* the Traversal?



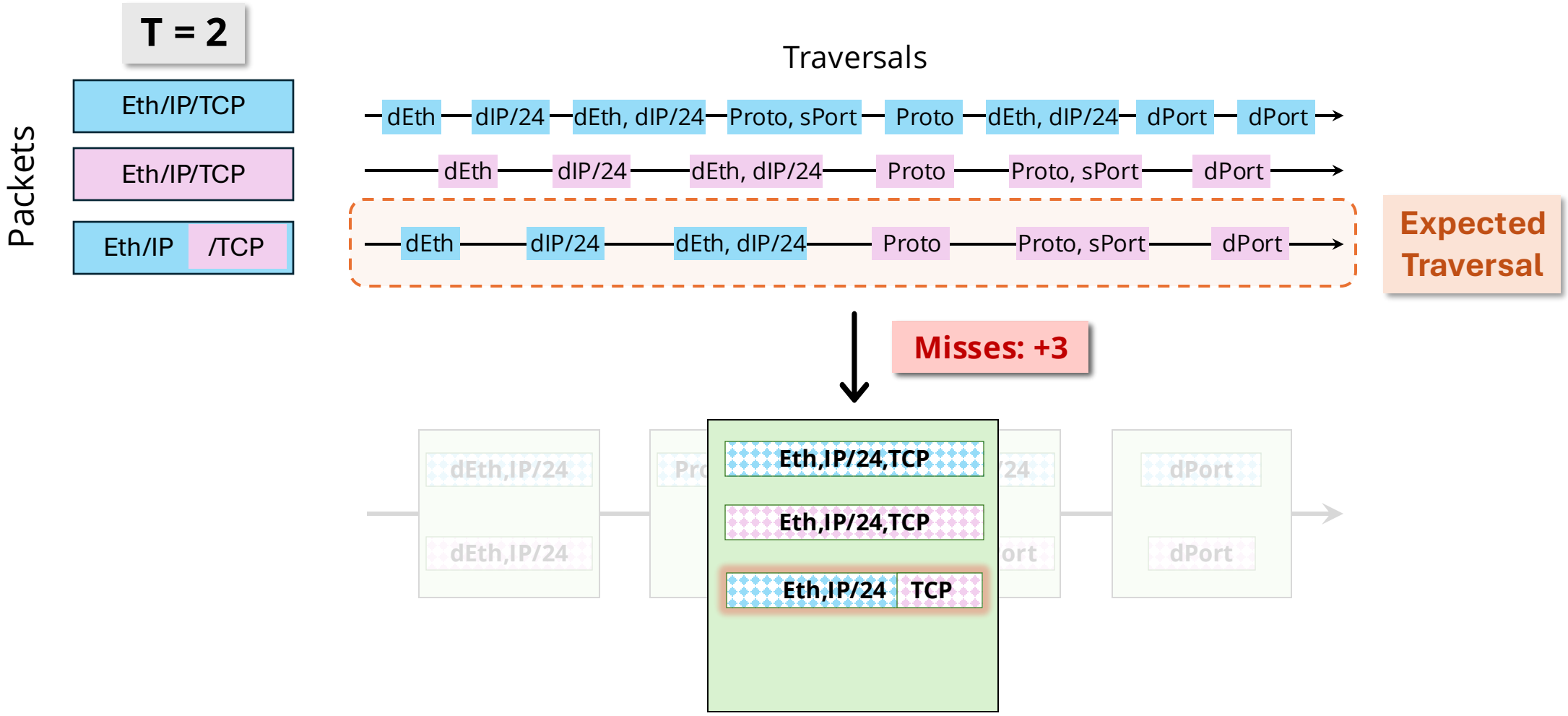
How about Smartly *Partitioning* the Traversal?



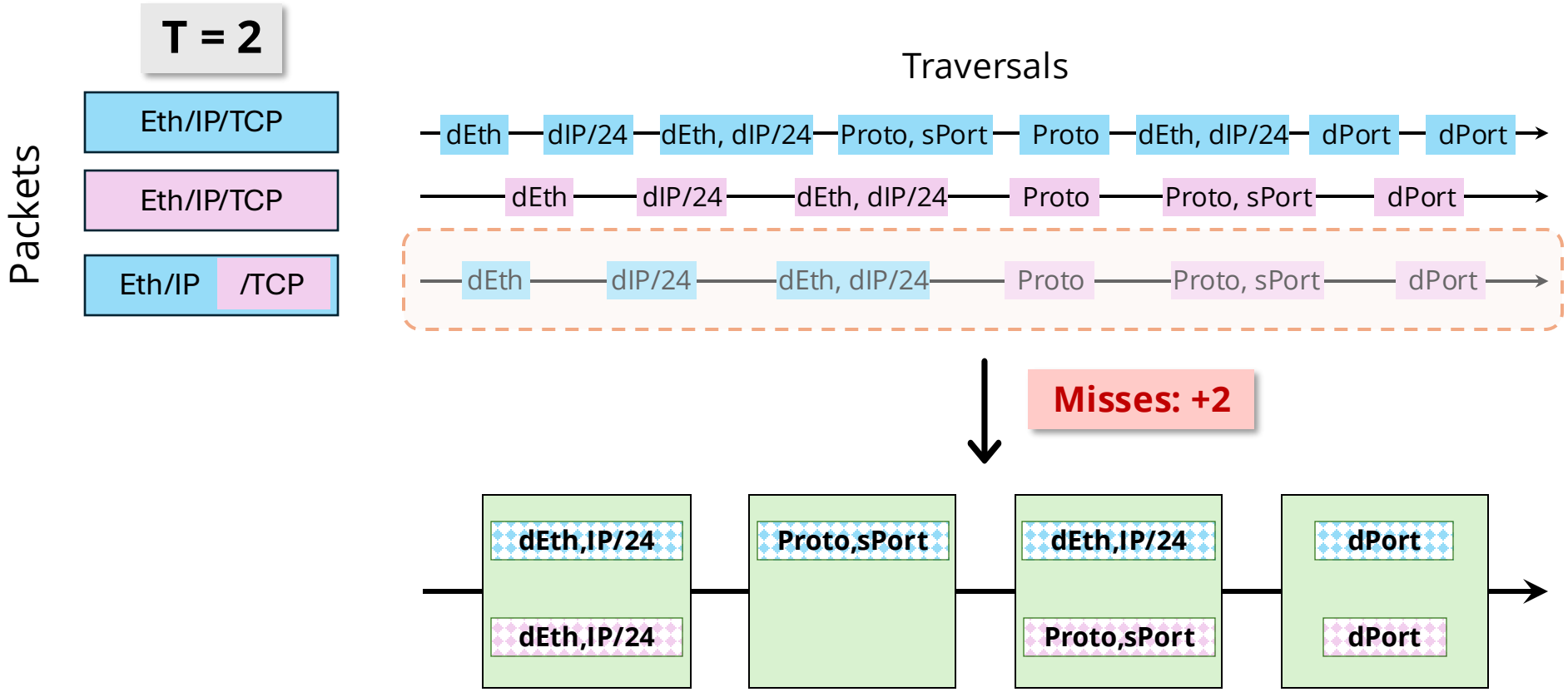
How about Smartly *Partitioning* the Traversal?



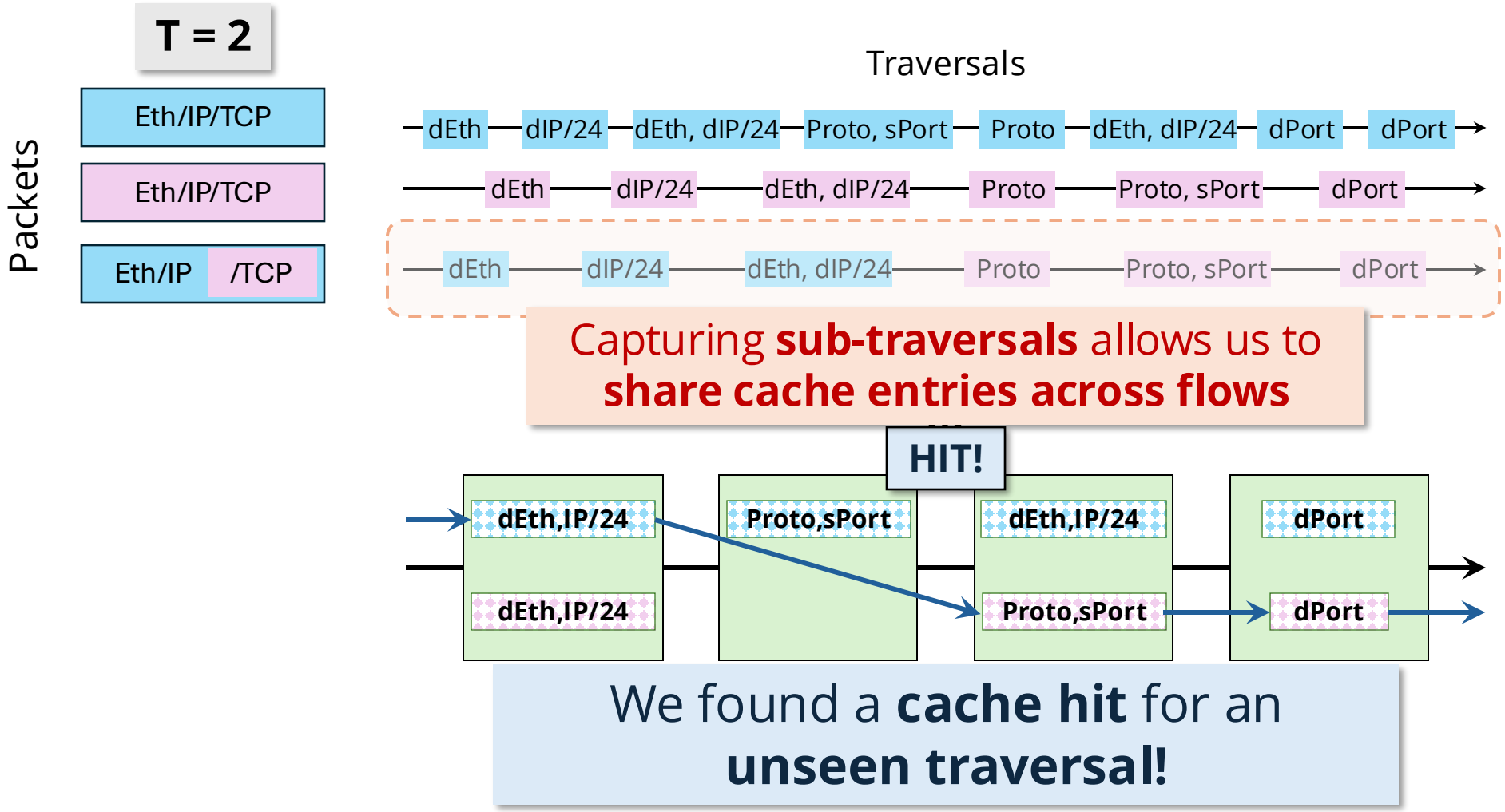
How about Smartly *Partitioning* the Traversal?



How about Smartly *Partitioning* the Traversal?



How about Smartly *Partitioning* the Traversal?

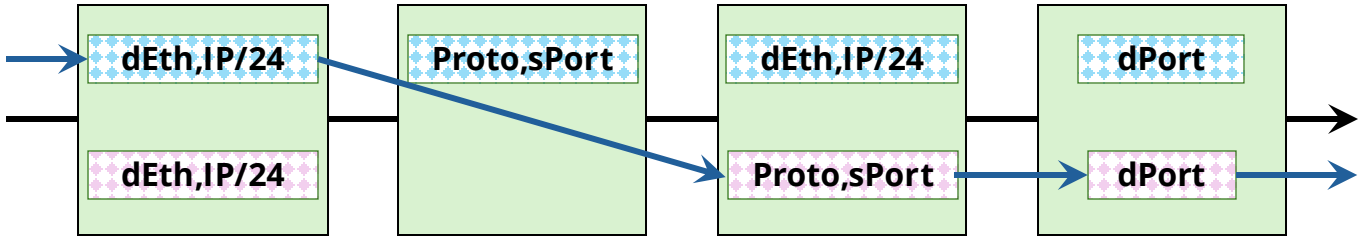


Sub-Traversal Caching with Gigaflow

Sub-traversals shared across flows
as cache entries in the SmartNIC

Optimal sub-traversals as cache entries
capture **pipeline-aware locality**

Up to **two orders of magnitude improvement** in **rule space coverage!**



Sub-Traversal Caching with Gigaflow

Optimal sub-traversals as cache entries capture **pipeline-aware locality**

Up to **two orders of magnitude improvement** in **rule space coverage!**

SmartNIC Cache	Real-World vSwitch Pipelines (Ps)				
	P1	**P2**	**P3**	**P4**	**P5**
MF	32K	32K	32K	32K	32K
GF	??				

Rule Space Coverage of Megaflow 32K (MF)
vs Gigaflow 4x8K (GF) Cache

Capturing Pipeline-Aware Locality

Algorithm 1 Strawman Approach

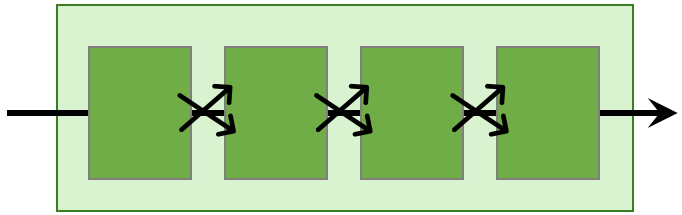
Input: Set of traversals $T = \{t_1, t_2, \ldots, t_N\}$
Output: Sub-traversals with maximum rule space coverage

This cost is **proportional** to **active flows!**
Keeps **increasing** with each **cache miss!**
Infeasible at line rate!

How to capture **pipeline-aware** locality ***efficiently*?**

Capturing Pipeline-Aware Locality *Efficiently*

To goal is to **maximize** the **rule space coverage** in the SmartNIC tables



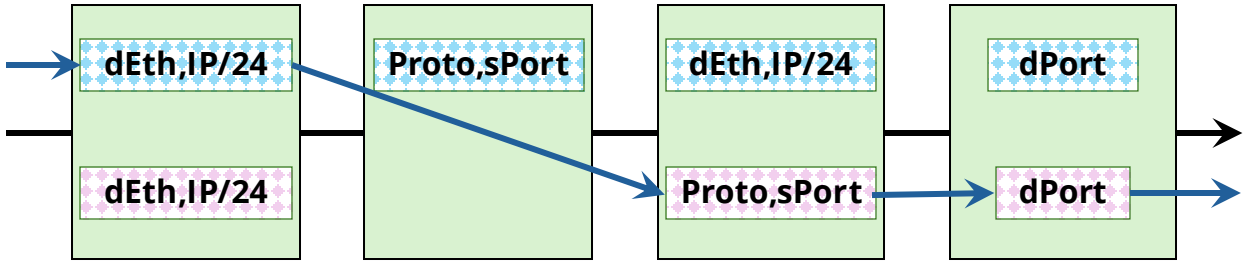
This **cross-product** rule space **should be maximized!**

Pipeline-Aware locality → **Disjointedness**

Maximize *the* ***field-level disjointedness*** *among consecutive sub-traversals*

Capturing Pipeline-Aware Locality *Efficiently*

To goal is to **maximize** the **rule space coverage** in the SmartNIC tables

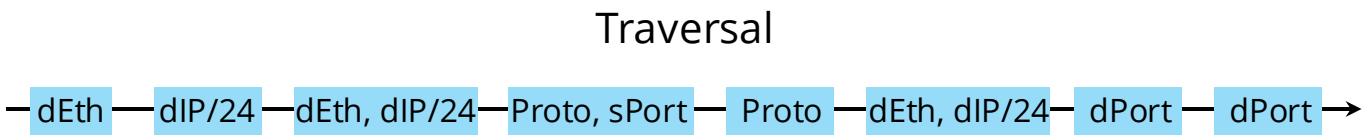


This **cross-product** rule space **should be maximized!**

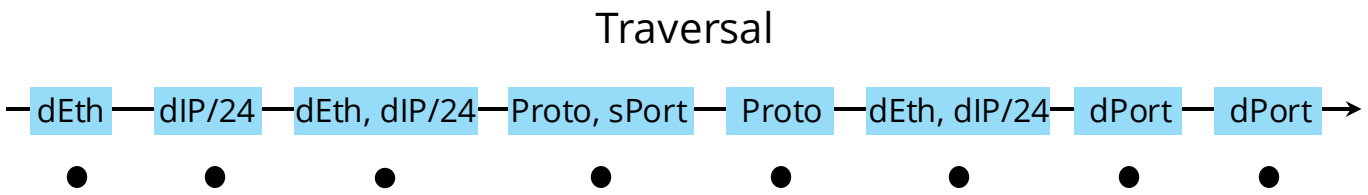
Pipeline-Aware locality → **Disjointedness**

Maximize *the* ***field-level disjointedness*** *among consecutive sub-traversals*

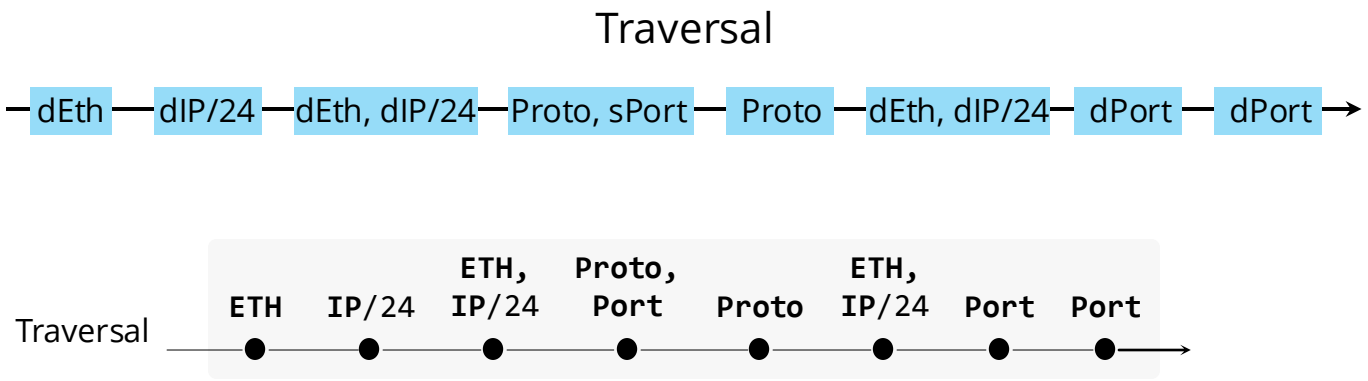
Disjointedness for Pipeline-Aware Locality



Disjointedness for Pipeline-Aware Locality

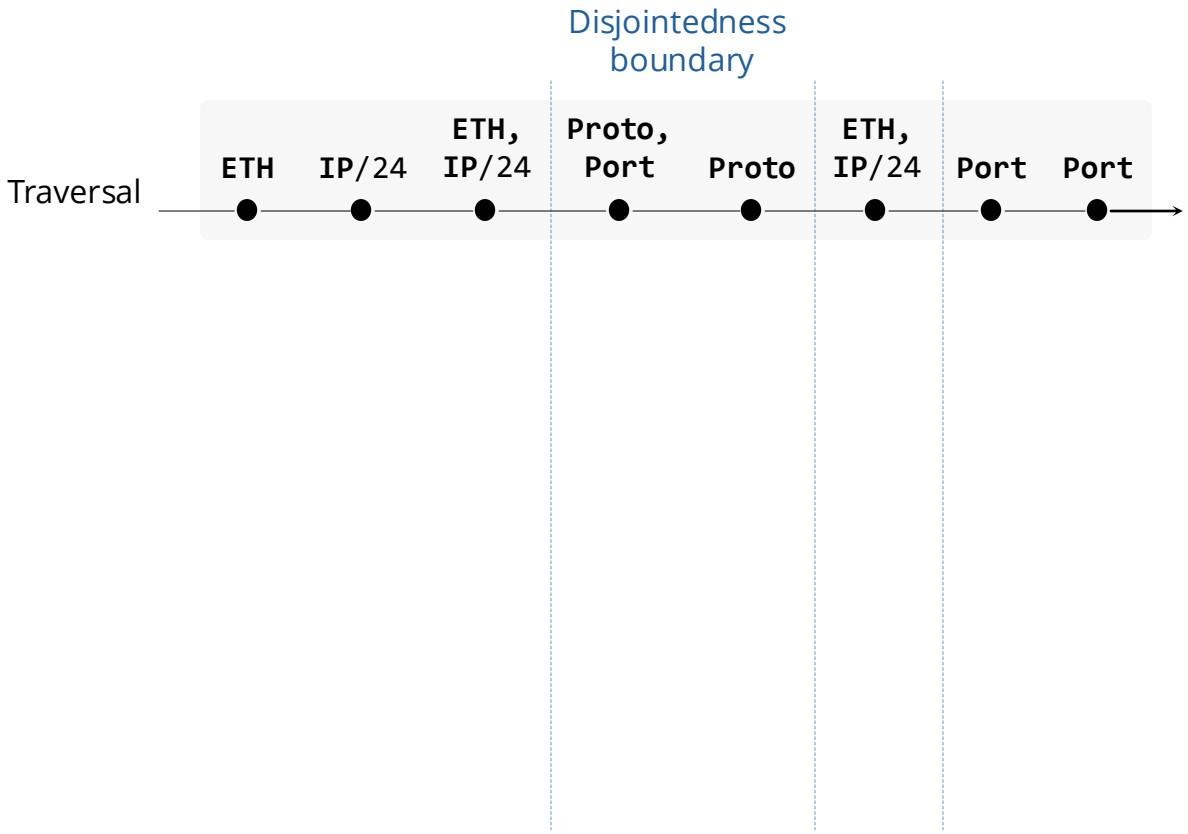


Disjointedness for Pipeline-Aware Locality

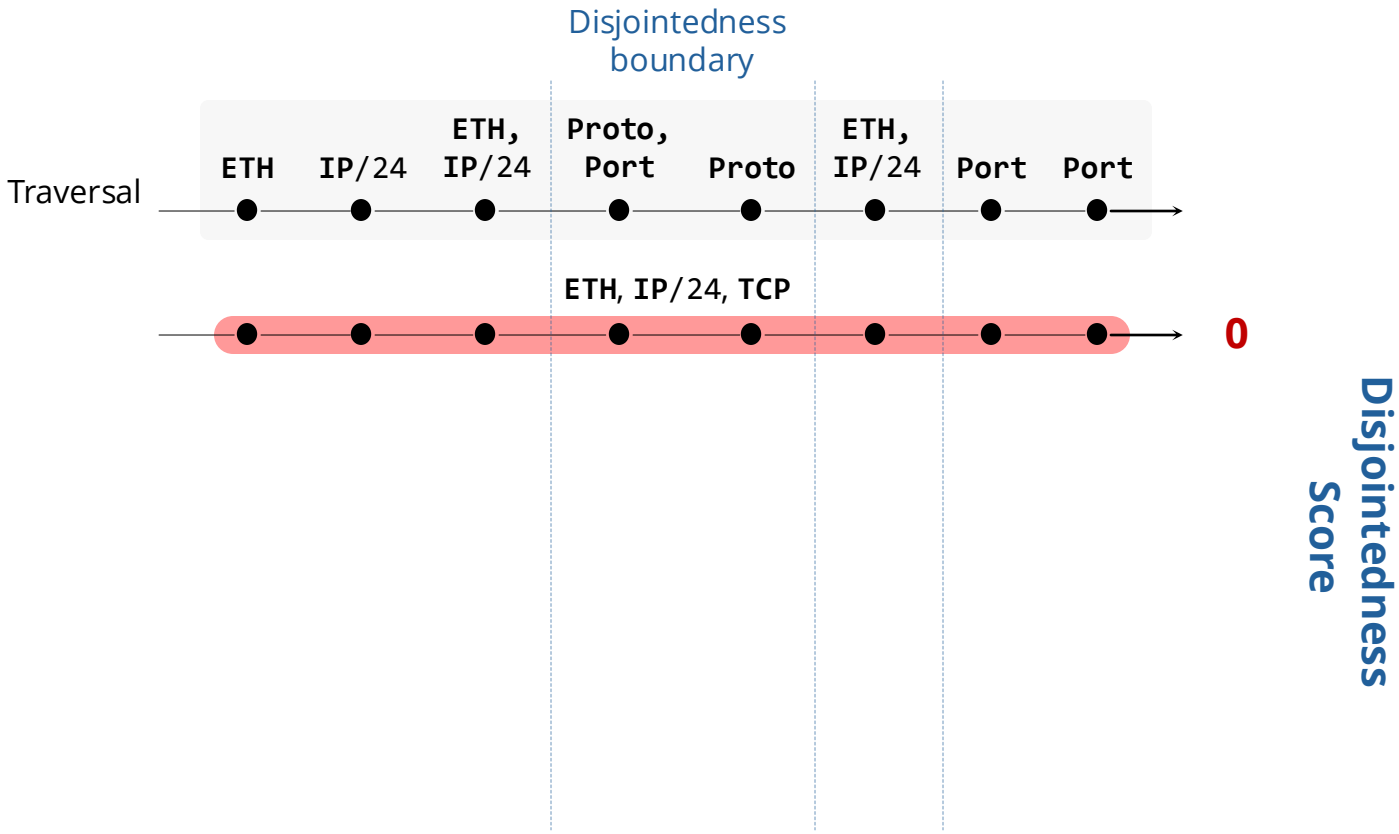


Disjointedness for Pipeline-Aware Locality

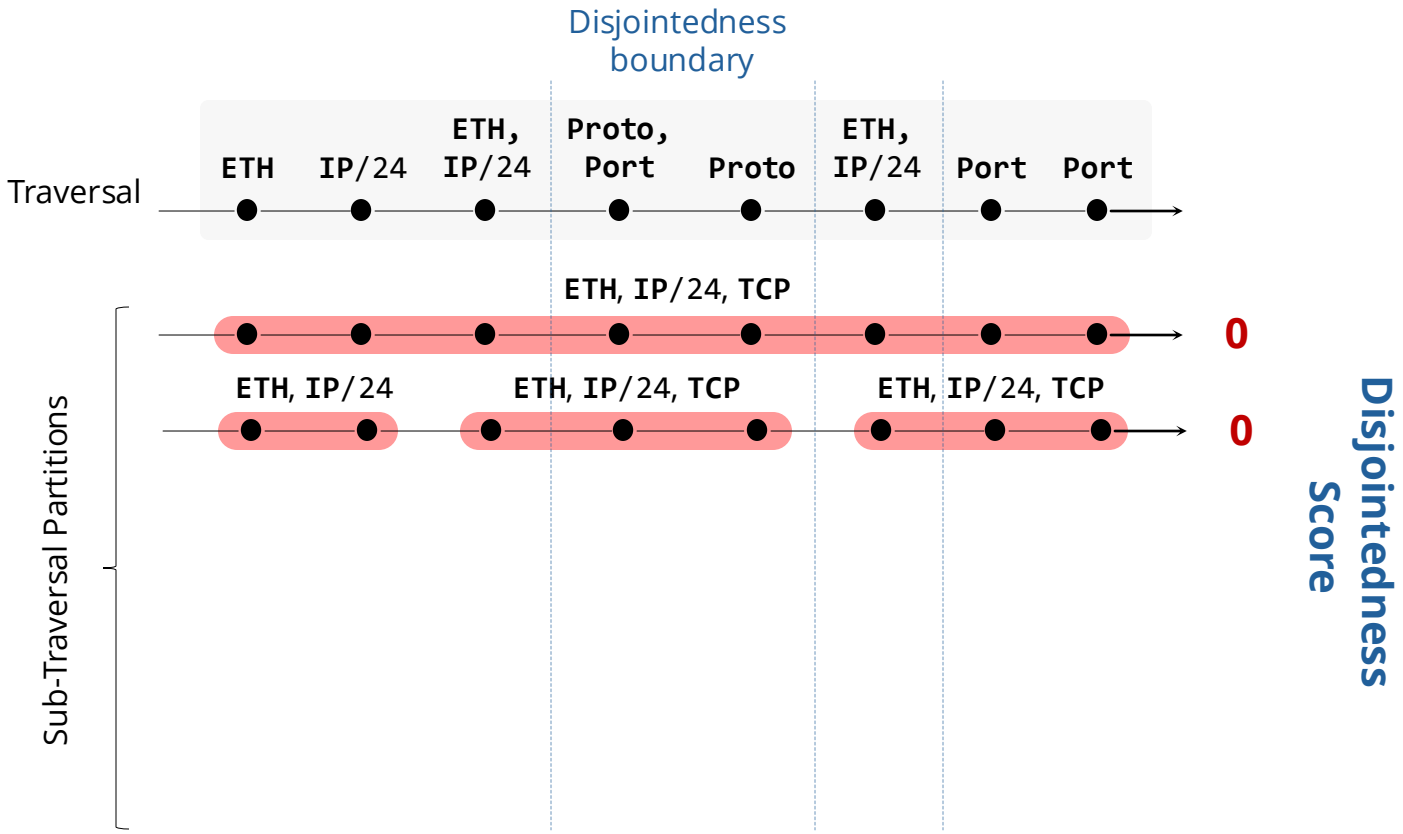
Disjointedness for Pipeline-Aware Locality



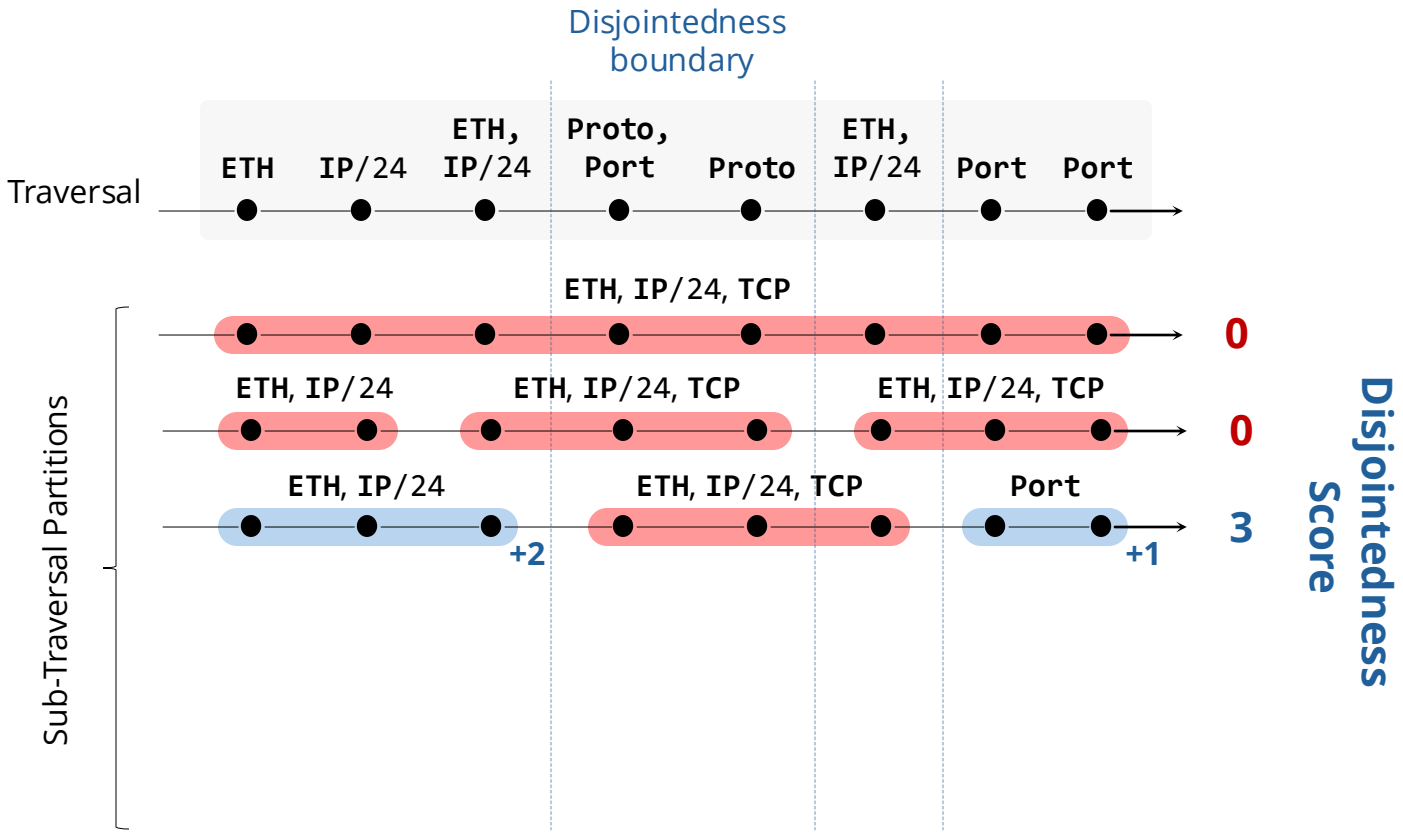
Disjointedness for Pipeline-Aware Locality



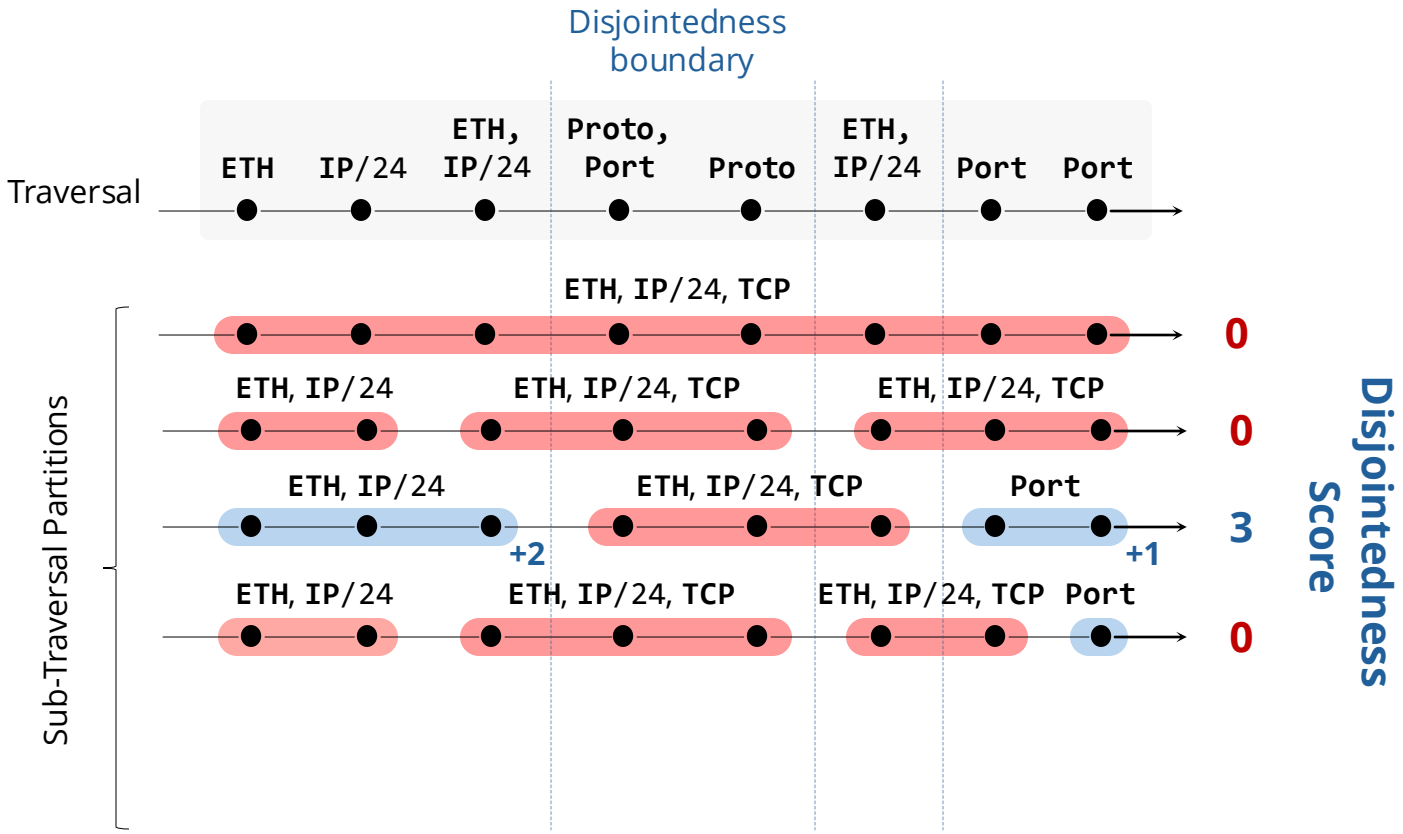
Disjointedness for Pipeline-Aware Locality



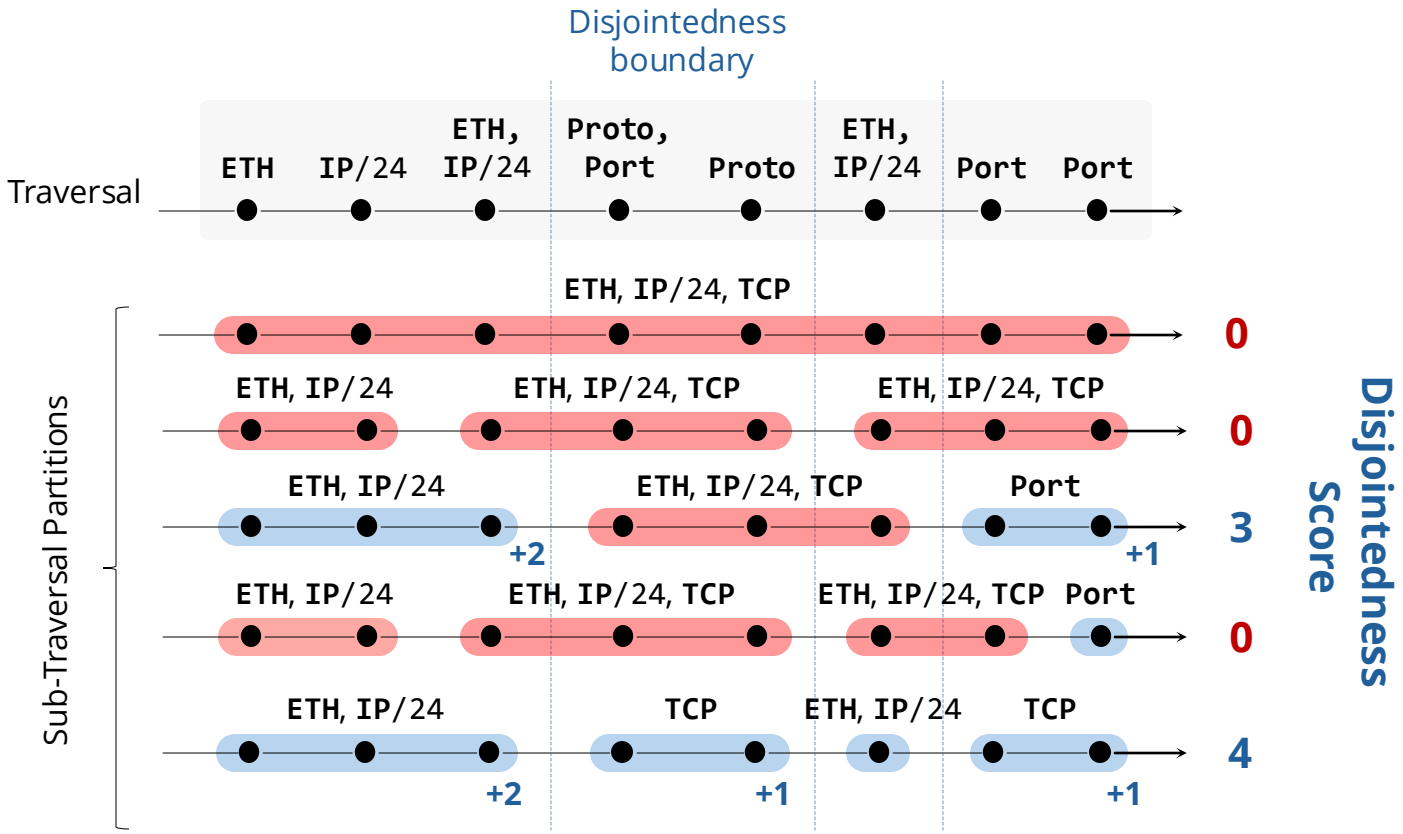
Disjointedness for Pipeline-Aware Locality



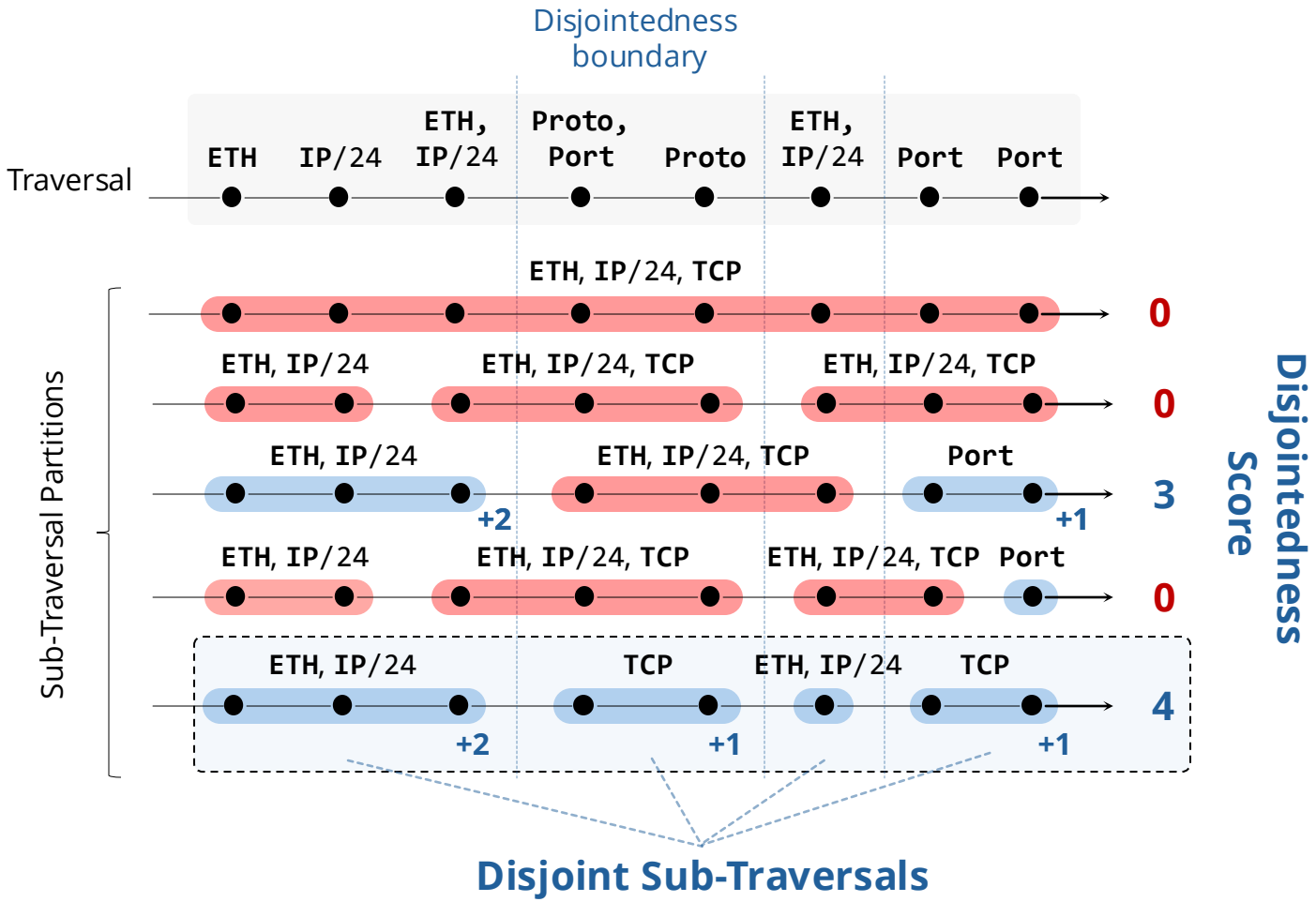
Disjointedness for Pipeline-Aware Locality



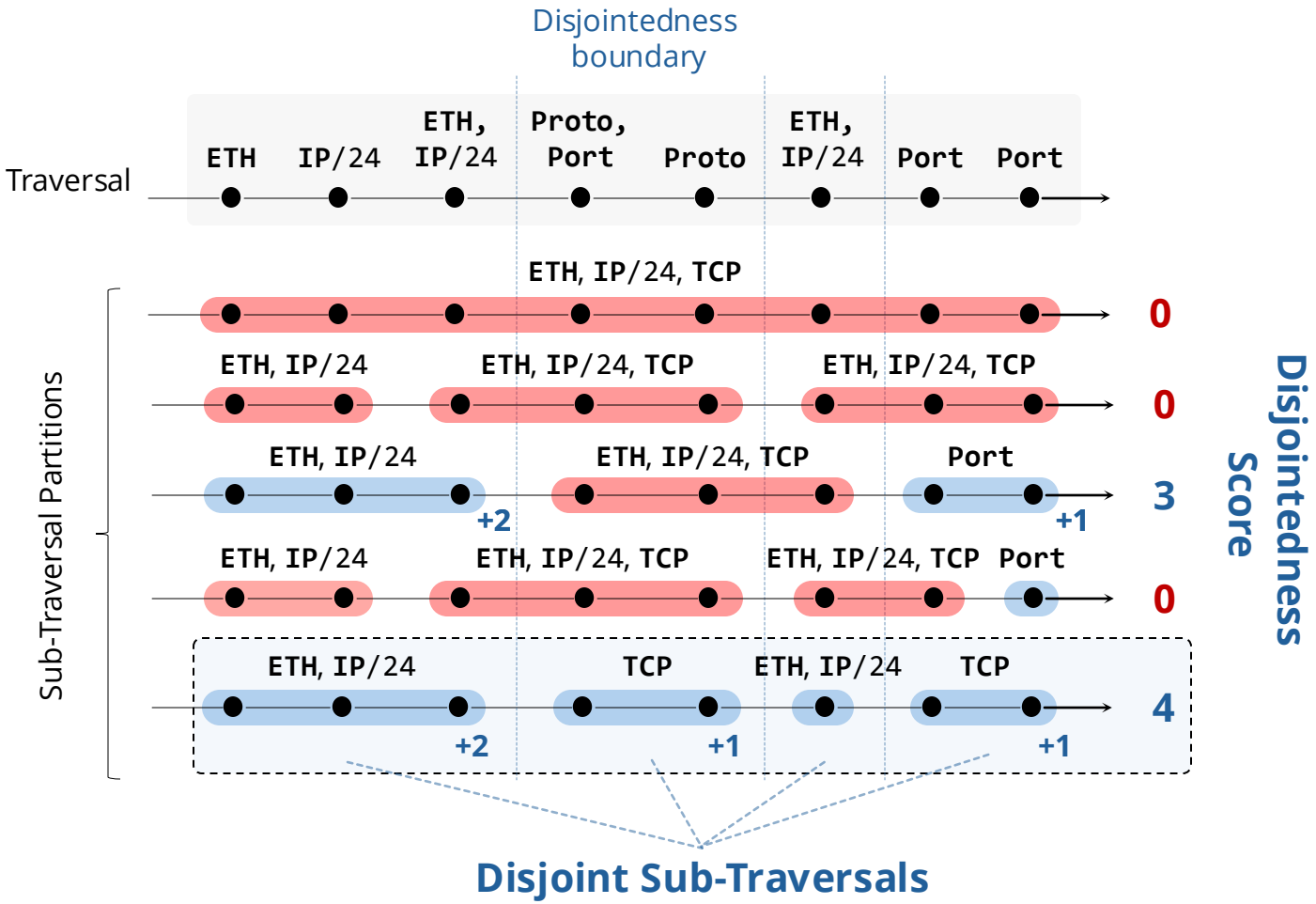
Disjointedness for Pipeline-Aware Locality



Disjointedness for Pipeline-Aware Locality



Disjointedness for Pipeline-Aware Locality

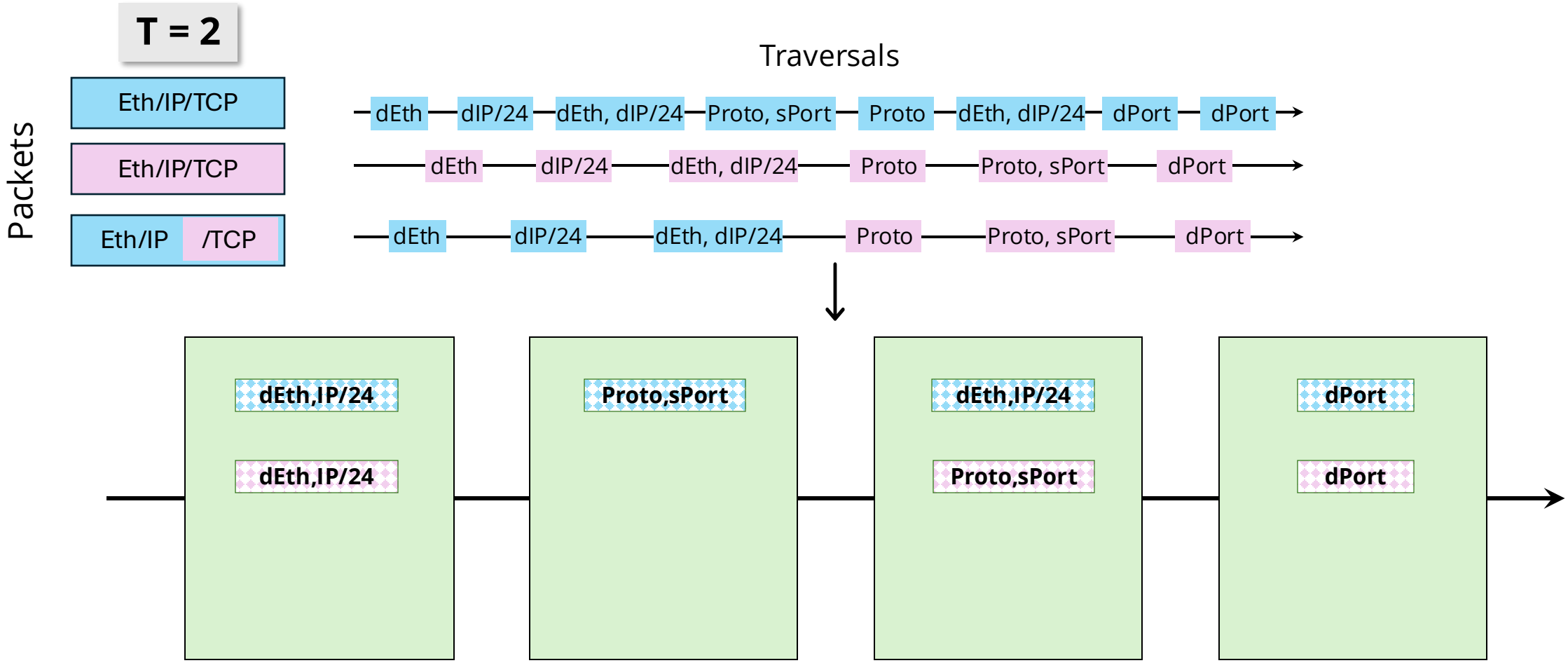


Worst Case Overhead	**5x**
Amortized Overhead	**2x**

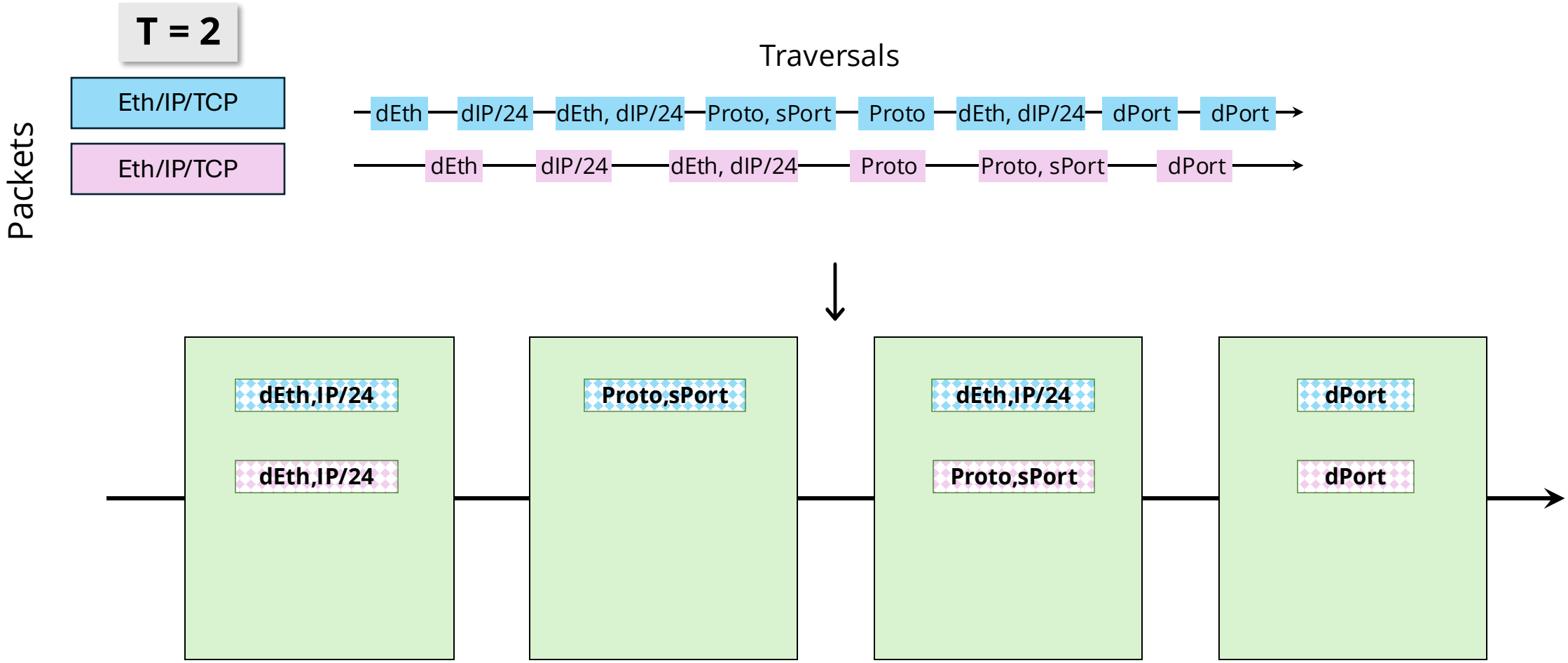
Software processing overhead of Disjoint Partitioning in Gigaflow

Overall cost is **negligible** because **90%** of **cache misses** are **reduced**

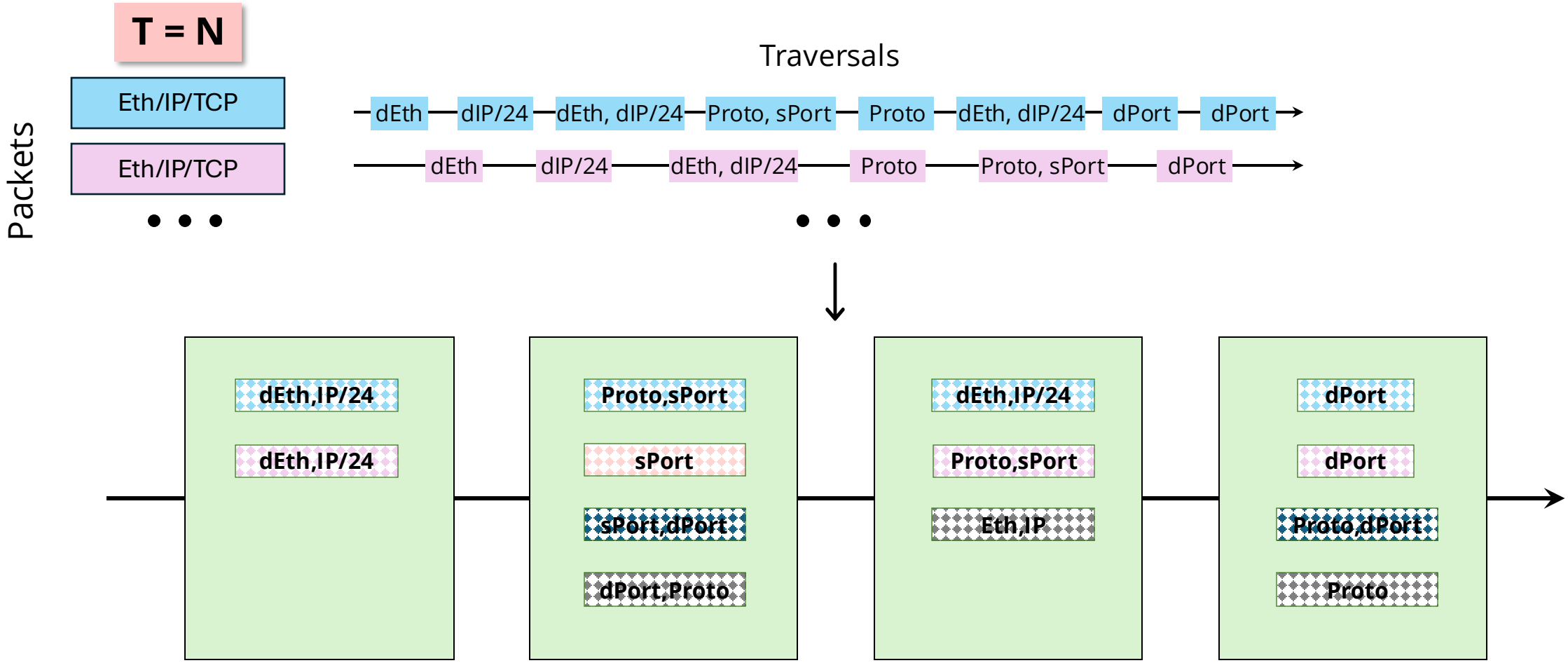
Performing a Cache Lookup with Gigaflow



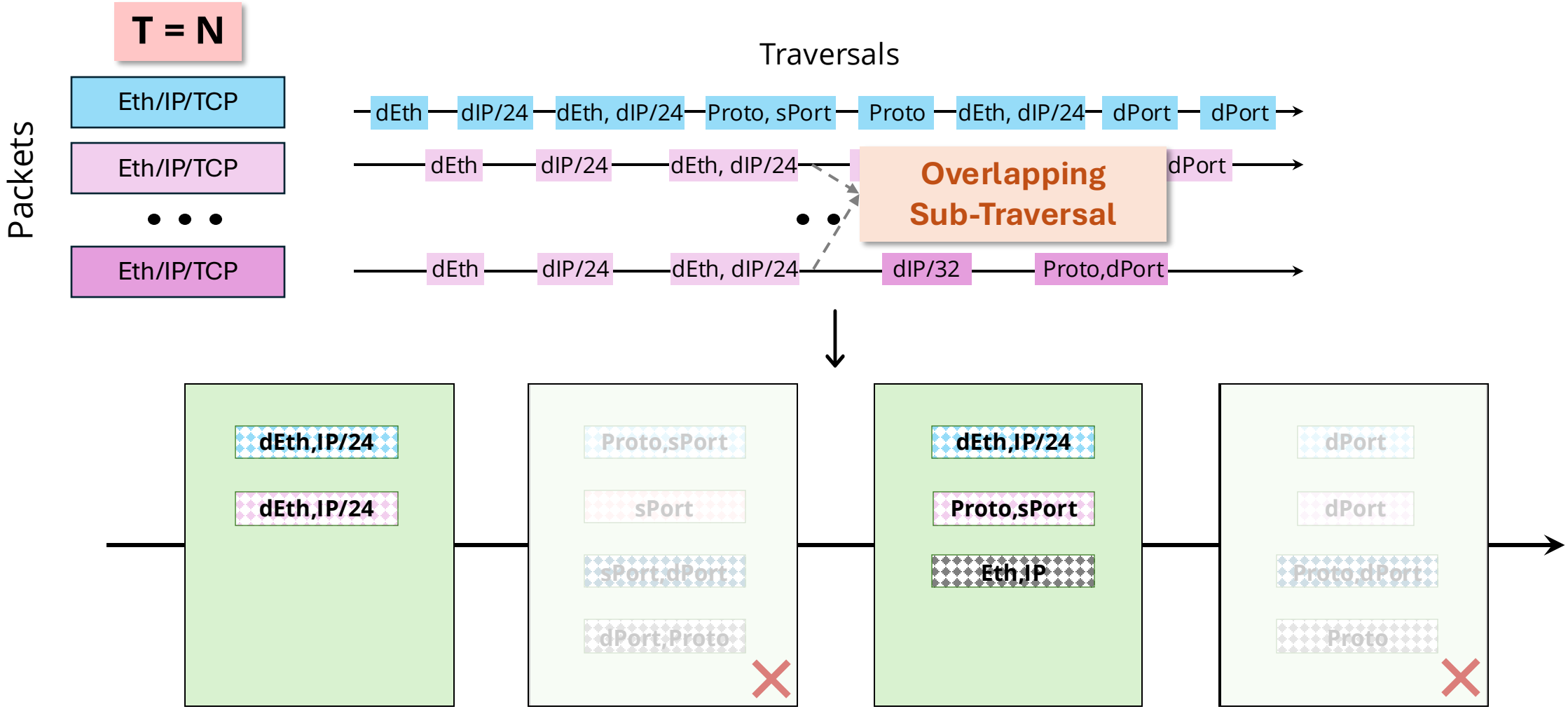
Performing a Cache Lookup with Gigaflow



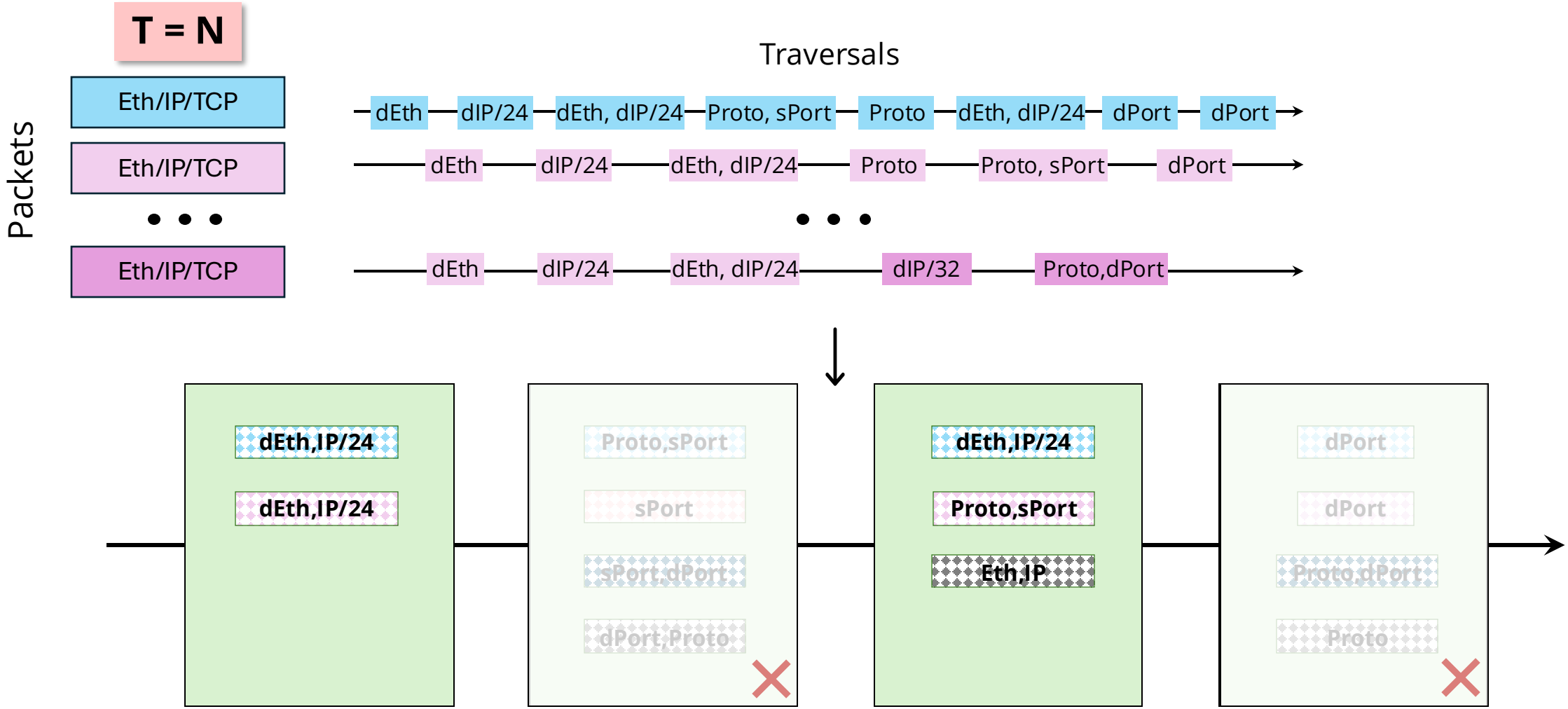
Performing a Cache Lookup with Gigaflow



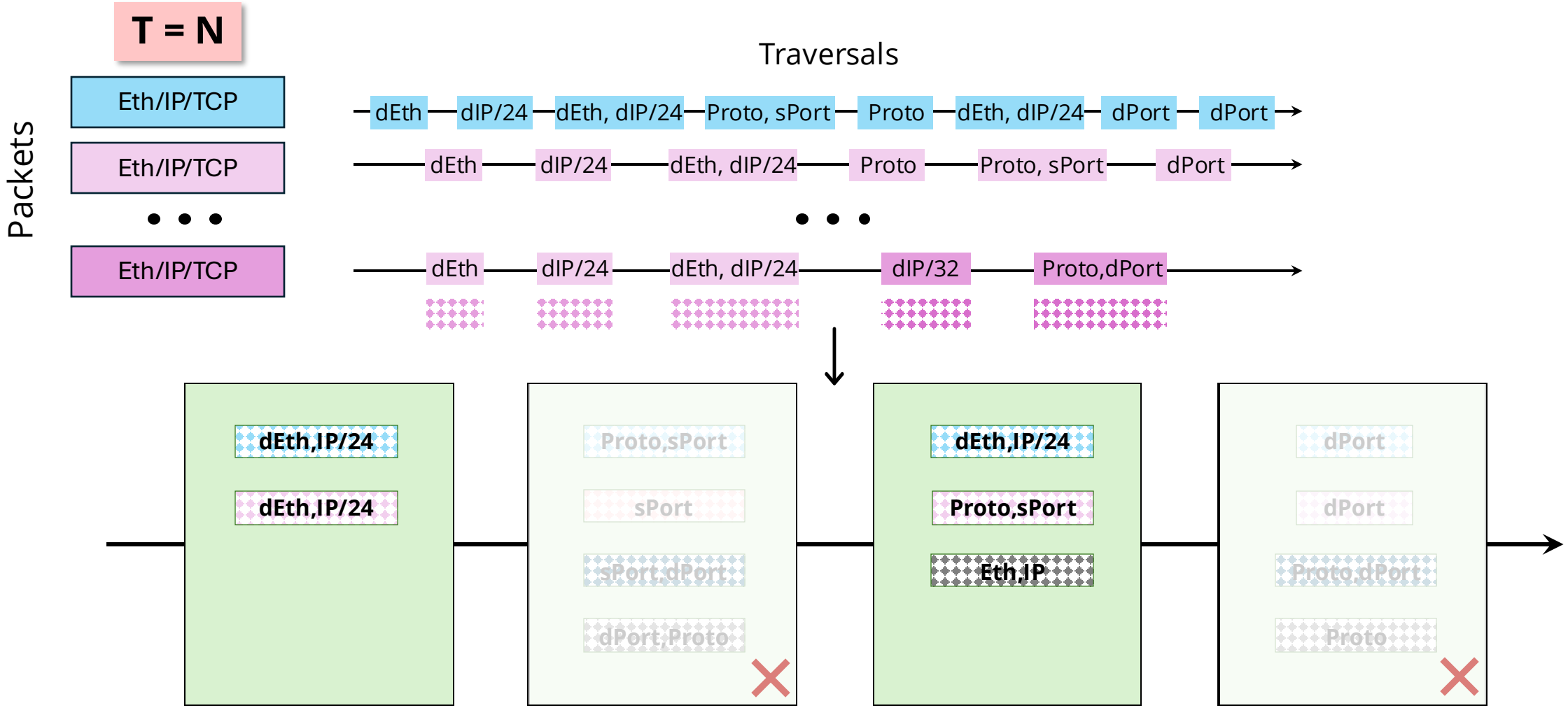
Performing a Cache Lookup with Gigaflow



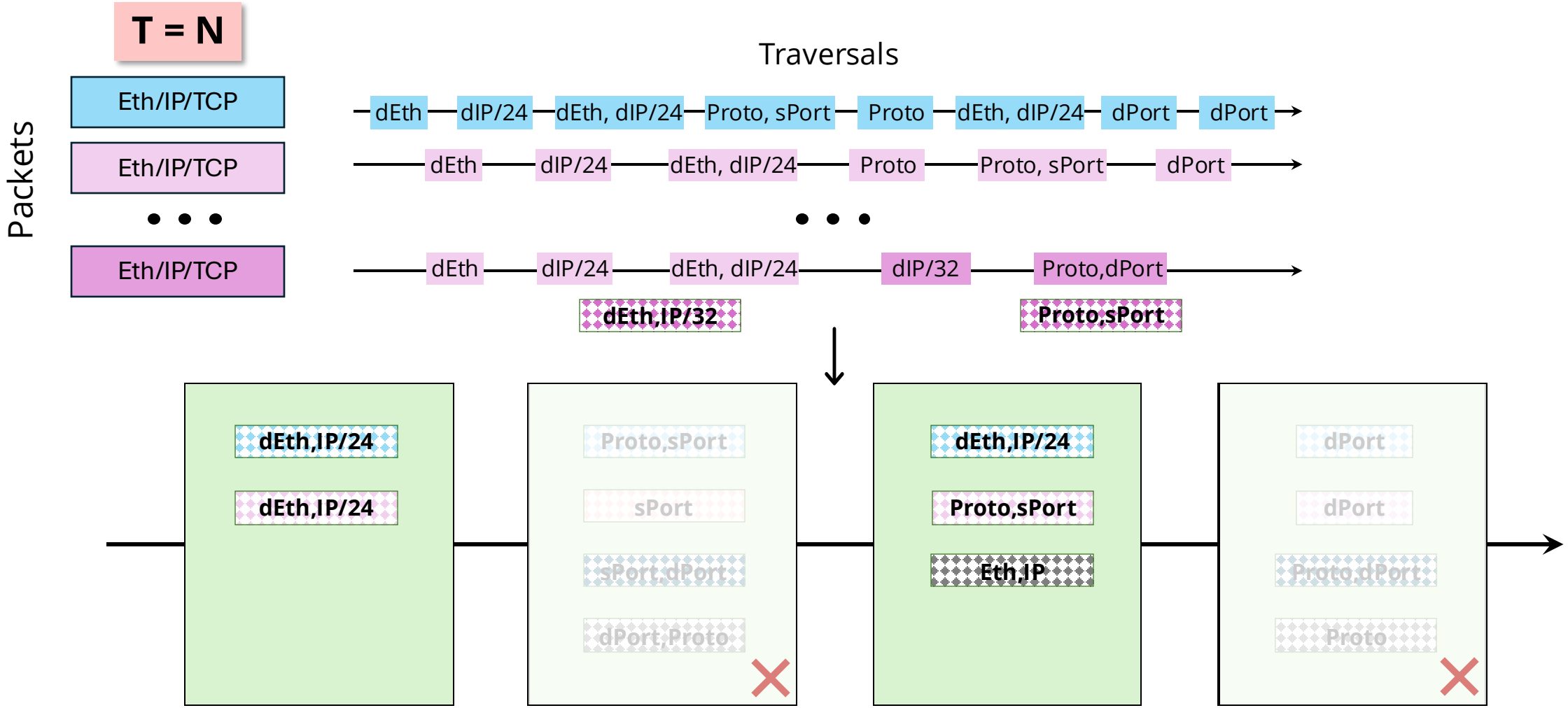
Performing a Cache Lookup with Gigaflow



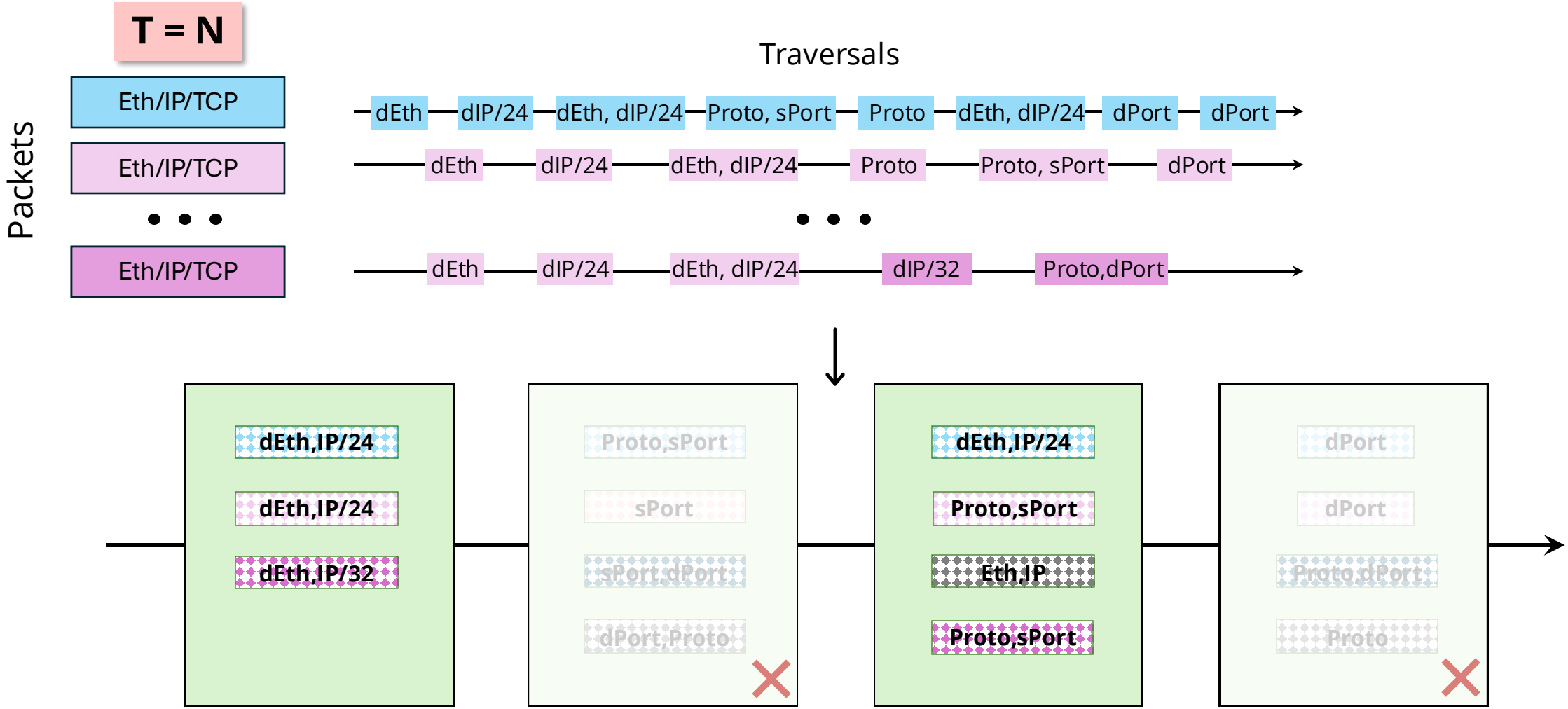
Performing a Cache Lookup with Gigaflow



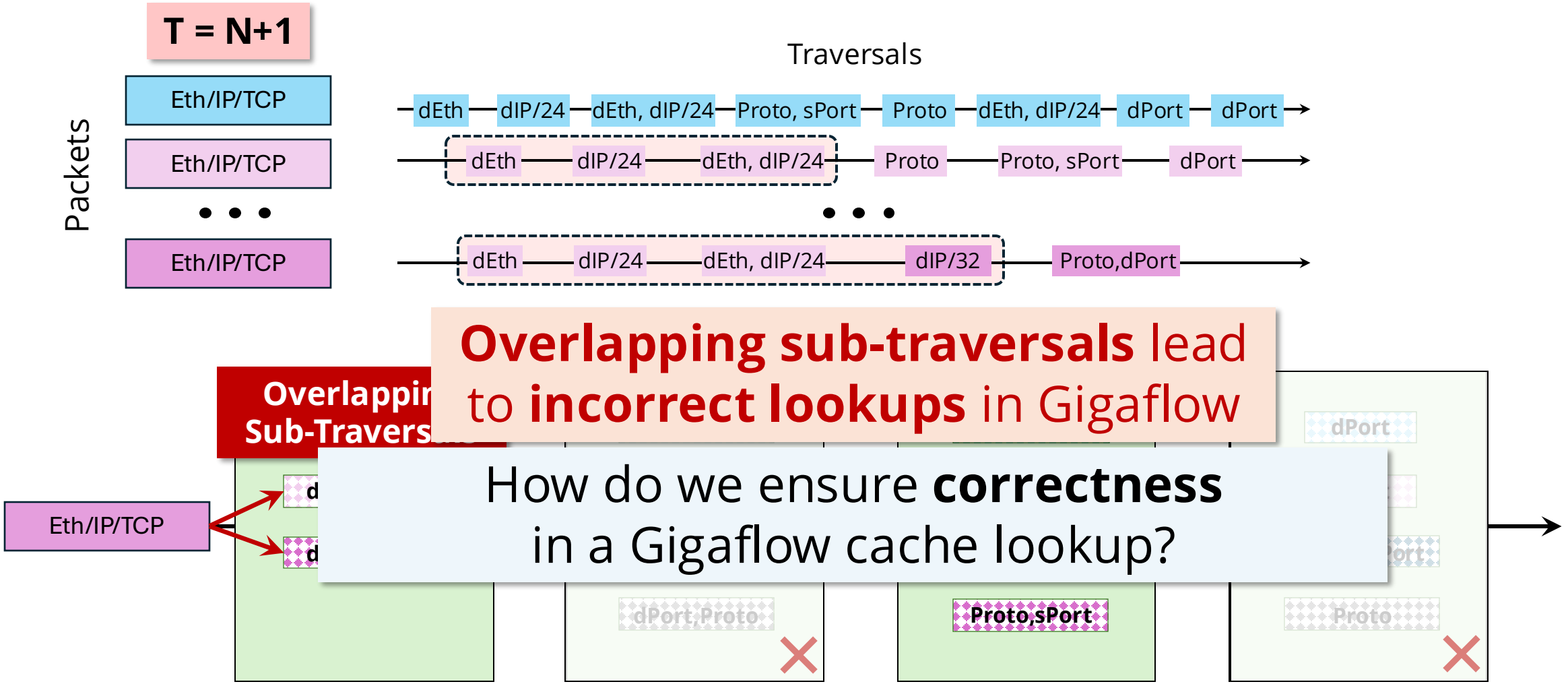
Performing a Cache Lookup with Gigaflow



Performing a Cache Lookup with Gigaflow



Performing a Cache Lookup with Gigaflow



Overlapping Rules in Routing Tables

Longest Prefix Match (LPM) in Routing

- **Challenge:**
  - A packet may match multiple destination prefixes
- **Goal:**
  - Route to the best-known destination
- **Rationale:**
  - Most specific match represents *targeted intent*
  - General matches are a *fallback* mechanism
- **Result:**
  - The packet follows the most precise path

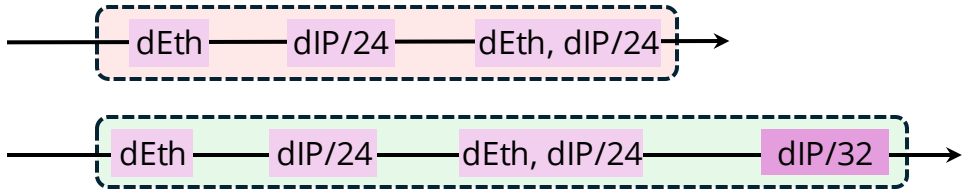
10.1.2.130

Table Rules

10.1.2.128/25 → **P:2**
10.1.2.0/24 → **P:4**
10.1.0.0/16 → **P:3**
10.0.0.0/8 → **P:5**

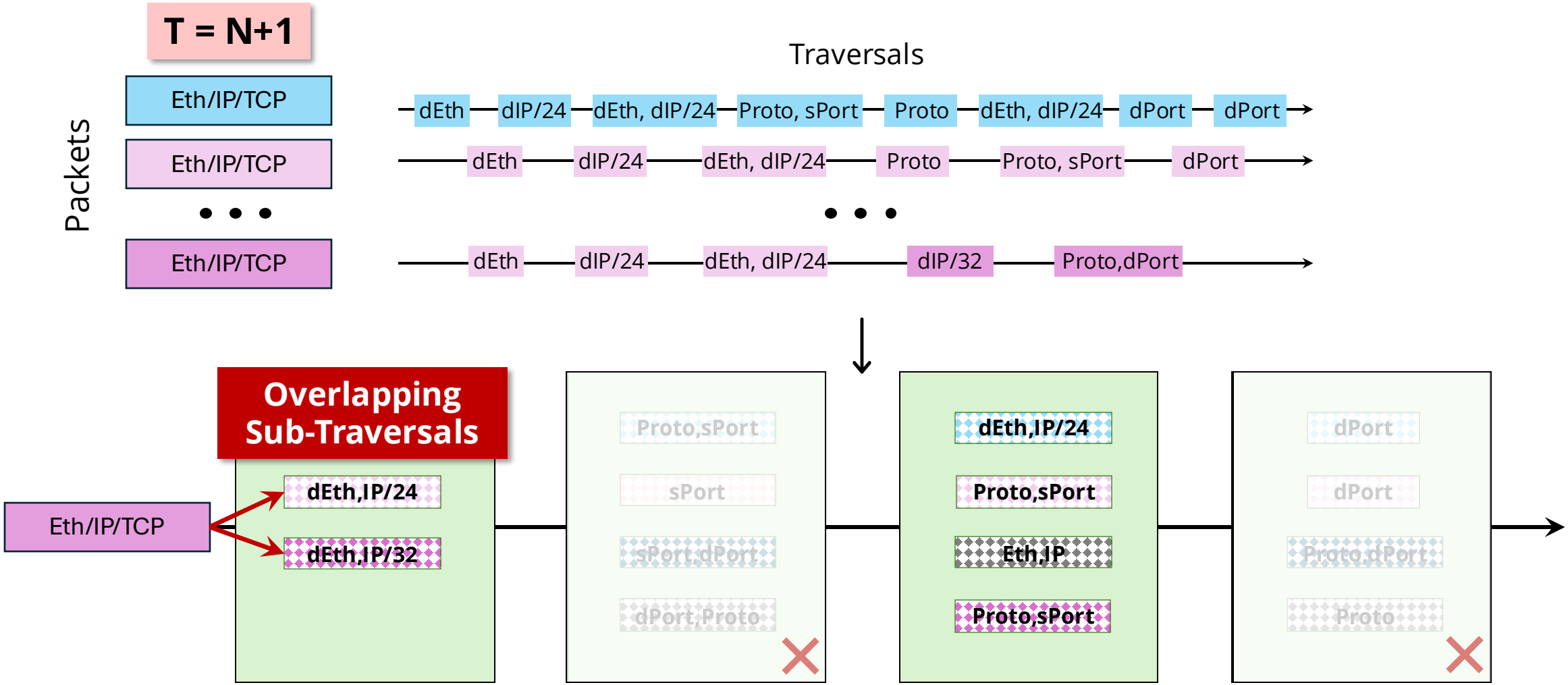
Overlapping Sub-Traversals in Gigaflow

- **Challenge:**
  - Packets match multiple overlapping sub-traversals
- **Goal:**
  - Highest priority processing for this packet
- **Rationale:**
  - Capture processing that might not appear later
  - Shorter sub-traversals are a *fallback* mechanism
- **Result:**
  - Gigaflow matches highest priority traversals

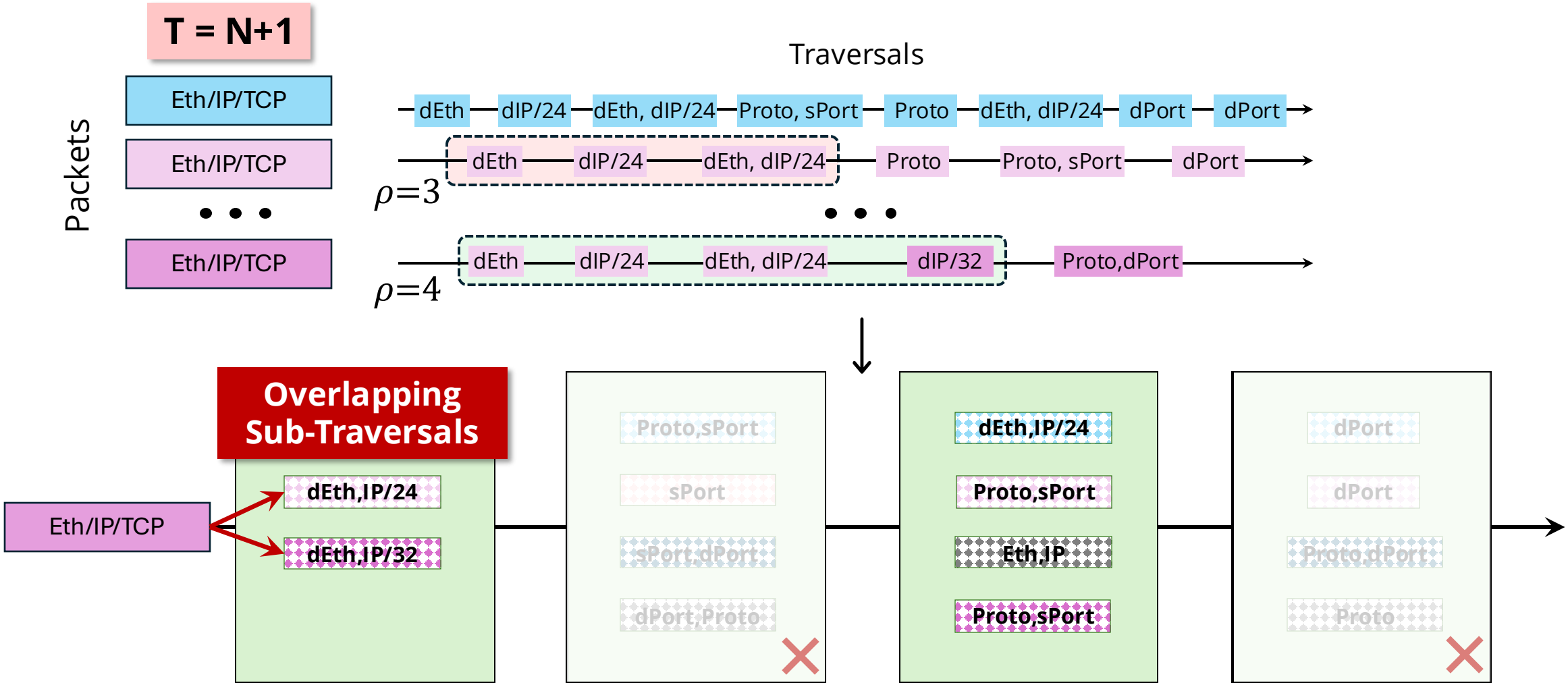


Longest Traversal Matching (LTM)

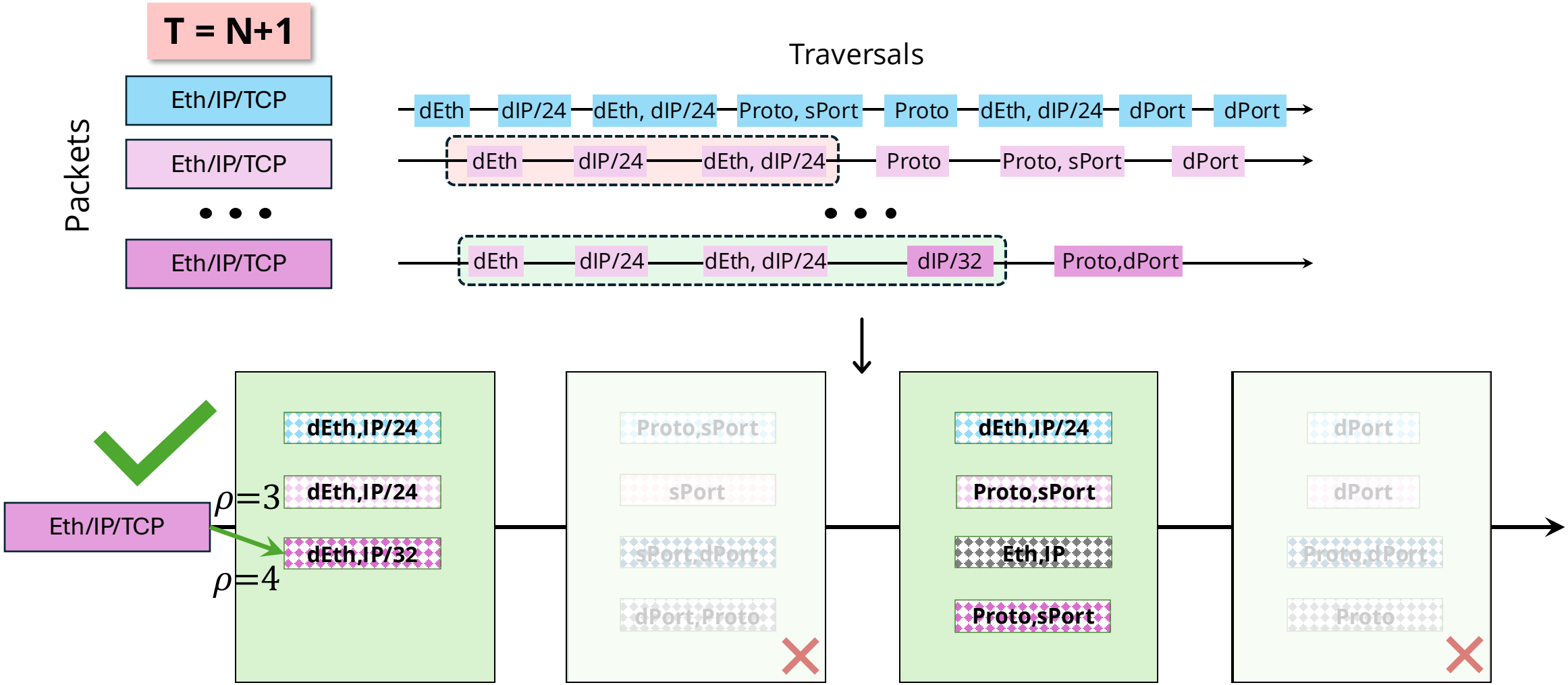
Performing a Cache Lookup with Gigaflow



Performing a Cache Lookup with Gigaflow

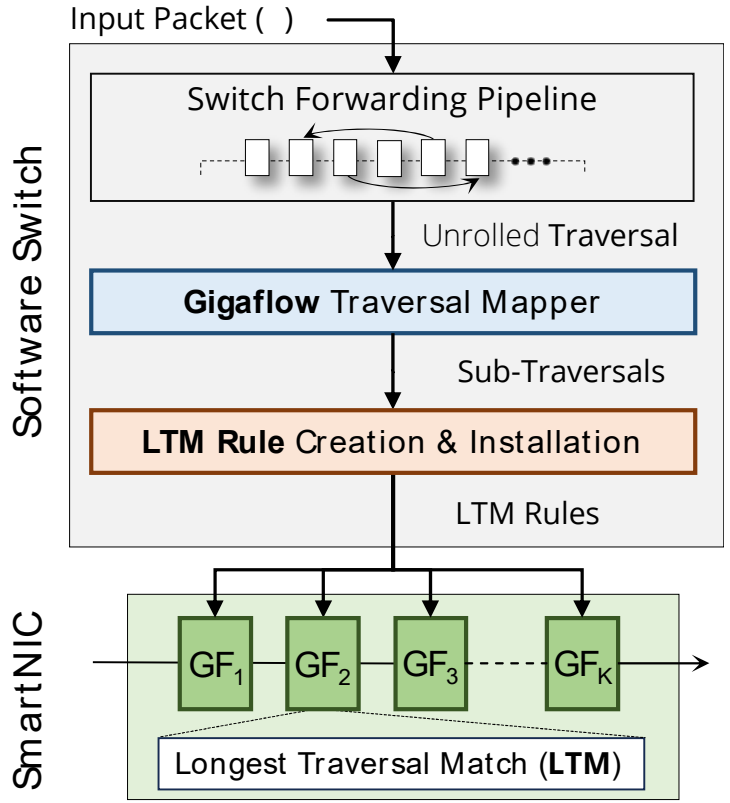


Performing a Cache Lookup with Gigaflow



Gigaflow Cache Implementation

- Integrated in the **Open vSwitch**
- Cache offload for **Xilinx Alveo U250**
- **Xilinx P4 SDNet** for hardware tables



High-level view of the Gigaflow in OVS

vSwitch Pipelines and Traffic Workloads

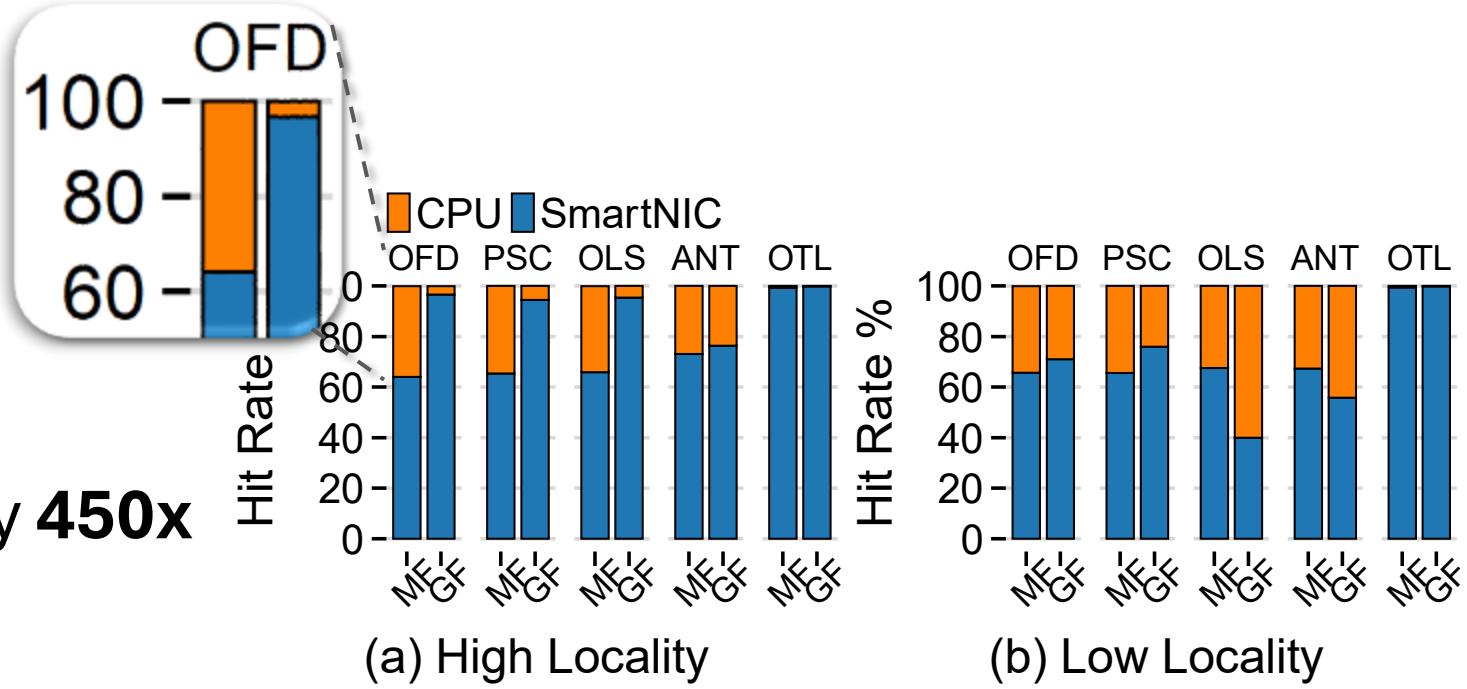
Pipeline	Description	Tables	Traversals
P1: OFD	CORD's Openflow data plane abstraction (OFDPA) for HW/SW switch integration	10	5
P2: PSC	An example L2L3-ACL pipeline implemented for the Pisces paper	7	2
P3: OLS	OVN Logical Switch pipeline to manage logical virtual network topologies in OVS	30	23
P4: ANT	Antrea pipeline implementing networking and security for Kubernetes clusters	22	20
P5: OTL	Openflow Table Type Patterns (TTP) to configure L2L3-ACL policies using OVS	8	11

- Real-world pipeline configurations
  - tables, matching fields, traversals
- Classbench rulesets to populate multi-table rules
- CAIDA traces for realistic traffic patterns
- High/low locality environments
  - high/low sub-traversal sharing opportunity

Performance Summary of Gigaflow

Gigaflow improves

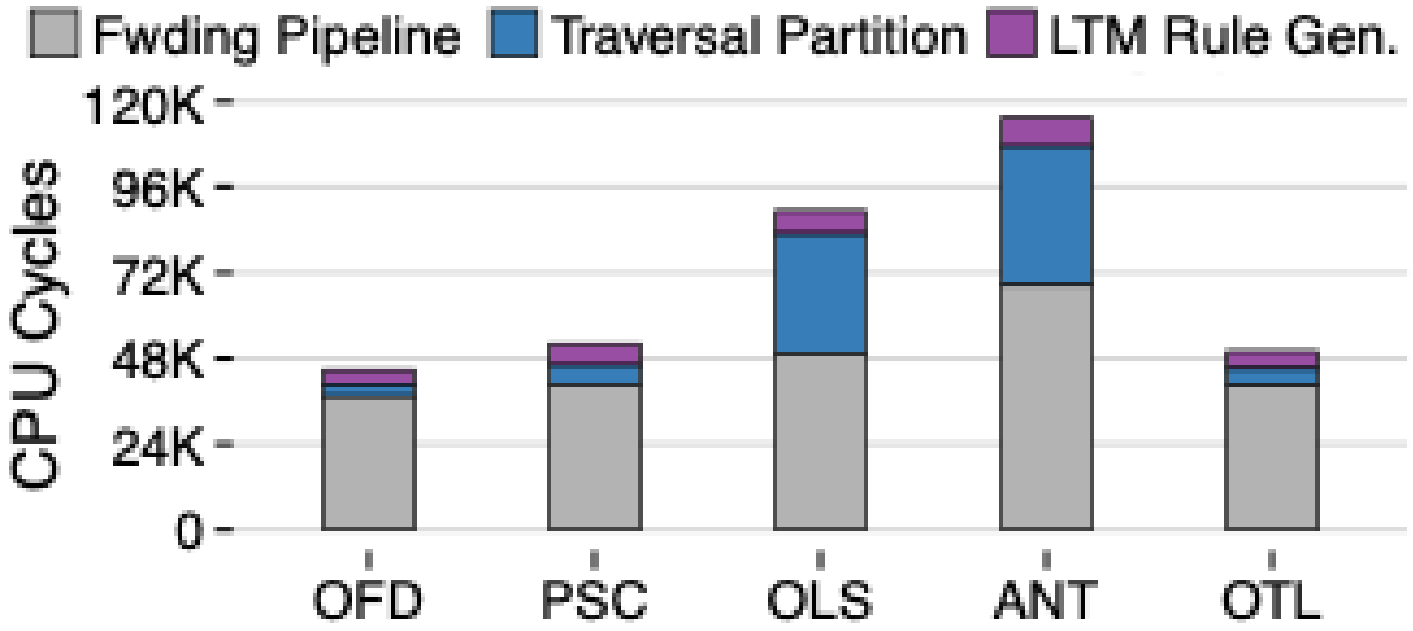
- **#Cache misses** by **90%**
- **Cache hit rate** by **51%**
- **Rule space coverage** by **450x**
- **#Cache entries** by **18%**
- **Forwarding latency** by **30%**



(a) High Locality (b) Low Locality

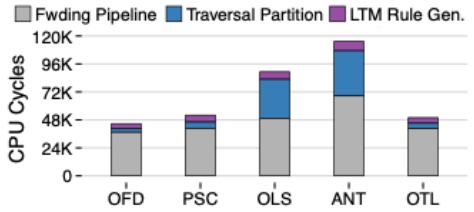
Cache hit rate of Megaflow 32K (MF)
vs Gigaflow 4x8K (GF) for real-world pipelines

Software Processing Overhead of Gigaflow

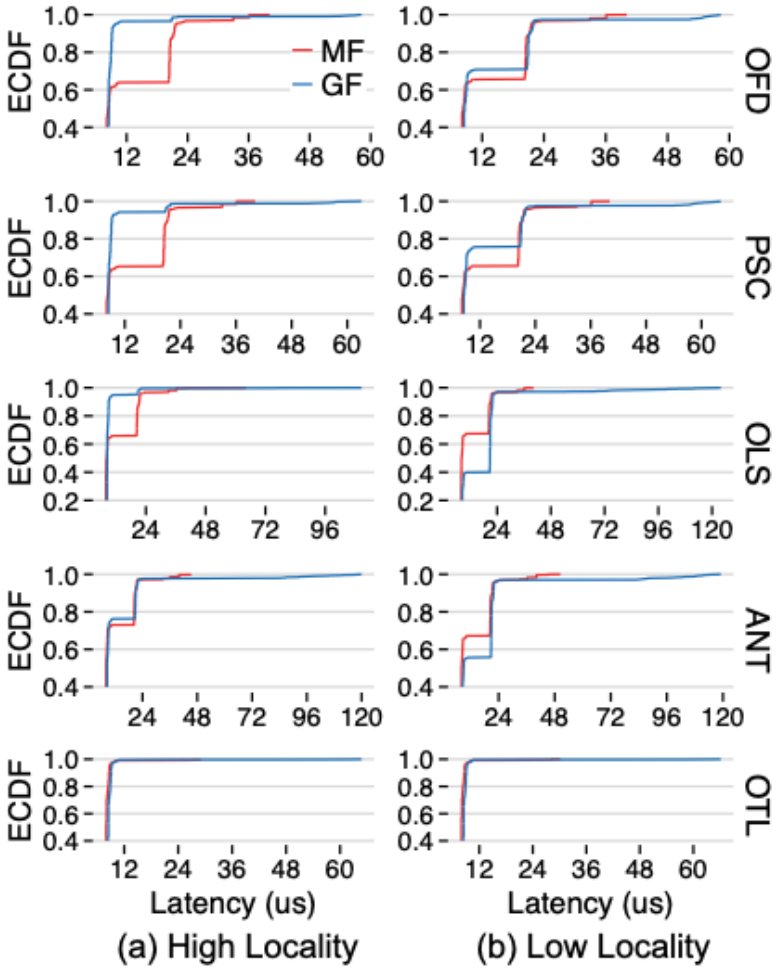


Software overhead of Gigaflow 4x8K (GF)
for five real-world pipelines

End-to-End Packet Latency with Gigaflow

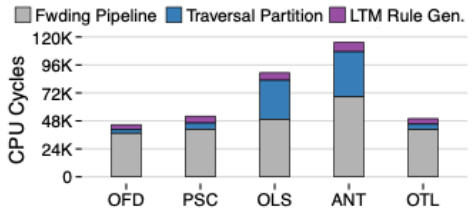


Software overhead of Gigaflow 4x8K (GF) for five real-world pipelines

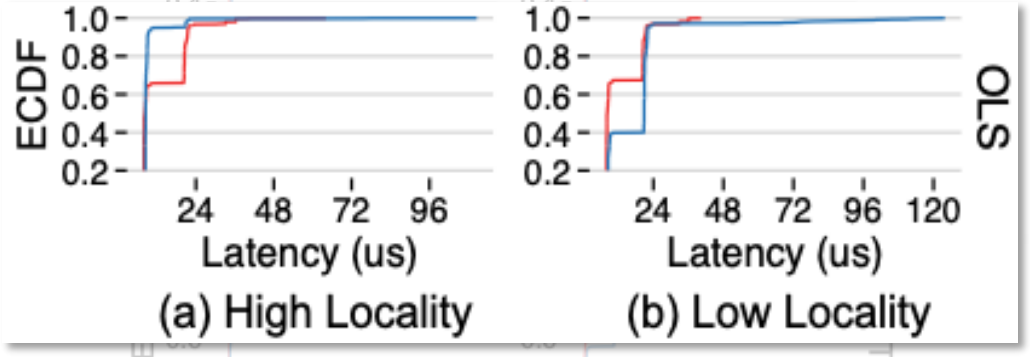


End-to-end latency: Megaflow 32K (MF) vs Gigaflow 4x8K (GF) in high/low locality

End-to-End Packet Latency with Gigaflow

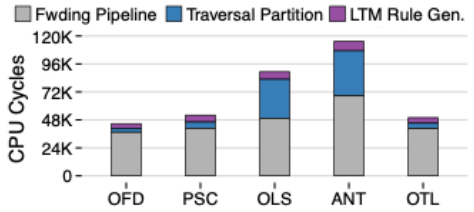


Software overhead of Gigaflow 4x8K (GF) for five real-world pipelines

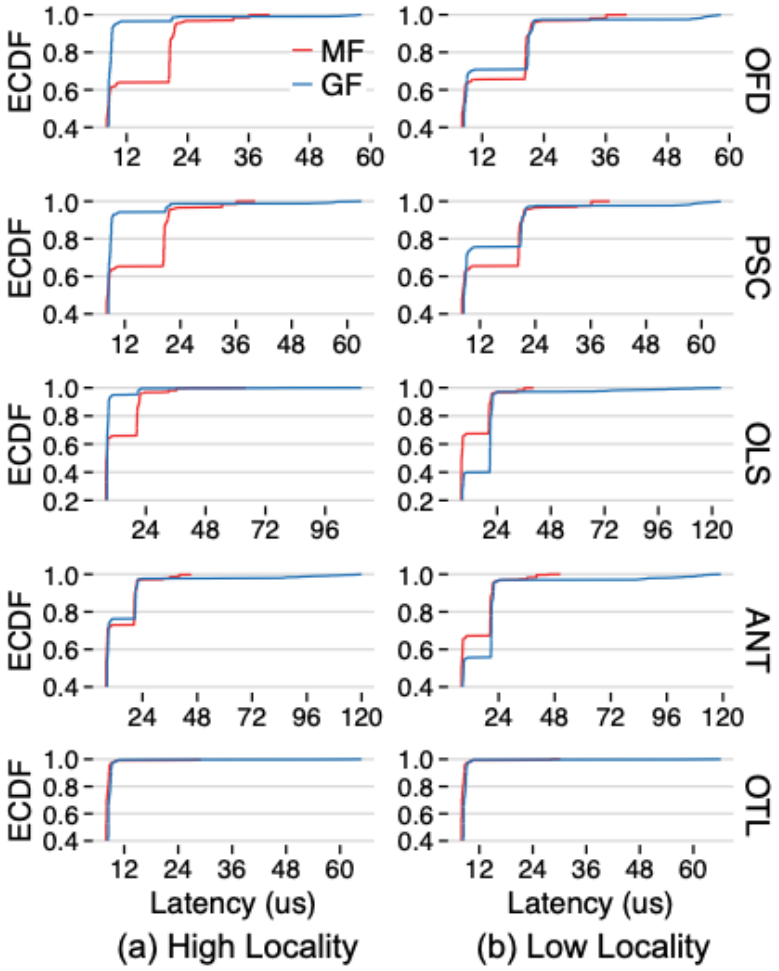


End-to-end latency: Megaflow 32K (MF) vs Gigaflow 4x8K (GF) in high/low locality

End-to-End Packet Latency with Gigaflow

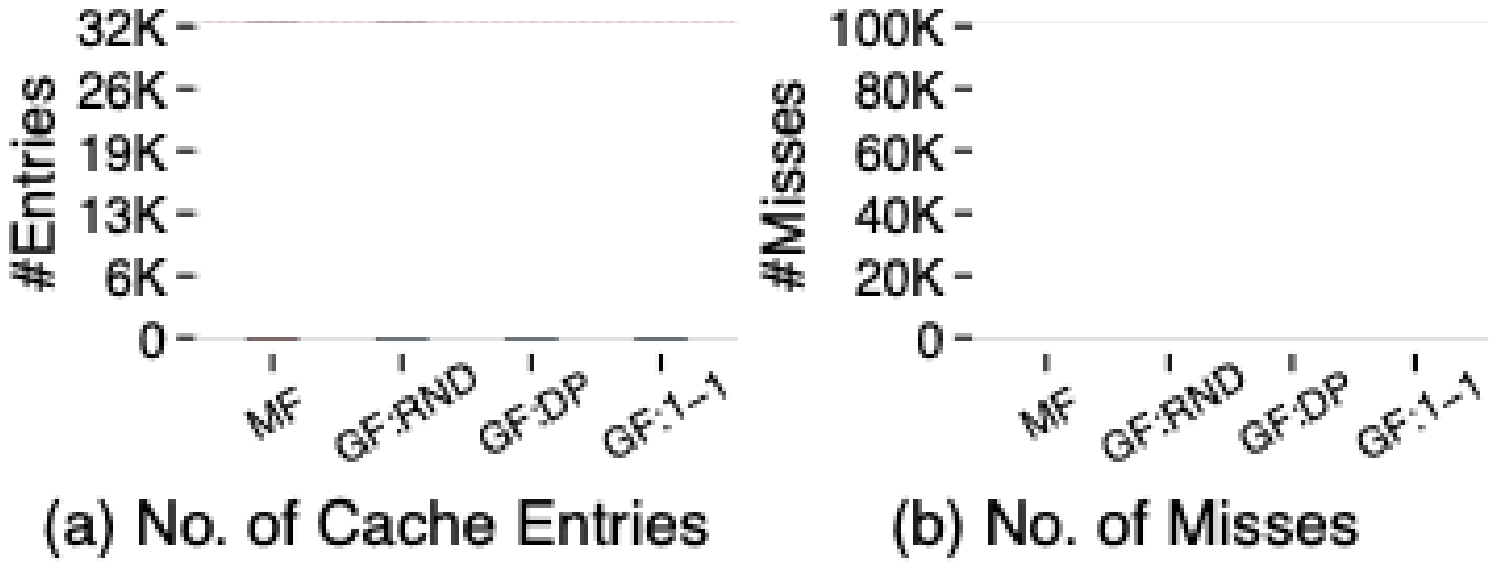


Software overhead of Gigaflow 4x8K (GF) for five real-world pipelines



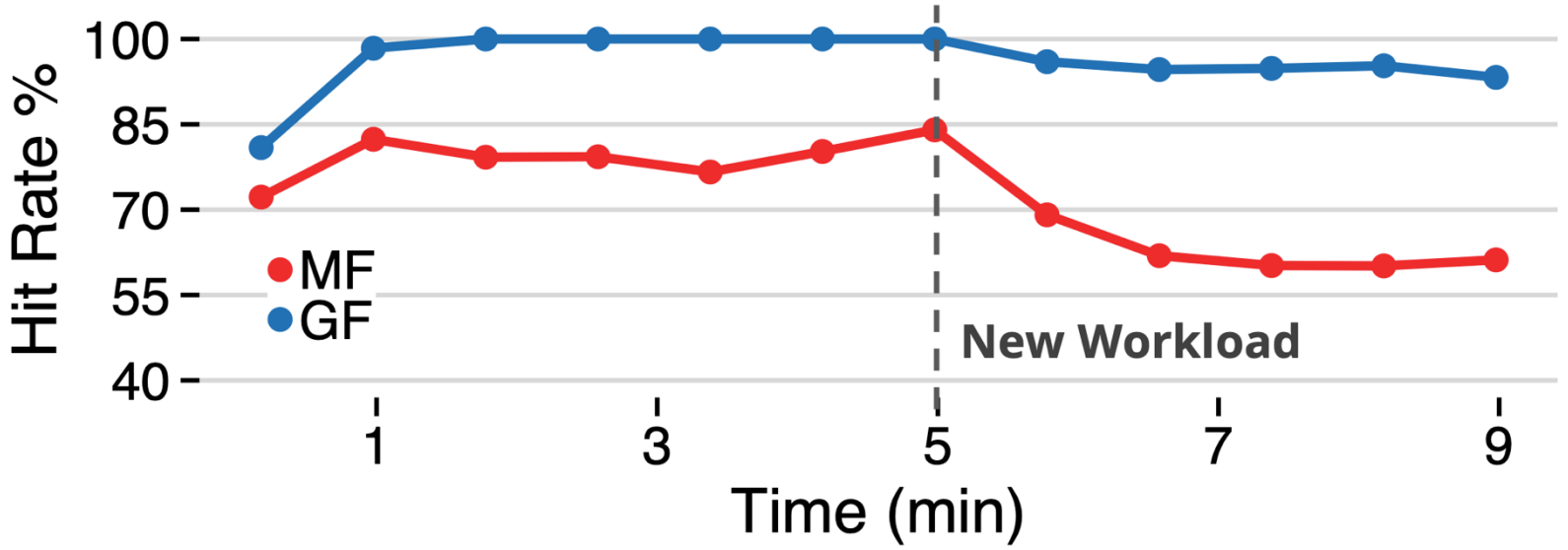
End-to-end latency: Megaflow 32K (MF) vs Gigaflow 4x8K (GF) in high/low locality

Gigaflow with Various Partitioning Strategies



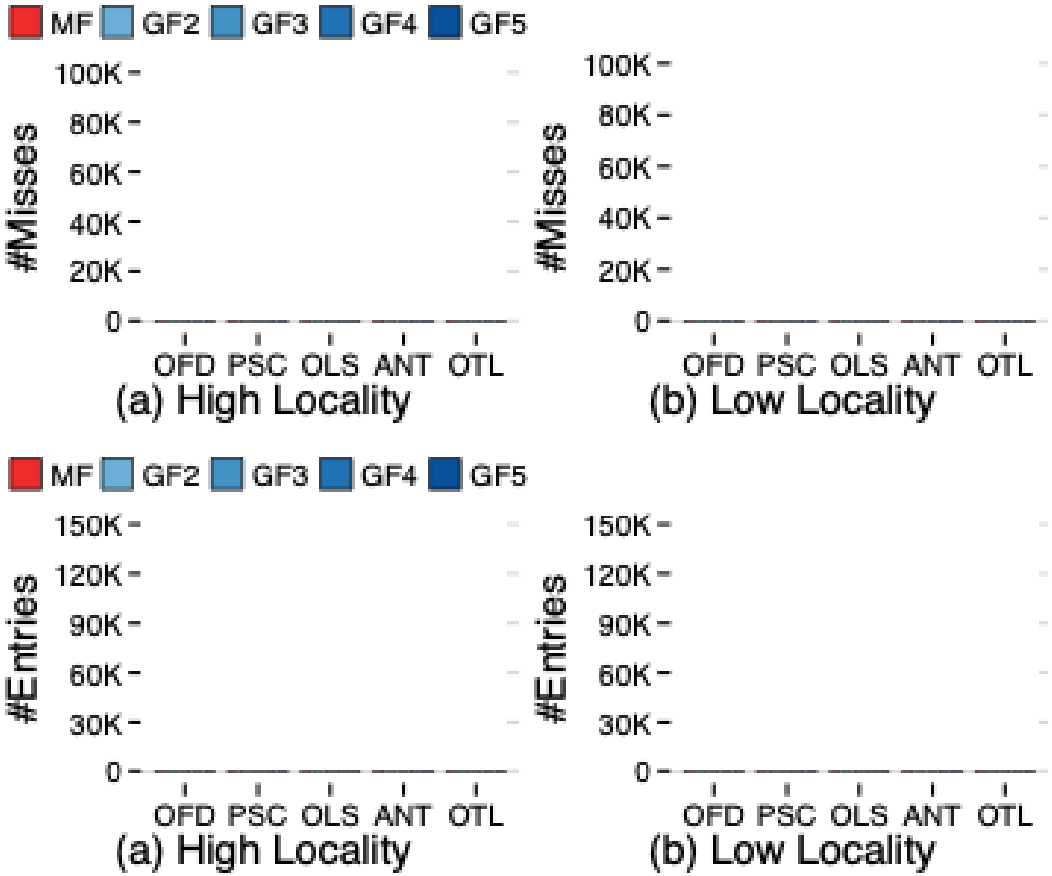
Number of cache entries and cache misses with various partitioning strategies in Gigaflow (GF)

Gigaflow With Dynamic Workloads



Cache hit rate of Megaflow 32K (MF)
vs Gigaflow 4x8K (GF) under dynamic workloads

Gigaflow with Increasing Tables



Cache misses and cache entries with increasing #of Gigaflow (GF) tables

Thank you!

- Gigaflow → a ***sub-traversal* cache**
- Captures **pipeline-aware locality**
- Delivers **higher rule space coverage**

NextGArch Lab – https://nextgarch.github.io/

Contact us: zulfiqaa@umich.edu