ACM SIGMETRICS 2025, Stony Brook, New York, USA

Revisiting Traffic Splitting for Software Switch in Datacenter

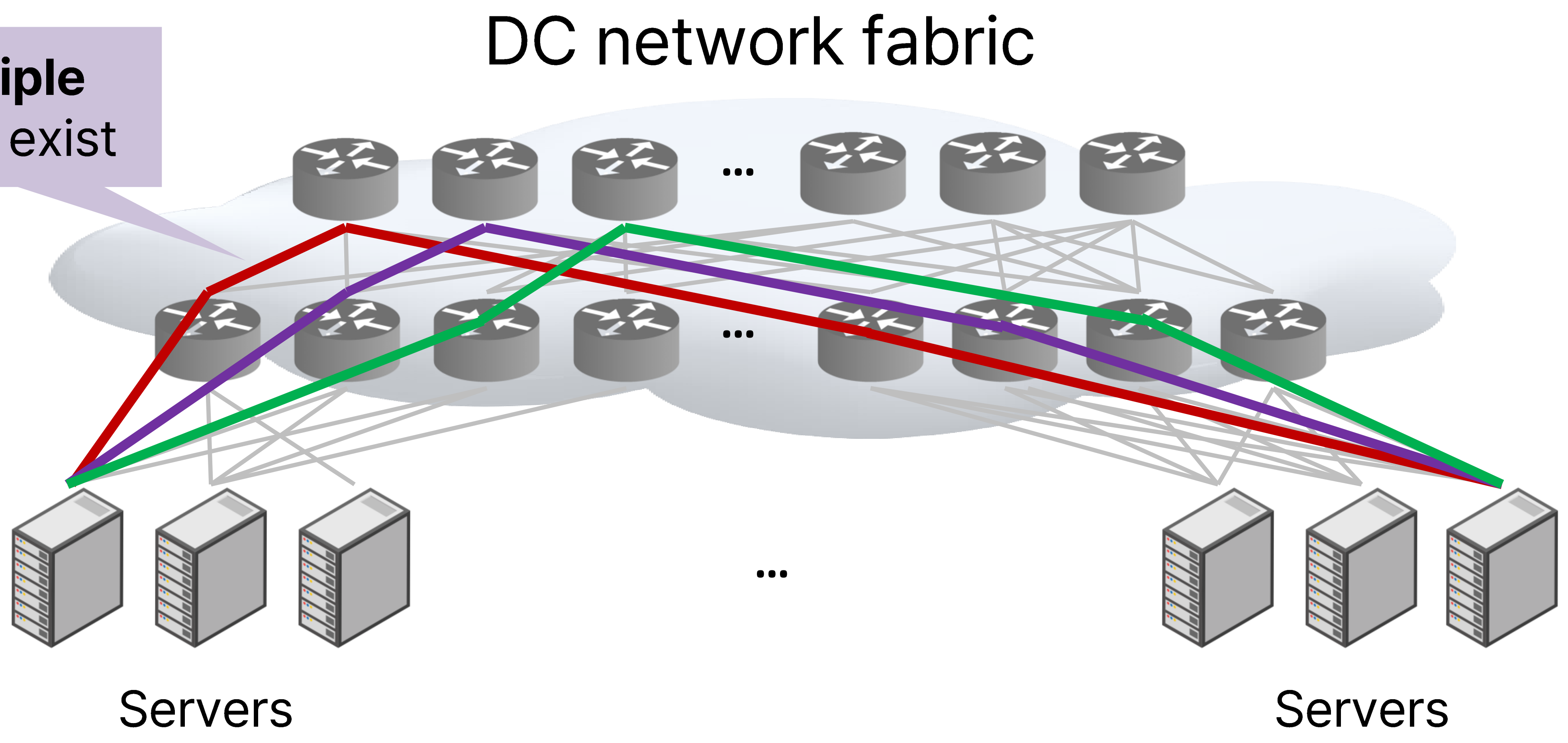
Yeonho Yoo[1], Gyeongsik Yang[1], Changyong Shin[1], Hwiju Cho[1], Wonmi Choi[1], Zhixiong Niu[2], Chuck Yoo[1]

[1]Korea University (Seoul, Republic of Korea)
[2]Microsoft Research (Beijing, China)

Multipath Networking in Datacenter

- Traffic in cloud datacenters (DCs) is increasing exponentially
  - Driven by web search, deep learning services, data mining, etc
- **Multiple paths** between servers for high throughput and reliability

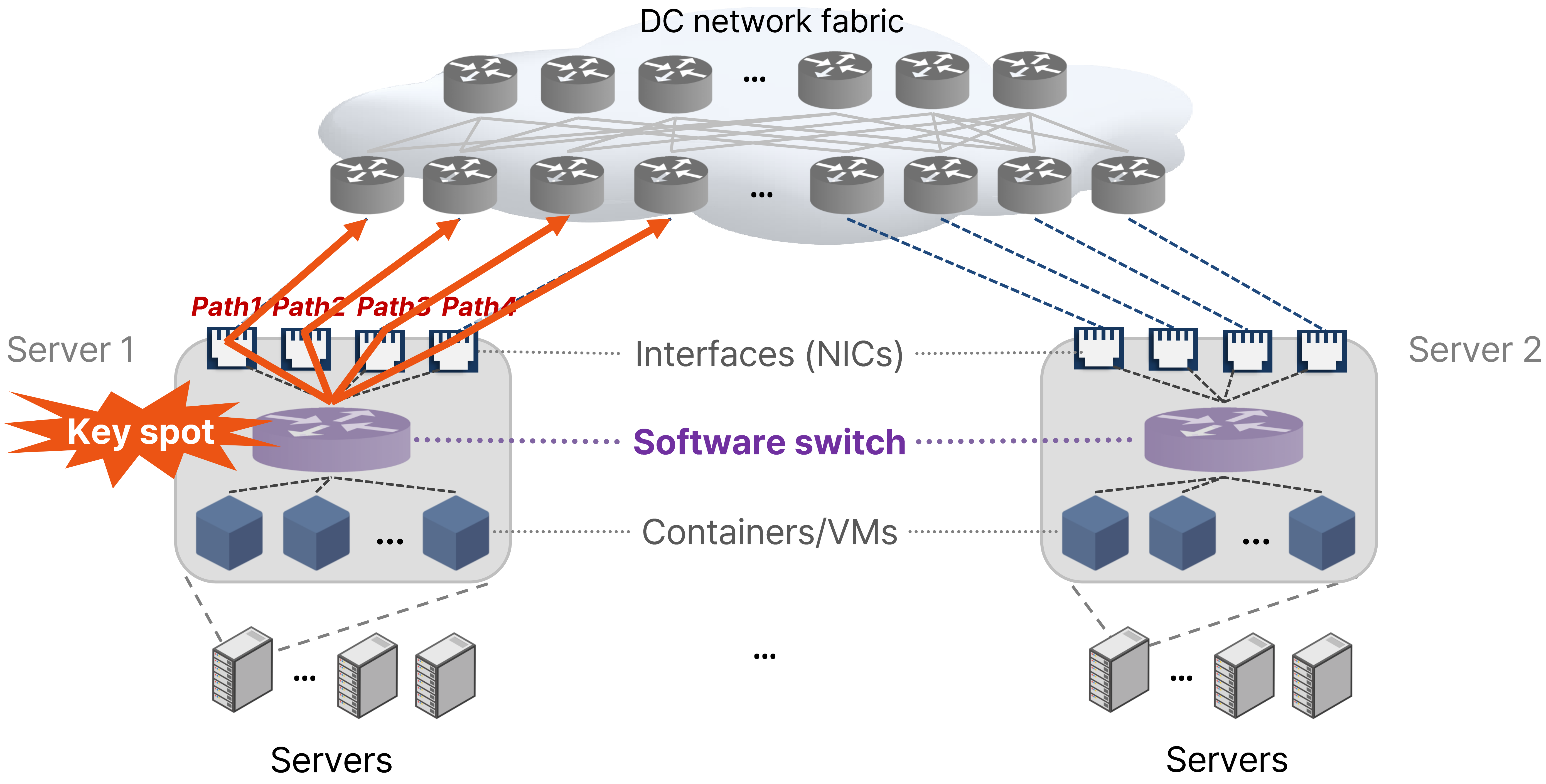
Multiple paths exist



Utilizing paths is critical

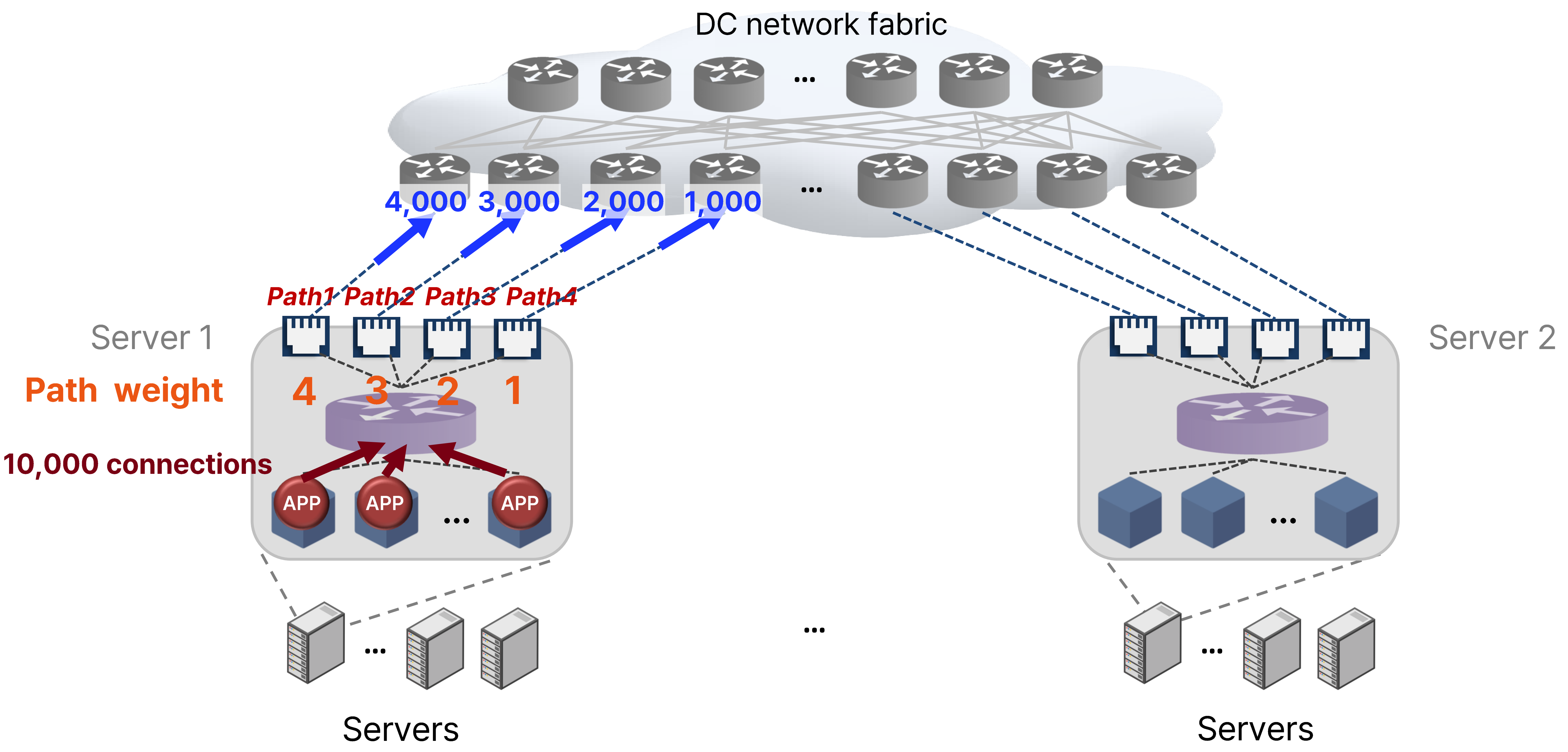
Software Switch in DC

- Multiple VMs and containers per server
- Software switch: bridge packets between 1) VMs/containers and 2) network interfaces
  - E.g., Open vSwitch (Linux Foundation), Apsara vSwitch (Alibaba), Hoverboard (Google)



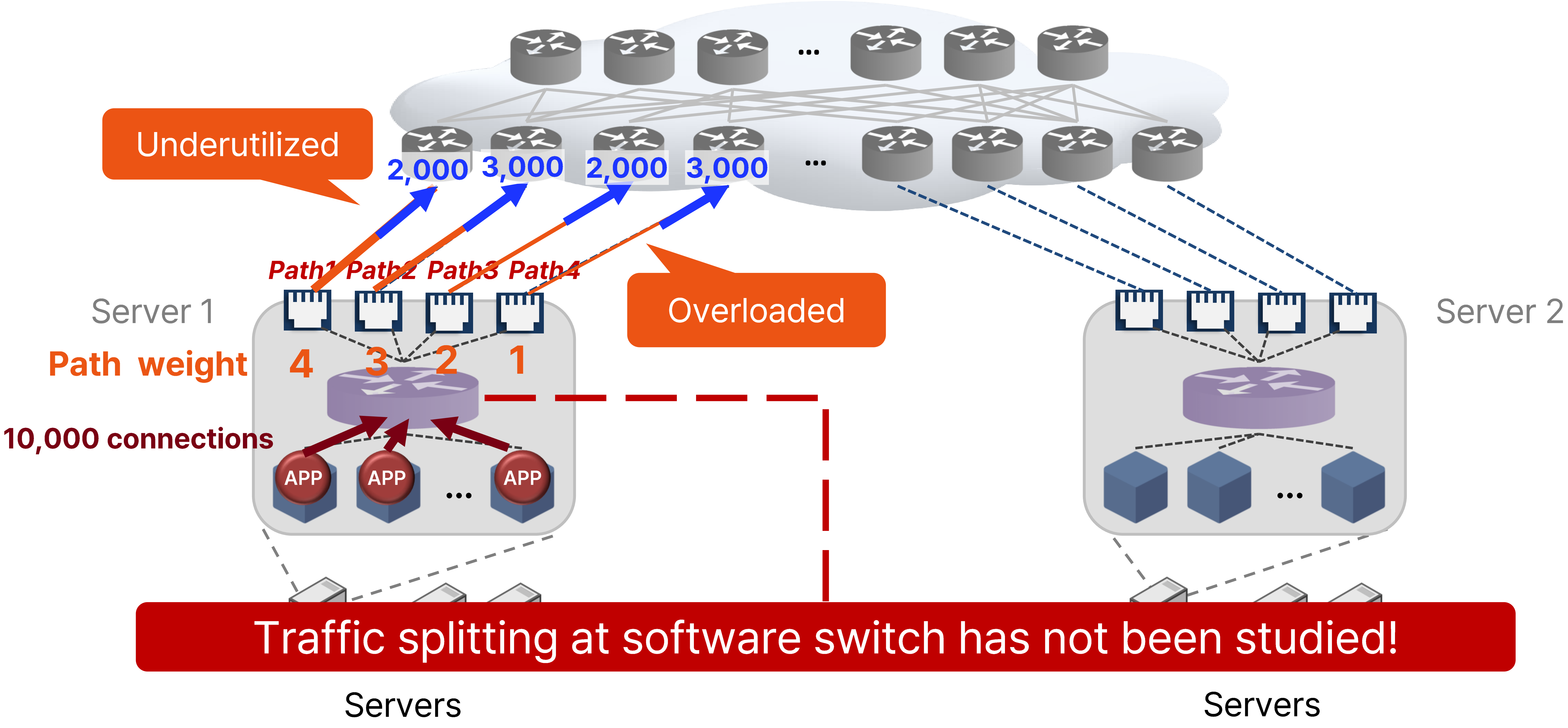
Traffic Splitting at Software Switch

- Thousands of network connections (TCP/UDP) from VMs/containers
- **Software switch splits connections across multiple paths based on weights**



Previous Studies on Software Switches

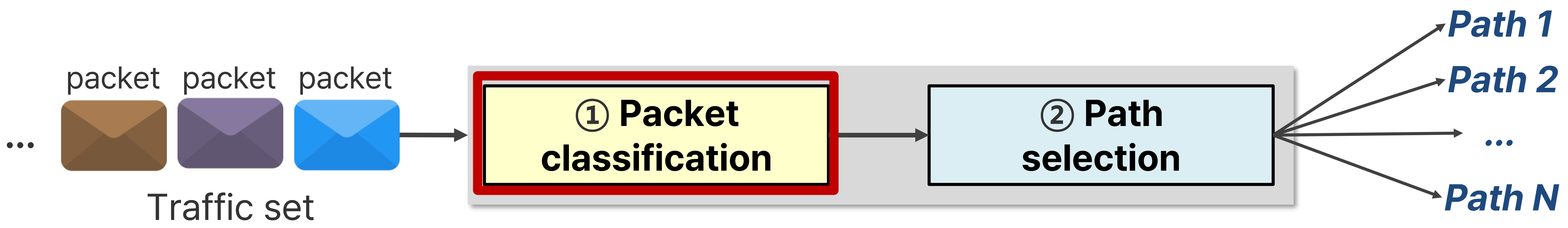
- Many studies have focused on how to determine path weights
  - CLOVE (CoNext`17): adjust weights based on congestion degree (e.g., # of ECN packets)
  - VMS (JSAC`20), TeaVisor (INFOCOM`21): path capacity (e.g., bandwidth, RTT)
- Accurate and efficient traffic splitting often overlooked



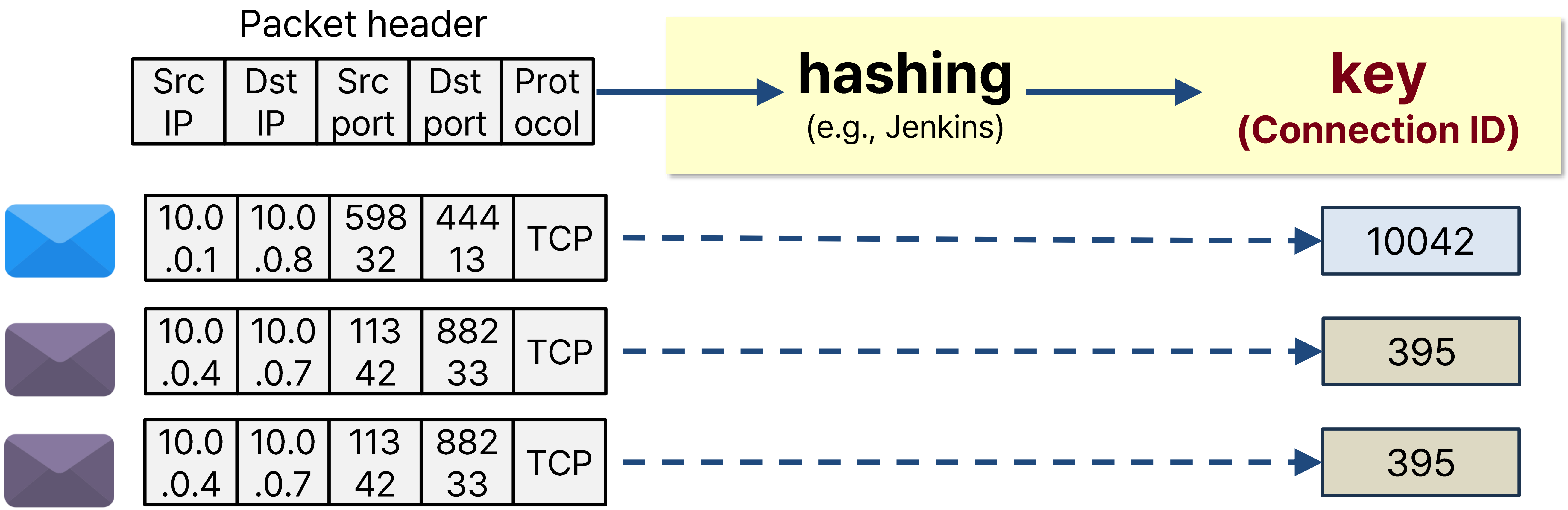
Traffic splitting at software switch has not been studied!

Background: Traffic Splitting Mechanism

- Two-stage process: ① Packet classification → ② Path selection

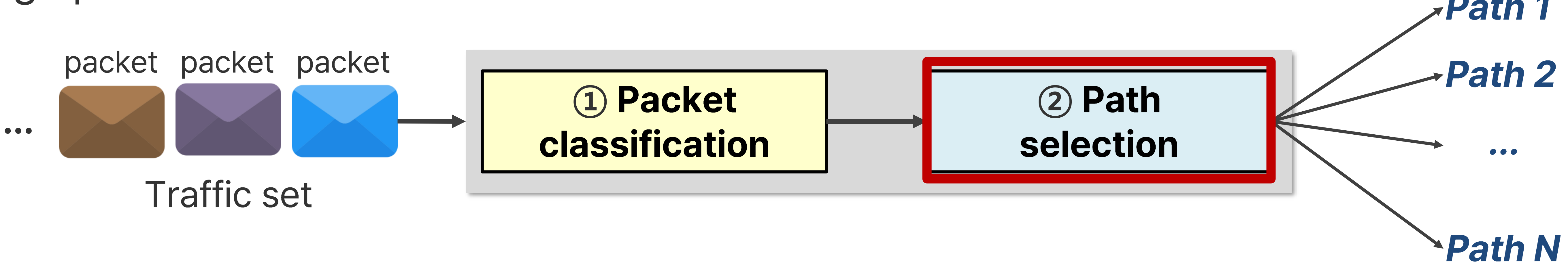


- **① Packet classification:** Identify network connection to which incoming packets belong
- Use a **hashing** on the **packet header`s 5-tuple** to determine "connection ID" (key)
- Ensure packets from the same connection follow the same path to **prevent out-of-order** packets

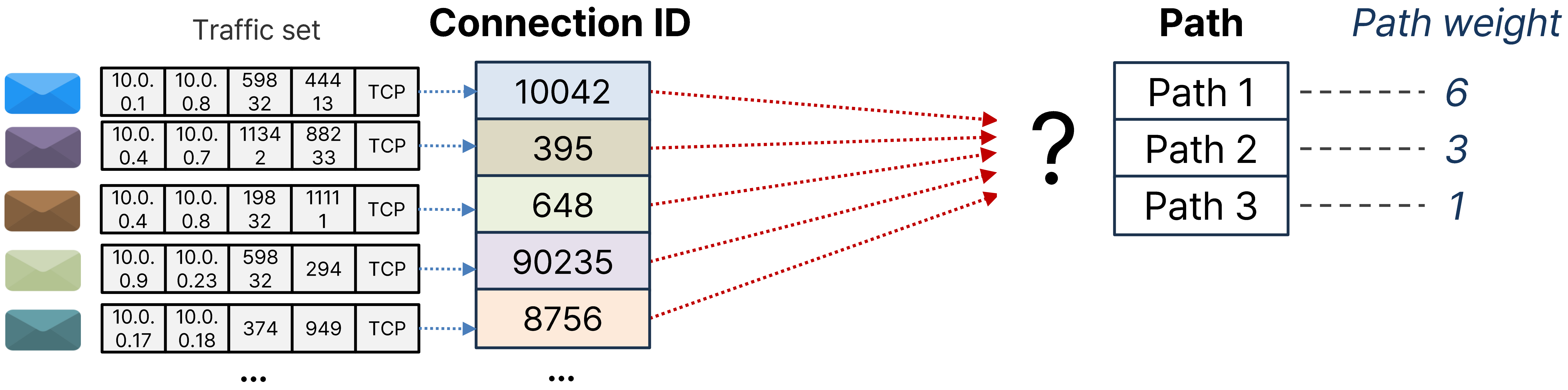


Background: Traffic Splitting Mechanism

- Two-stage process: ① Packet classification → ② Path selection



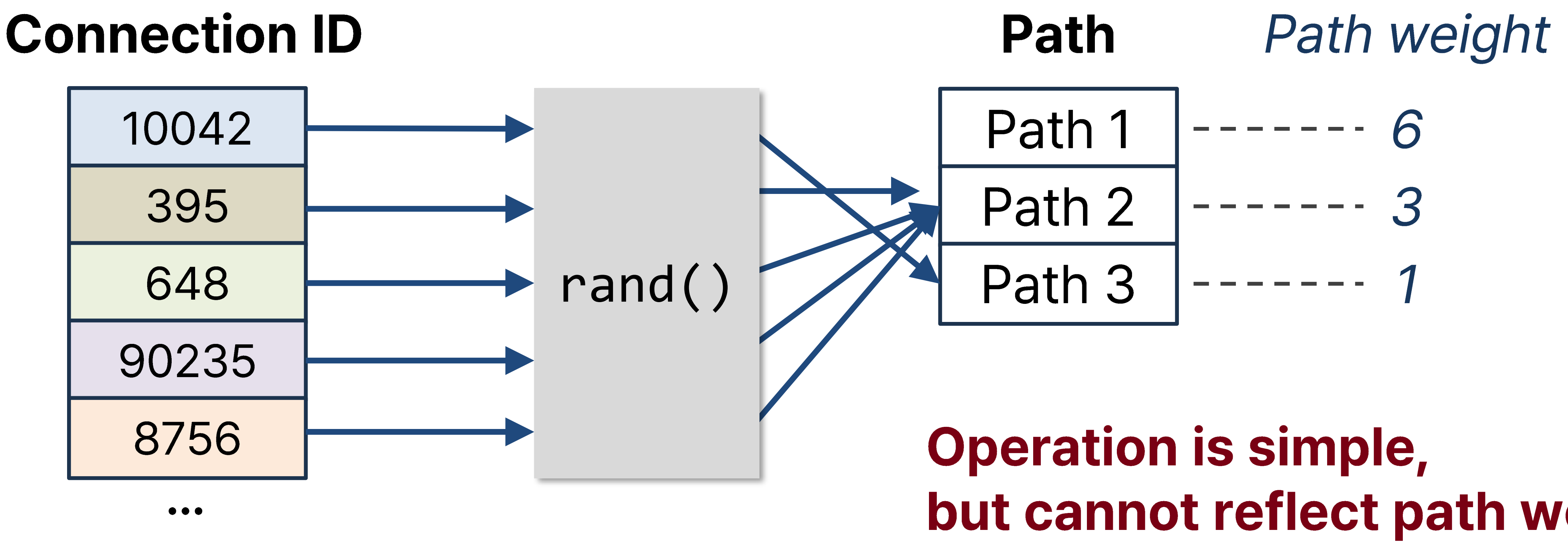
- **② Path selection:** assign a path to each connection considering path weights



- **<u>Four techniques</u>** used in software switches:
  - 1) random, 2) weighted round-robin (WRR), 3) weighted cost multipath (WCMP), 4) scoring

Path Selection: (1) Random

- Path is determined using a **random** distribution
- Connection ID serves as the random seed

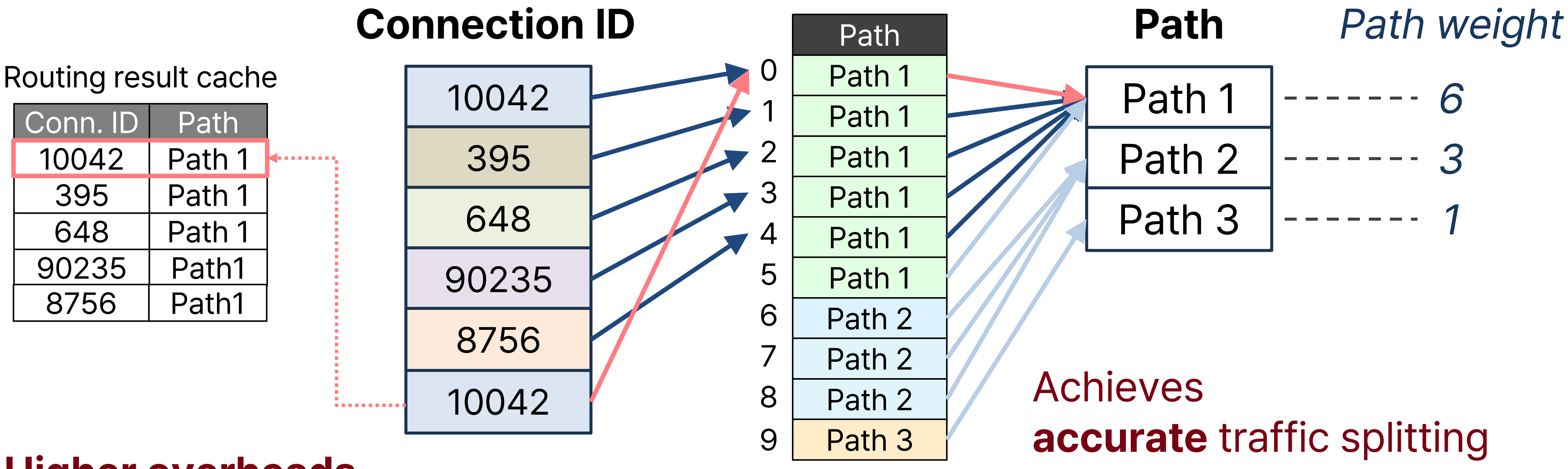


Operation is simple,
but cannot reflect path weight!

E.g., ECMP generally uses random

Path Selection: (2) WRR

- Select paths sequentially based on weights (round-robin manner)
  - **Weighted multipath table:** Contains multiple entries per path as much as weights (memory ↑)
  - **Routing result cache:** Stores routing decisions to prevent packet reordering issues (time complexity ↑)

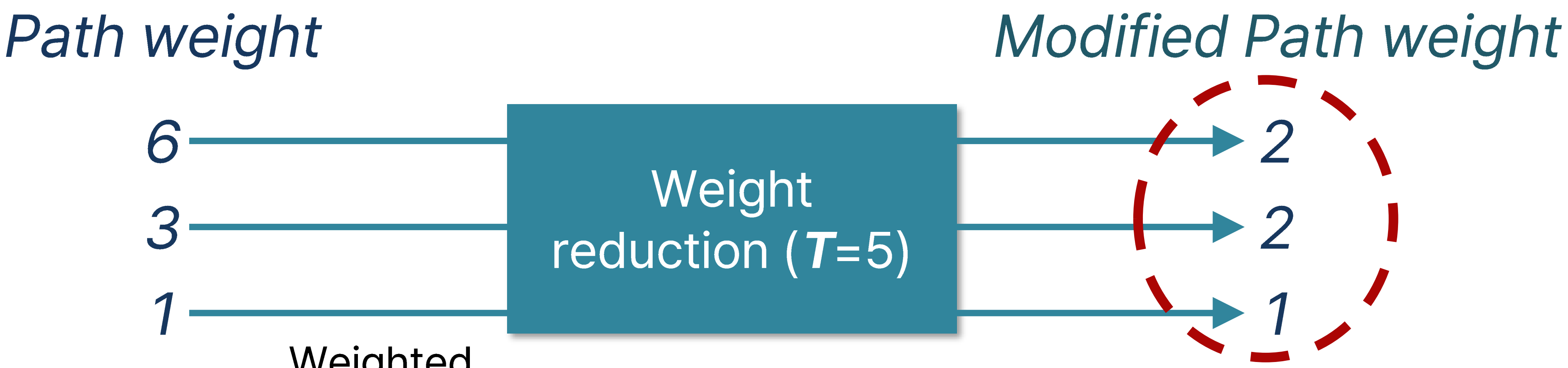


Achieves
accurate traffic splitting

Higher overheads
due to two additional structures

Path Selection: (3) WCMP

- Resource-efficient alternative to WRR
  - WRR becomes computationally heavier as number of path weights and connections increases
- **Weight reduction*:** change the weights into smaller scales under the threshold
  - E.g., 6, 3, 1 → 2, 2, 1 for threshold 5



Weighted multipath table

	Path
0	Path 1
1	Path 1
2	Path 1
3	Path 1
4	Path 2
5	Path 2
6	Path 2
7	Path 3
8	Path 3

Weighted multipath table

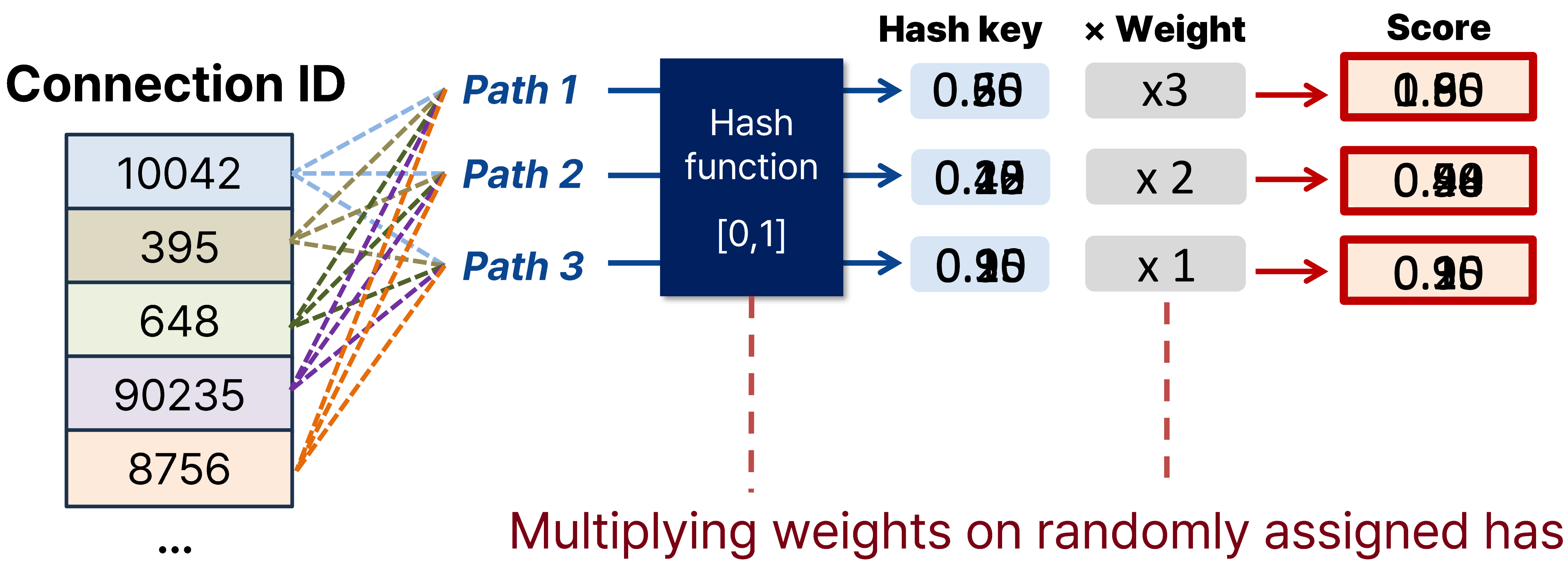
	Path
0	Path 1
1	Path 1
2	Path 2
3	Path 2
4	Path 3

Reduce overheads of WRR, but **distorts path weights**

*Junlan Zhou et al. 2014. WCMP: Weighted cost multipathing for improved fairness in data centers. In Ninth European Conference on Computer Systems. 1–14.

Path Selection: (4) Scoring

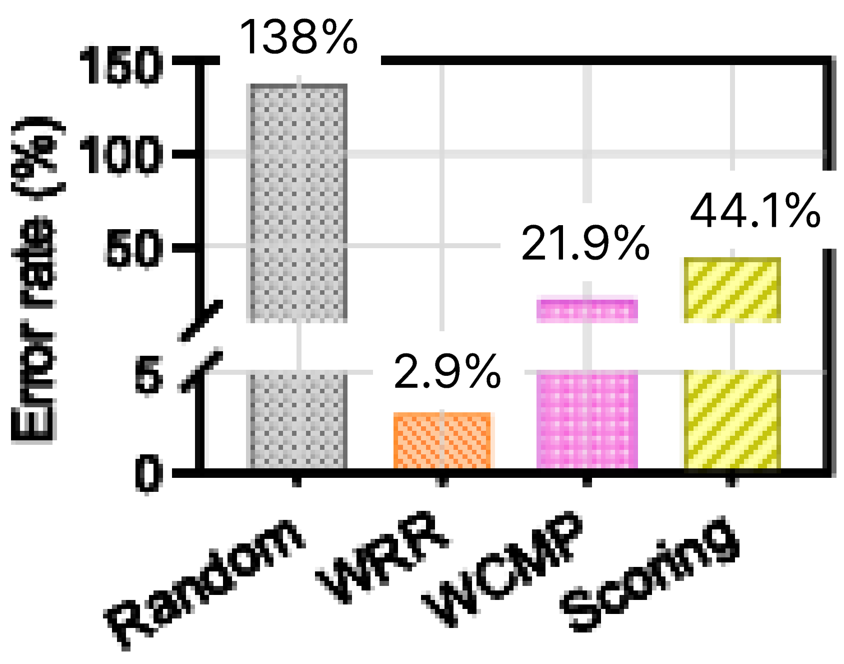
- De-facto technique
- For each connection, **examine all paths` score → select the one with the highest score**
  - **Score** calculation: hash key of (connection ID, path ID) × path weight



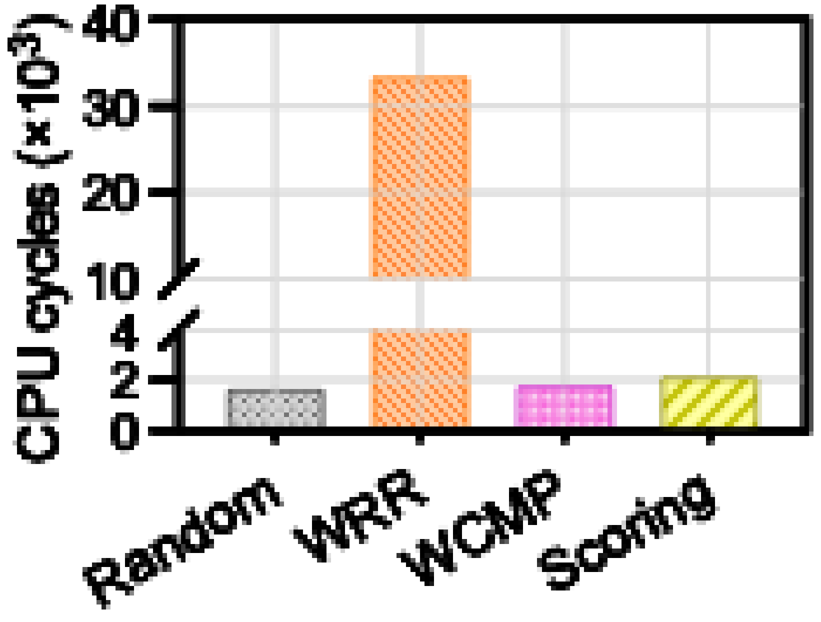
Multiplying weights on randomly assigned hash results
→ Reflecting path weights, but cannot satisfy weight exactly (known as black-box)

Problem: Inaccuracy and Resource-inefficiency

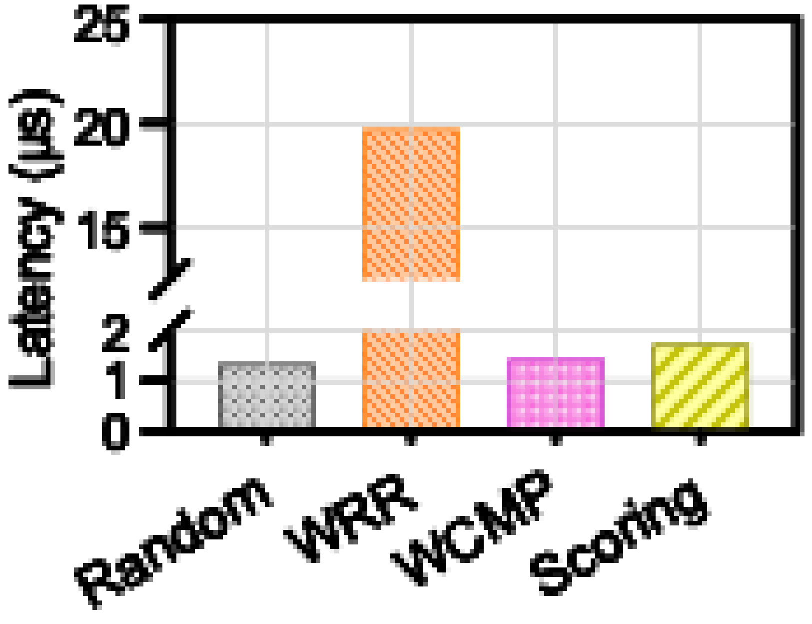
- Measure accuracy and resource-efficiency (CPU cycles and per-packet latency)
  - Dataset: Real-world DC traces from CAIDA* and ClassBench**
  - 200 trials with varying path weights
- **Accuracy:** Error rate (%) between actual and ideal connection counts per path
- **Resource-efficiency**: CPU cycles and per-packet latency



Accuracy (error rate)



CPU cycles



Per-packet latency

	Accuracy	Resource-efficiency
Random	X	O
WRR	O	X
WCMP	X	O
Scoring	X	O

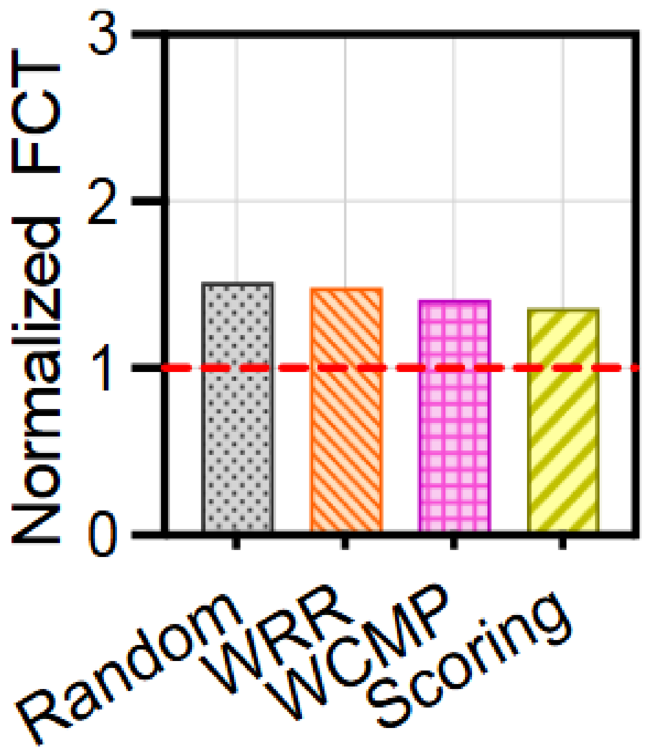
No technique achieves both high accuracy and resource-efficiency!

* CAIDA: The CAIDA UCSD Anonymized Internet Traces. https://www.caida.org/catalog/datasets/passive_dataset
** ClassBench: Jiří Matoušek, et. al. 2017. ClassBench-ng: Recasting `ClassBench after a decade of network evolution. In 2017 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS). IEEE, 204–216.

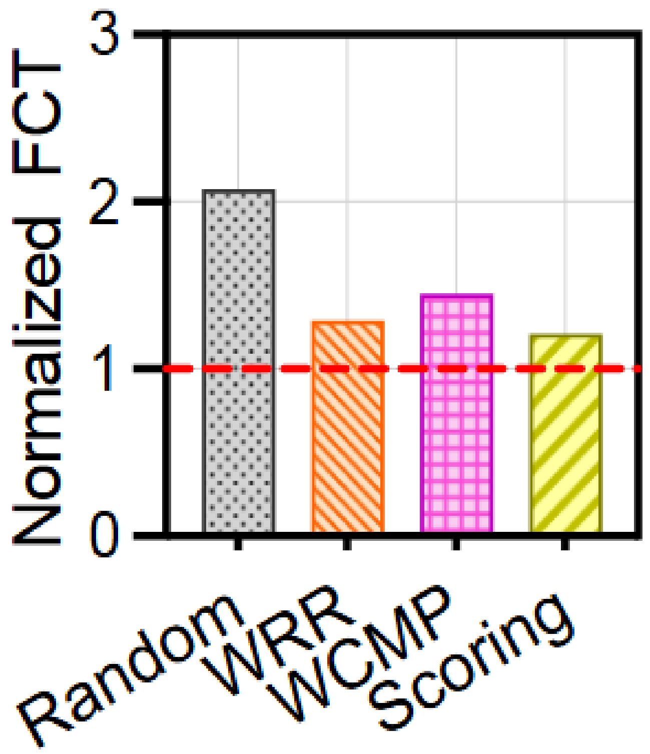
Lead to Poor DC Networking Services

- Inaccuracy ↑→ Leads to path overloading, resulting in slower speeds and packet retransmissions
- Per-packet overhead ↑→ Accumulates across thousands of packets, increasing total service latency

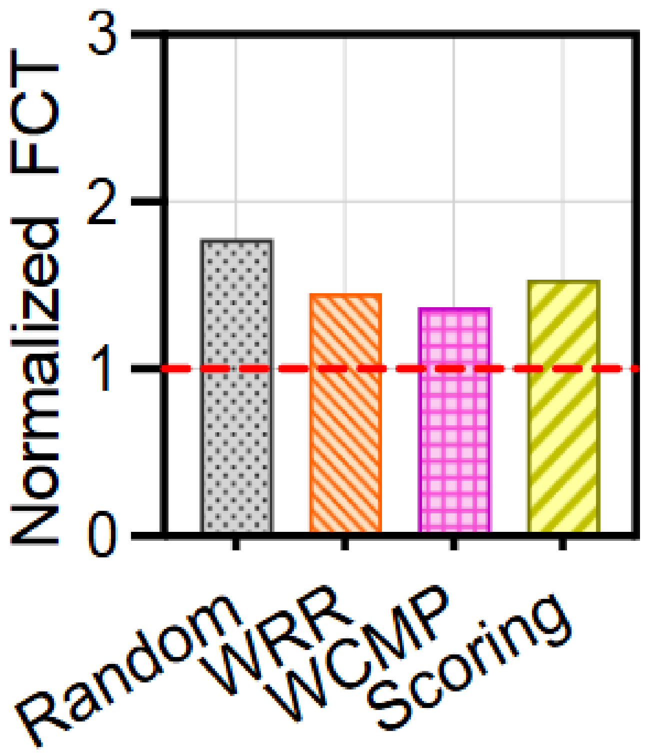
- **Our experiment: Significantly delays DC services** (~2.8× compared to ideal*)



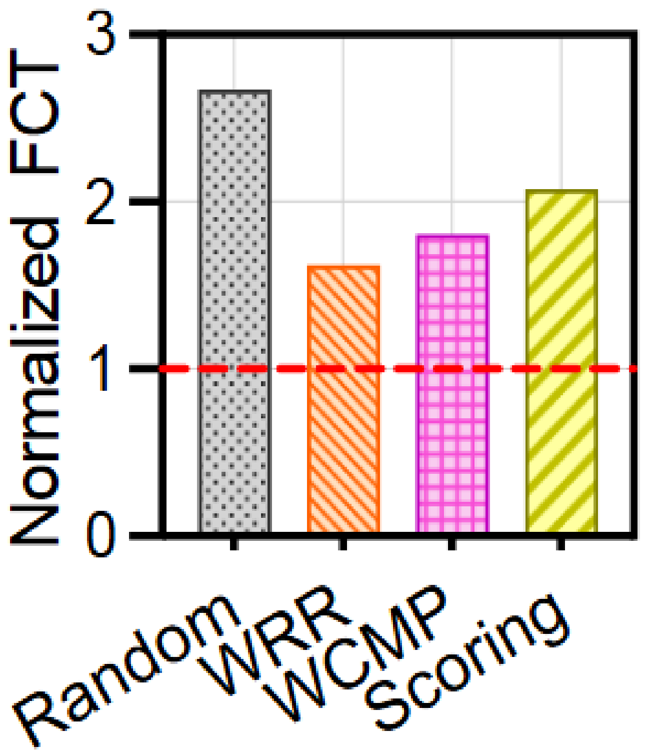
Web search



Data mining



Deep learning

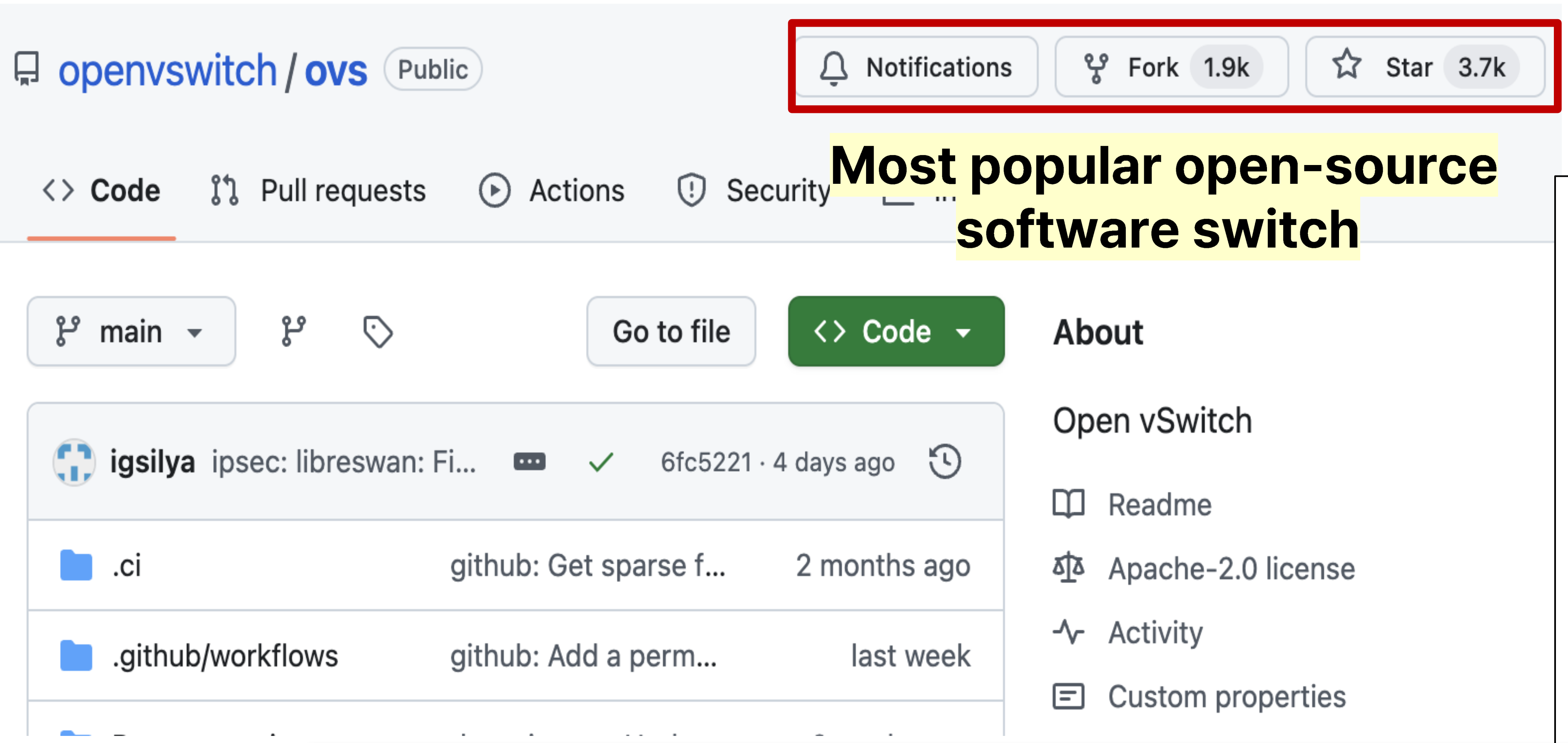


In-memory cache
(Twitter)

* Ideal: factitious scenario where software switches perform perfectly and efficiently accurate traffic splitting

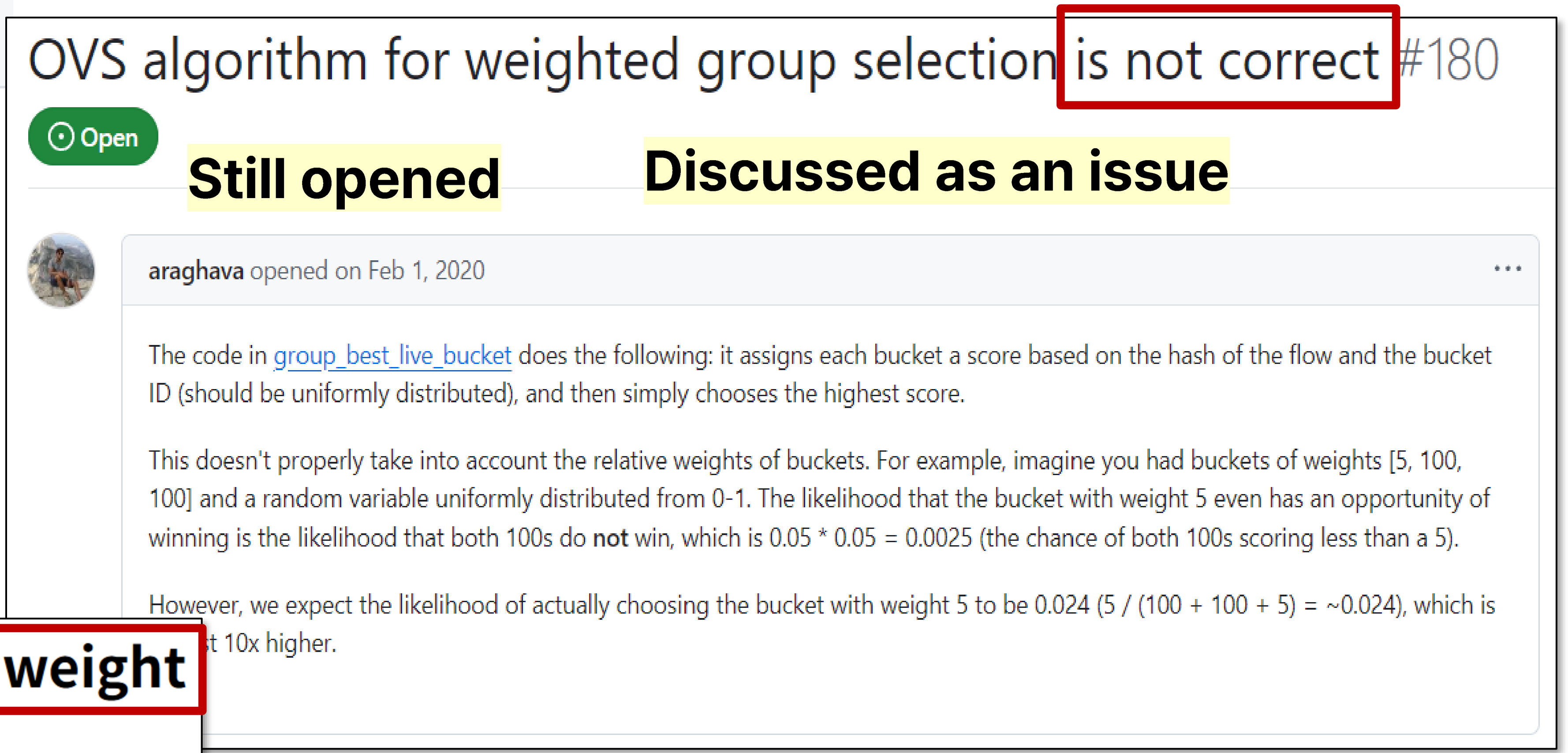
Widely Recognized Issue in Practice

- The problem has been widely known in the open-source community, **<u>no solution exists yet</u>**
  - Inaccuracy of **<u>scoring</u>** is treated as black-box



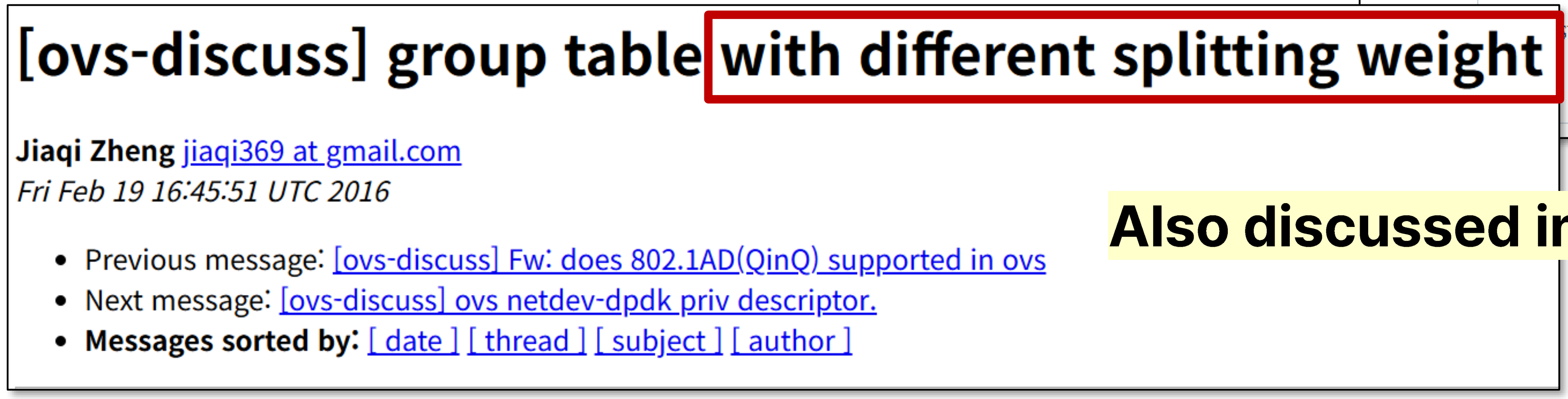
Most popular open-source software switch

Open vSwitch



Still opened

Discussed as an issue



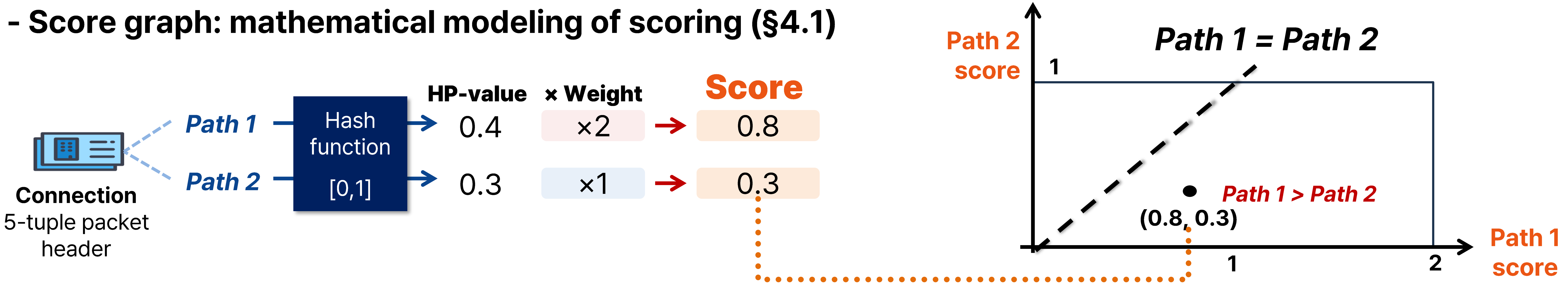
Also discussed in mailing list

Propose VALO: New Traffic Splitting Mechanism

Our Goal: achieves both high accuracy and resource-efficiency (by improving scoring)

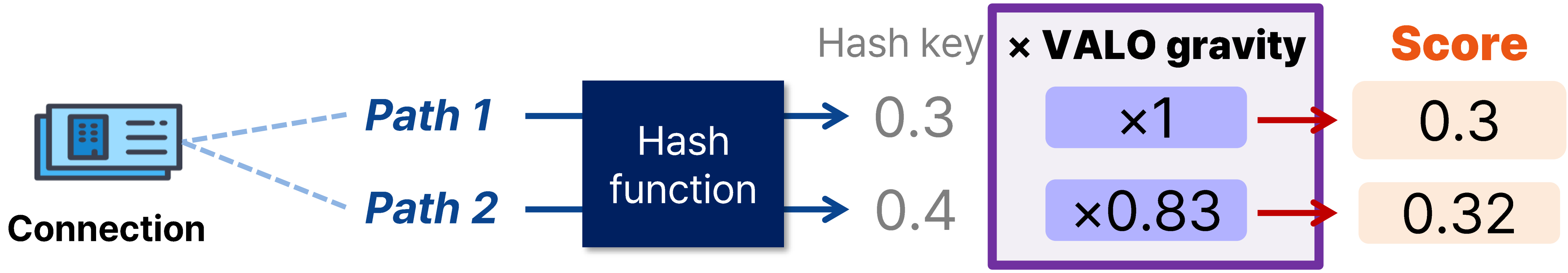
1. Modeling of scoring to identify root causes of its inaccuracy

- Score graph: mathematical modeling of scoring (§4.1)



2. Devise VALO, incorporating its novel parameter, "VALO gravity"

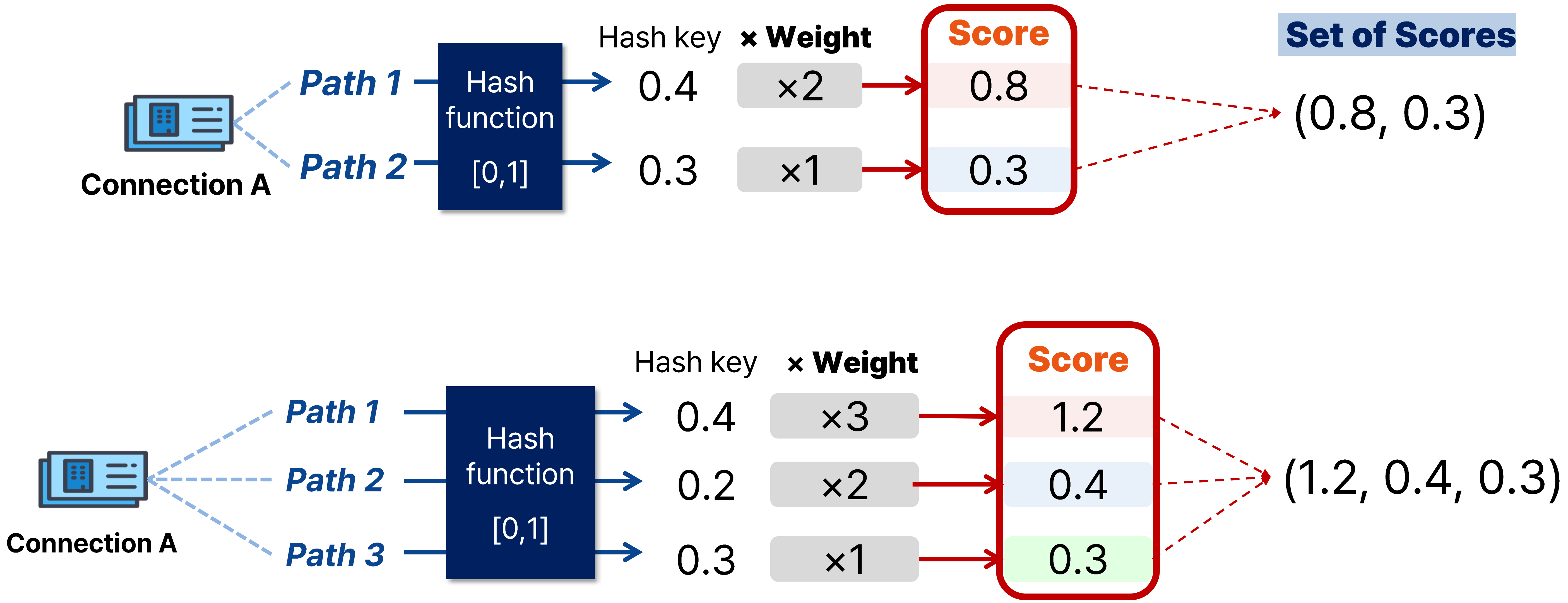
- Formulate resource-efficient method for calculating VALO gravity (§4.1)



$$\frac{x_i}{x_1} = \prod_{k=2}^{i}\left(\frac{kw_k + w_{k+1} + \cdots + w_n}{(k-1)w_{k-1} + w_k + \ldots + w_n}\right)^{\frac{1}{k-1}}$$

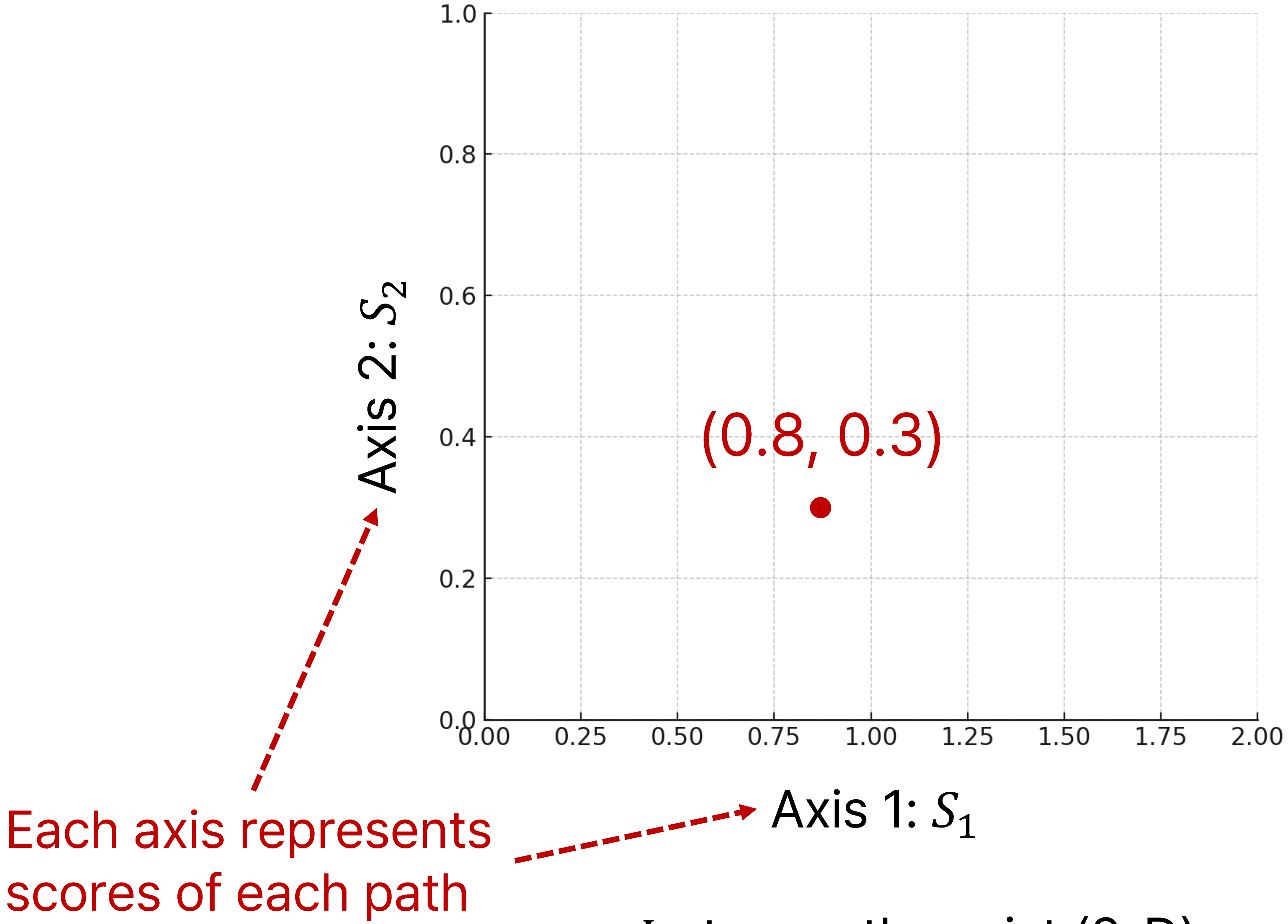
Mathematical Modeling of Scoring

- Scoring calculates scores for all paths
- Each connection has *N* scores corresponding to *N* possible paths

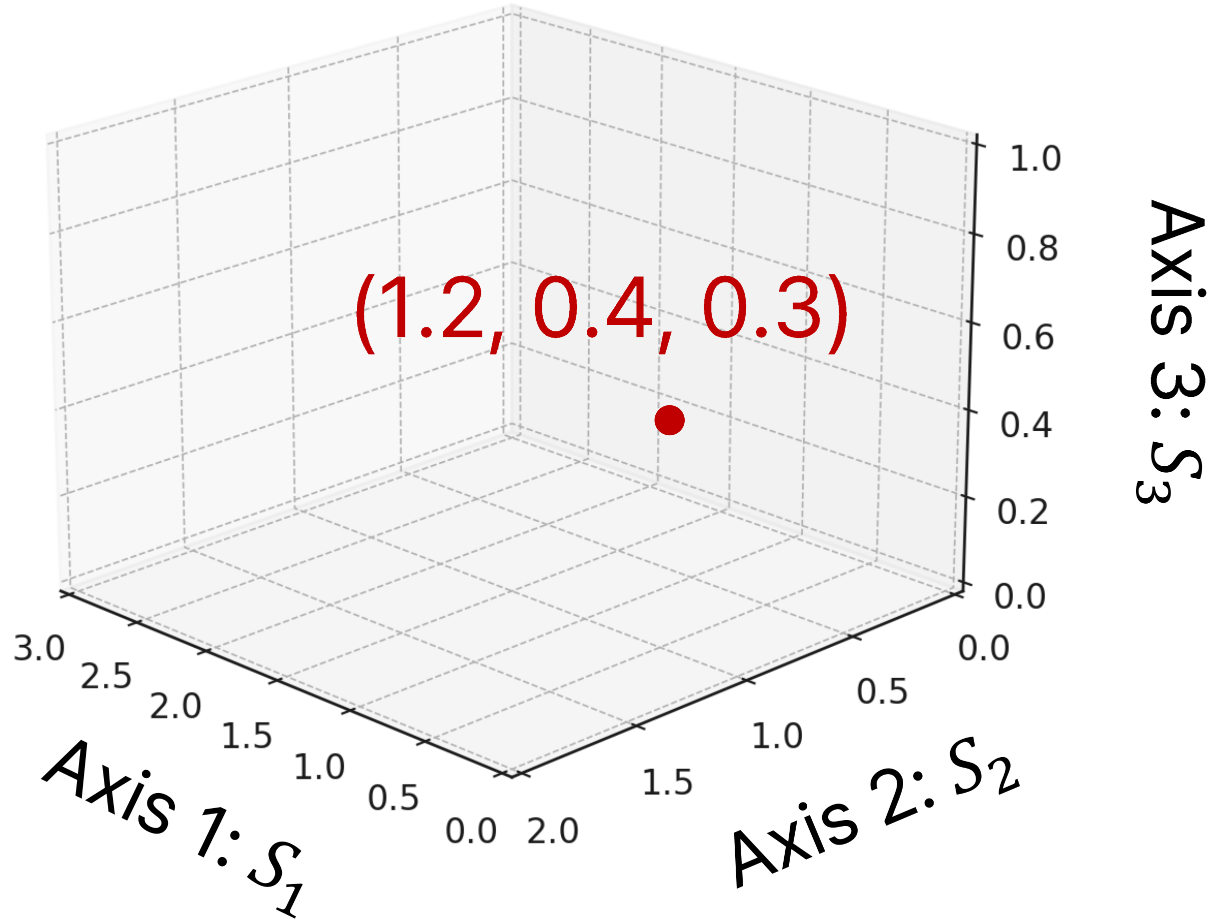


Each connection as a Coordinate

- Represent a connection by scores assigned to each path, denoted as s_i (score for path i): $(s_1, s_2, \ldots, s_n)$
  - E.g., (0.8, 0.3) or (1.2, 0.4, 0.3)
- *N*-dimension Coordinate space:



I_2: two paths exist (2-D)

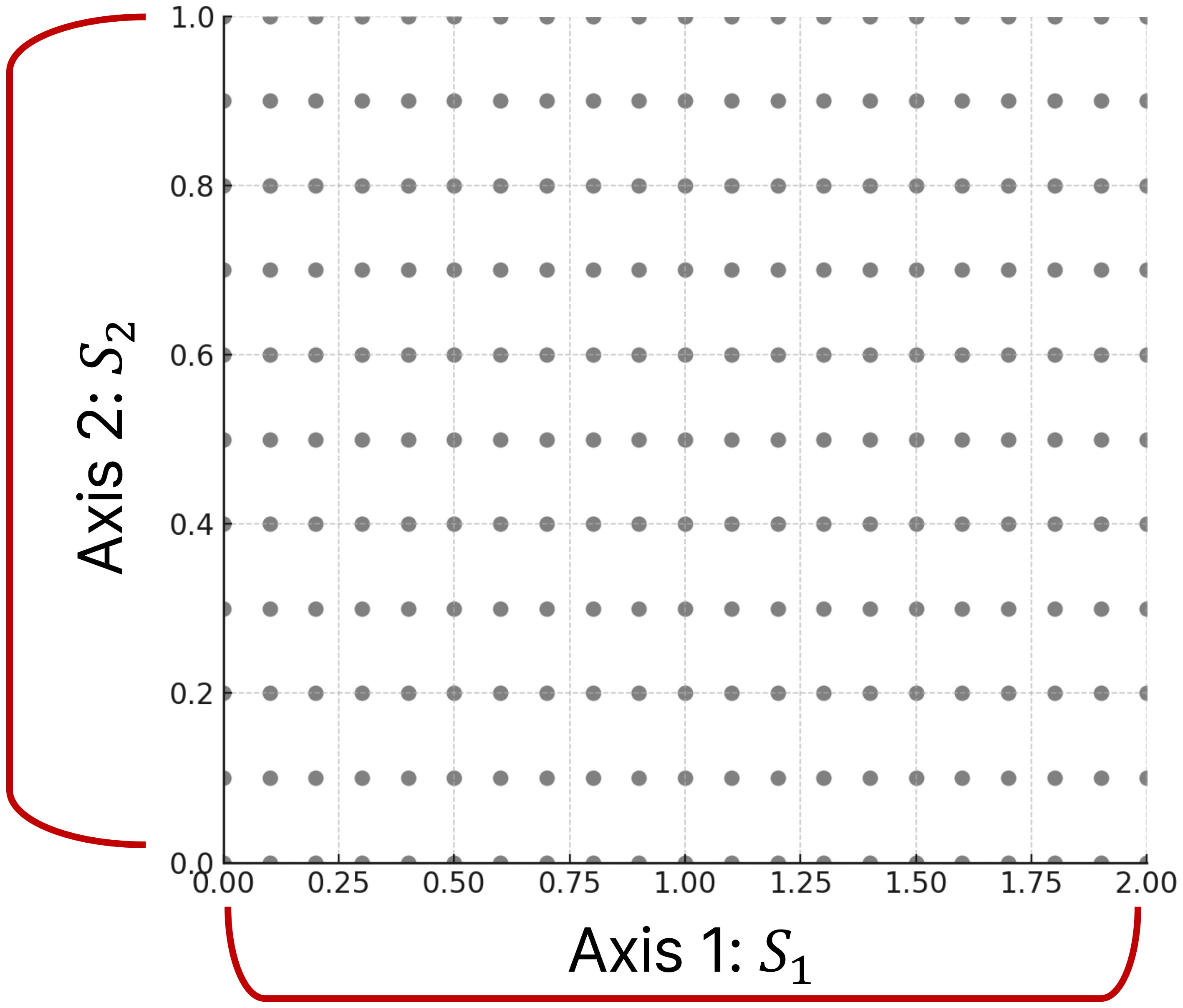


I_3: three paths exist (3-D)

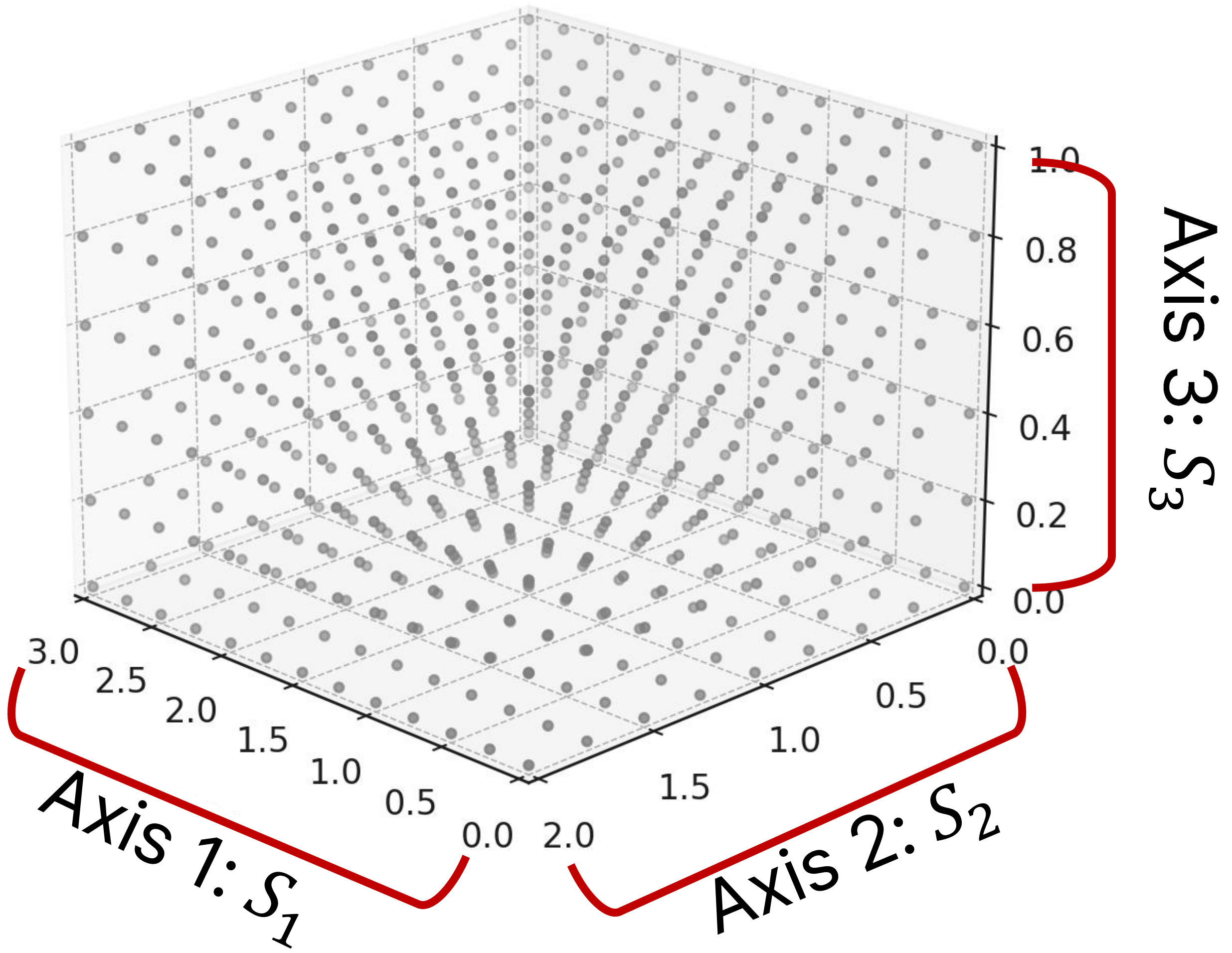
Score Graph: for All Possible Connections

- Score graph can represents all possible network connections
  - Each axis represents the range of all possible scores for each path

All possible range of scores



I_2: two paths exist (2-D)

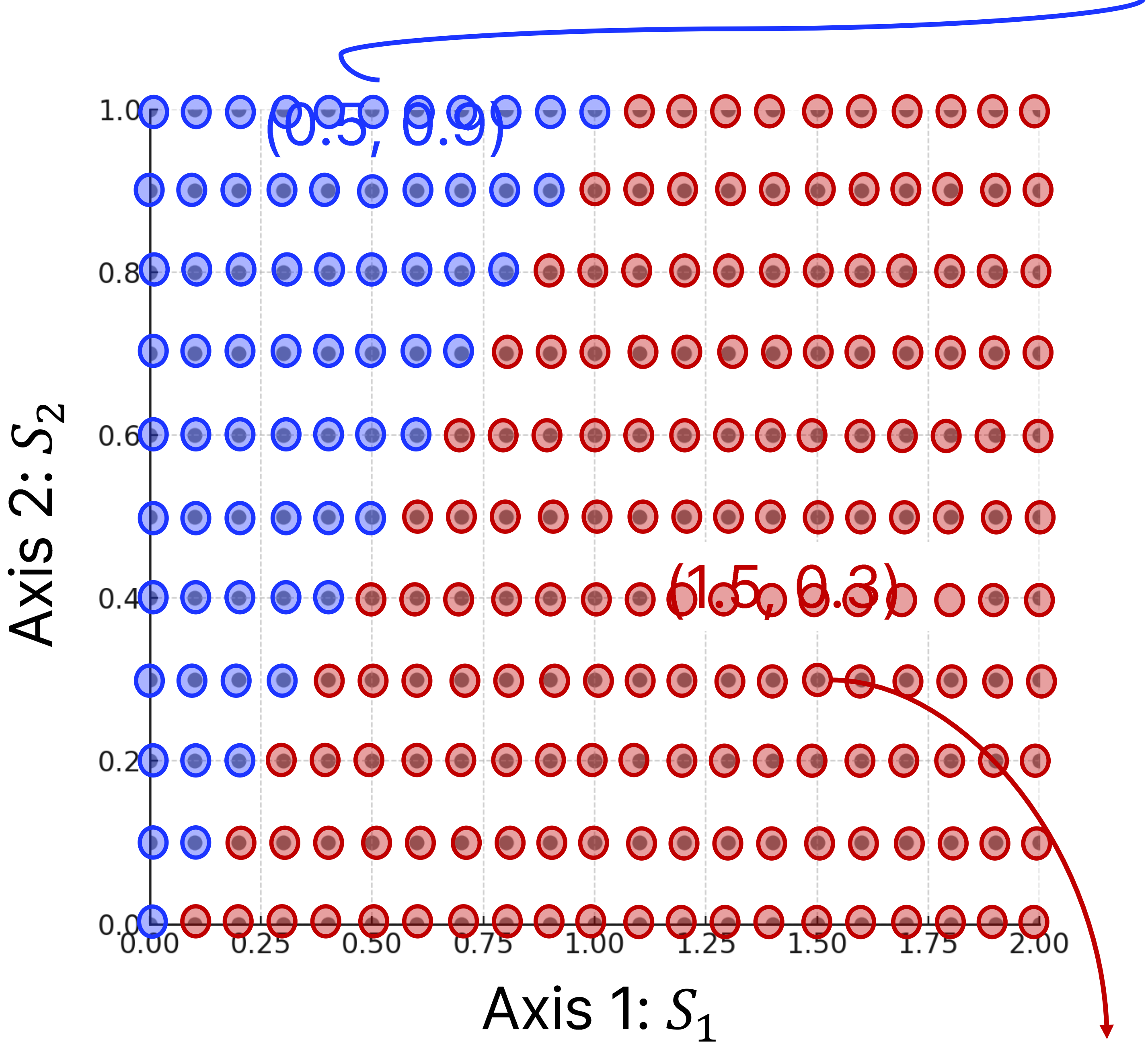


I_3: three paths exist (3-D)

Path Selection in Score Graph

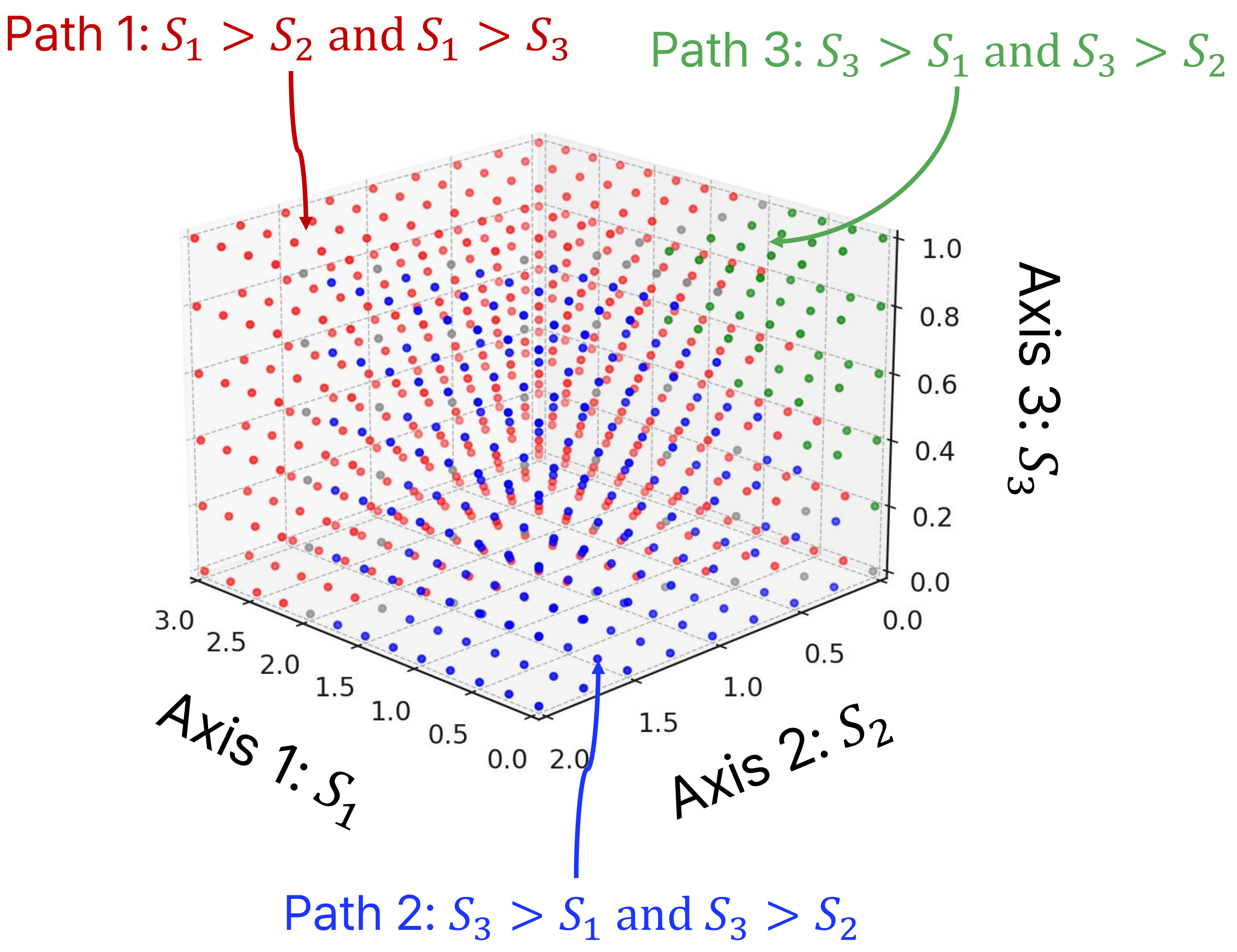
- Connections (points) are distinguished by scoring: **assigned to path with the highest score**

score of path 1 (0.5) < score of path 2 (0.9) → Select Path 2



score of path 1 (1.5) > score of path 2 (0.3) → Select Path 1

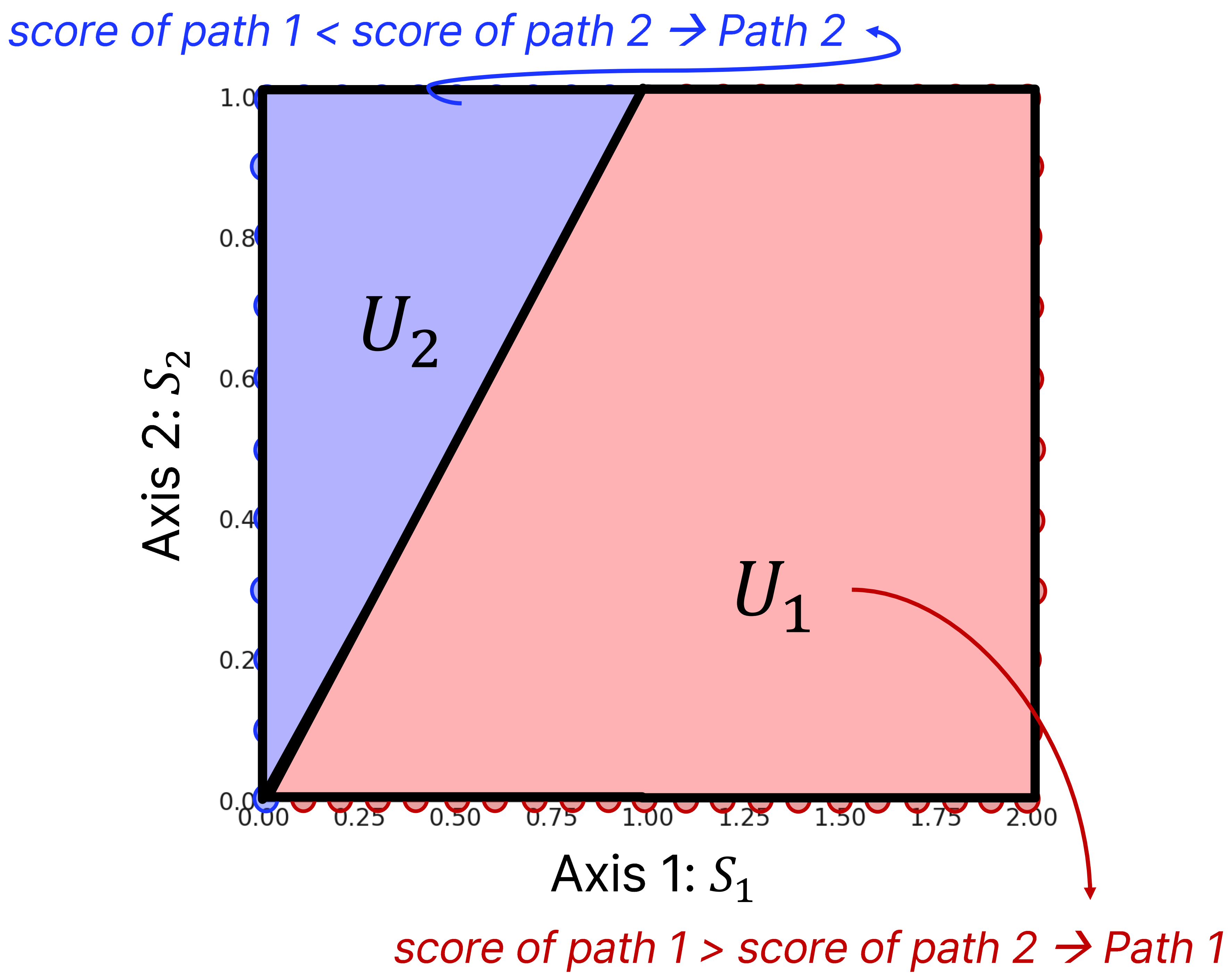
I_2:Two paths exist



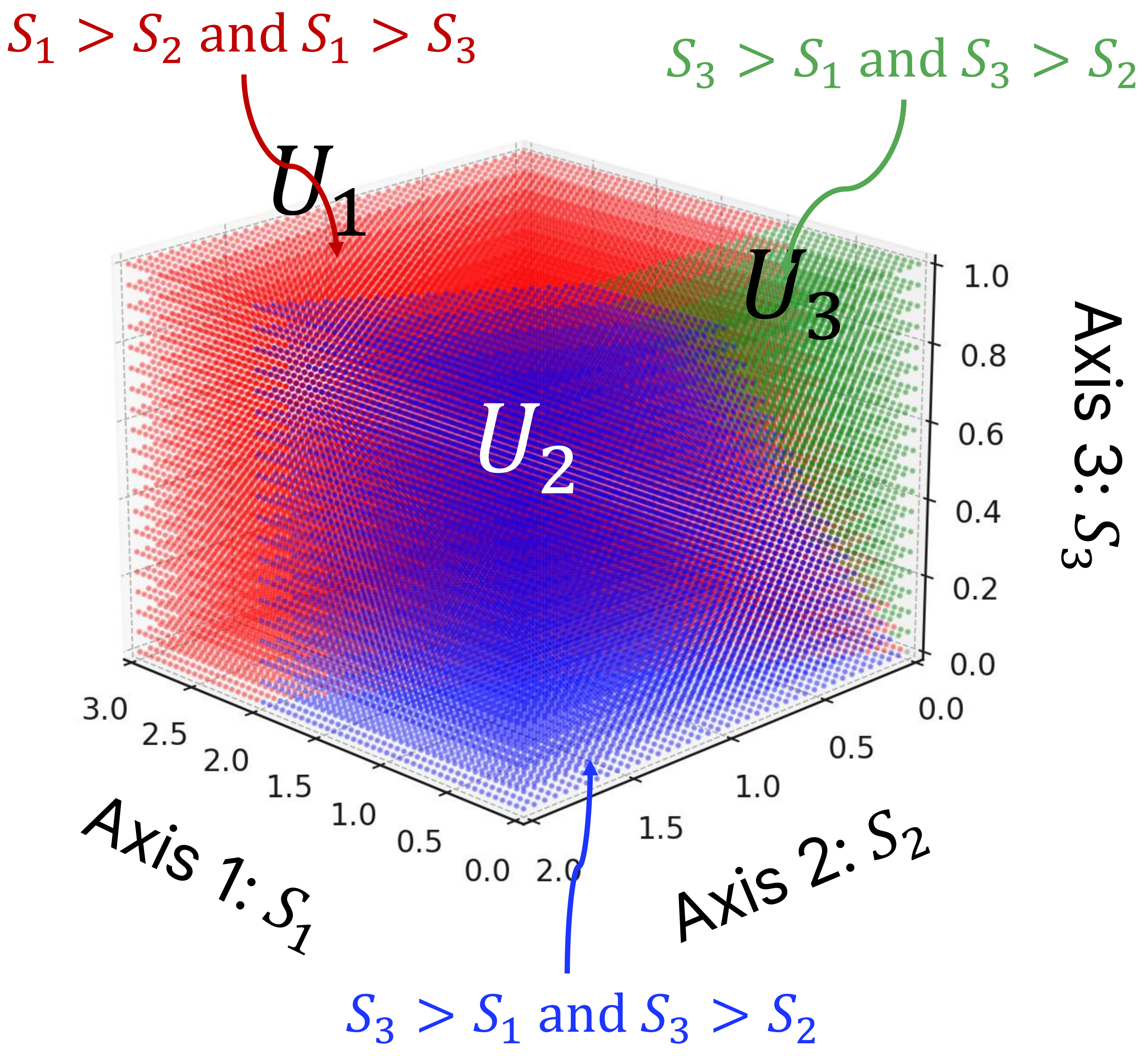
I_3: Three paths exist

Volume: The Number of Connections per Path

- Connections are divided into distinct **sub-area** (U_i), each selected path i
- **Volume: Total number of connections assigned to a path (e.g., calculated as the integral over area)**



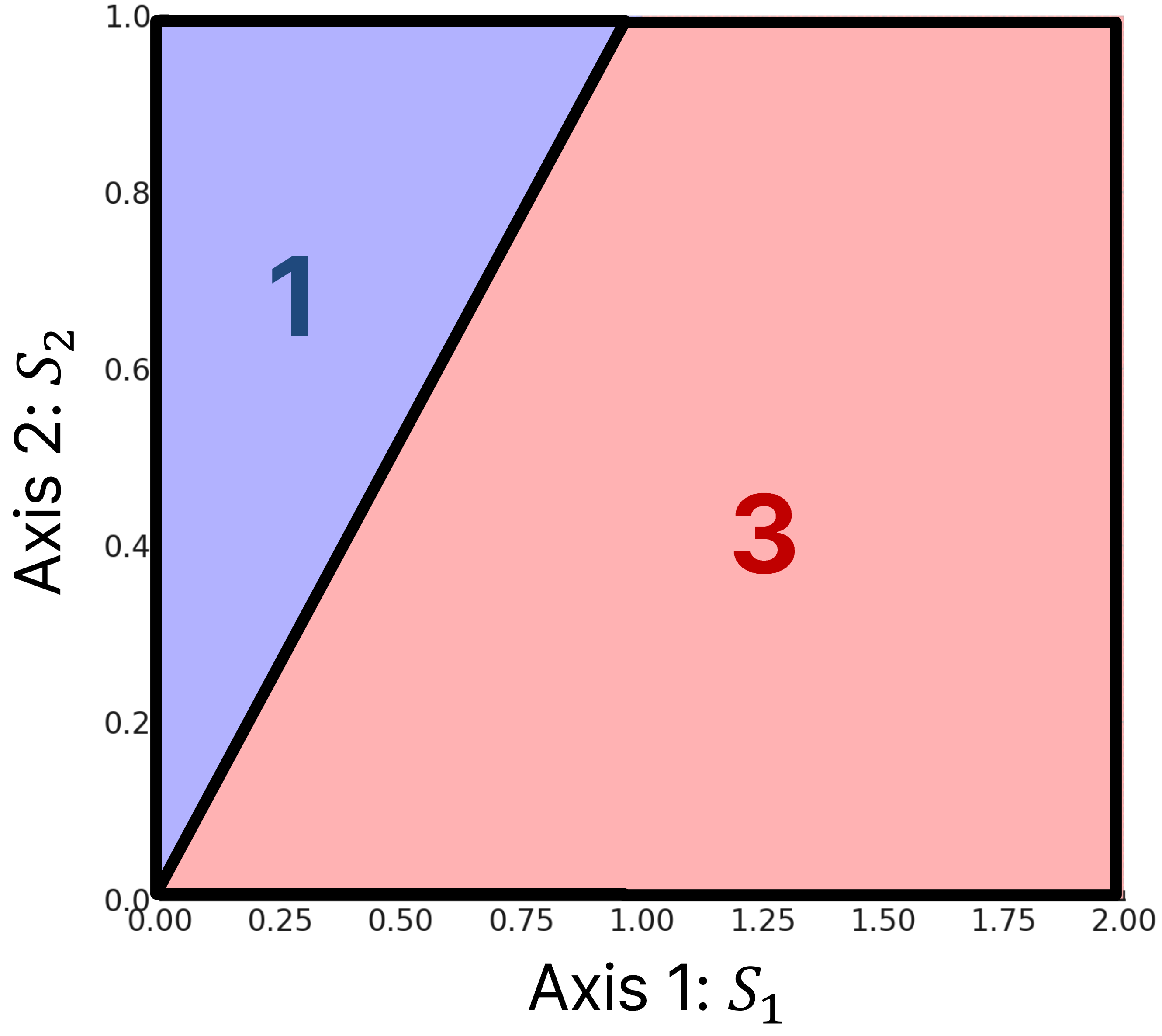
I_2: two paths exist



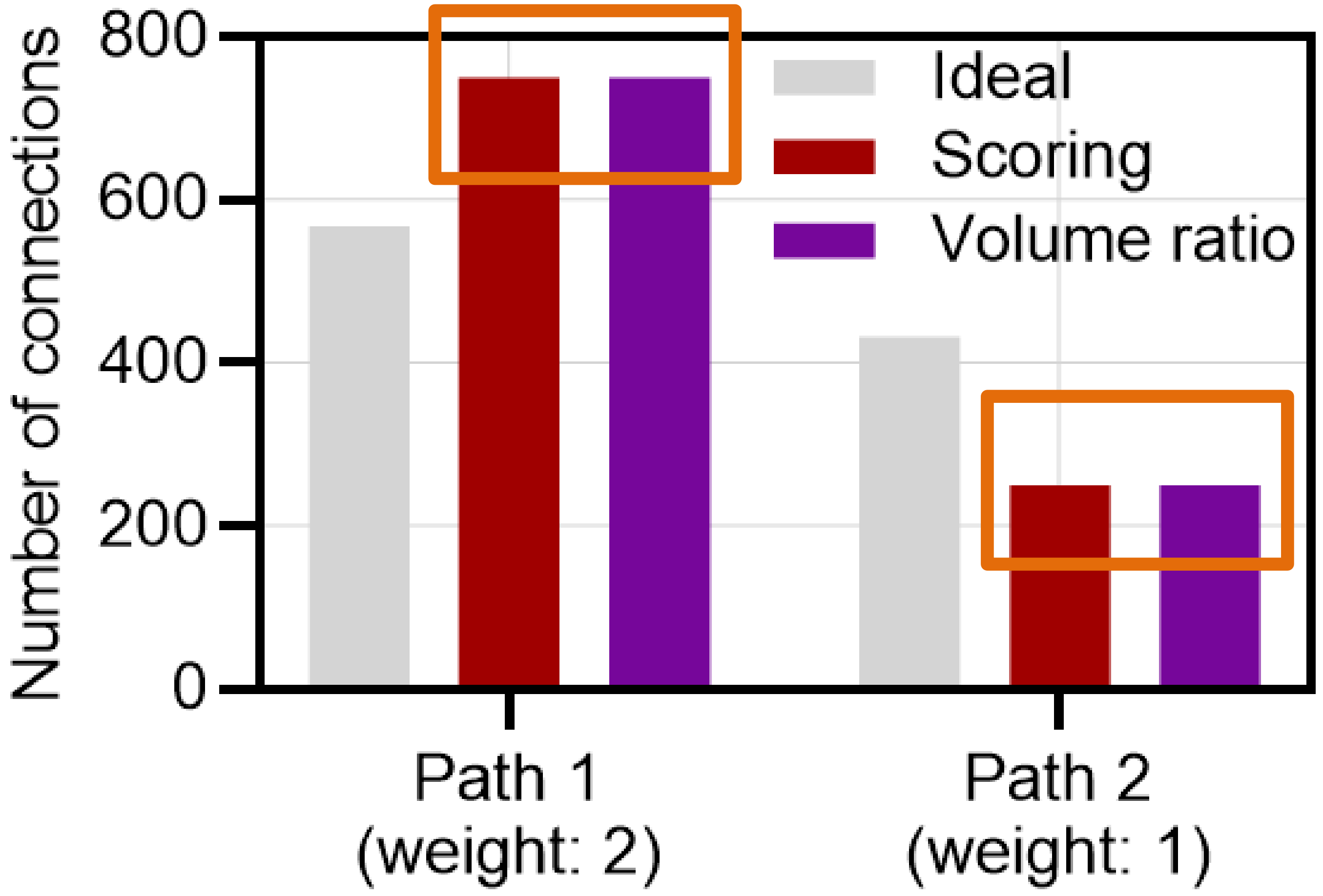
I_3: three paths exist

Mismatch between Volumes and Weights

- Given two paths with weights **2:1** → the volume **3:1 (mismatch)**
- Actual connection split is 3:1 ratio: matching volume, not weights

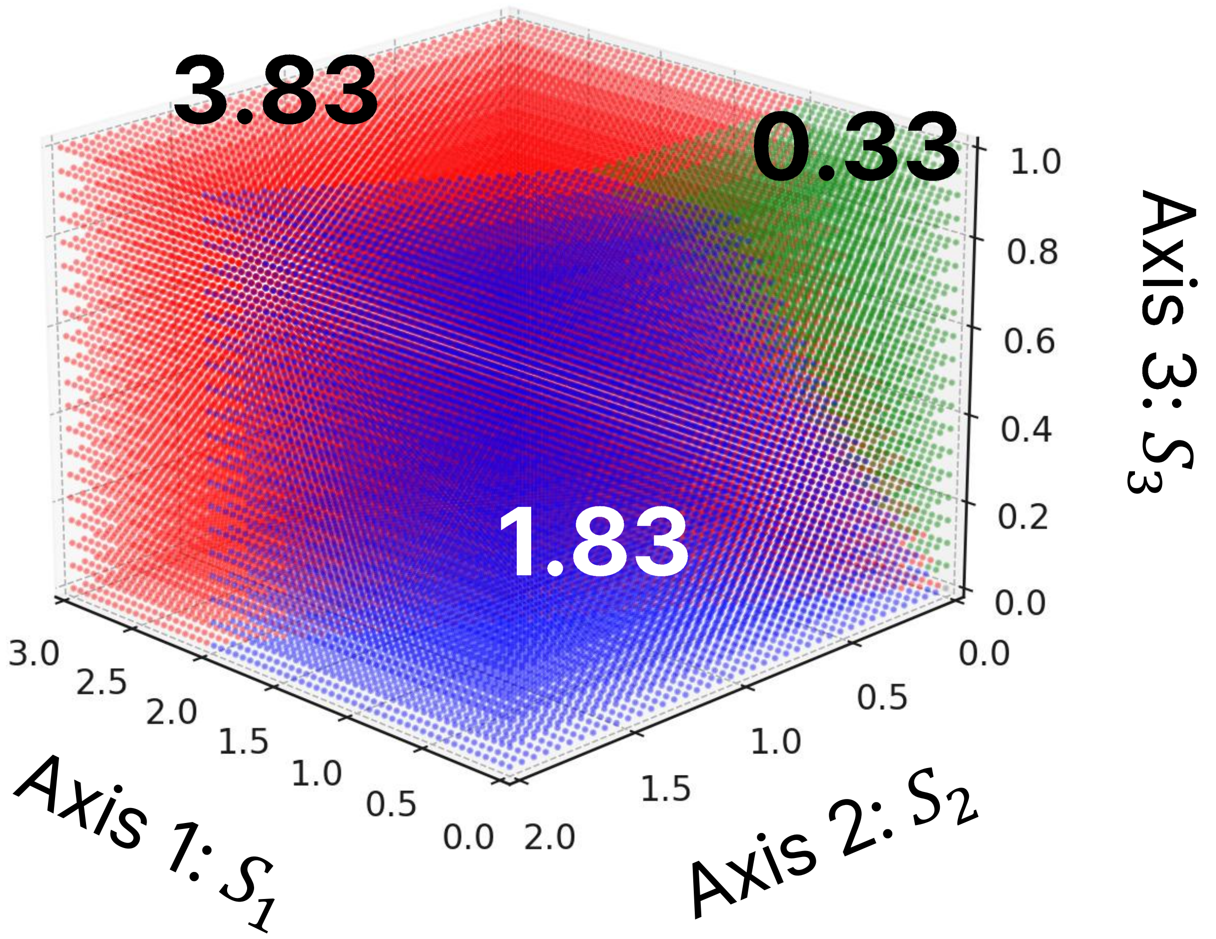


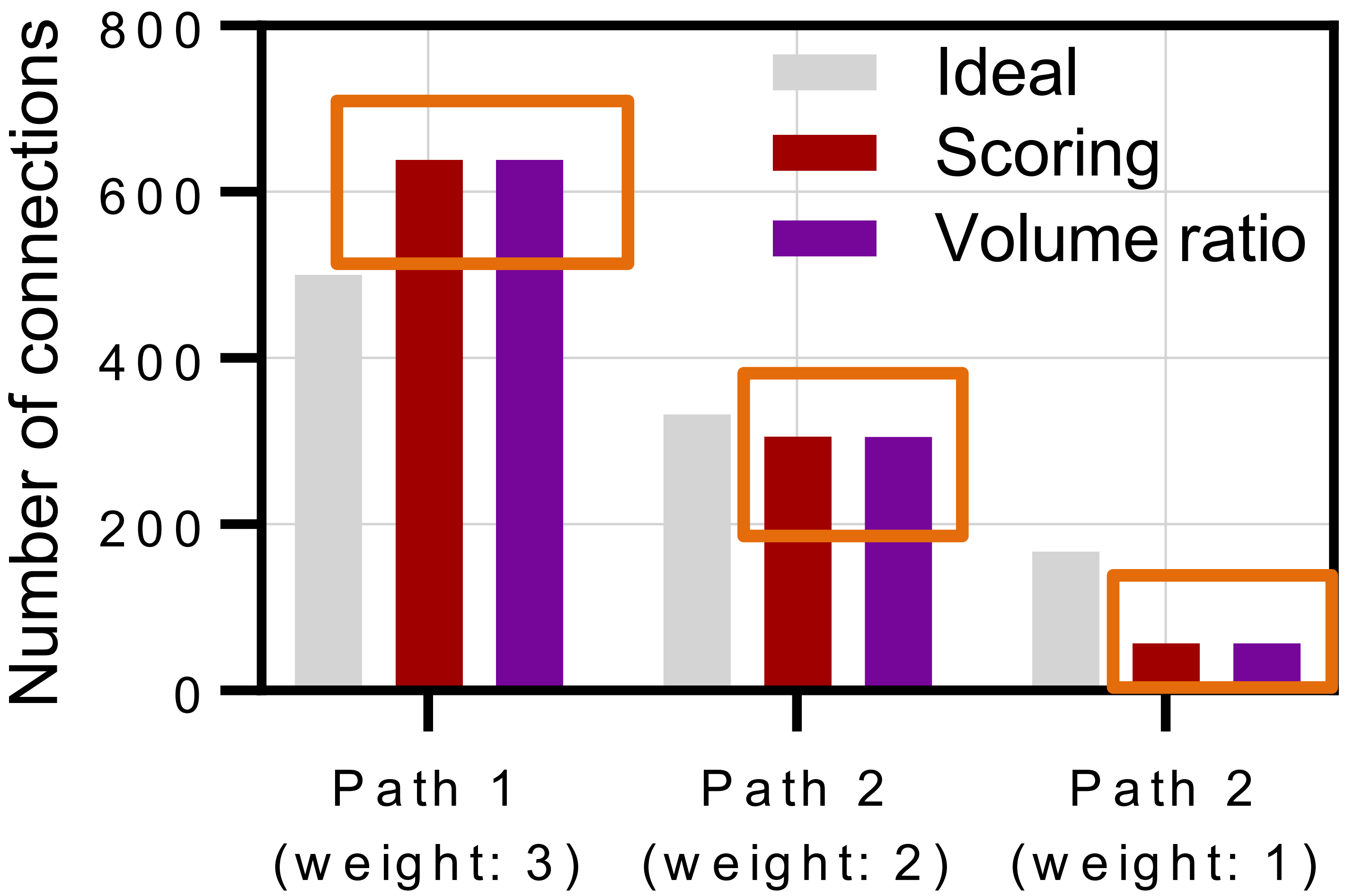
I_2: two paths exist



Mismatch between Volumes and Weights

- Similar mismatch for three paths:
  - For given weights **3 : 2 : 1,** the volume ratios becomes **3.83 : 1.83: 0.33**
  - Actual connection split observed: **3.8 : 1.8 : 0.3**

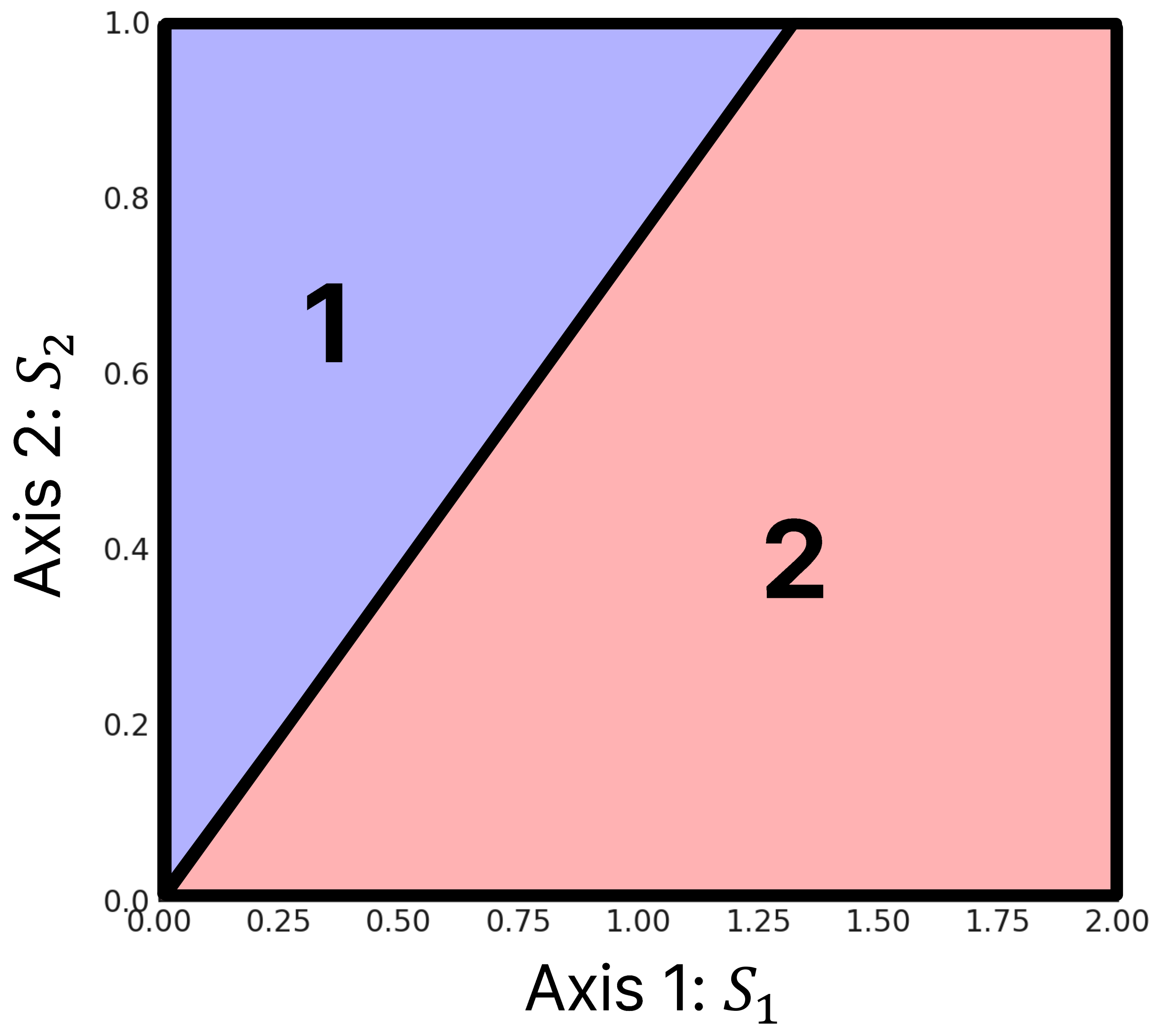




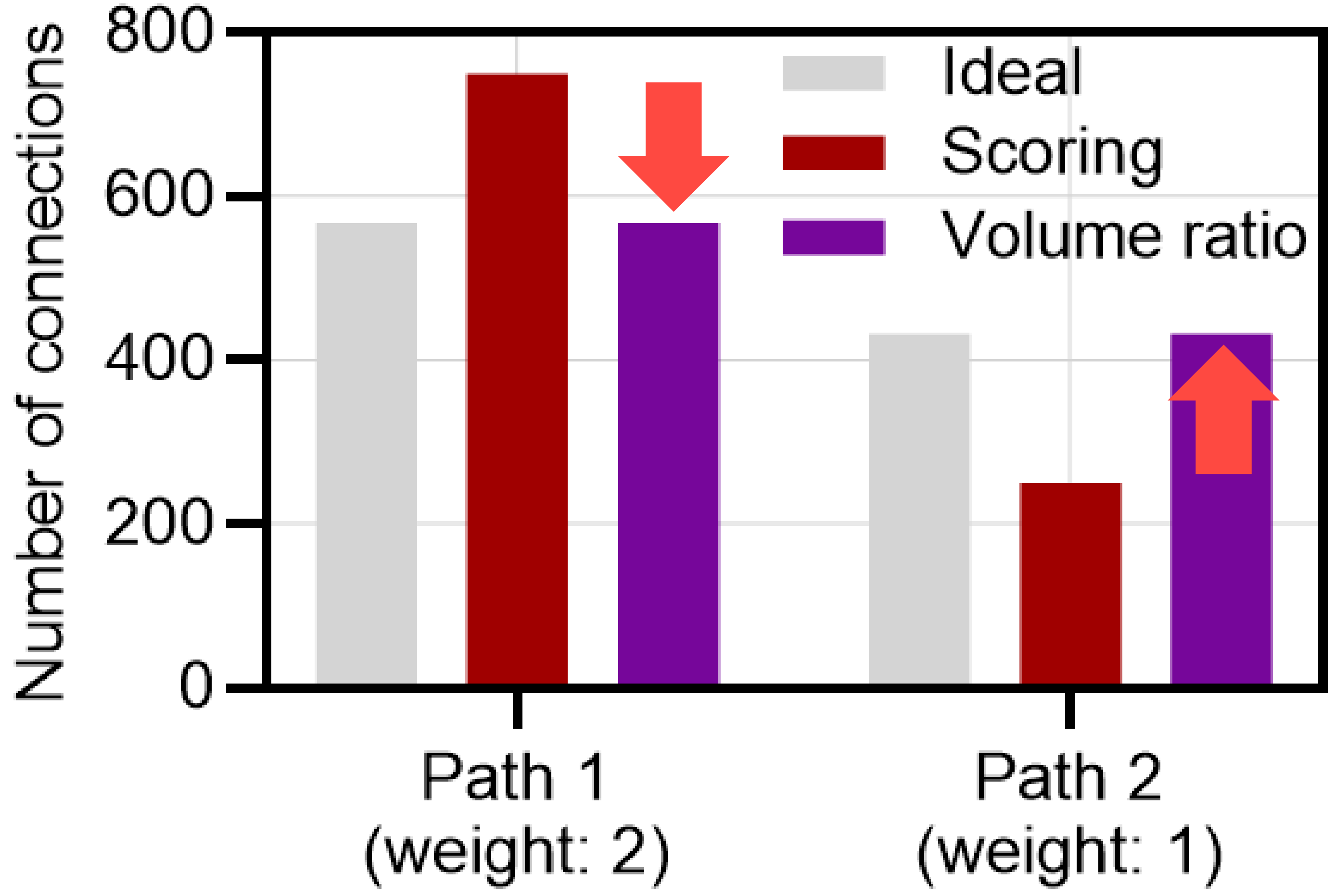
Actual splitting ratio matches volume, not weight!
: Cause of inaccuracy

Our idea: Align Volumes with Path Weights

- Our observation: actual splitting ratio matches to "**volume**"
- **Adjust sub-areas to ensure their volumes match path weights**

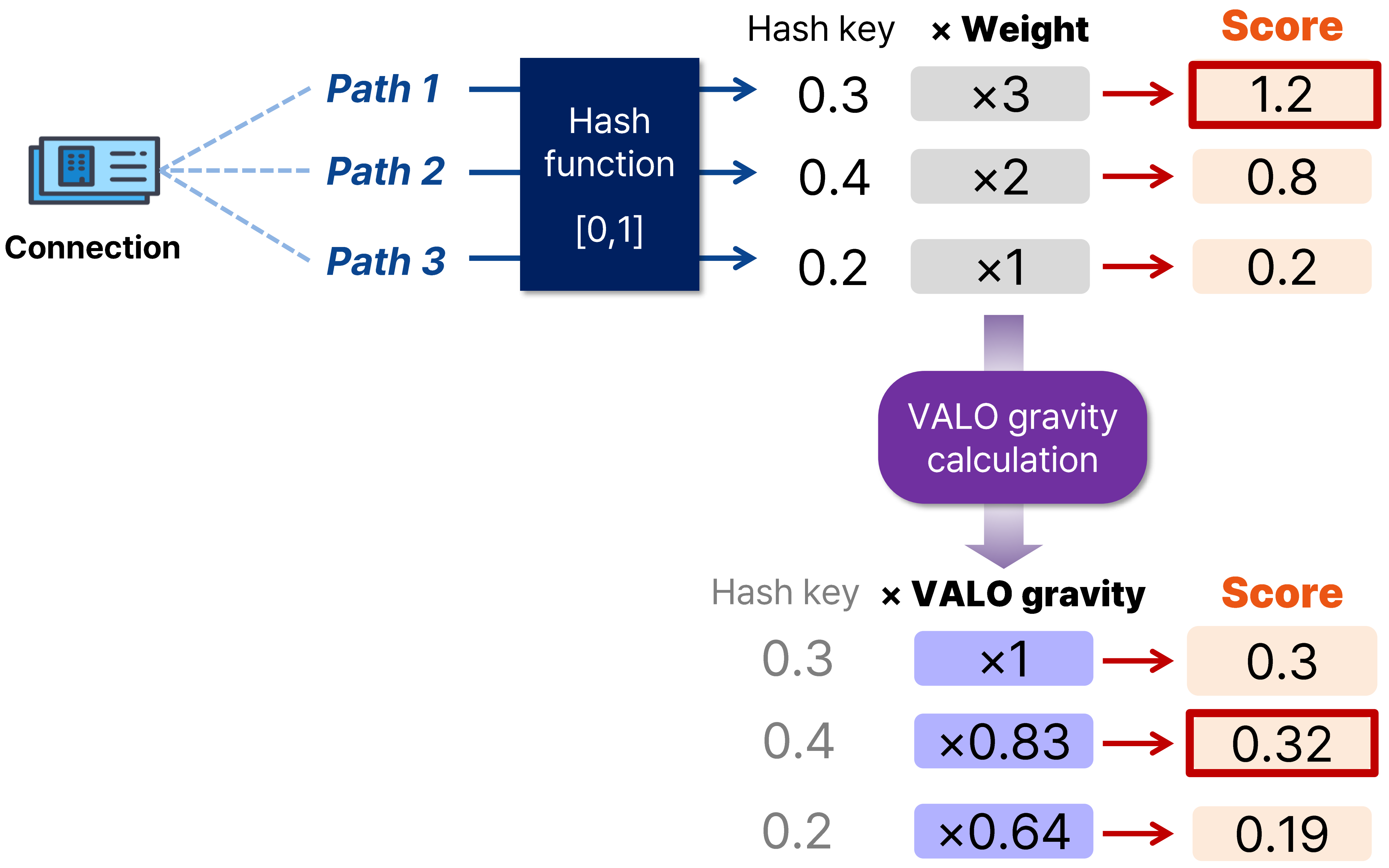


I_2: two paths exist



VALO Gravity

- Simple, but powerful parameter **to align volumes accurately with given path weights**
- Instead of directly multiplying path weights, **multiply by "VALO gravity"**



VALO Gravity Calculation

- **VALO gravity: New weight to align volumes accurately with given path weights**
- **Optimized calculation: Efficient method with low complexity**
  - Robust to higher dimension paths (e.g., 4D graph for four paths, 8D graph for eight paths)
- **Derivation steps:**
  - (1) Calculate path i`s volume ($\boldsymbol{vol}(U_i)$) with variable path weight (x_i)

$$vol(U_{n,i}) = \sum_{m=i}^{n} \frac{1}{m}(X_{n,m} - X_{n,m+1}), \quad X_{n,m} = \begin{cases} x_m^m \, x_{m+1} \ldots x_n & n > m \\ x_n^n & n = m \\ 0 & n < m \end{cases}$$

  - (2) Rearrange to above equation, express x_i in terms of the volume ($\boldsymbol{vol}(U_i)$)

$$\frac{x_i}{x_1} = \prod_{k=2}^{i} \left(\frac{k\,vol(U_{n,k}) + vol(U_{n,k+1}) + \cdots + vol(U_{n,n})}{(k-1)vol(U_{n,k-1}) + vol(U_{n,k}) + \cdots + vol(U_{n,n})} \right)^{\frac{1}{k-1}}$$

  - (3) Substituting volumes with path weights (w_i), calculating x_i as VALO gravity

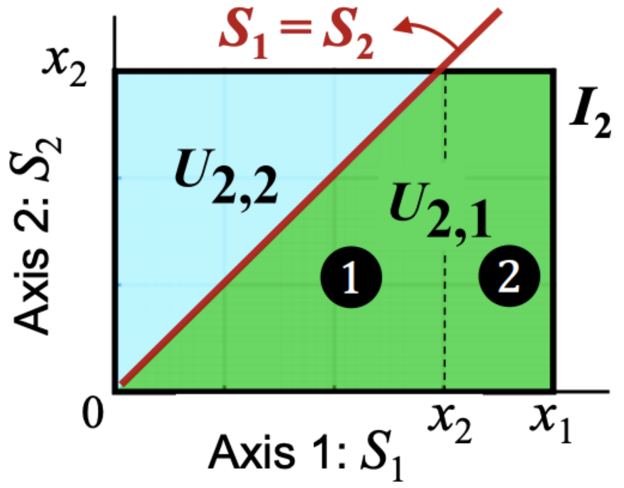
Optimize calculation by simple arithmetic operations

$$\frac{x_i}{x_1} = \prod_{k=2}^{i} \left(\frac{k w_k + w_{k+1} + \cdots + w_n}{(k-1) w_{k-1} + w_k + \ldots + w_n} \right)^{\frac{1}{k-1}}$$

(1) Volume Calculation

- Derive path volume ($\boldsymbol{vol}(U_i)$) from path weight (x_i)
- Use mathematical induction to generalize volume patterns as the number of paths increases

(2 Paths exist)



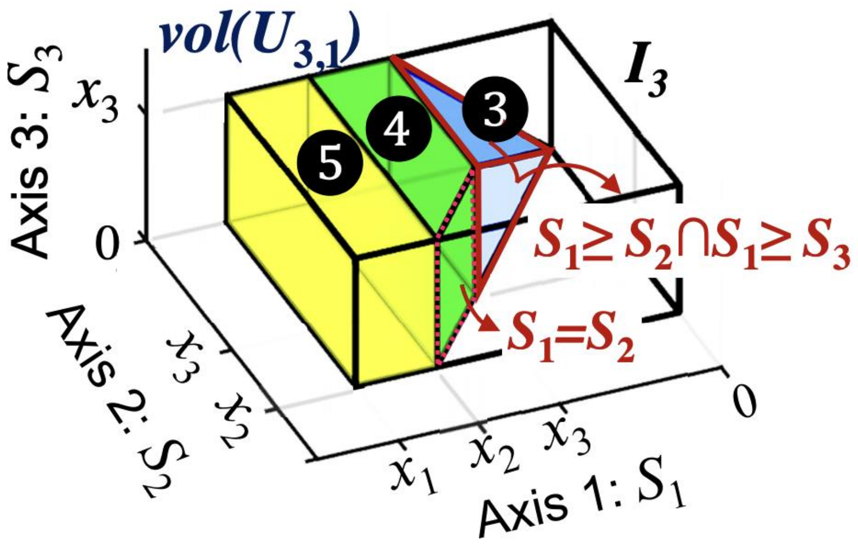
❶ + ❷

$$vol(U_{2,1}) = \frac{1}{2}x_2^2 + (x_1x_2 - x_2^2)$$

❶

$$vol(U_{2,2}) = \frac{1}{2}x_2^2$$

(3 Paths exist)



❸ + ❹ + ❺

$$vol(U_{3,1}) = \frac{1}{3}x_3^3 + \frac{1}{2}(x_2^2x_3 - x_3^3) + (x_1x_2x_3 - x_2^2x_3)$$

❸ + ❹

$$vol(U_{3,2}) = \frac{1}{3}x_3^3 + \frac{1}{2}(x_2^2x_3 - x_3^3)$$

❸

$$vol(U_{3,3}) = \frac{1}{3}x_3^3$$

Mathematical induction (proof in §4.2.2)

(n Paths exist)

$$vol(U_{n,i}) = \sum_{m=i}^{n} \frac{1}{m}(X_{n,m} - X_{n,m+1}), \quad X_{n,m} = \begin{cases} x_m^m x_{m+1} \ldots x_n & n > m \\ x_n^n & n = m \\ 0 & n < m \end{cases}$$

(2) Weight Calculation

- **Reorganize** equations to express x_i explicitly in terms of the volume ($\boldsymbol{vol}(U_i)$)
- Using **Inverse matrix** operations to simplify complex calculations

$$vol(U_{n,i}) = \sum_{m=i}^{n} \frac{1}{m}\left(X_{n,m} - X_{n,m+1}\right), \quad X_{n,m} = \begin{cases} x_m^m x_{m+1} \dots x_n & n > m \\ x_n^n & n = m \\ 0 & n < m \end{cases}$$

Express as matrix

$$\begin{bmatrix} vol(U_{n,1}) \\ vol(U_{n,2}) \\ vol(U_{n,3}) \\ \vdots \\ vol(U_{n,n}) \end{bmatrix} = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n} \\ & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n} \\ & & \frac{1}{3} & \cdots & \frac{1}{n} \\ & & & \ddots & \vdots \\ & & & & \frac{1}{n} \end{bmatrix} \begin{bmatrix} 1 & -1 & & & \\ & 1 & -1 & & \\ & & \ddots & & \\ & & & 1 & -1 \\ & & & & 1 \end{bmatrix} \begin{bmatrix} X_{n,1} \\ X_{n,2} \\ X_{n,3} \\ \vdots \\ X_{n,n} \end{bmatrix}$$

Inverse matrix

$$\begin{bmatrix} X_{n,1} \\ X_{n,2} \\ X_{n,3} \\ \vdots \\ X_{n,n} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ & 1 & 1 & \cdots & 1 \\ & & 1 & \cdots & 1 \\ & & & \ddots & \vdots \\ & & & & 1 \end{bmatrix} \begin{bmatrix} 1 & -1 & & & \\ & 2 & -2 & & \\ & & \ddots & & \\ & & & (n-1) & (-n+1) \\ & & & & n \end{bmatrix} \begin{bmatrix} vol(U_{n,1}) \\ vol(U_{n,2}) \\ vol(U_{n,3}) \\ \vdots \\ vol(U_{n,n}) \end{bmatrix} = \begin{bmatrix} vol(U_{n,1}) + vol(U_{n,2}) + vol(U_{n,3}) + \cdots + vol(U_{n,n}) \\ 2vol(U_{n,2}) + vol(U_{n,3}) + \cdots + vol(U_{n,n}) \\ 3vol(U_{n,3}) + \cdots + vol(U_{n,n}) \\ \vdots \\ nvol(U_{n,n}) \end{bmatrix}$$

(2) Weight Calculation

$$\begin{bmatrix} X_{n,1} \\ X_{n,2} \\ X_{n,3} \\ \vdots \\ X_{n,n} \end{bmatrix} = \begin{bmatrix} vol(U_{n,1}) + vol(U_{n,2}) + vol(U_{n,3}) + \cdots + vol(U_{n,n}) \\ 2vol(U_{n,2}) + vol(U_{n,3}) + \cdots + vol(U_{n,n}) \\ 3vol(U_{n,3}) + \cdots + vol(U_{n,n}) \\ \vdots \\ nvol(U_{n,n}) \end{bmatrix}$$

Define weights as $\frac{x_i}{x_1}$

$$\frac{x_2}{x_1} = \frac{x_2^2 \times \cdots \times x_n}{x_1 \times x_2 \times \cdots \times x_n} = \frac{X_{n,2}}{X_{n,1}} = \frac{2vol(U_{n,2}) + \cdots + vol(U_{n,n})}{vol(U_{n,1}) + \cdots + vol(U_{n,n})}$$

$$\frac{x_3}{x_1} = \frac{X_{n,2}}{X_{n,1}} \times \left(\frac{X_{n,3}}{X_{n,2}}\right)^{\frac{1}{2}} = \frac{x_2}{x_1} \times \left(\frac{3vol(U_{n,3}) + \cdots + vol(U_{n,n})}{2vol(U_{n,2}) + \cdots + vol(U_{n,n})}\right)^{\frac{1}{2}}$$

$$\frac{x_4}{x_1} = \frac{X_{n,2}}{X_{n,1}} \times \left(\frac{X_{n,3}}{X_{n,2}}\right)^{\frac{1}{2}} \times \left(\frac{X_{n,4}}{X_{n,3}}\right)^{\frac{1}{3}} = \frac{x_3}{x_1} \times \left(\frac{4vol(U_{n,4}) + \cdots + vol(U_{n,n})}{3vol(U_{n,3}) + \cdots + vol(U_{n,n})}\right)^{\frac{1}{3}}$$

Generalize to n paths

$$\frac{x_i}{x_1} = \prod_{k=2}^{i} \left(\frac{kvol(U_{n,k}) + vol(U_{n,k+1}) + \cdots + vol(U_{n,n})}{(k-1)vol(U_{n,k-1}) + vol(U_{n,k}) + \cdots + vol(U_{n,n})}\right)^{\frac{1}{k-1}}$$

(3) VALO Gravity Calculation

- Finally, weights are expressed in terms of the volume of each path

$$\frac{x_i}{x_1} = \prod_{k=2}^{i} \left(\frac{k\,vol(U_{n,k}) + vol(U_{n,k+1}) + \cdots + vol(U_{n,n})}{(k-1)vol(U_{n,k-1}) + vol(U_{n,k}) + \cdots + vol(U_{n,n})} \right)^{\frac{1}{k-1}}$$

Instead of volume ($vol(U_{n,i})$), substitute given path weights (w_i)

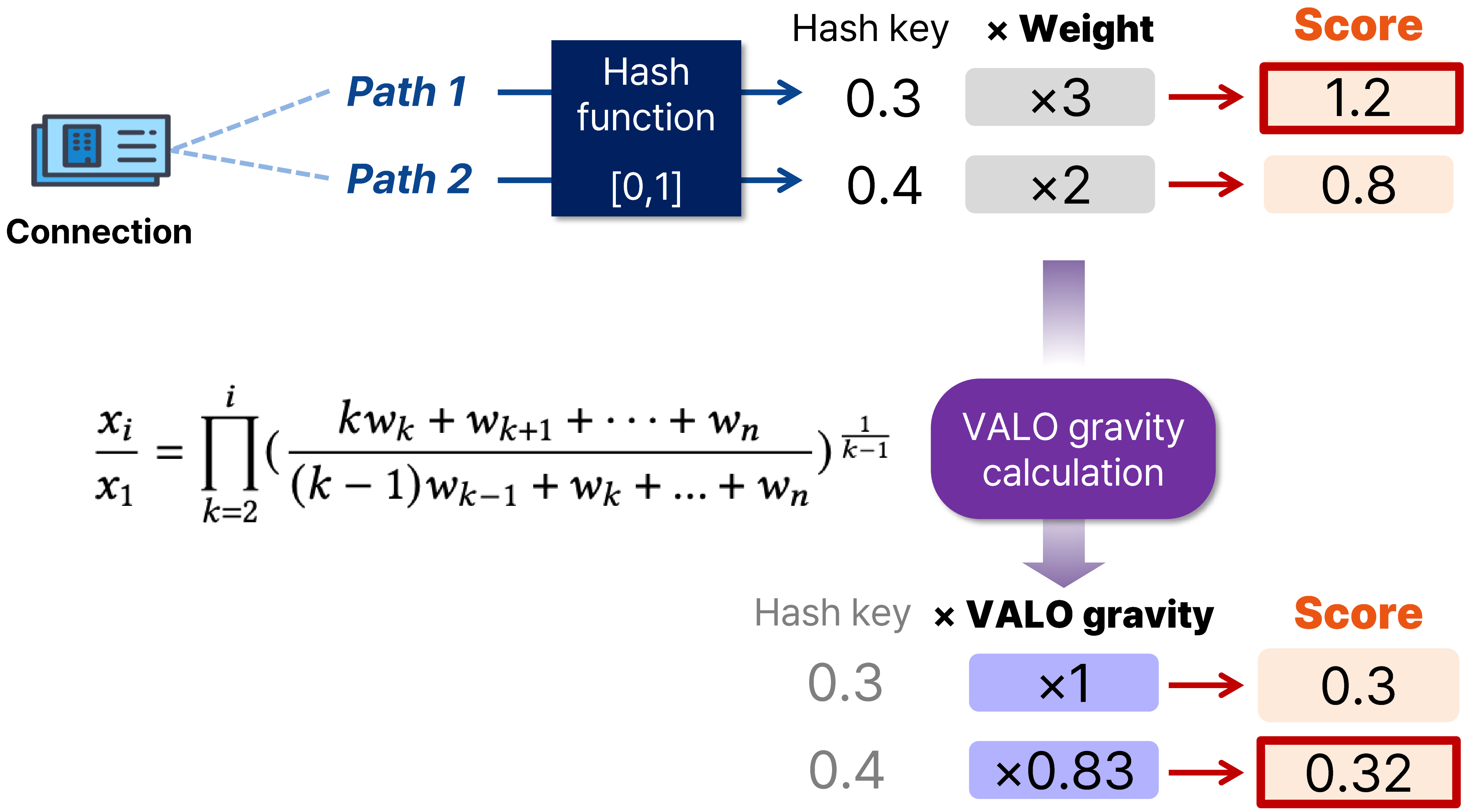
$$\frac{x_i}{x_1} = \prod_{k=2}^{i} \left(\frac{k w_k + w_{k+1} + \cdots + w_n}{(k-1) w_{k-1} + w_k + \ldots + w_n} \right)^{\frac{1}{k-1}}$$

Now, we get new weight $\frac{x_i}{x_1}$, VALO gravity

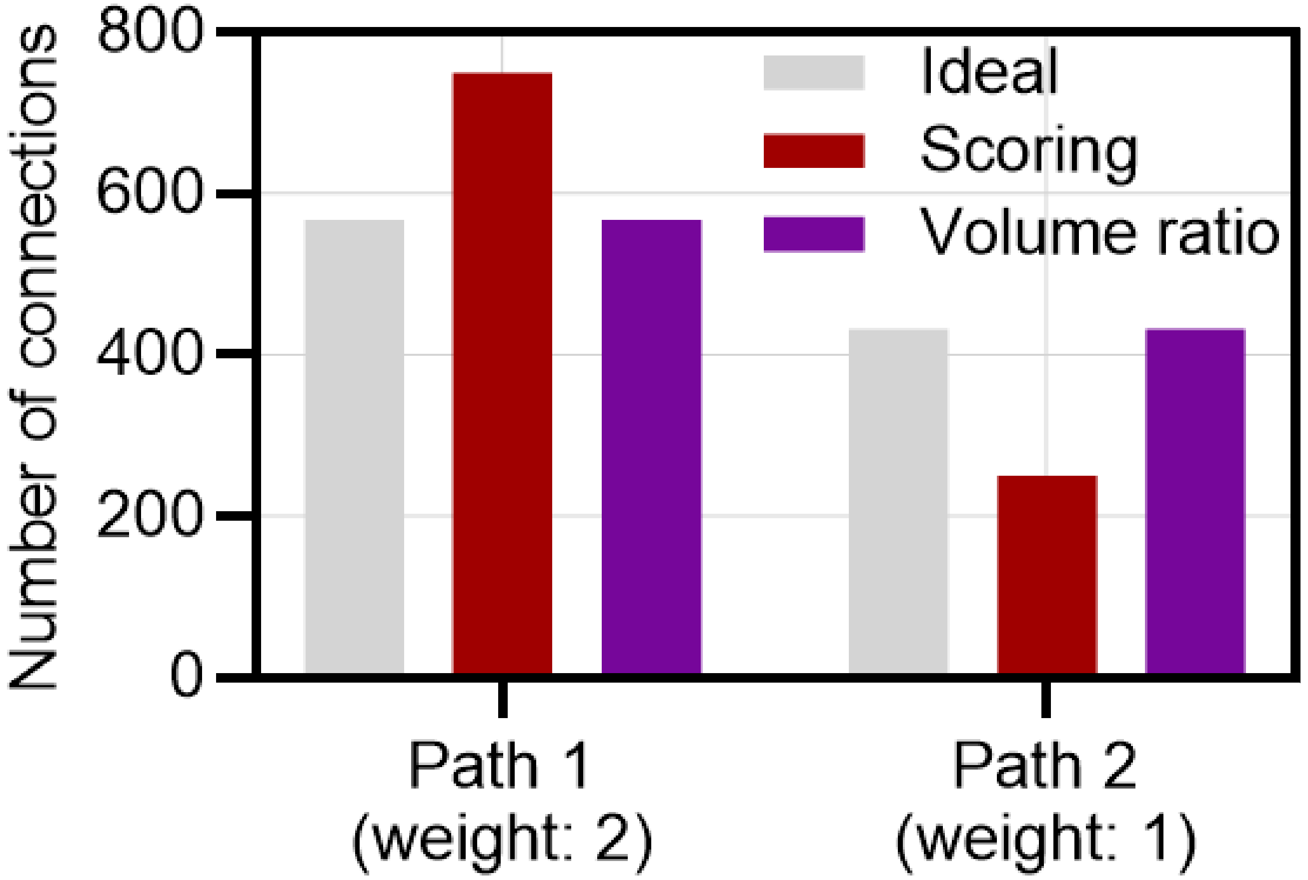
Calculated by only simple arithmetic operations

VALO Workflow Summary

- Through VALO gravity, the number of connections matches with the path weights

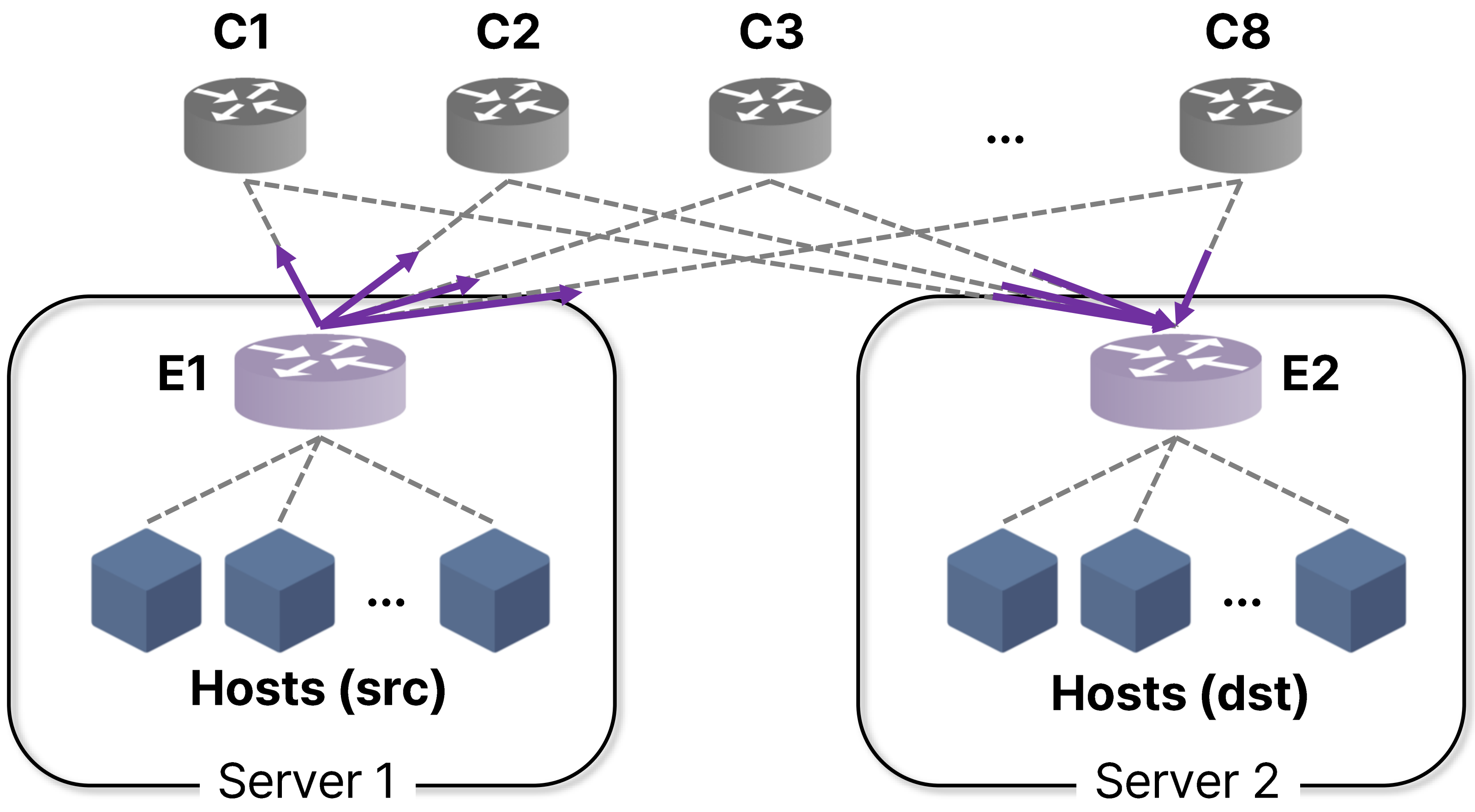


$$\frac{x_i}{x_1} = \prod_{k=2}^{i} \left(\frac{k w_k + w_{k+1} + \cdots + w_n}{(k-1) w_{k-1} + w_k + \ldots + w_n} \right)^{\frac{1}{k-1}}$$



Evaluation

- **Topology:** Two-tier DC topology (core switches increased 2–8), using ~32 containers
- **Compare five techniques:** random, WRR, WCMP, scoring, and VALO (implemented on OVS)
- **Workloads:** 1) DC traffic traces (CAIDA, ClassBench) and 2) Real-world DC workloads
- **Measurement:** 1) accuracy, resource-efficiency and 2) end-to-end latency of DC services



- 2–8 paths exist

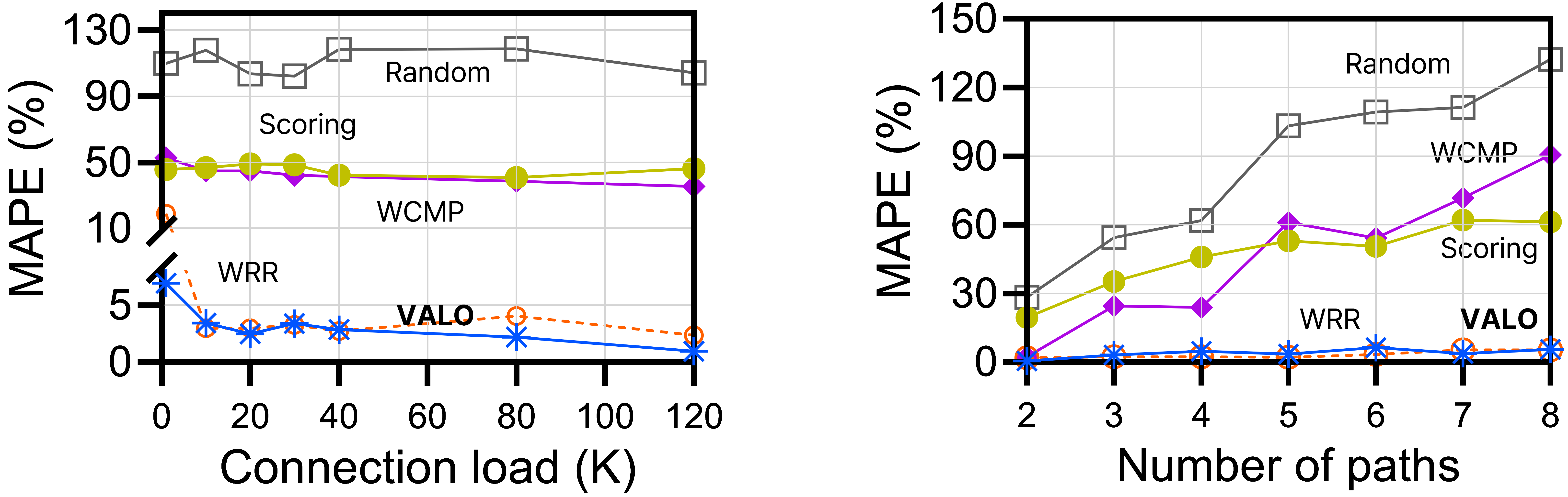
- Random, WRR, WCMP, scoring, VALO

- CAIDA, ClassBench
- Web search, data mining, deep learning, in-memory cache

Traffic Splitting Accuracy

- Measure error rate (MAPE) between ideal (C_i) and actual ($\widehat{C}_i$) connection distributions

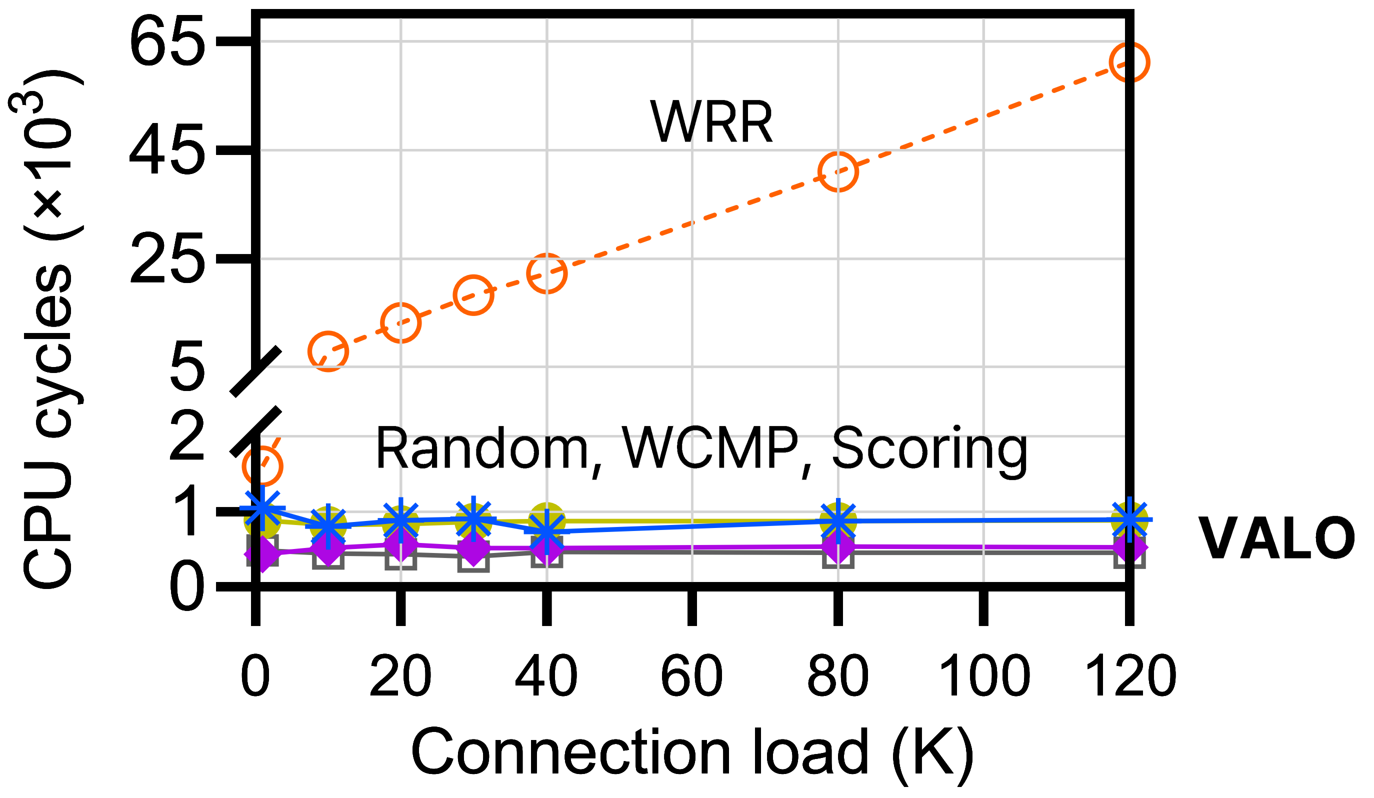
$$MAPE = \frac{100}{n} \times \sum_{i=1}^{n} \frac{|C_i - \hat{C}_i|}{C_i}$$



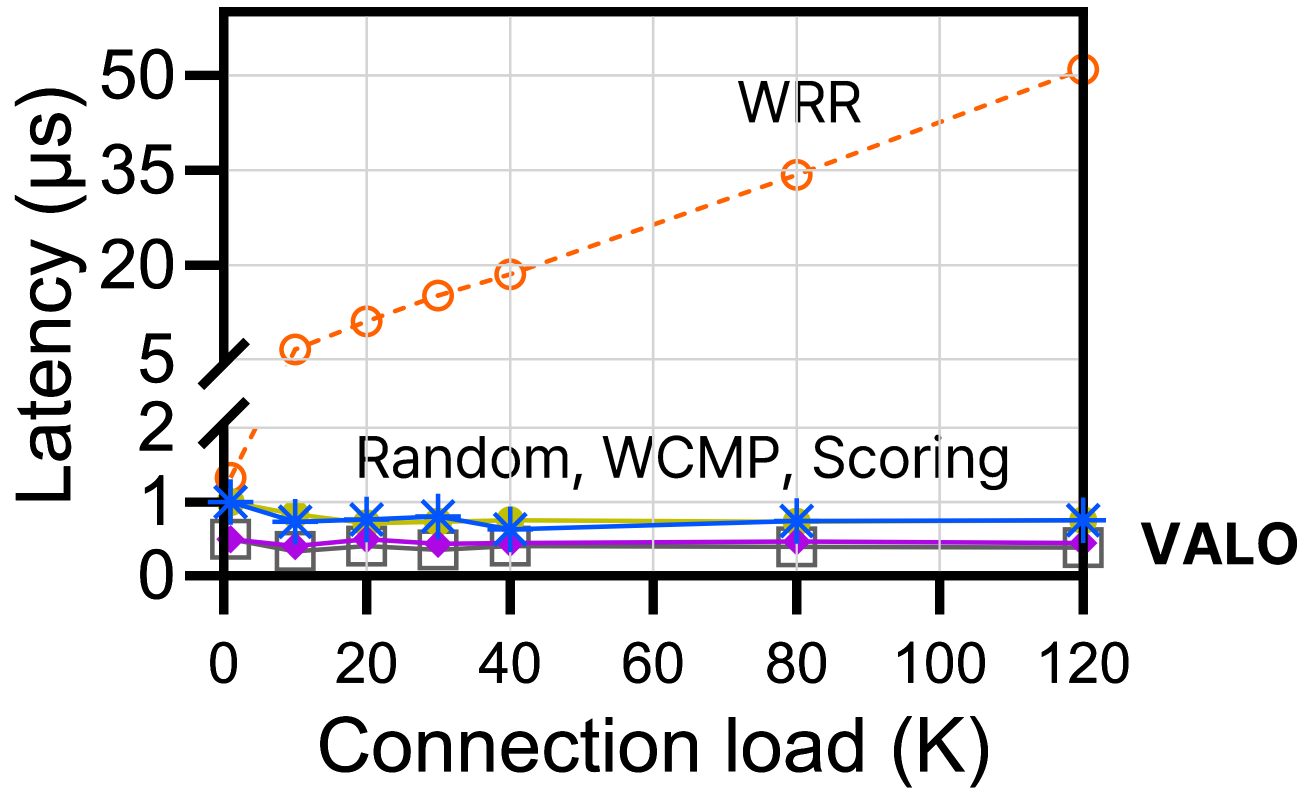
VALO keeps low accuracy (3.1% on average) in all cases (~46.3× improvement)

Resource-efficiency

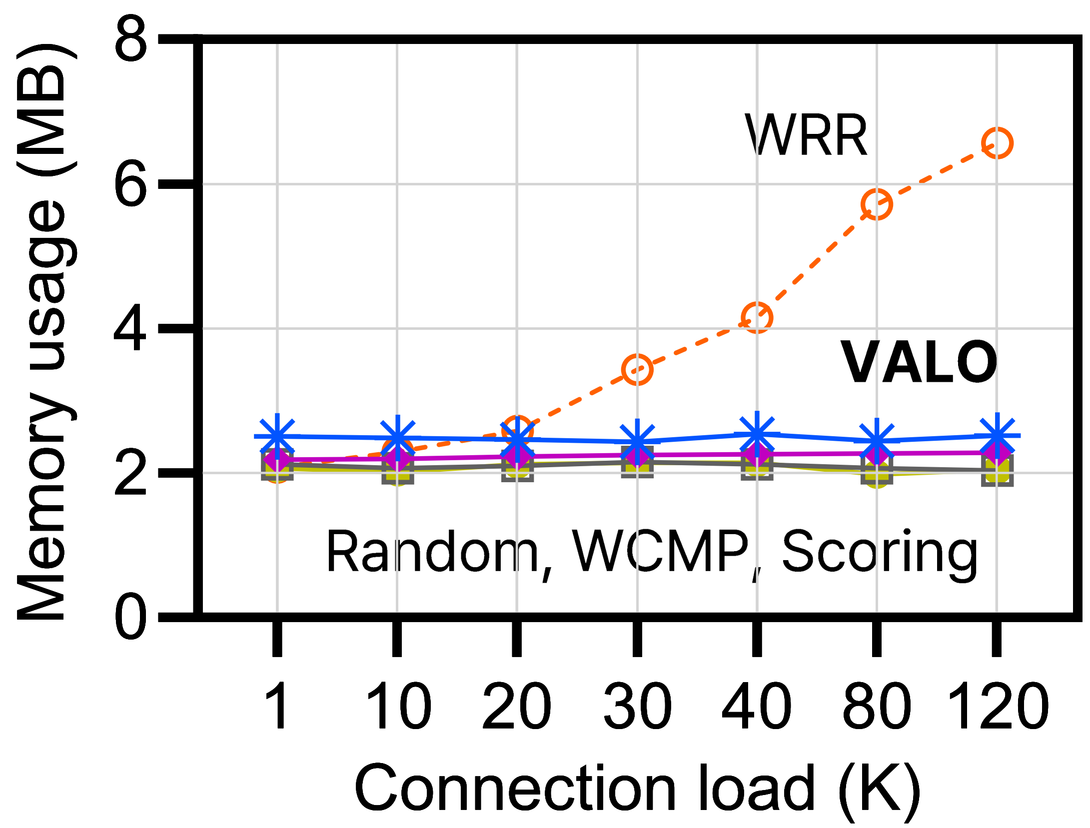
- CPU usage



- Per-packet latency



- Memory usage

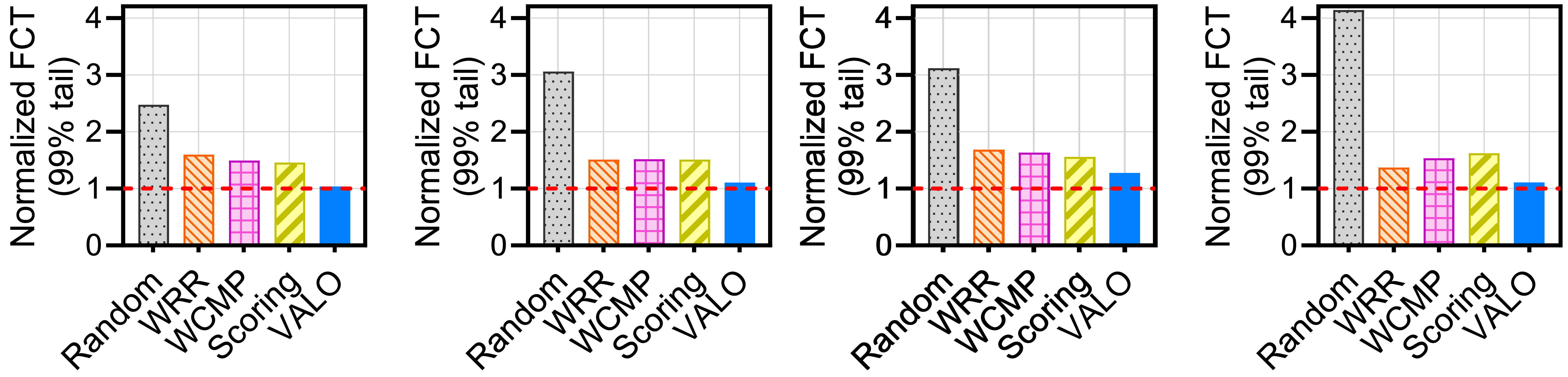


VALO shows a similar level of resource-efficiency as random, WCMP, and scoring

VALO does not sacrifice efficiency to achieve high accuracy!

Flow Completion Time of DC Services

- VALO achieves the shortest flow completion time (FCT) on real-world DC service workloads
  - VALO improves 99% tail latency by ~2.8×



Web search

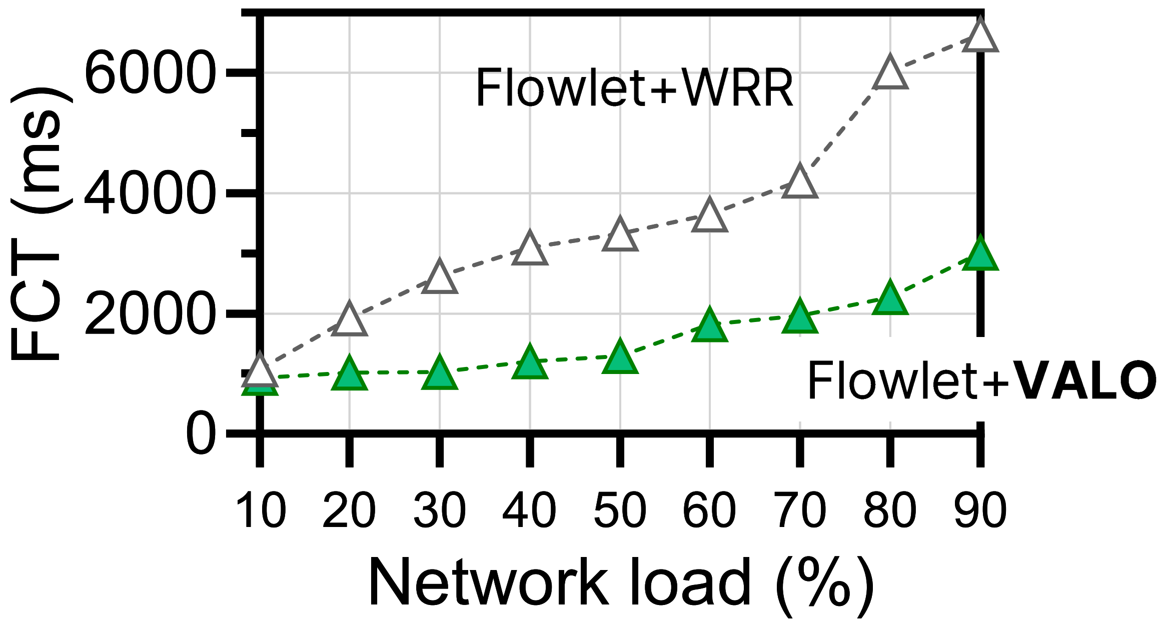
Data mining

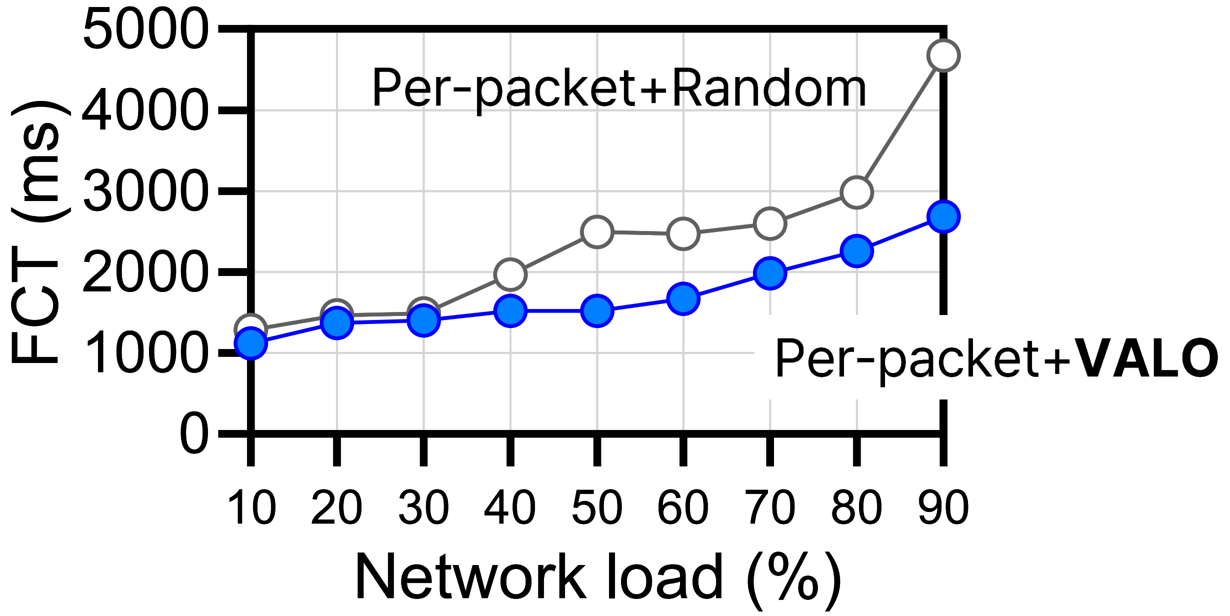
Deep learning

In-memory cache (Twitter)

Conclusion

- **Summary**
  - Major result: Achieves **accuracy** by ~46.3× and **CPU usage** by ~10.7× → accelerate DC services (~2.8×)
  - Approach 1: Analyze root-cause of inaccuracy of scoring with **score graph**
  - Approach 2: Find new internal weight, **VALO gravity** that align volumes to path weights
- **VALO can integrate with other load balancing techniques (§5.5)**
  - Not only per-connection, VALO can work at finer granularity (e,g., flowlet, per-packet)





- **Artifact**
  - VALO implemented and evaluated on de-facto software switch (Open vSwitch of Linux foundation)
  - Our codes are available at GitHub!
    - https://github.com/yeonhooy/VALO-OVS-SIGMETRICS25.git

Thank you

Yeonho Yoo (yhyoo@os.korea.ac.kr)