Longer Stay Less Priority: Flow length approximation used for traffic scheduling in Data Centers

Muhammad Shahid Iqbal
EECS International Graduate Program
National Yang-Ming Chiao Tung University
Hsinchu, Taiwan
shahid2636.eed07g@nctu.edu.tw

Chien Chen
Dept. of Computer Science
National Yang-Ming Chiao Tung University
Hsinchu, Taiwan
chienchen@nctu.edu.tw

Flow Completion Time (FCT) is Key

- Data center applications
  - Desire low latency for short messages
  - App performance & user experience

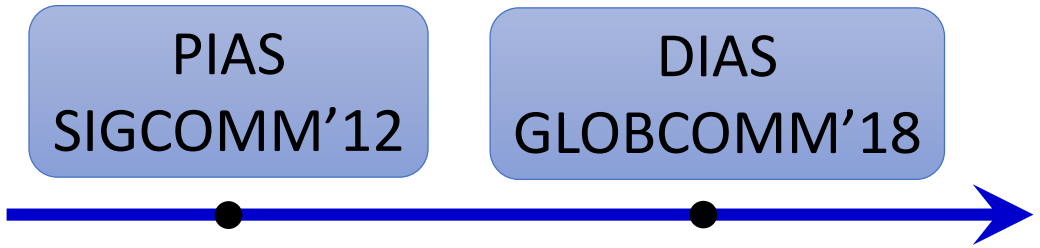
- Goal of DCN transport: **minimize FCT**
  - Many flow scheduling proposals

Information aware solution

All assume ~~prior knowledge of flow size information~~ to approximate ideal preemptive Shortest Job First (SJF) with/without customized network elements

- **Not feasible for many applications**
- **Hard to deploy in practice**

Information-agnostic solutions



PIAS,DIAS ~~install a kernel space application~~ at end-hosts (servers) to improve(reduce) FCT without prior knowledge of flow size information

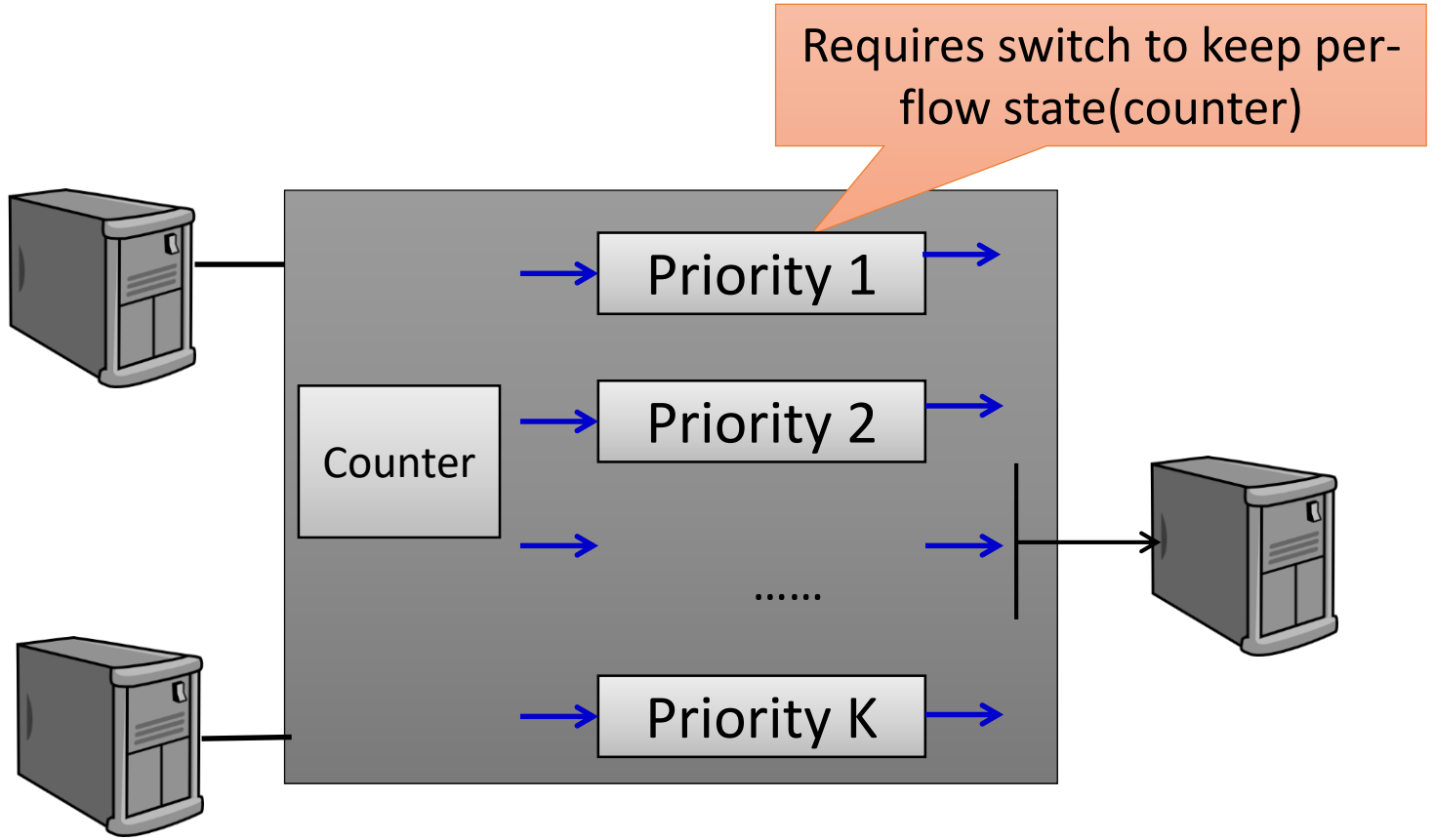
- **Hard to deploy in practice**

Constraints in PIAS

- Install a kernel space application at thousands of end hosts
  - time consuming
  - not easy to adapt
- Counting the number of bytes via software
  - adds delay.
- Low sending rate → less data received → judged as short flow
  - Remains active longer → must be large flow

PIAS at P4

- Implementing PIAS directly at P4 switch
  - Low register memory



Question

Without prior knowledge of flow size information and without installing an application at the servers, how to minimize FCT in data centers using P4?

Our Answer

Without prior knowledge of flow size information and without installing an application at the servers, how to minimize FCT in data centers using P4?

Longer Stay Less Priority: Flow length approximation used for traffic scheduling in Data Centers

Design Goal 1

Without prior knowledge of flow size information and installing an application, how to minimize FCT in data centers using P4?

Information-agnostic: not assume a priori knowledge of flow size information available from the applications

Design Goal 2

Without prior knowledge of flow size information and installing an application, how to minimize FCT in data centers using P4?

FCT minimization: minimize the average and tail FCTs of short flows & not adversely affect FCTs of large flows

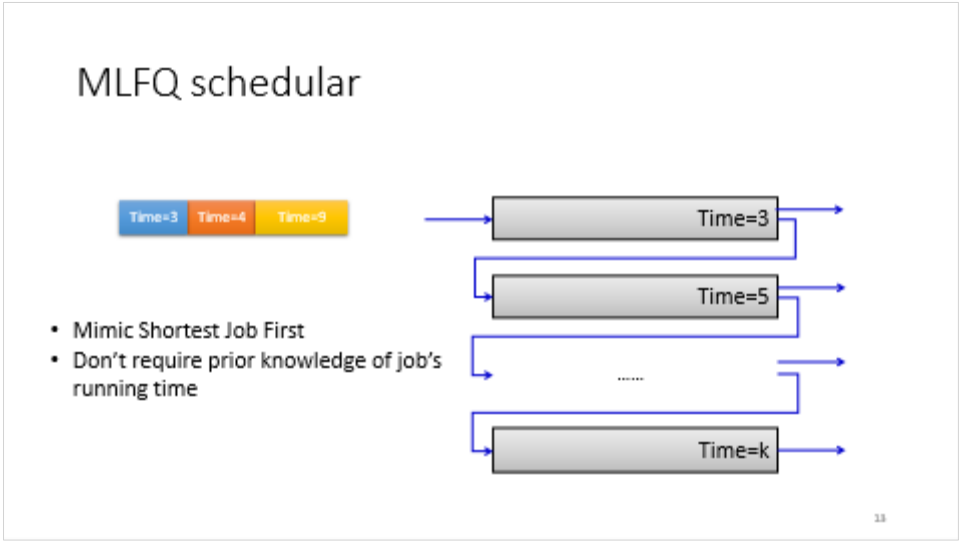
Design Goal 3

Without prior knowledge of flow size information and without installing an application, how to minimize FCT in data centers using P4?

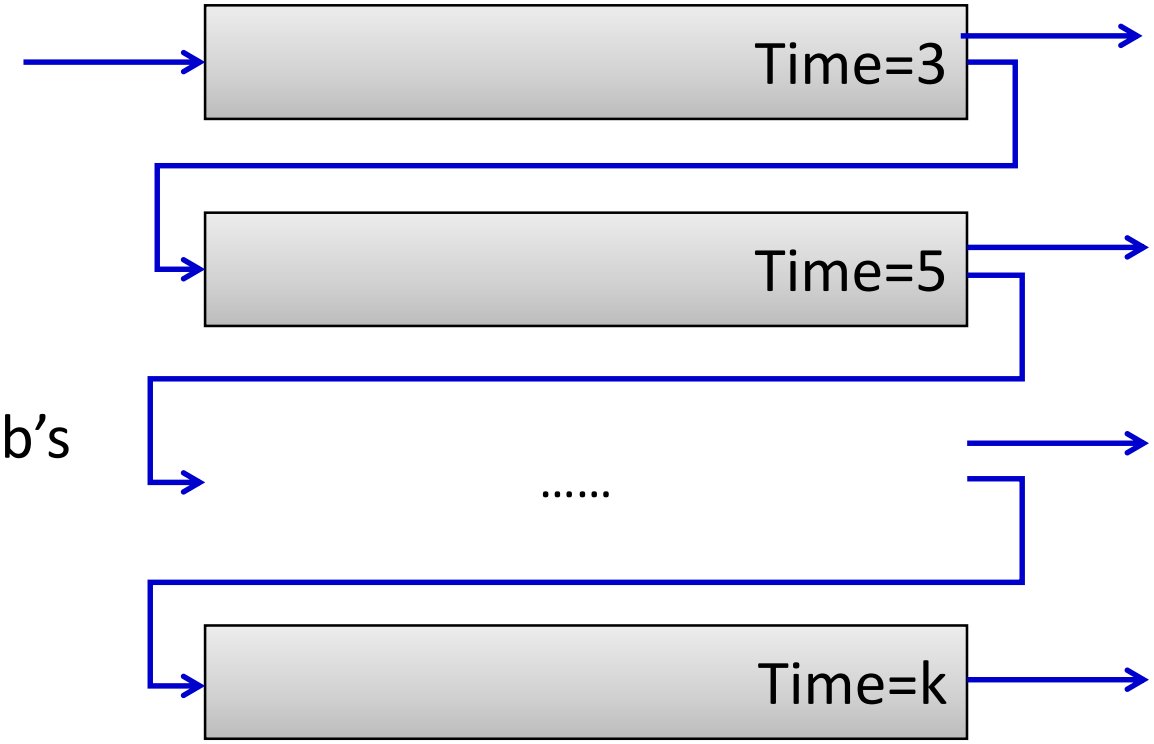
Easily Deployable: Should be easily deployable across data center networks, without requiring the end hosts to install/update any application

LSLP key idea

- LSLP→ Longer stay Less priority
  - If a flows stays longer, its priority will keep decreasing as time passes
- LSLP's idea is similar to OS's MLFQ scheduler
  - Optimize turnaround time
  - Minimize response time for short job



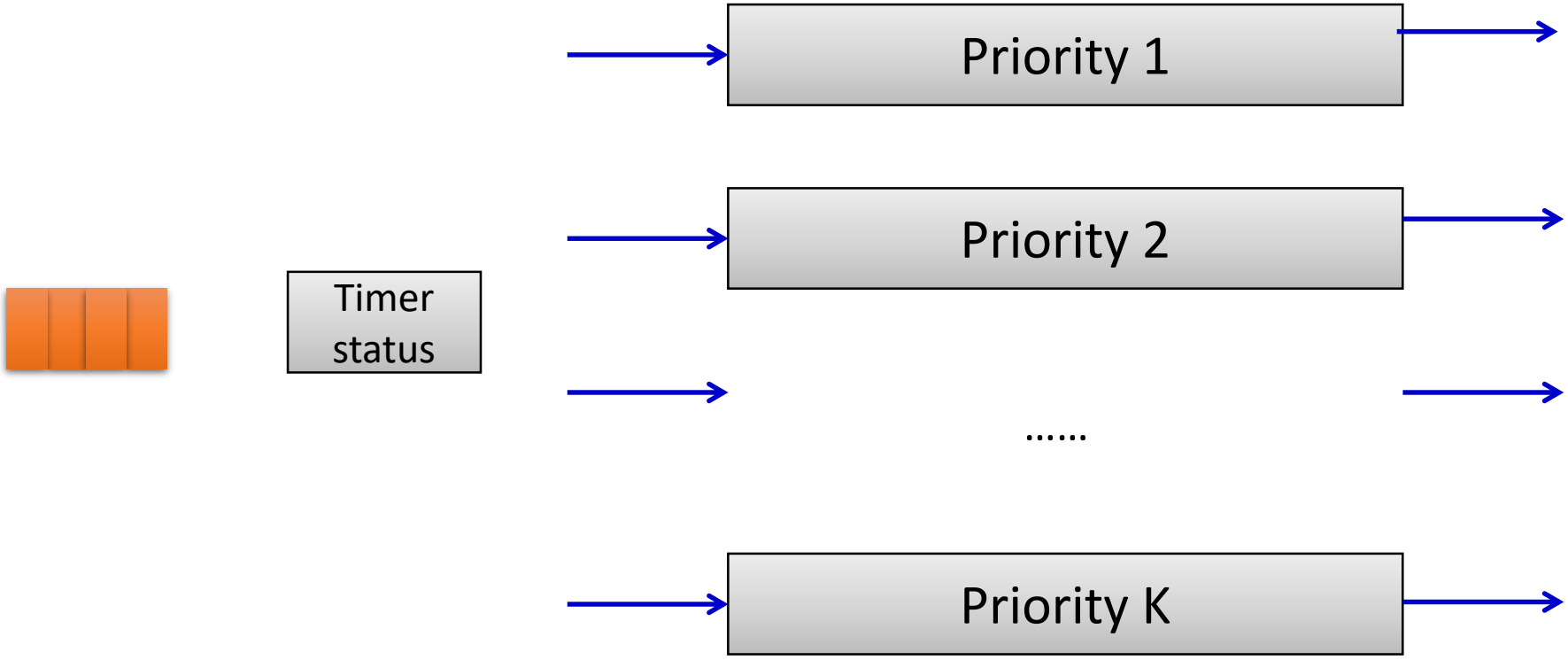
MLFQ schedular



- Mimic Shortest Job First
- Don’t require prior knowledge of job’s running time

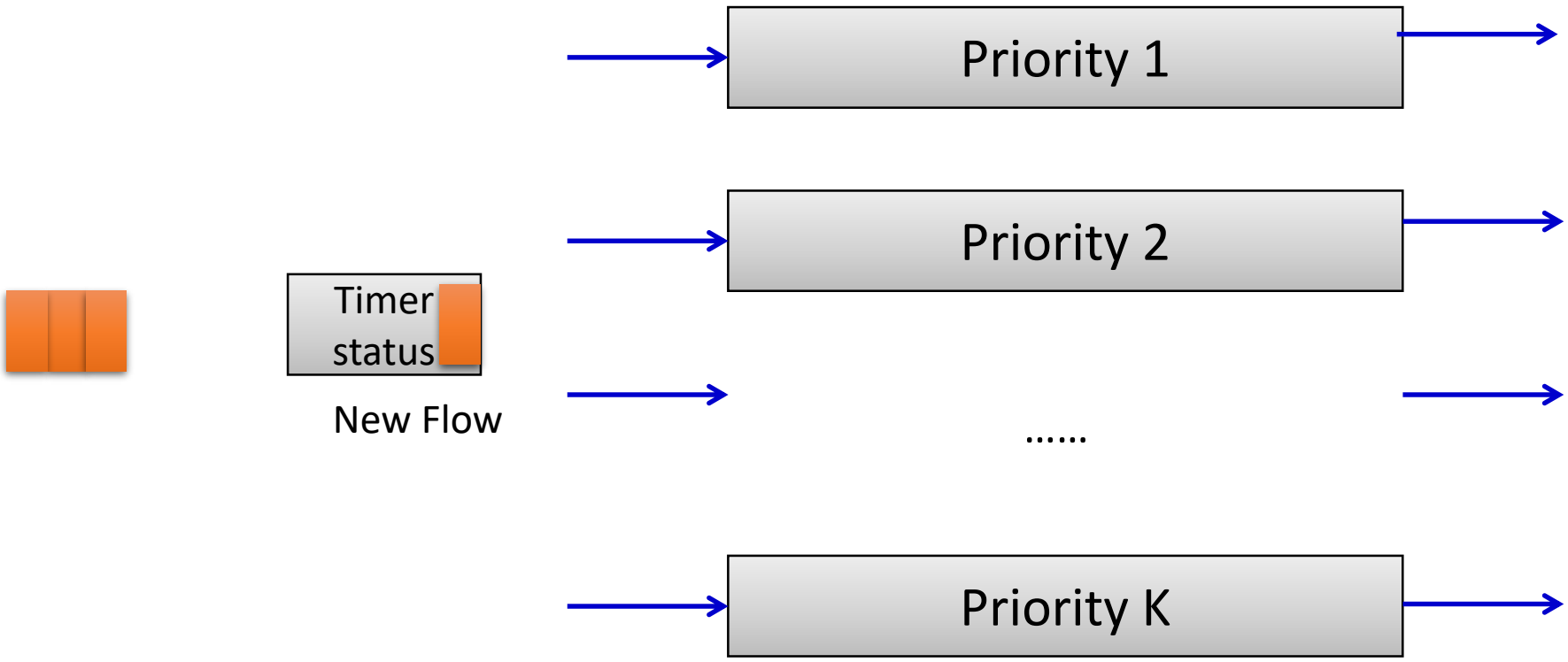
LSLP Key Idea

- LSLP performs Multi-Level Feedback Queue (MLFQ) to emulate Shortest Job First (SJF)



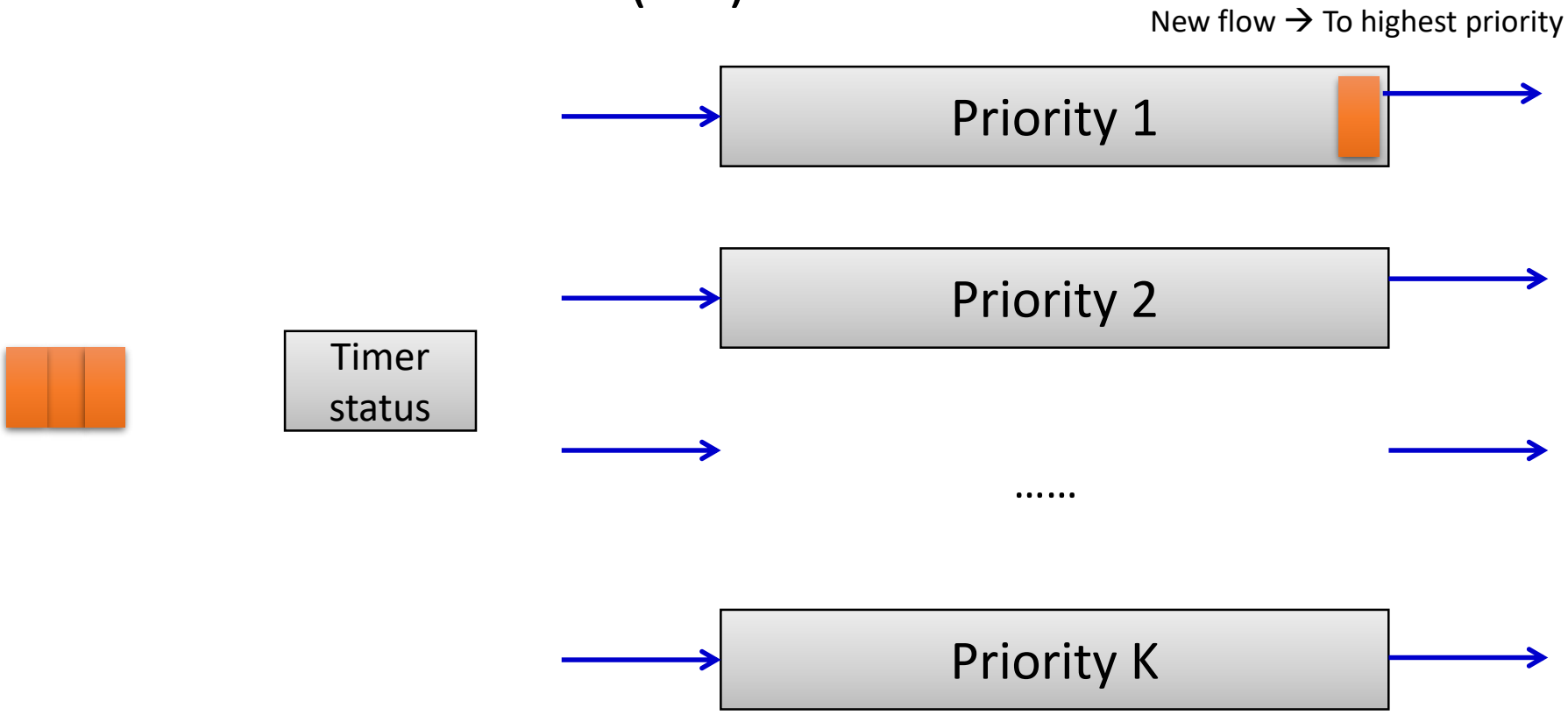
LSLP Key Idea

- LSLP performs Multi-Level Feedback Queue (MLFQ) to emulate Shortest Job First (SJF)



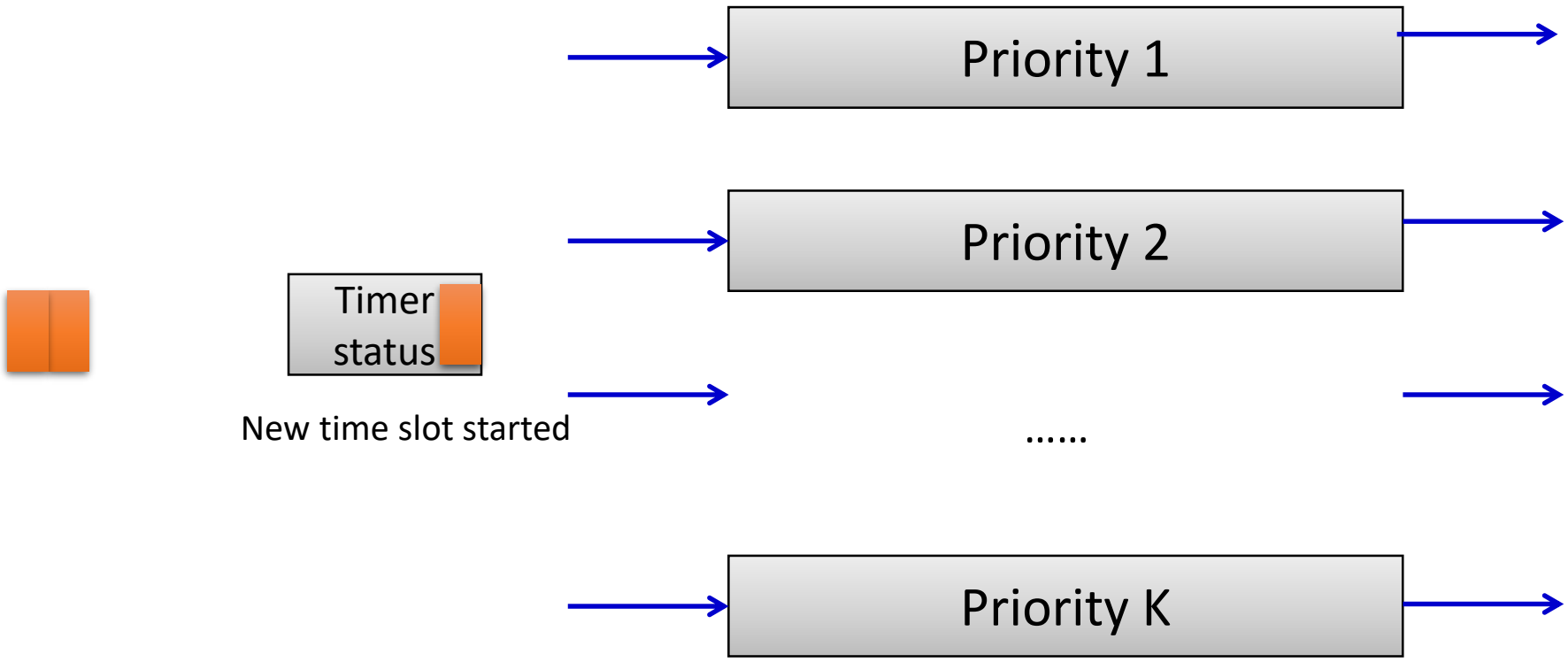
LSLP Key Idea

- LSLP performs Multi-Level Feedback Queue (MLFQ) to emulate Shortest Job First (SJF)



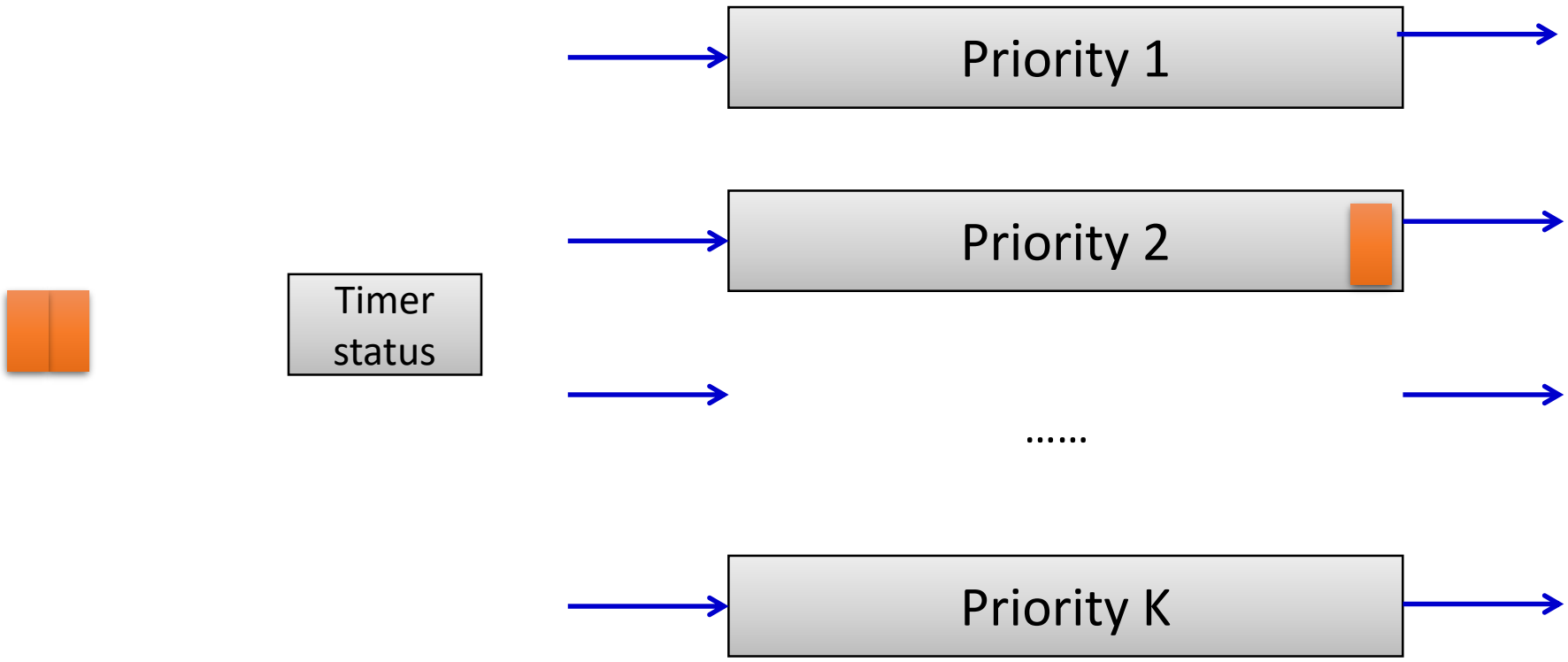
LSLP Key Idea

- LSLP performs Multi-Level Feedback Queue (MLFQ) to emulate Shortest Job First (SJF)



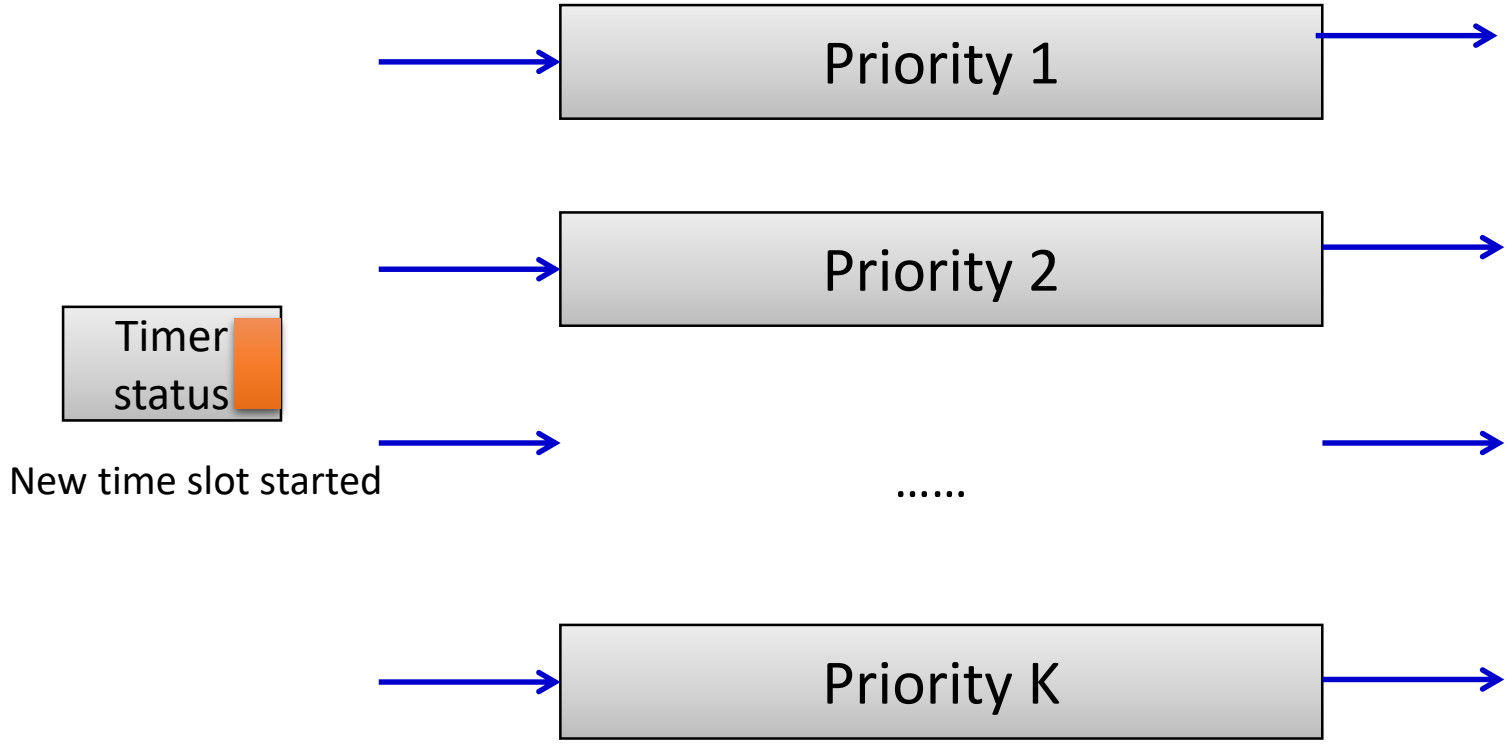
LSLP Key Idea

- LSLP performs Multi-Level Feedback Queue (MLFQ) to emulate Shortest Job First (SJF)



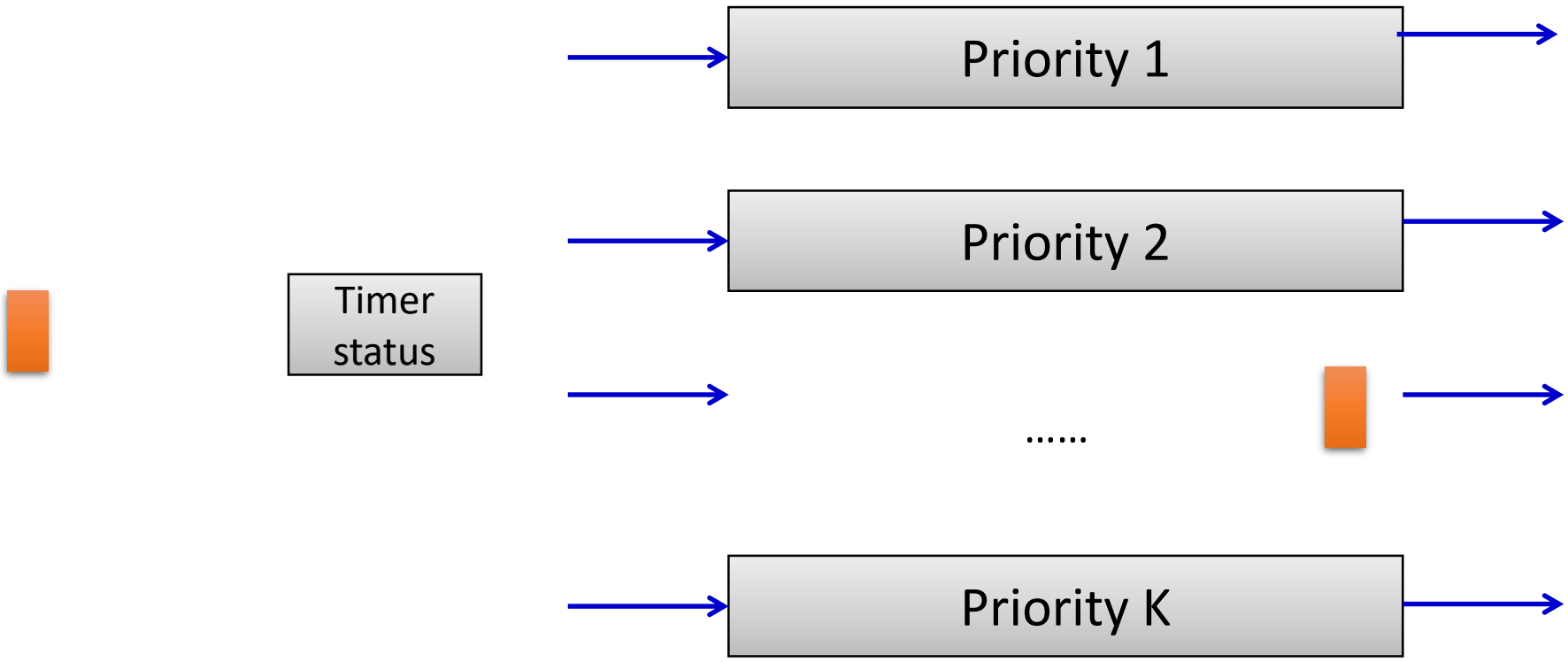
LSLP Key Idea

- LSLP performs Multi-Level Feedback Queue (MLFQ) to emulate Shortest Job First (SJF)



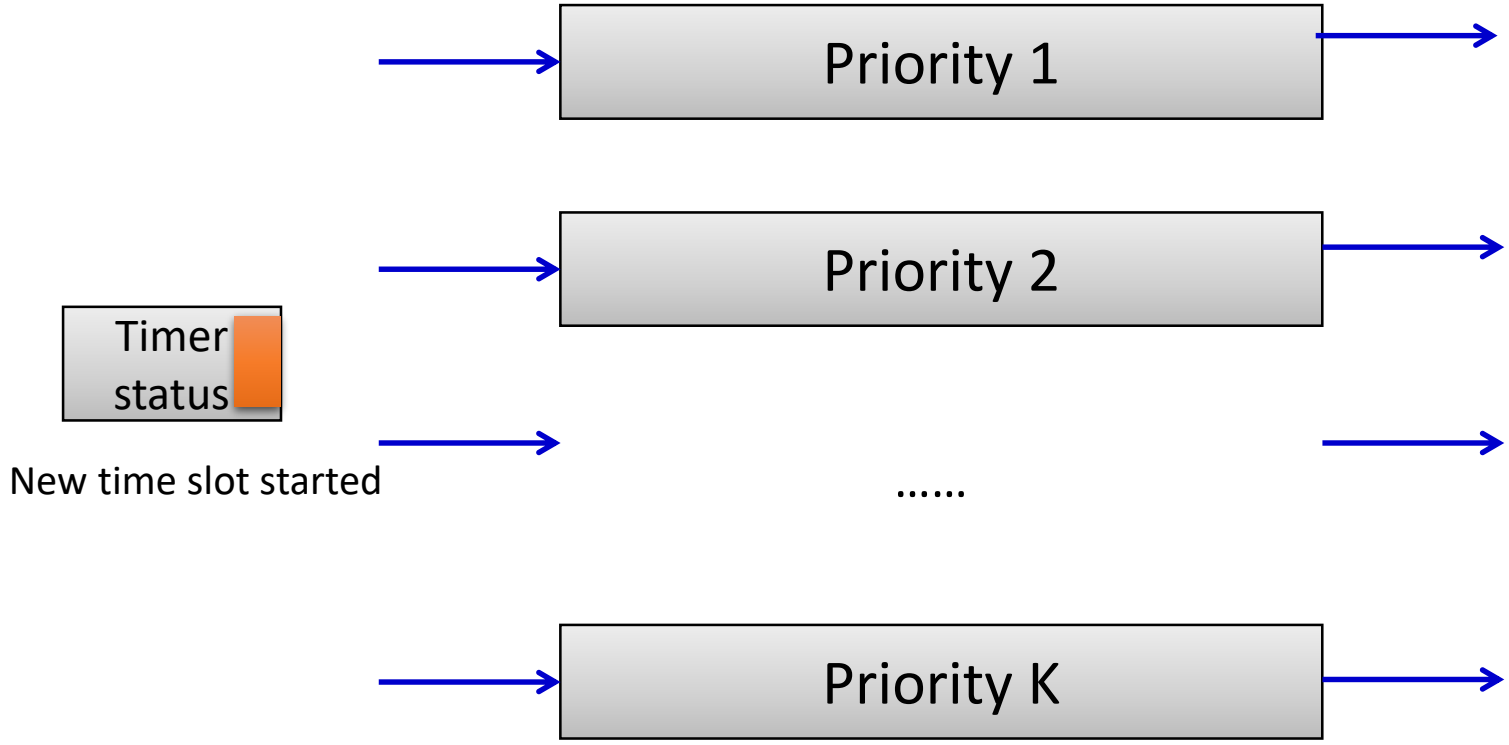
LSLP Key Idea

- LSLP performs Multi-Level Feedback Queue (MLFQ) to emulate Shortest Job First (SJF)



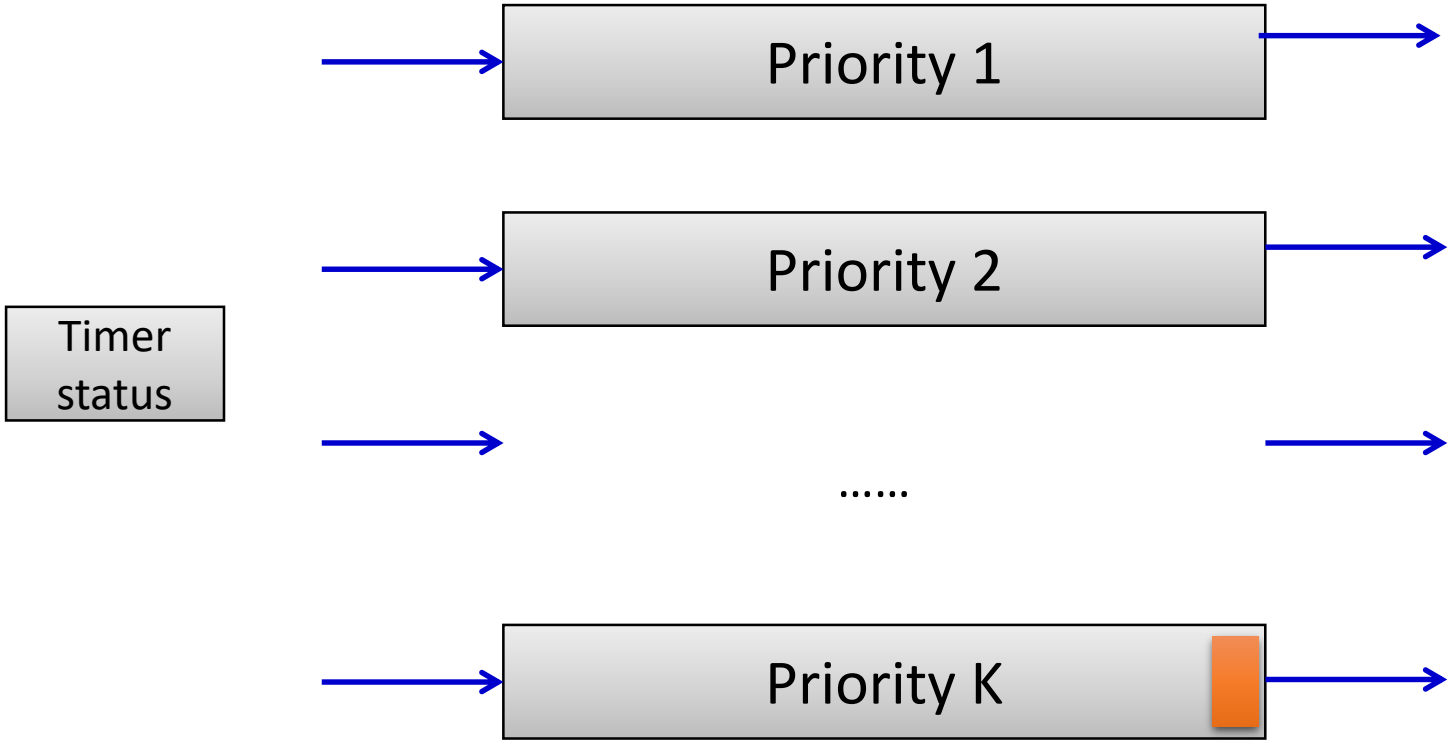
LSLP Key Idea

- LSLP performs Multi-Level Feedback Queue (MLFQ) to emulate Shortest Job First (SJF)



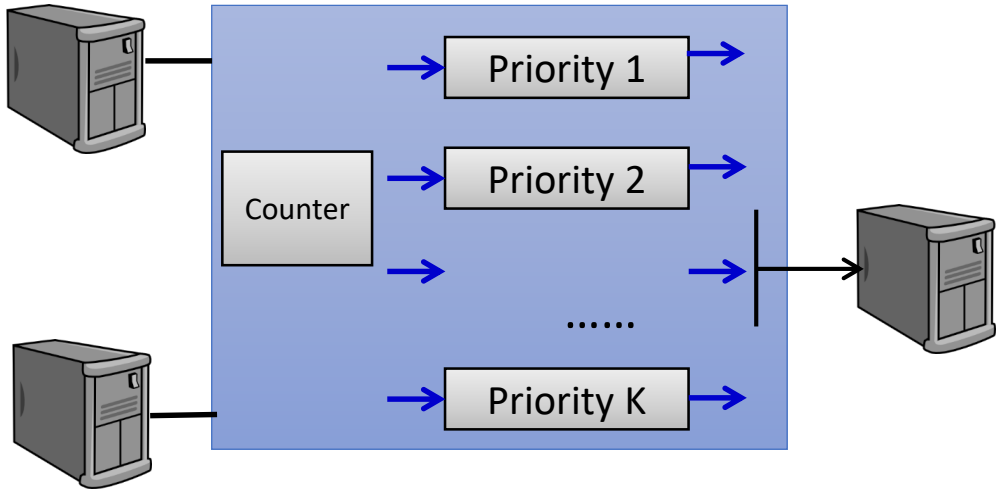
LSLP Key Idea

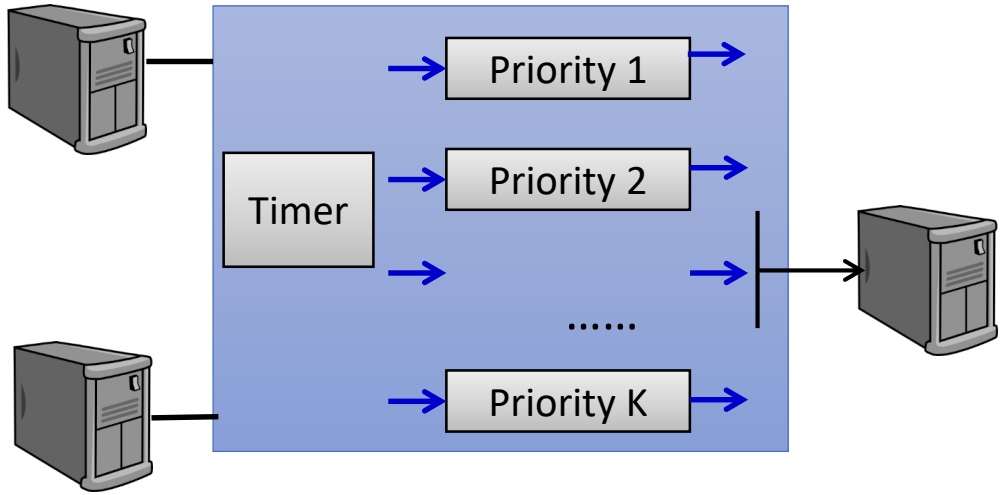
- LSLP performs Multi-Level Feedback Queue (MLFQ) to emulate Shortest Job First (SJF)



How to implement LSLP?

- Implementing PIAS directly at P4 switch
  - Low register memory
    - Counter can not be stored
- Counting bytes is directly proportional to the time elapsed
  - Can use timer instead of counter
    - Provides approximate results to counter
    - Still not suitable for million of flows as timer status can not be stored due to low register memory

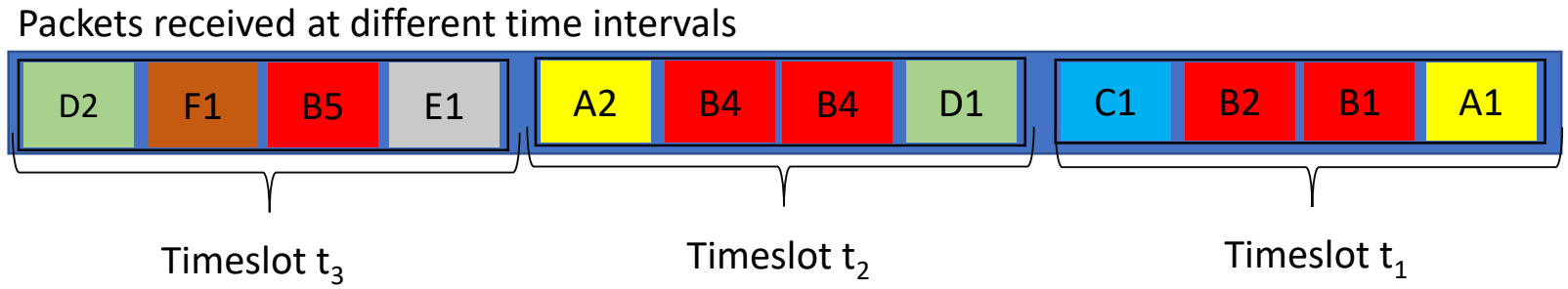




How to implement LSLP in P4?

Solution

- Instead of using timer for every single flow
  - Divide the time in time slots
  - Treat every time-slot as a version
  - Use version id to prioritize traffic
  - Newer version → higher priority

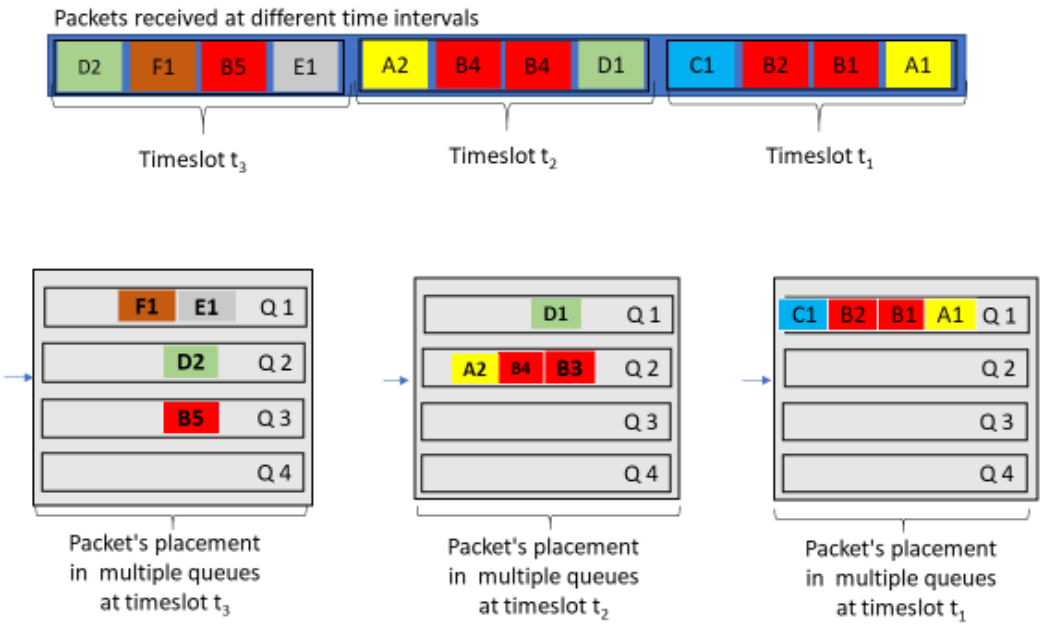


Benefits

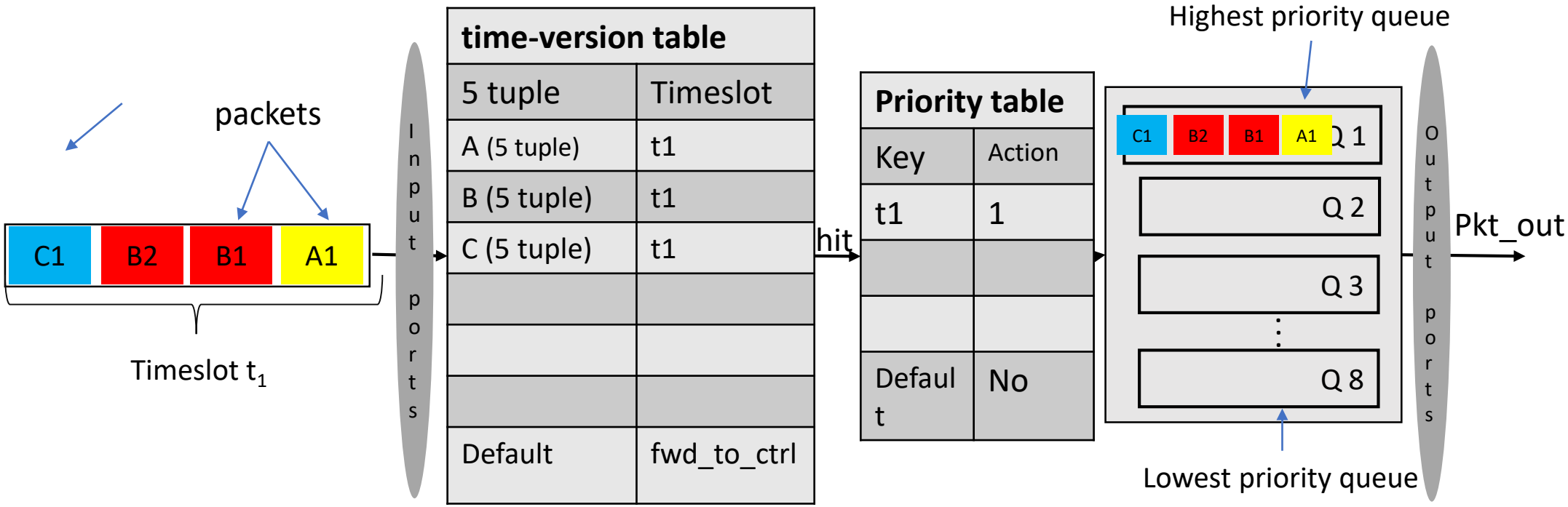
- Don’t require a lot of resources
  - Only one timer can be used to handle all the traffic

Proposed LSLP method using P4

- Beginning of new time slot
  - All new flows are group in one version
    - Packets assigned same priority
  - All existing flows
    - Demoted by one level except already in lowest priority

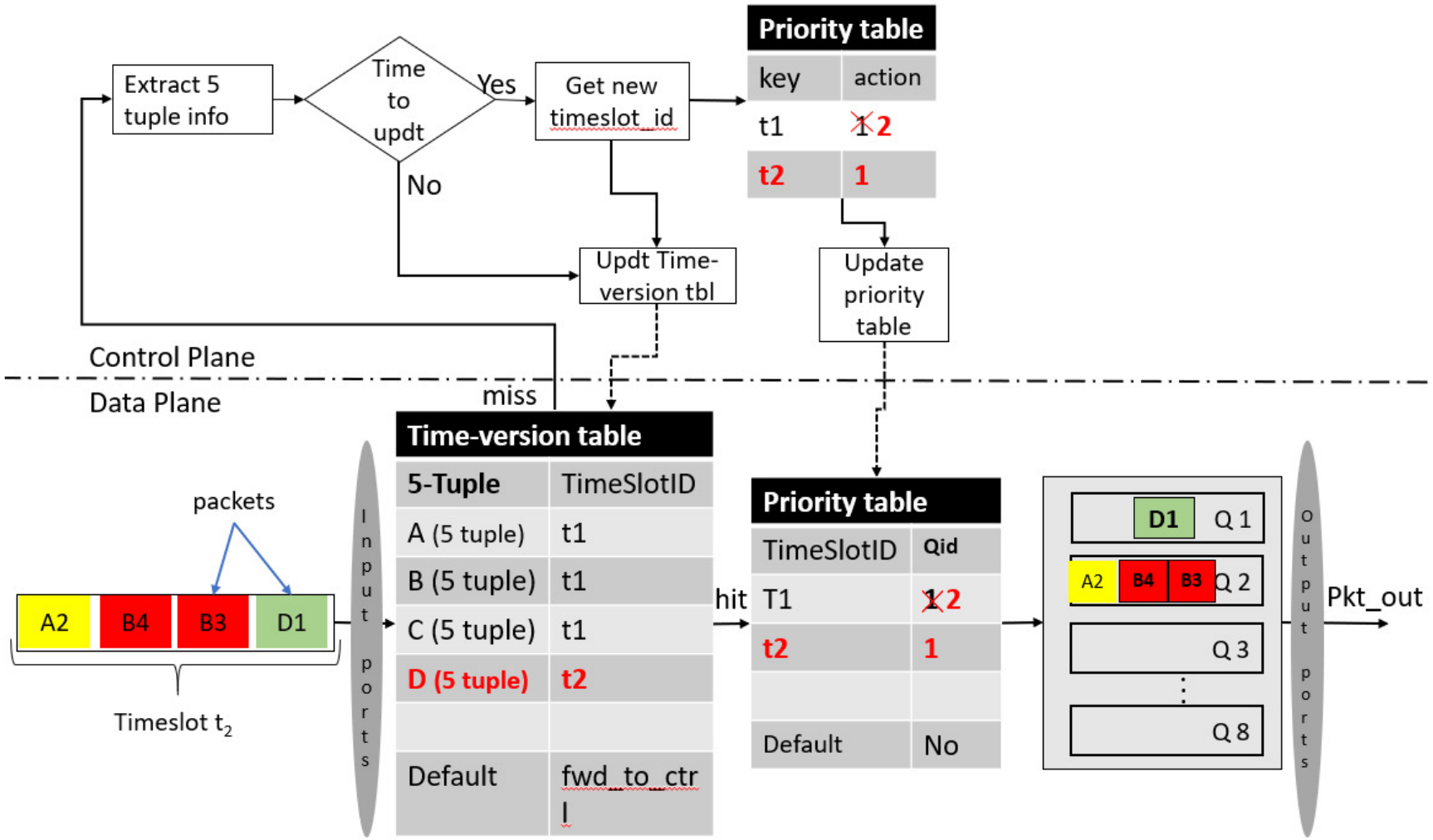


Implementation



P4 Data plane pipeline

Implementation

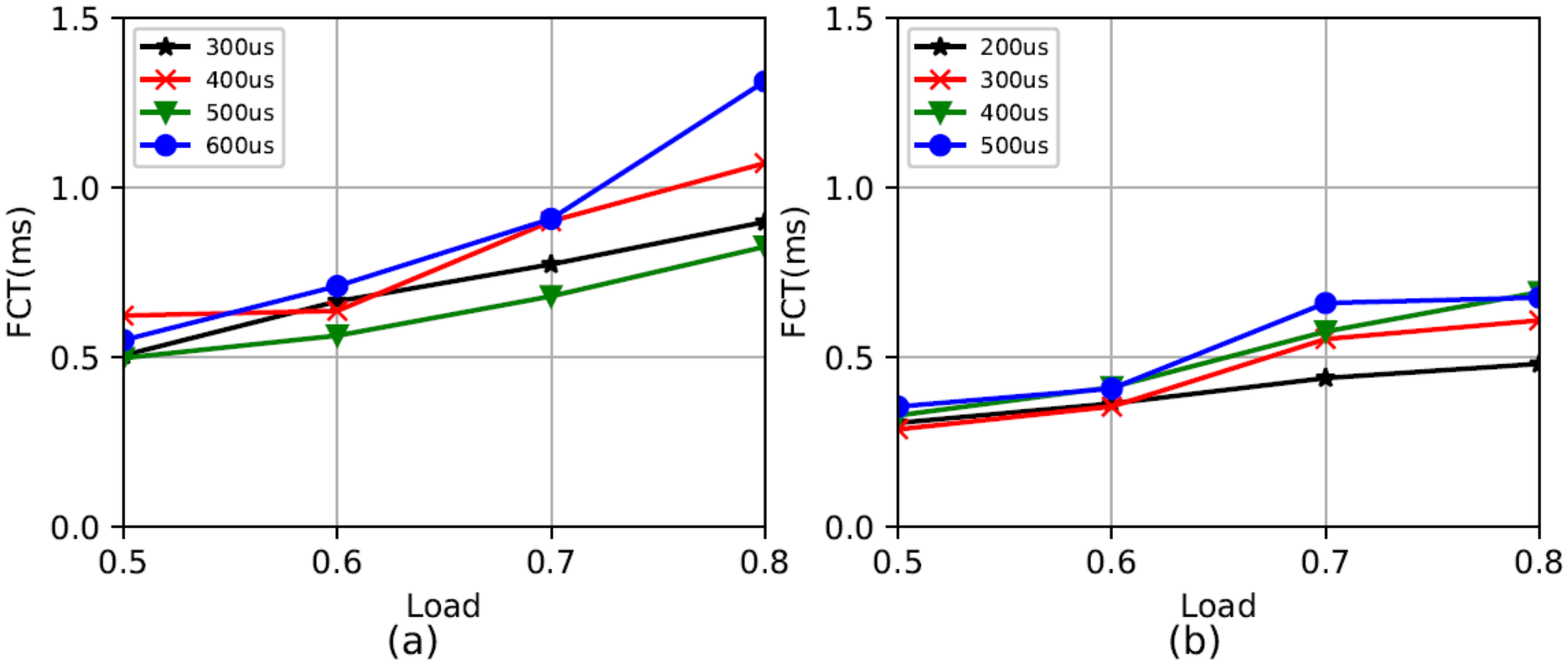


Data plane and control plane pipelines

Demotion Threshold

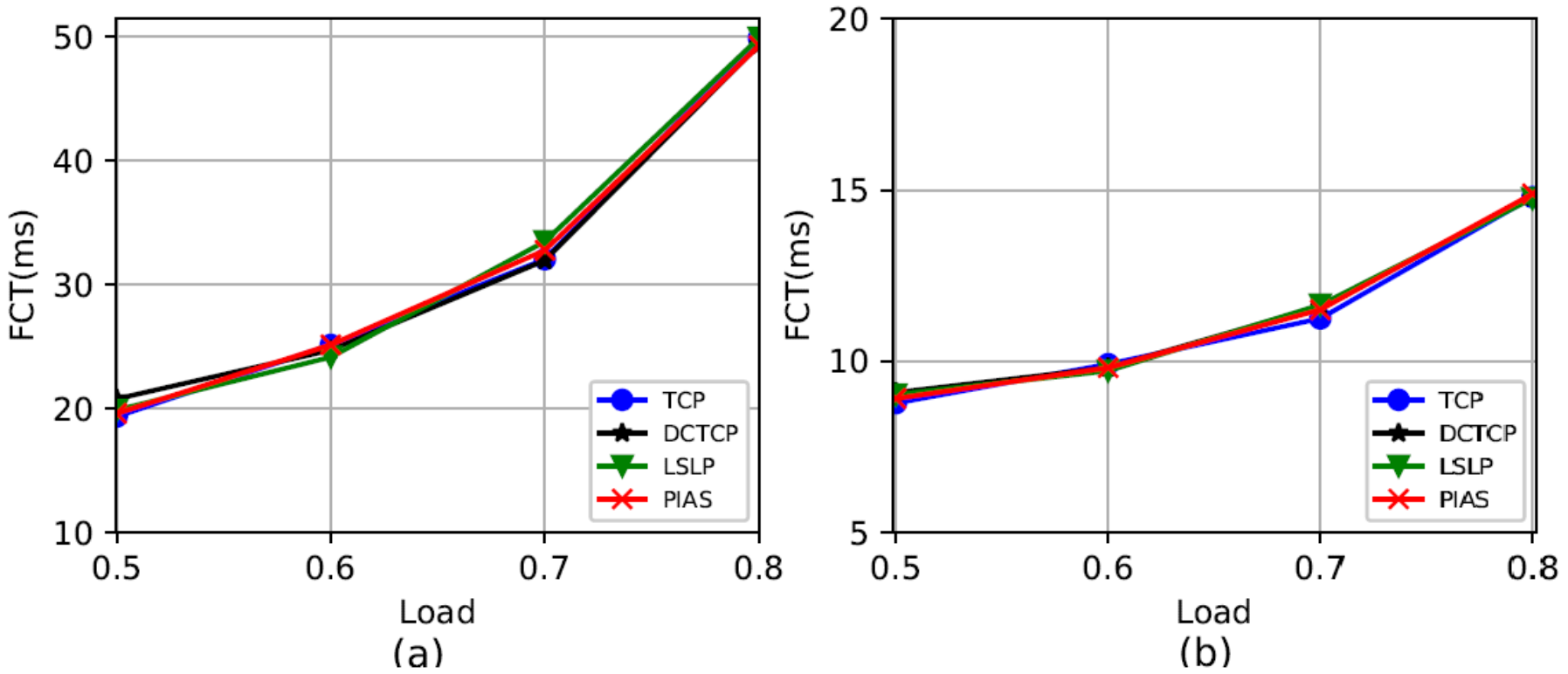
- A time interval after which a flow is demoted by one level
- Current settings
  - K priorities P_i, $1 <= i <= k$, while we consider $P_1 > P_2 > \ldots > P_k$ and α_j demotion thresholds where $j = 1, 2, \ldots, k-1$.
  - Currently, we are using a fixed demotion threshold such that
    - $\alpha_1 = \alpha_2, \ldots, = \alpha_{k-1}$

Demotion Threshold Selection



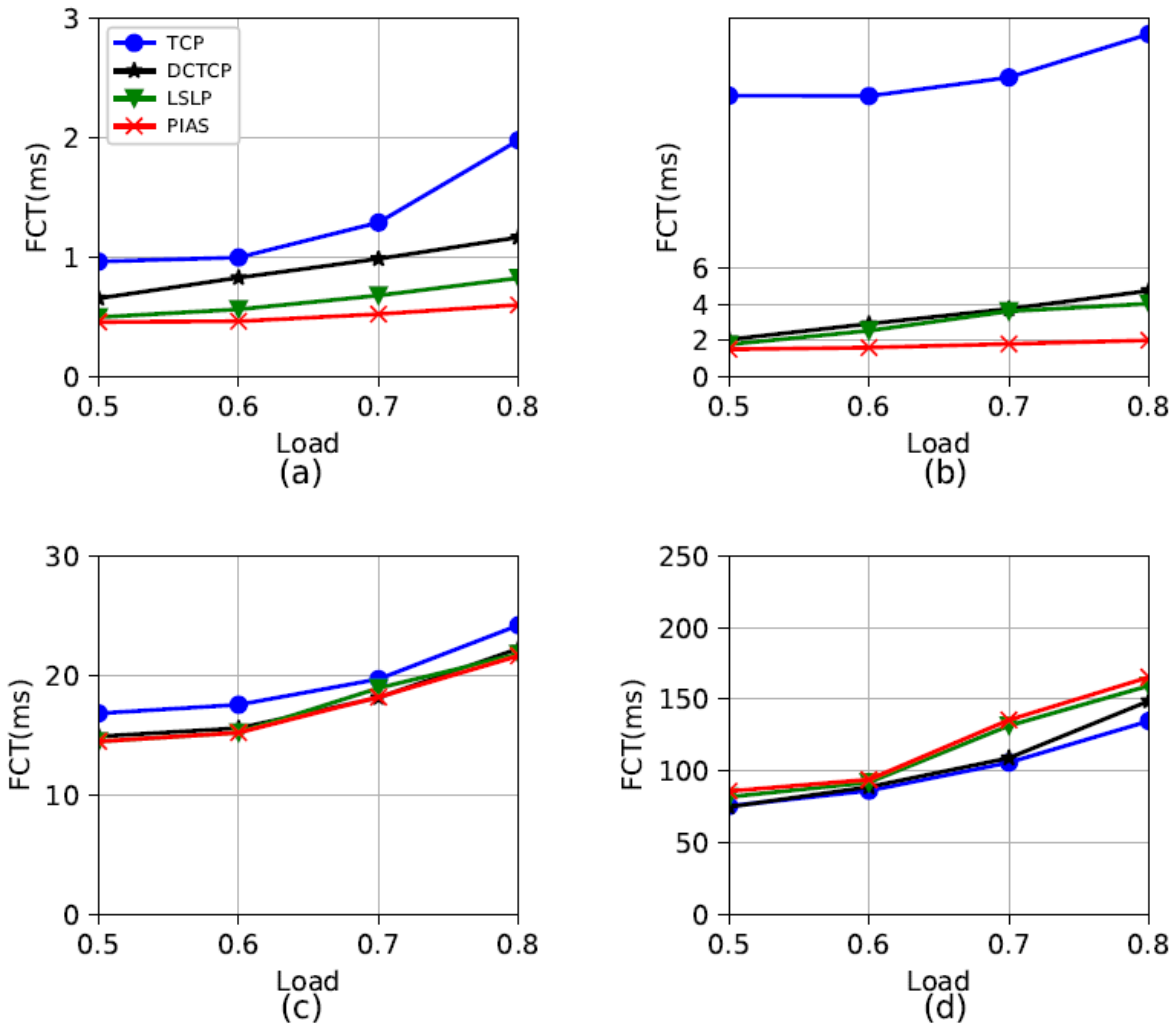
Demotion Threshold Selection (a): web search workload (b) Datamining workload

Overall average FCT



Overall average FCT at different load: (a) Web search workload, (b) Data mining workload

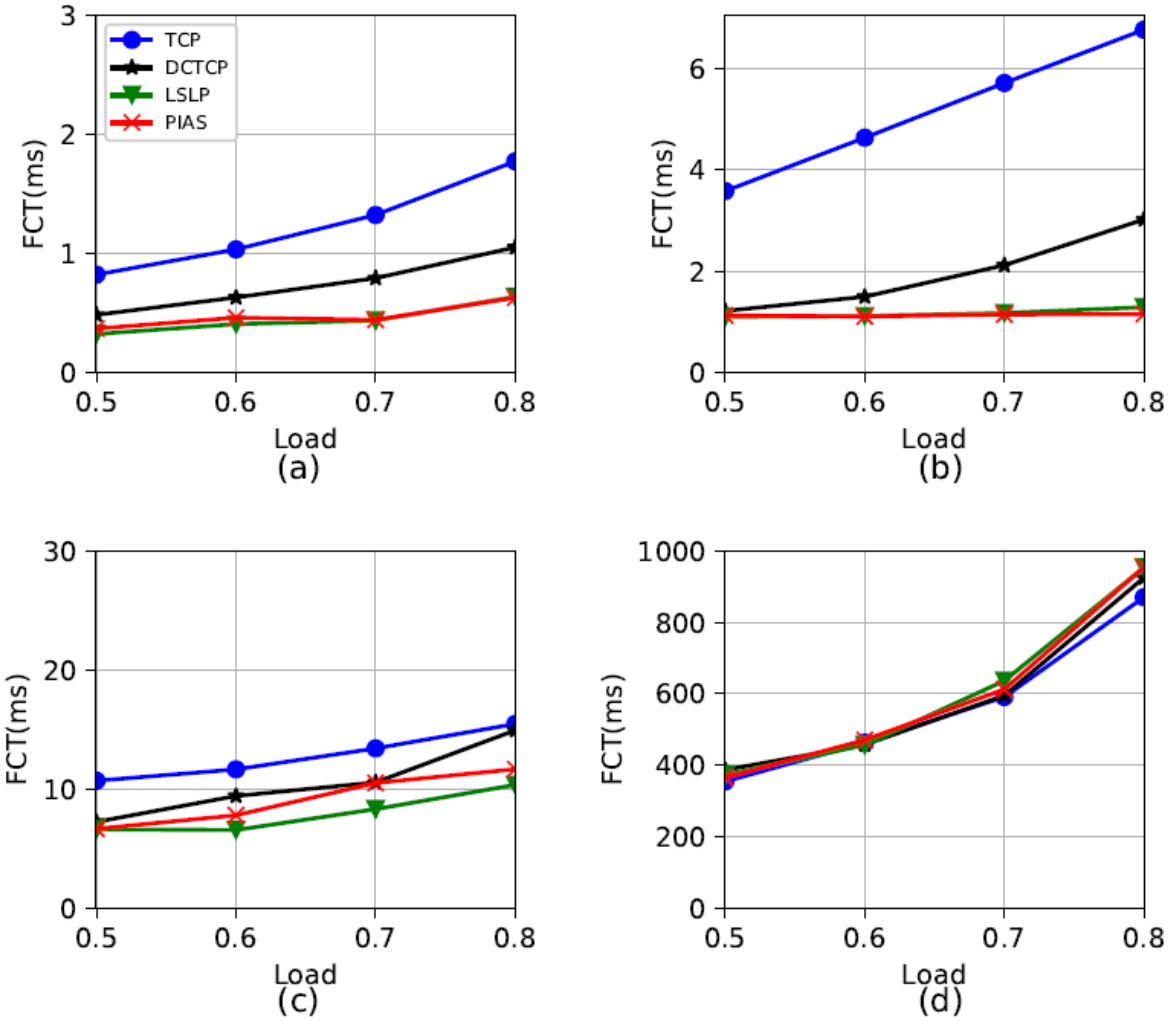
Web search workload FCT



Web Search workload: FCT across different flow sizes

- (a)(0; 100KB] Avg.
- (b) (0; 100KB] 99th Percentile
- (c) (100KB; 10MB]:
- (d)(10MB;∞]

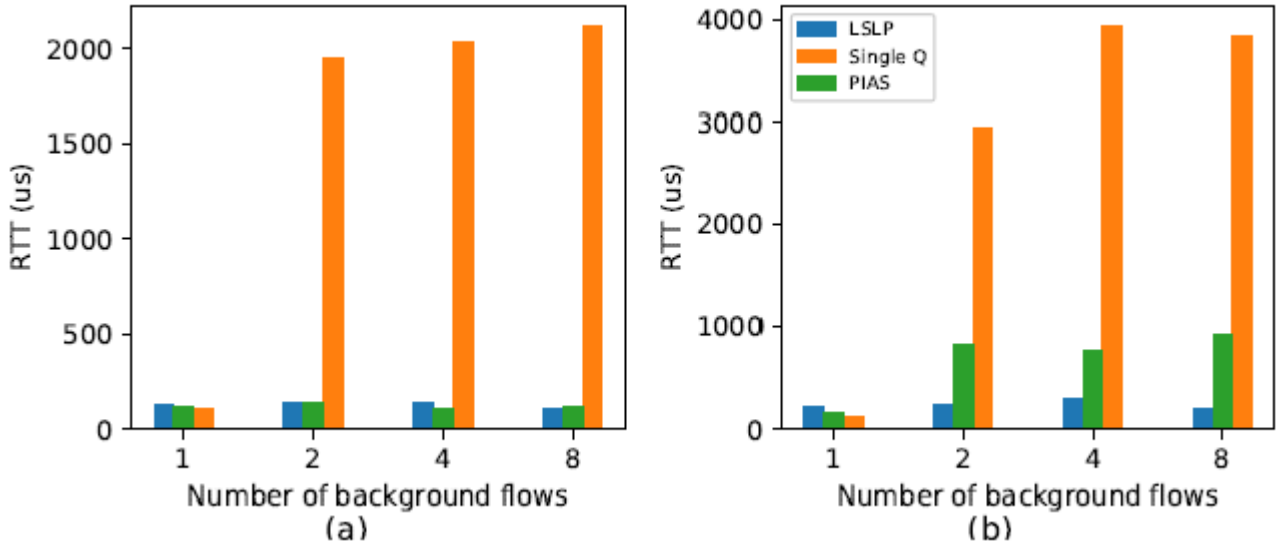
Data Mining workload FCT



Data Mining workload: FCT across different flow sizes

- (a)(0; 100KB] Avg.
- (b) (0; 100KB] 99th Percentile
- (c) (100KB; 10MB]:
- (d)(10MB; ∞]

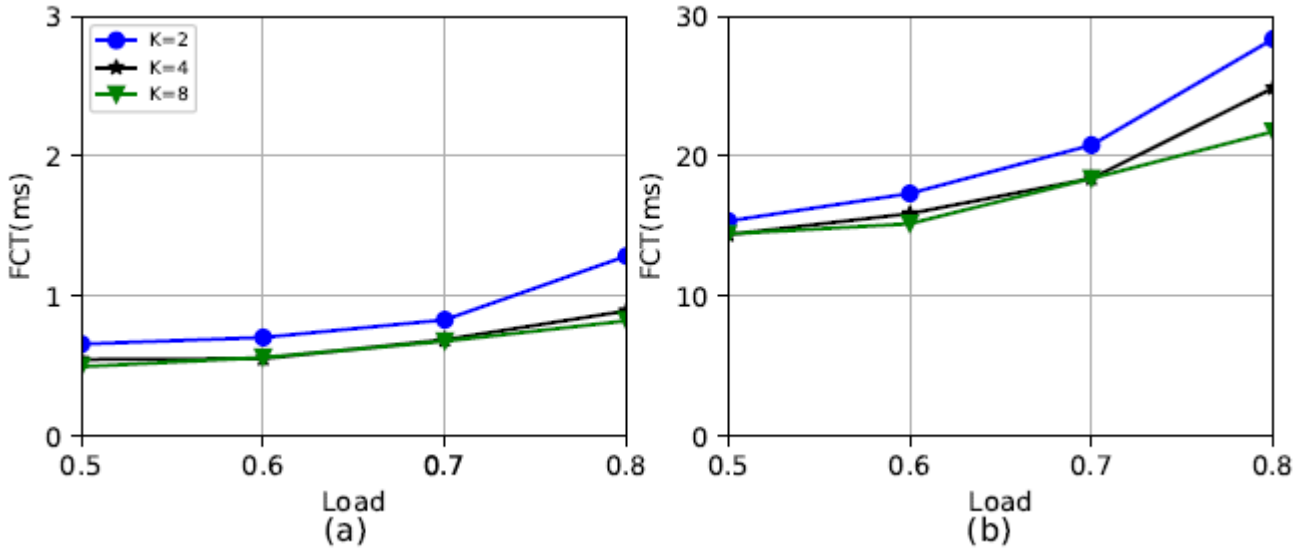
RTT with background traffic



RTT with background flows

- (a) Average RTT
- (b) 99th percentile

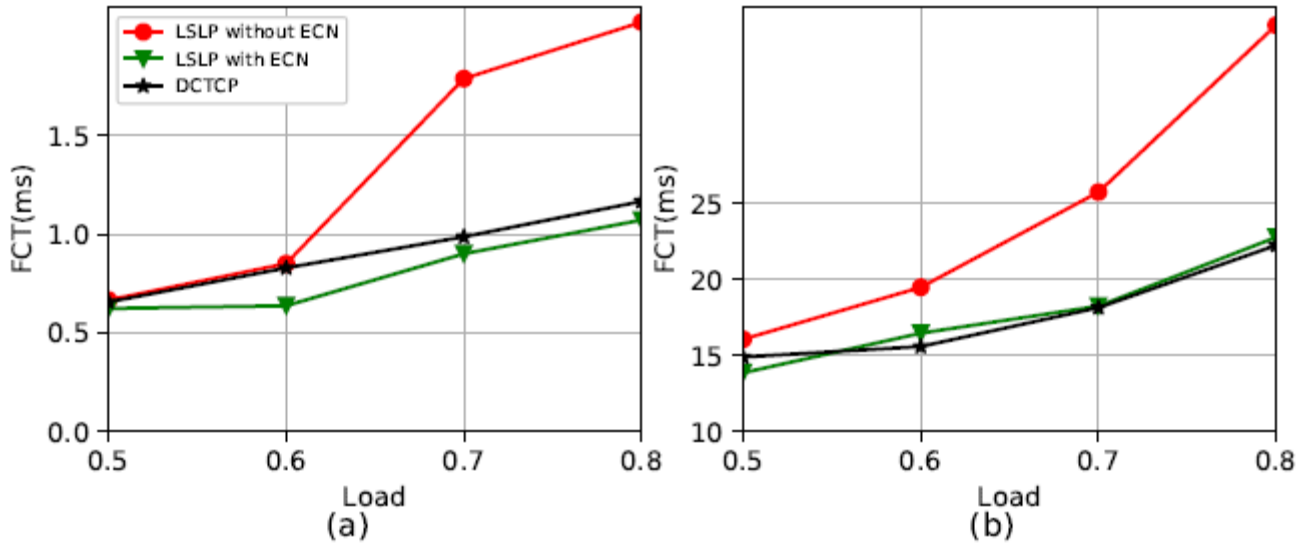
Results with different # of queues



Web search workload with different number of queues.

- (a) short(0; 100KB] Average FCT:
- (b) medium (100KB; 10MB] Average FCT

Mismatch demotion threshold



Web search workload with mismatch demotion thresholds.

- (a) short (0; 100KB] Average FCT
- (b) medium (100KB; 10MB] Average FCT

Conclusion

- Aim to minimize FCT for short flows in data center networks
  - by mimicking SJF using P4 programable switches.
- Approximates the active time of flows at the P4 switch
  - schedules in a strict priority queue.
- Evaluation shows that LSLP reduces the average FCT compared with DCTCP
  - for web search up to 29%
  - for data mining workload up to 39%
- Improves the average FCT for medium flows
  - by up to 30% for data mining workload
  - By up to 2% for web search workload
- Slightly worse than PIAS for web search workload
  - For short flows since it diminishes the counting efforts added by PIAS at thousands of end servers.
- Mismatched demotion threshold also show promising results