SplitCast: Optimizing Multicast Flows in Reconfigurable Datacenter Networks

Long Luo **Hongfang Yu**

University of Electronic Science and Technology of China
P.R. China

Klaus-Tycho Foerster **Stefan Schmid**

Faculty of Computer Science
University of Vienna
Austria

Multicast workload in datacenter network

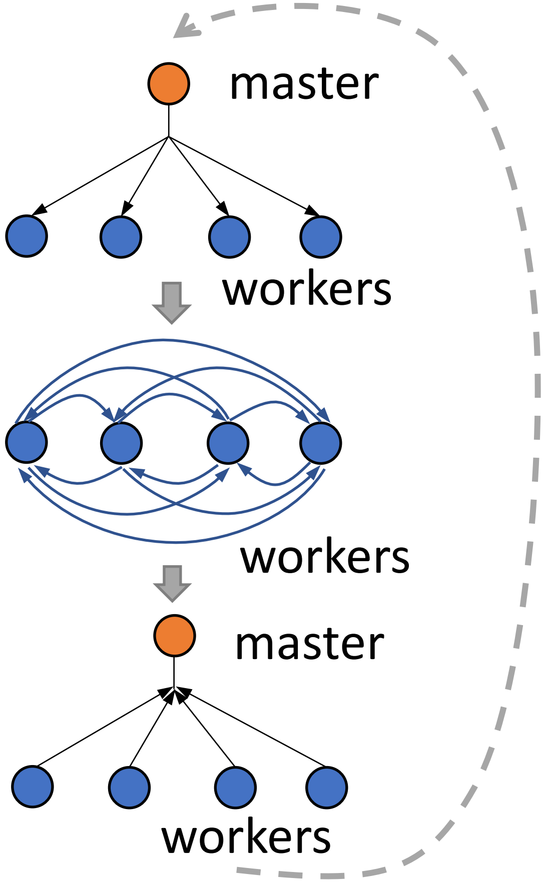
Modern datacenter applications are rife with point-to-multipoint communication patterns---multicast workloads

Database query

Data analytics applications

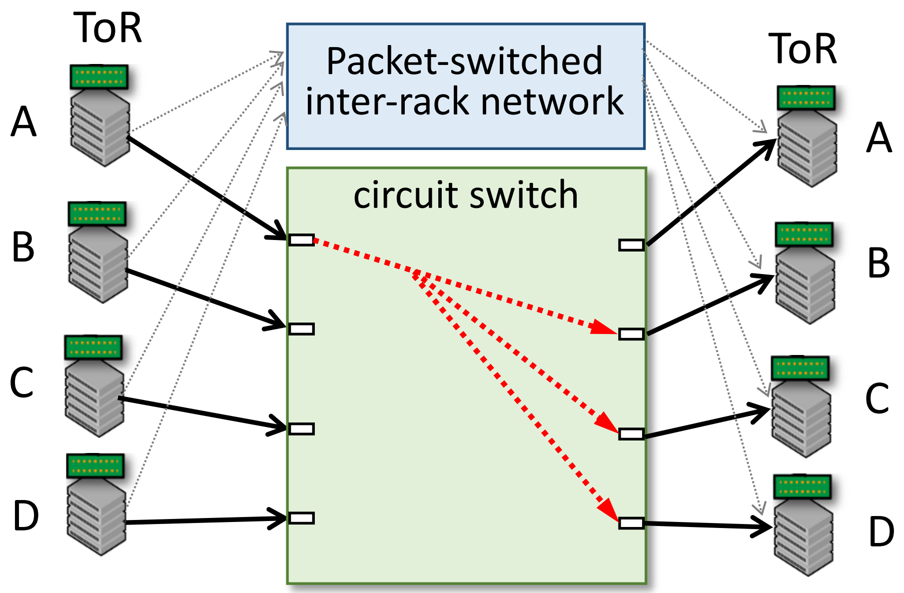
Google Cloud Pub/Sub

Publish-subscribe systems



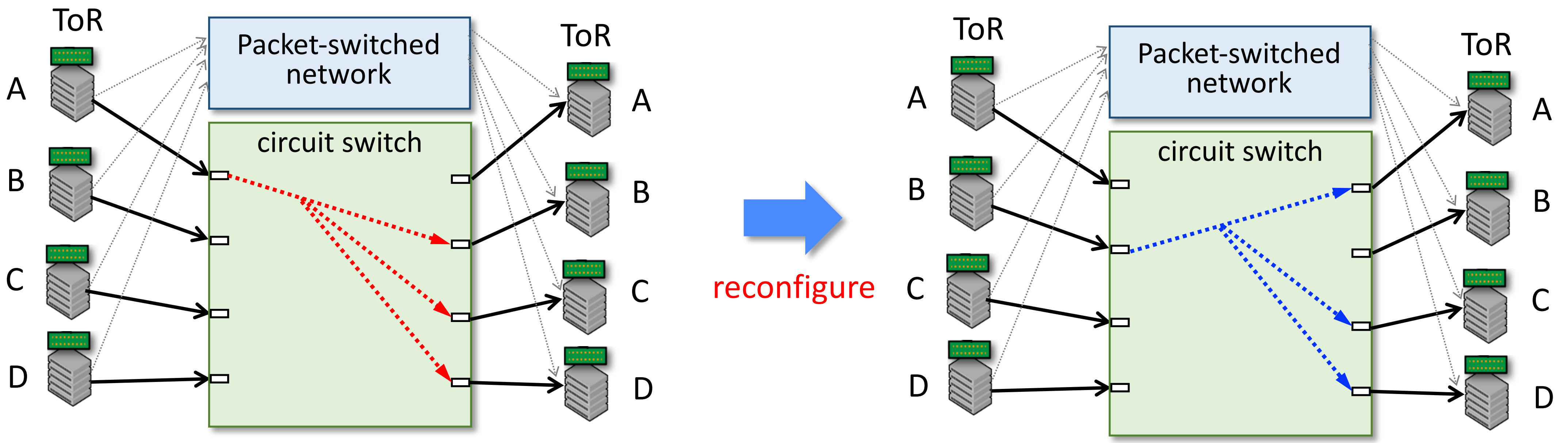
Iterative machine learning jobs

Hybrid datacenter network



The circuit switch can build directed port-to-port or port-to-multiport circuit connections between the ToRs.

Hybrid datacenter network



The circuit switch can be reconfigured to change circuit connections between the ToRs.

Intuition

- Physical layer multicasting improves the performance in transferring multicast flows
  - packets can be delivered to multiple ToRs in a single transmission
  - high-bandwidth up to 40GbE or 100GbE

- Physical layer multicasting **>** IP unicasting

- Physical layer multicasting **>** IP multicasting

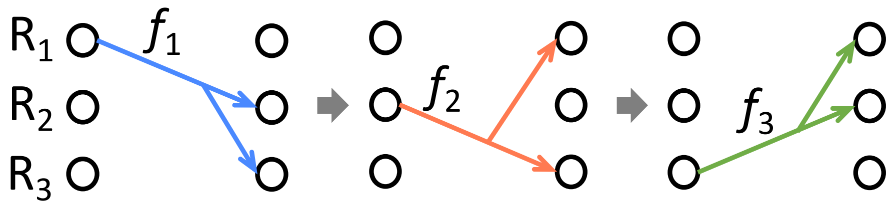
Challenges

- How to schedule multicast flows efficiently?
  - fully use the network bandwidth

Intuition #1

- unit flows f_1, f_2, f_3, each transfer data from one rack to another two

port capacity =1

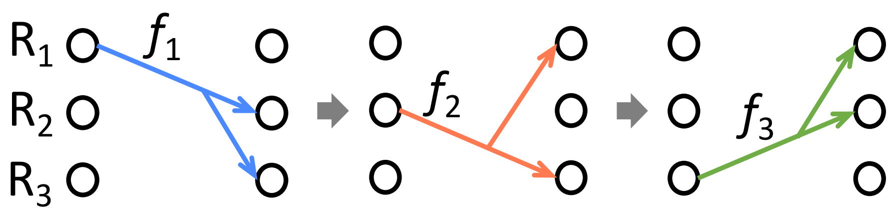


average flow time: $\frac{1+2+3}{3}=2$

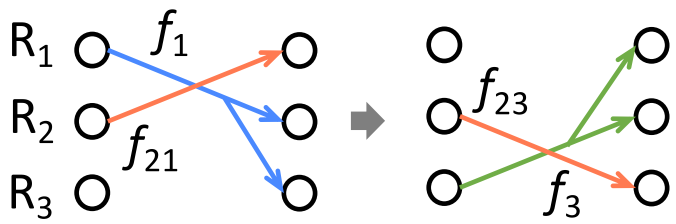
Intuition #1

- unit flows f_1, f_2, f_3, each transfer data from one rack to another two

port capacity =1



average flow time: $\frac{1+2+3}{3} = 2$



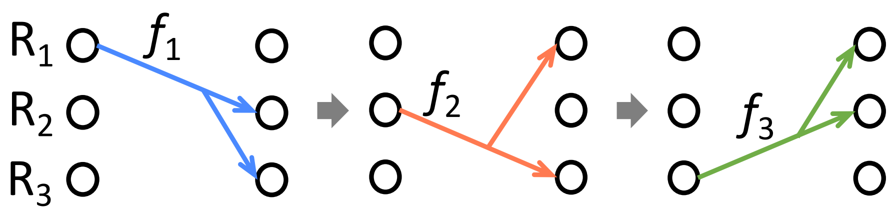
split f_2 into f_{21} and f_{23}

average flow time: $\frac{1+2+2}{3} \approx 1.67$

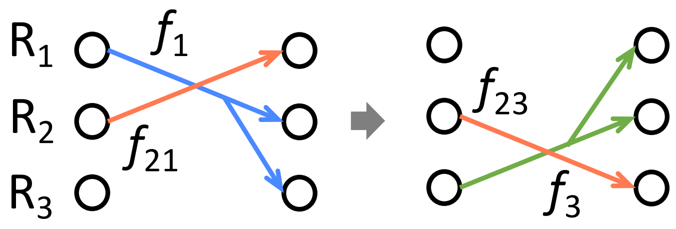
Intuition #1

- unit flows f_1, f_2, f_3, each transfer data from one rack to another two

port capacity =1



average flow time: $\frac{1+2+3}{3}=2$



split f_2 into f_{21} and f_{23}

average flow time: $\frac{1+2+2}{3}\approx 1.67$

unsplittable multicast < splittable multicast

Insights into the fundamental static problem

- The (**unsplittable**) multicast matching problem
  - Equivalent to a specific hypergraph matching problem
  - **NP-hard** even for **k = 2** receivers per transfer
  - If each source has at most one transfer:
    - **Polynomial-time** for **k = 2** receivers per transfer
    - **NP-hard** for every **k > 2**

Insights into the fundamental static problem

- The (**unsplittable**) multicast matching problem
  - Equivalent to a specific hypergraph matching problem
  - **NP-hard** even for **k = 2** receivers per transfer
  - If each source has at most one transfer:
    - **Polynomial-time** for **k = 2** receivers per transfer
    - **NP-hard** for every **k > 2**

When considering **splittable** case:

- **Polynomial-time** for **any k**

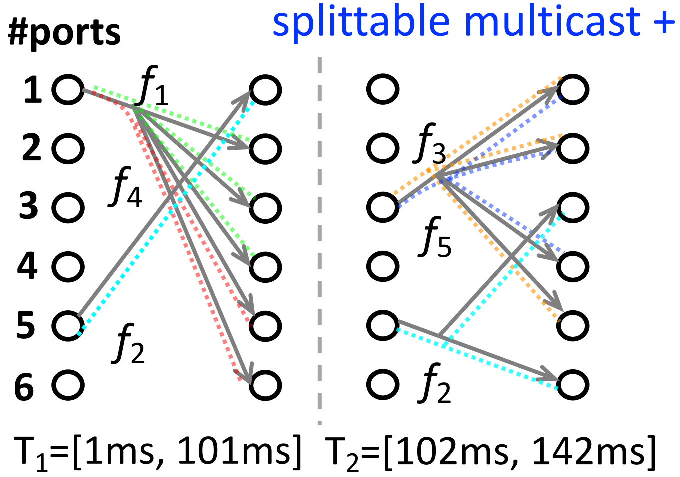
Multicast scheduling problem

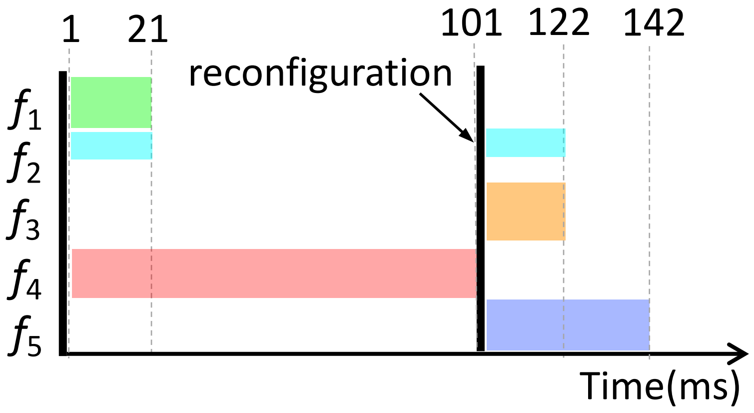
- Objectives
  - Maximizing the network throughput
  - Minimizing the flow time

- Which circuit connections should be configured?
- When to preempt flows and reconfigure circuit connections?

Intuition #2

splittable multicast + non-preemptive scheduling



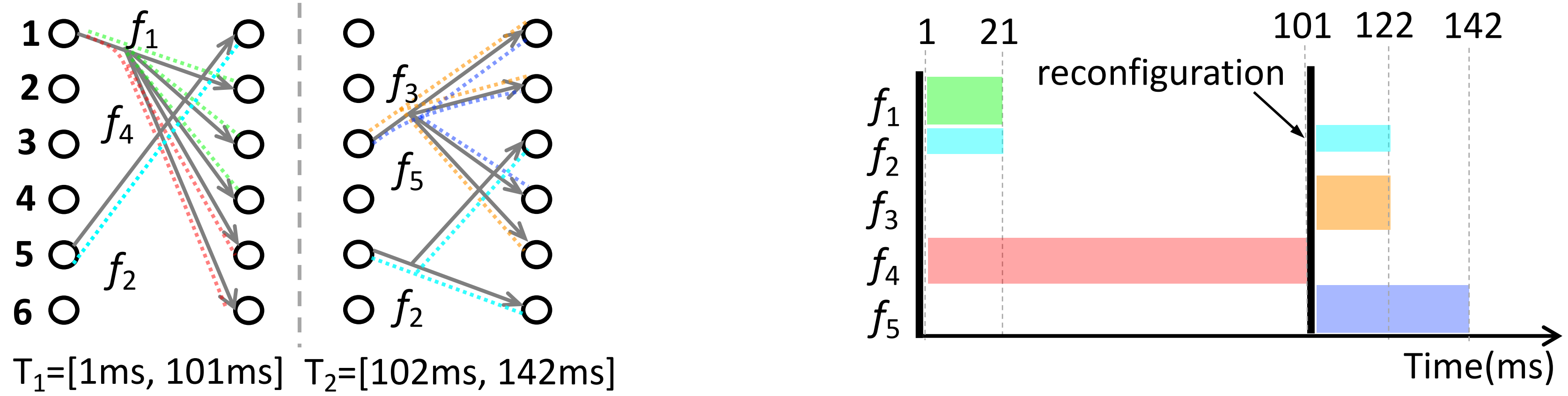


#Flows	sender-receivers	time needs to complete
f_1	1→2,3,4	20ms
f_2	5→1,3,6	20ms
f_3	3→1,2,5	20ms
f_4	1→5,6	100ms
f_5	3→1,2,4	40ms

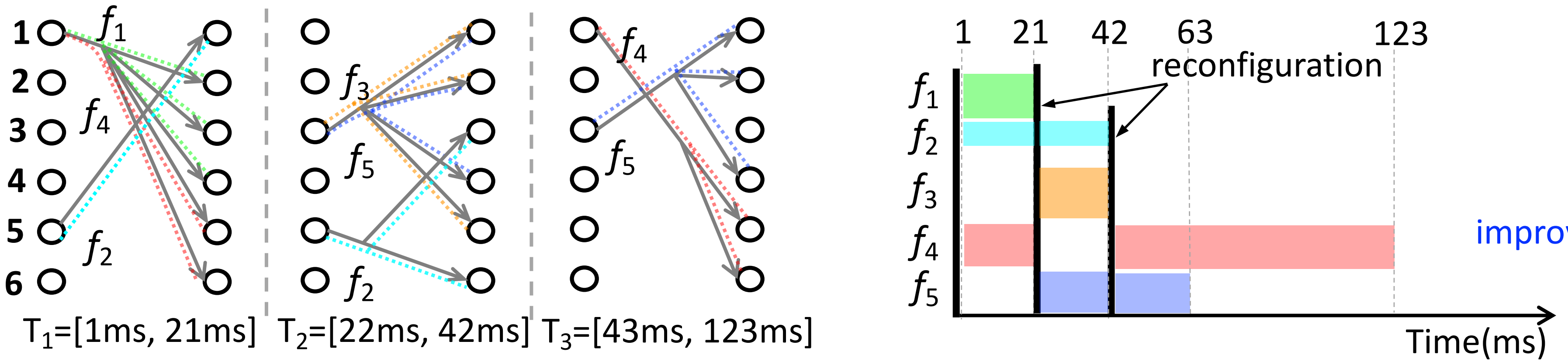
Intuition #2

#Flows	sender-receivers	time needs to complete
f_1	1→2,3,4	20ms
f_2	5→1,3,6	20ms
f_3	3→1,2,5	20ms
f_4	1→5,6	100ms
f_5	3→1,2,4	40ms

splittable multicast + non-preemptive scheduling



splittable multicast + preemptive scheduling



improve average flow time by 1.74×

Intuition #2

splittable multicast > unsplittable multicast

preemptive scheduling > non-preemptive scheduling

Solution #1

- Formulate as an optimization problem

Constraints

- Circuit switch port: each port can be involved in one connect
- Link and port capacities
- Flow sizes

Maximizing the network throughput

$$\max g(\boldsymbol{w}^t, \theta^t) = \frac{\sum_f \sum_{d \in \boldsymbol{d}_f} \min(v_{f,d}^t, b_s \theta^t) w_{f,d}^t}{(\theta^t + \delta)}$$

Minimizing the flow time

$$\min h(\boldsymbol{w}^t, \theta^t) = \sum_f \sum_{d \in \boldsymbol{d}_f} (I(v_{f,d}^t > b_s \theta^t w_{f,d}^t)(\theta^t + \delta)$$
$$+ I(v_{f,d}^t < b_s \theta^t w_{f,d}^t) \frac{v_{f,d}^t}{b_s} + t^{\text{start}} - t_f^{\text{arr}})$$

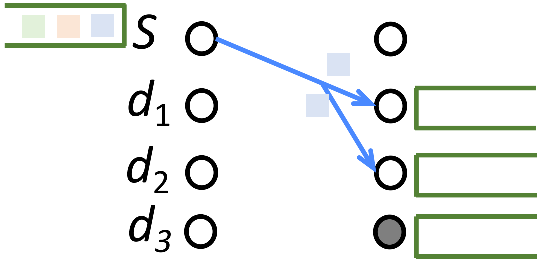
Solution #1

- Algorithm design
  - Hierarchically creating circuit connections and scheduling flows
  - Calculating the epoch length to maximize the network throughput or minimize the flow time

Challenge #2

- Receiver asynchronization

An example of transferring three units of data to three receivers

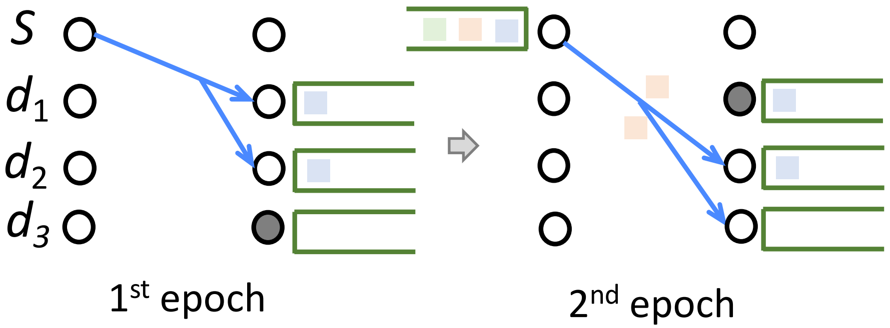


1^{st} epoch

Challenge #2

- Receiver asynchronization

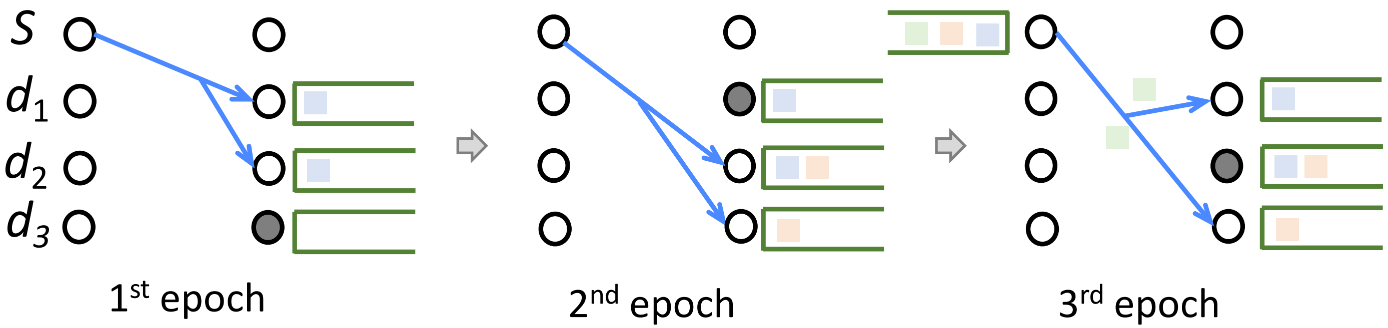
An example of transferring three units of data to three receivers



Challenge #2

- Receiver asynchronization

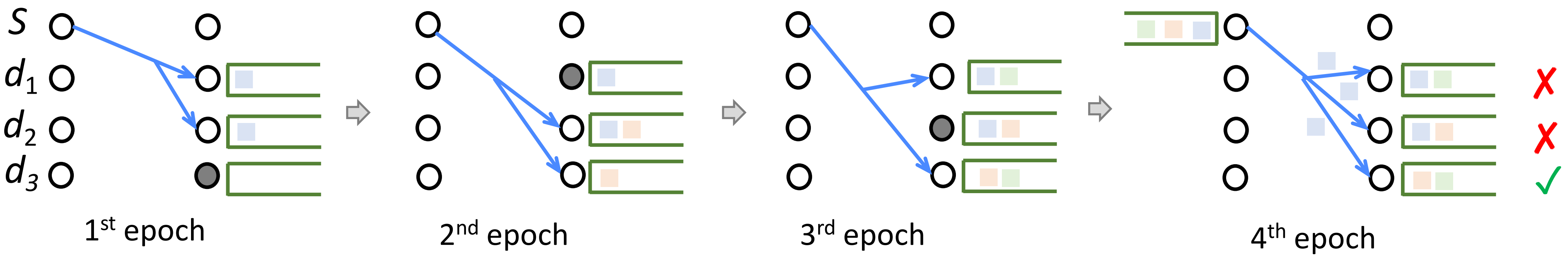
An example of transferring three units of data to three receivers



Challenge #2

- Receiver asynchronization

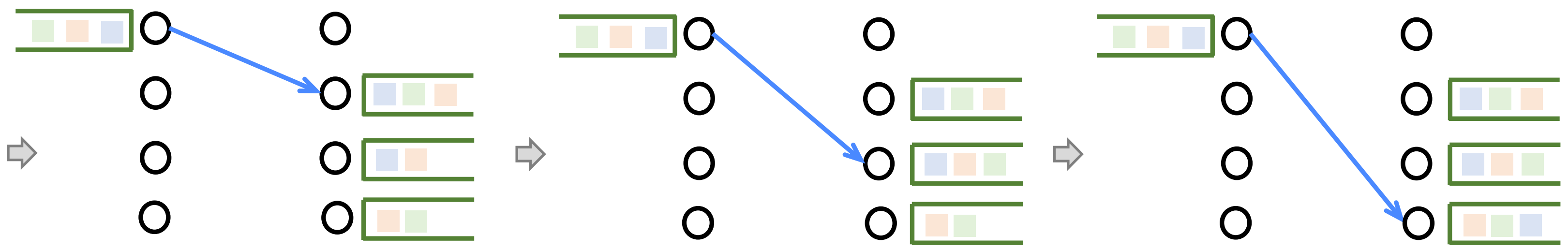
An example of transferring three units of data to three receivers



Solution #2

- Receiver asynchronization

An example of transferring three units of data to three receivers



4th epoch
duration: 1

5th epoch
duration: 1

6th epoch
duration: 1

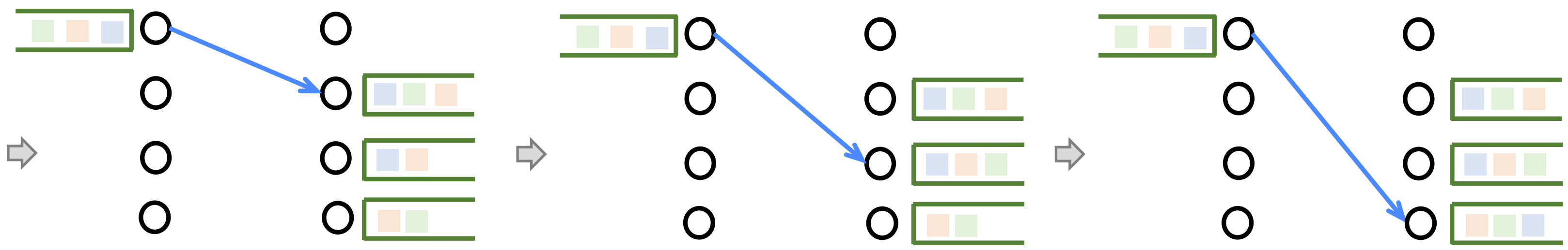
reconfigure three times

flow time: 3+3δ

Solution #2

- Receiver asynchronization

An example of transferring three units of data to three receivers



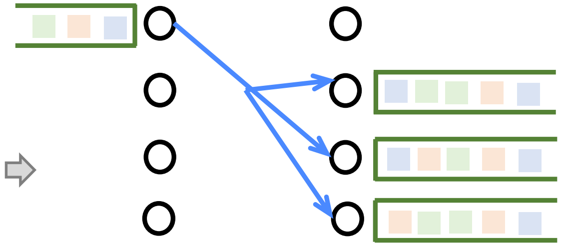
4^{th} epoch
duration: 1

5^{th} epoch
duration: 1

6^{th} epoch
duration: 1

reconfigure three times

additional flow time: 3+3δ



4^{th} epoch duration: 3

reconfigure once

additional flow time: 3+δ

tradeoff between reconfiguration and
the amount of data sent to network

Evaluation

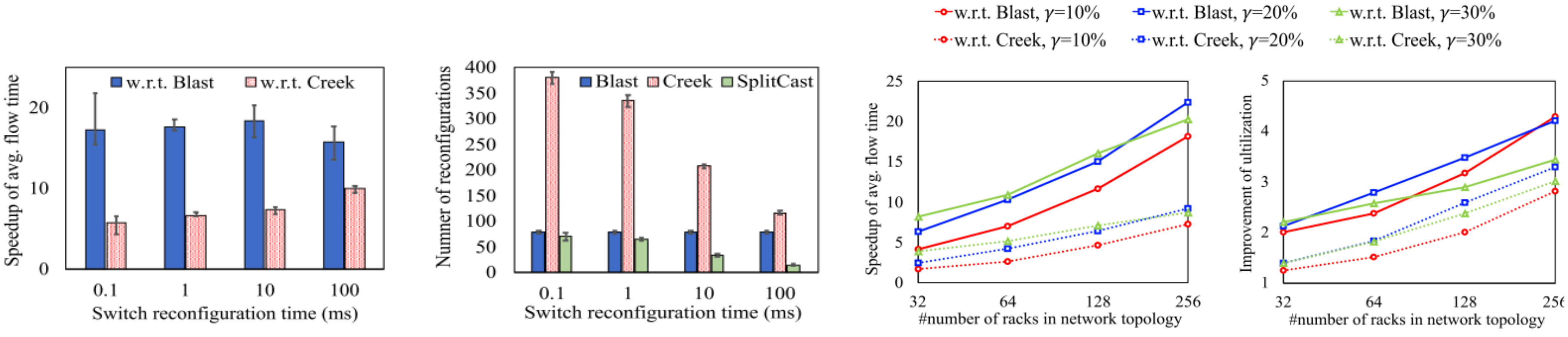
- Comparison
  - Blast: non-preemptive scheduling + unsplittable multicast

    Xia, Yiting, et. al. "Blast: Accelerating high-performance data analytics applications by optical multicast." 2015 INFOCOM.
  - Creek: preemptive scheduling + unsplittable multicast

    Sun, Xiaoye Steven, et.al. "When creek meets river: Exploiting high-bandwidth circuit switch in scheduling multicast data." 2017 ICNP
  - Splitcast: preemptive scheduling + splittable mutlicast

Evaluation

- Splitcast vs. Creek vs. Blast



(a) Speedup of average flow time (b) Number of circuit reconfigurations (c) Speedup of average flow time (d) Improvement of circuit utilization

Summary and Outlook

- We exploit enablers of reconfigurable datacenter networks:
  - in-network multicast
  - splittable multicasting
  - preemptive scheduling
  - simulations show good performance in flow time and throughput

- Outlook:
  - find and test further realistic workloads
  - Extend to multi-hop routing

Thanks!