IC2E'18

Deadline-Aware Scheduling and Routing for Inter-Datacenter Multicast Transfers

Siqi Ji*, Shuhao Liu, Baochun Li

UNIVERSITY OF TORONTO | iQua

Inter-Datacenter Traffic

- Interactive
  - Highly sensitive to loss and delay
  - Should be delivered instantly with strictly higher priority
- Elastic
  - Requires timely delivery— prior to a deadline
- Background
  - No explicit deadline or a long deadline

Why we need to consider deadlines?

- Total demand for inter-DC transfers typically far exceeds the available capacity

- Cloud providers set different data replication SLAs (or deadlines) based on delay tolerance.

- Customers are willing to pay more for guaranteed deadlines.

Multicast Transfers

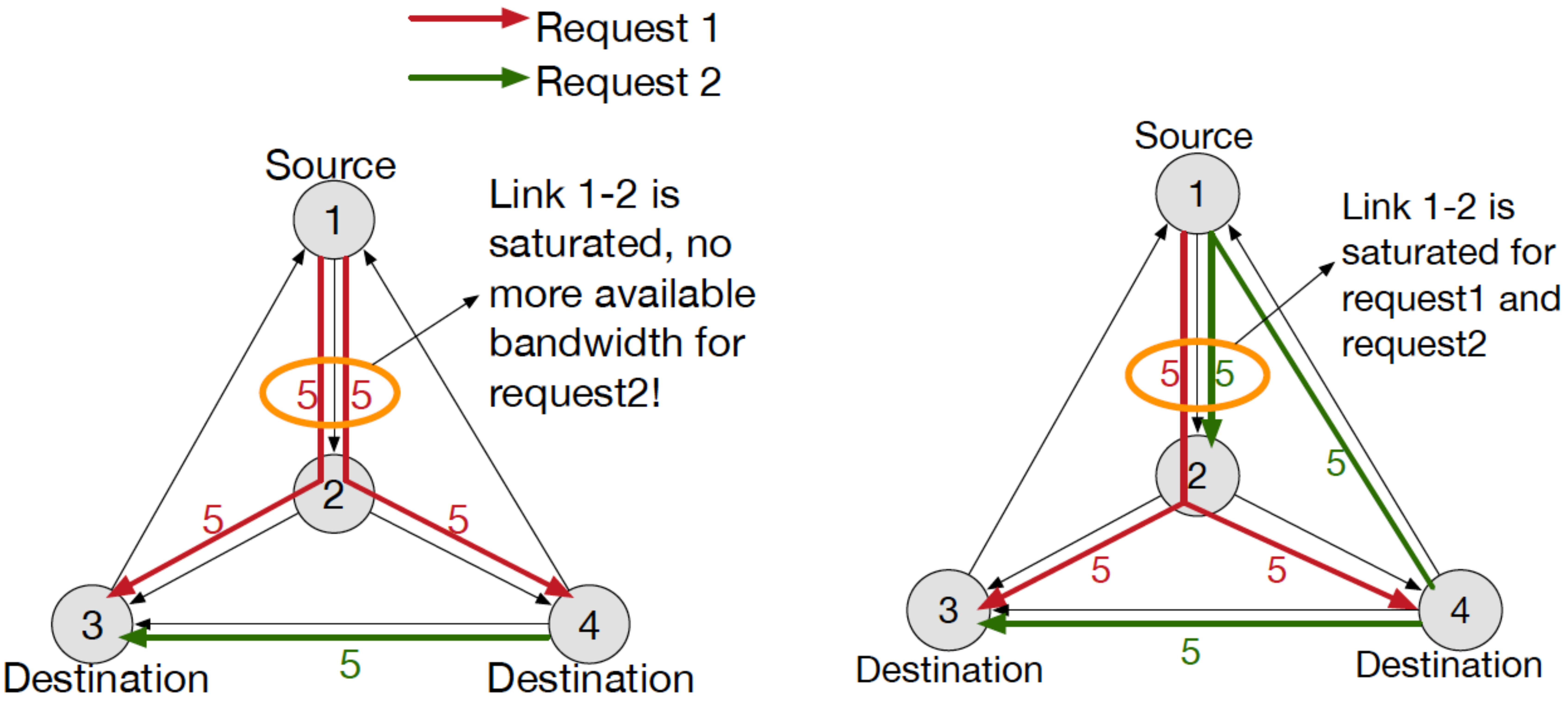
- Deliver data from one datacenter to multiple datacenters
  - Fault tolerance, availability and high service quality.
- Examples: data replication, database synchronization…
- Most of them have deadlines.

The Problem?

- Scheduling and allocating bandwidth for multiple inter-datacenter multicast transfers.
  - Meet deadline requirements.
  - Maximize throughput.

Motivation Example

Requests	Source	Destinations	Volume (MB)	Deadlines (seconds)
R_1	1	3, 4	200	40
R_2	4	2, 3	200	40



Previous Work

- Unicast transfers:
  - SWAN [sigcomm’13], B4 [sigcomm’13], BwE [sigcomm’15], Tempus [sigcomm’14], Amoeba [eurosys’15]
- DCCast [hotcloud’17] and DDCCast[tech report]:
  - Did not maximize throughput
  - Not effective for requests that require high bandwidth

Deadline Transfers

- Considering there are n transfers, a transfer request i can be specified as a tuple $\{S^i, \mathrm{R}^i, \mathrm{Q}^i, \mathrm{D}^i\}$:
  - S^i: source datacenter of request i
  - R^i: destination datacenters of request i
  - Q^i and D^i : data volume and deadline requirements of request i
- Objective: Maximize throughput for all transfers with the consideration of meeting deadlines.

Linear Program

$T^i = \{t \mid t \text{ is a Steiner tree (or multicast tree) from } S^i \text{ to } R^i\}$.

$$\begin{aligned}
&\text{maximize} \quad \chi && (1)\\
&\text{subject to} \quad \chi \le \sum_{t \in T^i} x^i(t), \forall \mathrm{i} = 1, \ldots, \mathrm{n}, && (2)\\
&\sum_{i=1}^{n} \sum_{t \in T^i} x^i(t)\, \phi(t, e) \le C(e), \forall \mathrm{e} \in E, && (3)\\
&D^i \sum_{t \in T^i} x^i(t) \ge Q^i, \forall \mathrm{i} = 1, \ldots, \mathrm{n}, && (4)\\
&x^i(t) \ge 0, \chi \ge 0, \forall t \in T^i, \forall \mathrm{i} = 1, \ldots, \mathrm{n}. && (5)
\end{aligned}$$

Maximize throughput: the sum of flow rates in all selected trees

The summation of trees' flow rates that use edge e should not exceed the edge capacity

All transfers will complete prior to deadlines

where ϕ is defined as:

$$\phi(t, \mathrm{e}) = \begin{cases} 1, & \text{if } \mathrm{e} \in t, \\ 0, & \text{otherwise.} \end{cases}$$

Sparse Solution

- Reduce splitting and packet reordering overhead
- We add a penalty function at the objective to get a sparse solution

$$\text{maximize} \quad \chi - \mu \sum_{i=1}^{n} \sum_{t \in T^i} g\left(x^i\left(t\right)\right),$$

$$g\left(x^i\left(t\right)\right) = \begin{cases} 0, & \text{if } x^i\left(t\right) = 0, \\ 1, & \text{if } x^i\left(t\right) > 0. \end{cases}$$

Sparse Solution

- We can linearize the penalty function by using a l1-norm weighted heuristic.

$$\text{maximize} \quad \chi - \mu \sum_{i=1}^{n} \sum_{t \in T^i} \left(W^i(t) \cdot x^i(t) \right),$$

- In each iteration we recalculate the weight function W^i where:

$$W^i(t) = \frac{1}{\left(x^i(t)\right)^k + \delta}.$$

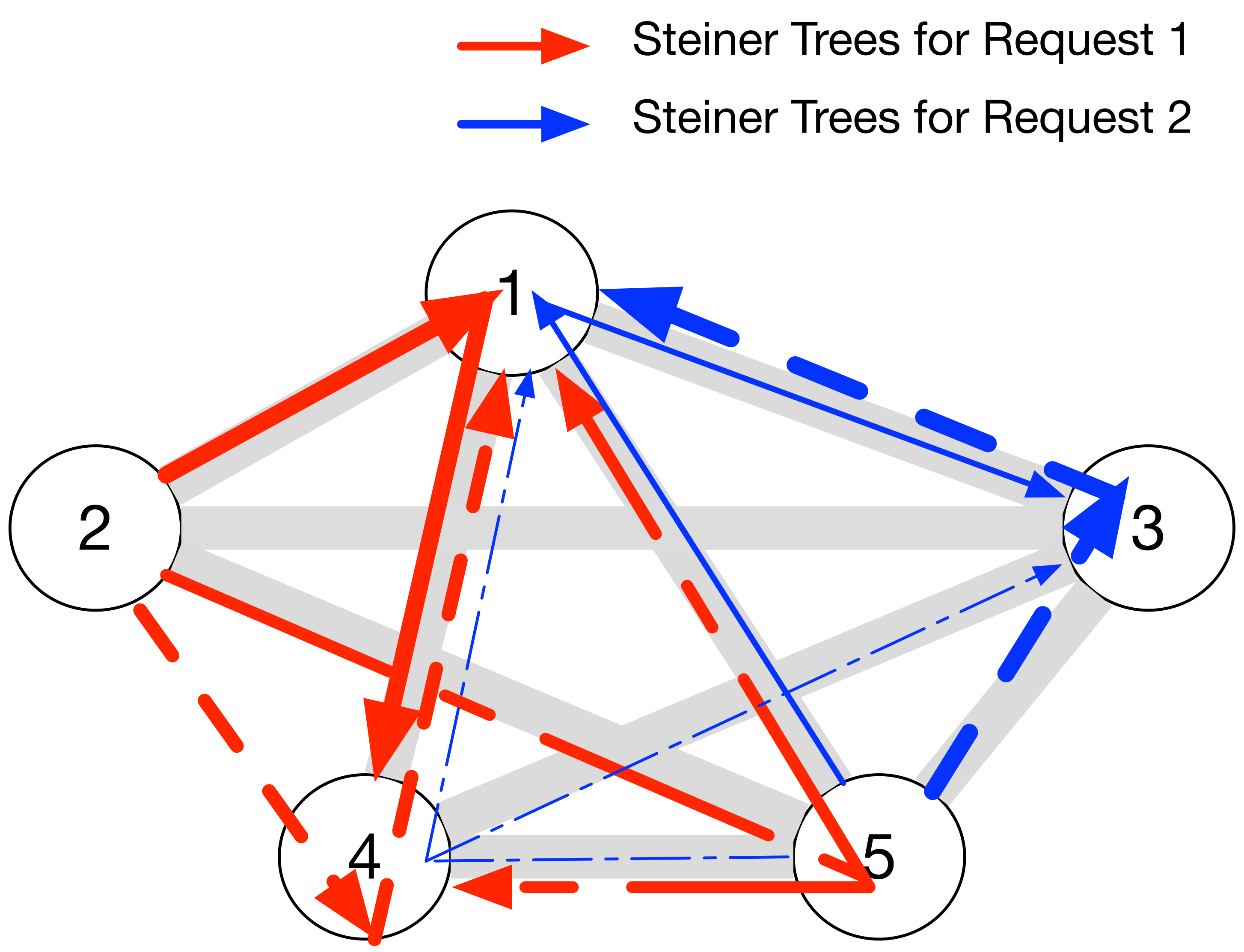
Sparse Solution

- Upon convergence, $\left(x^i(t)\right)^k \approx \left(x^i(t)\right)^{k+1} = \left(x^i(t)\right)^*$, for i = 1, ..., n, t $\in T^i$

$$W^i(t) \cdot \left(x^i(t)\right)^* = \frac{\left(x^i(t)\right)^*}{\left(x^i(t)\right)^k + \delta} = \begin{cases} 0, & \text{if } \left(x^i(t)\right)^* = 0, \\ 1, & \text{if } \left(x^i(t)\right)^* > 0. \end{cases}$$

- Eventually, the transformed problem approaches the original problem and yield a sparse solution

An example of the optimal solution obtained by our linear program:



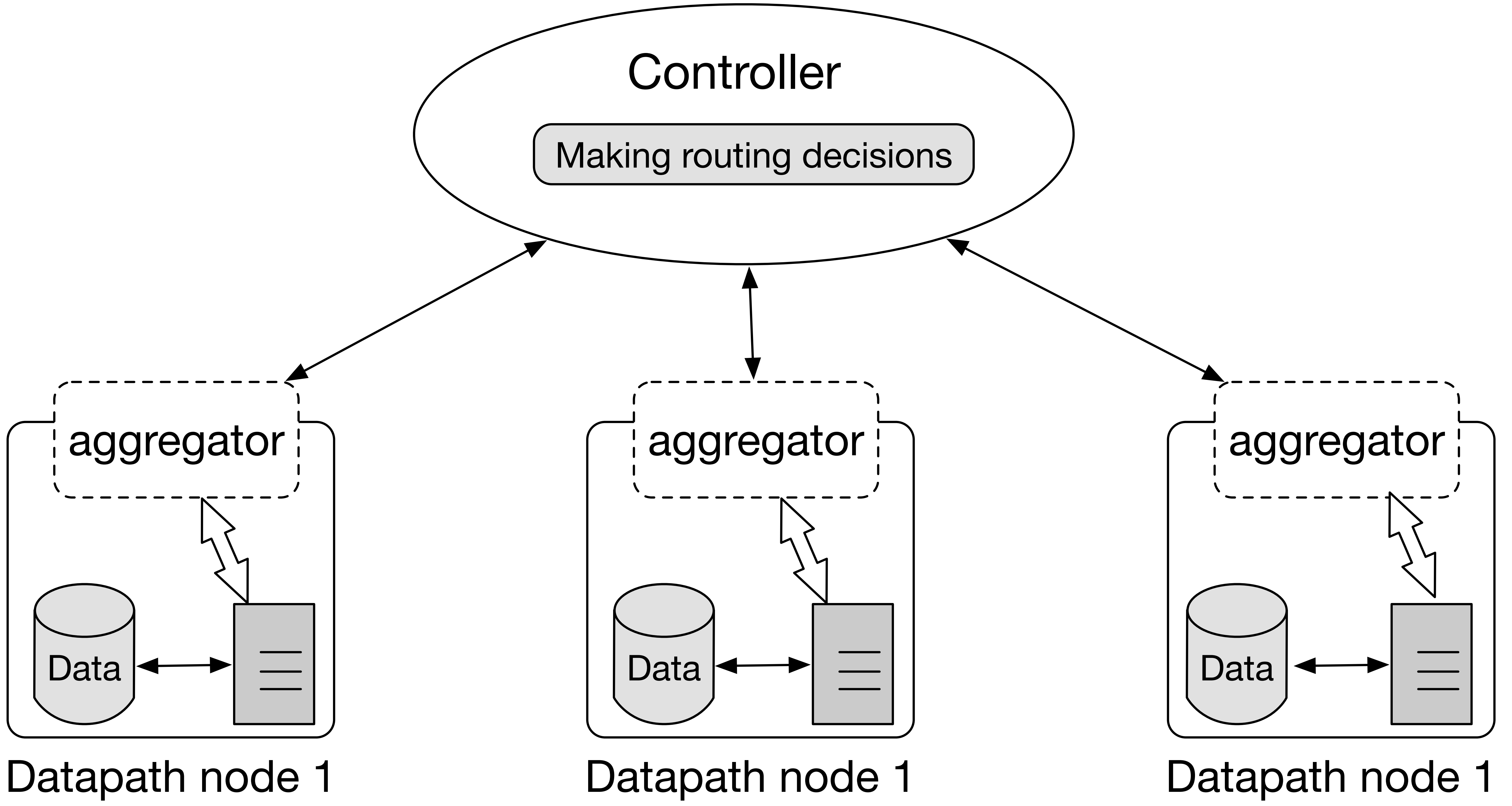
Request 1		Request 2	
Trees	Rate	Trees	Rate
2 → 1 → 4	15	5 → 1 → 3	4.56
2 → 4 → 1	12.06	5 → 3 → 1	15
2 → 5 → 1, 4	10.44	5 → 4 → 3, 1	2.94

Requests	**Source**	**Destinations**	**Volume (MB)**	**Deadline (seconds)**
R_1	2	1, 4	300	8
R_2	5	1, 3	300	18

- Assume all link capacities are 15MB/s
- If we use only one tree, the shortest completion time is 20s, all requests will miss their deadlines
- Maximize throughput, request R2 can even finish the transfer before its deadline.

Implementation

- We have completed a real-world implementation in a software-defined overlay network testbed at the application layer.

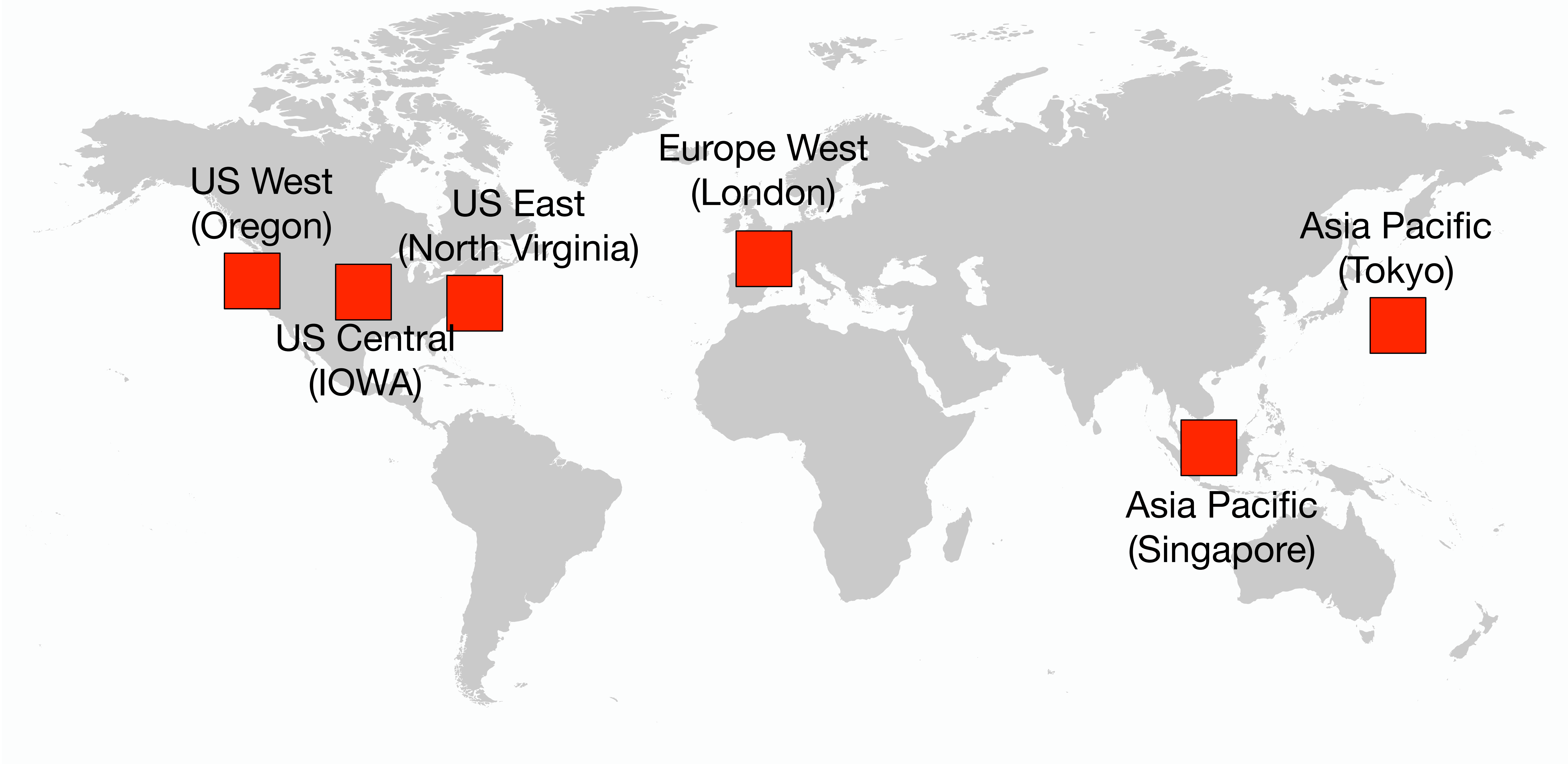


How a transfer is routed and completed through our application-layer SDN testbed?

- Destinations: subscribe to a specific channel
- Source: publish its data, destinations and deadline requirements to the channel
- Aggregator: consult the controller for routing rules
- Controller: routing rules — next hop and sending rate to each datapath node

Experiment

- Google Cloud Platform
  - Six Virtual Machines (VM) instances located in six different datacenters, and one of the VMs is also launched as the central controller.

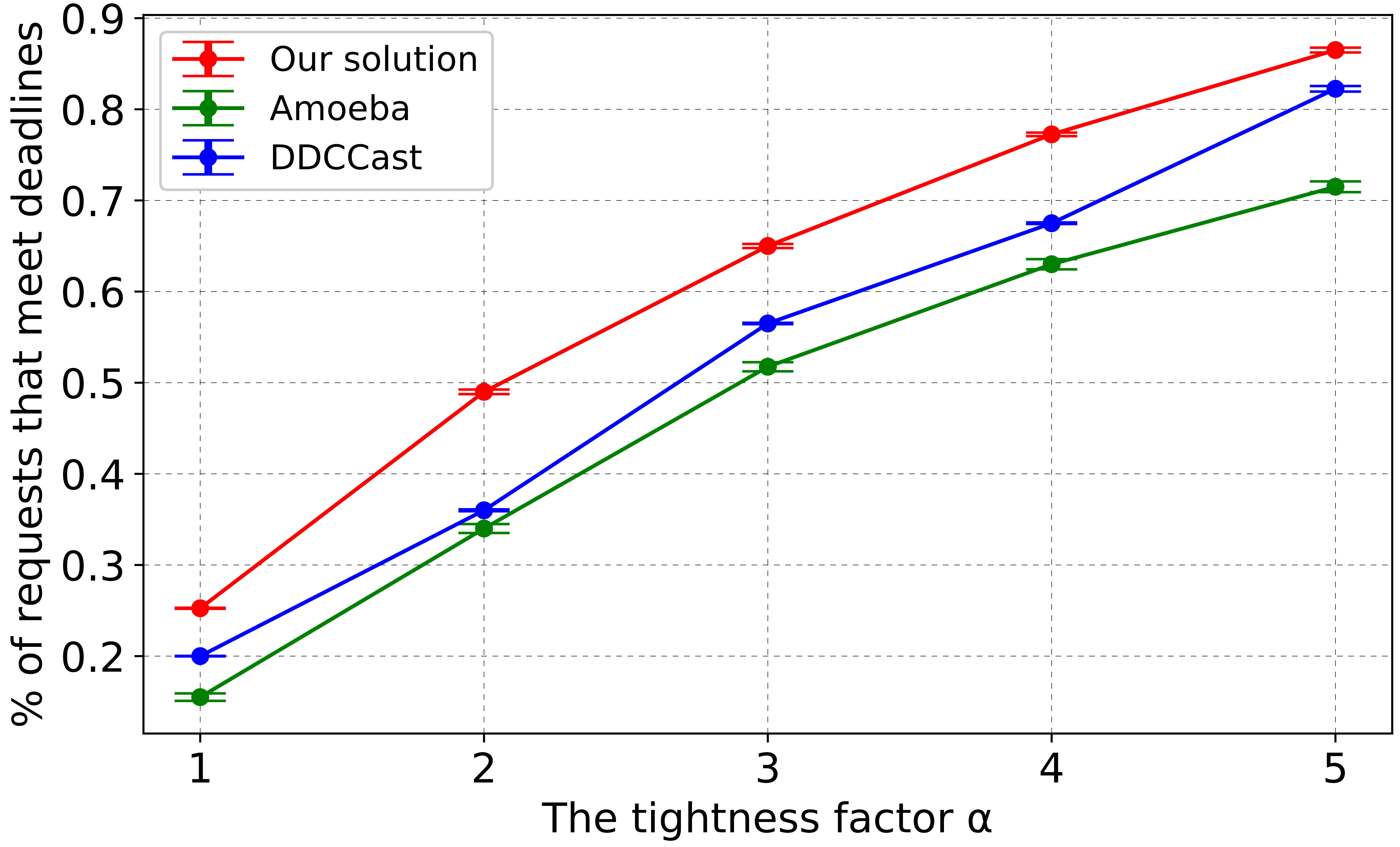


Experiment

- We use file replication as inter-datacenter traffic
  - The volume of each file is set to be 300MB
  - Deadlines: generate from a uniform distribution between $[T, \alpha\mathrm{T}]$
    - α represents the tightness of deadlines for generated transfers

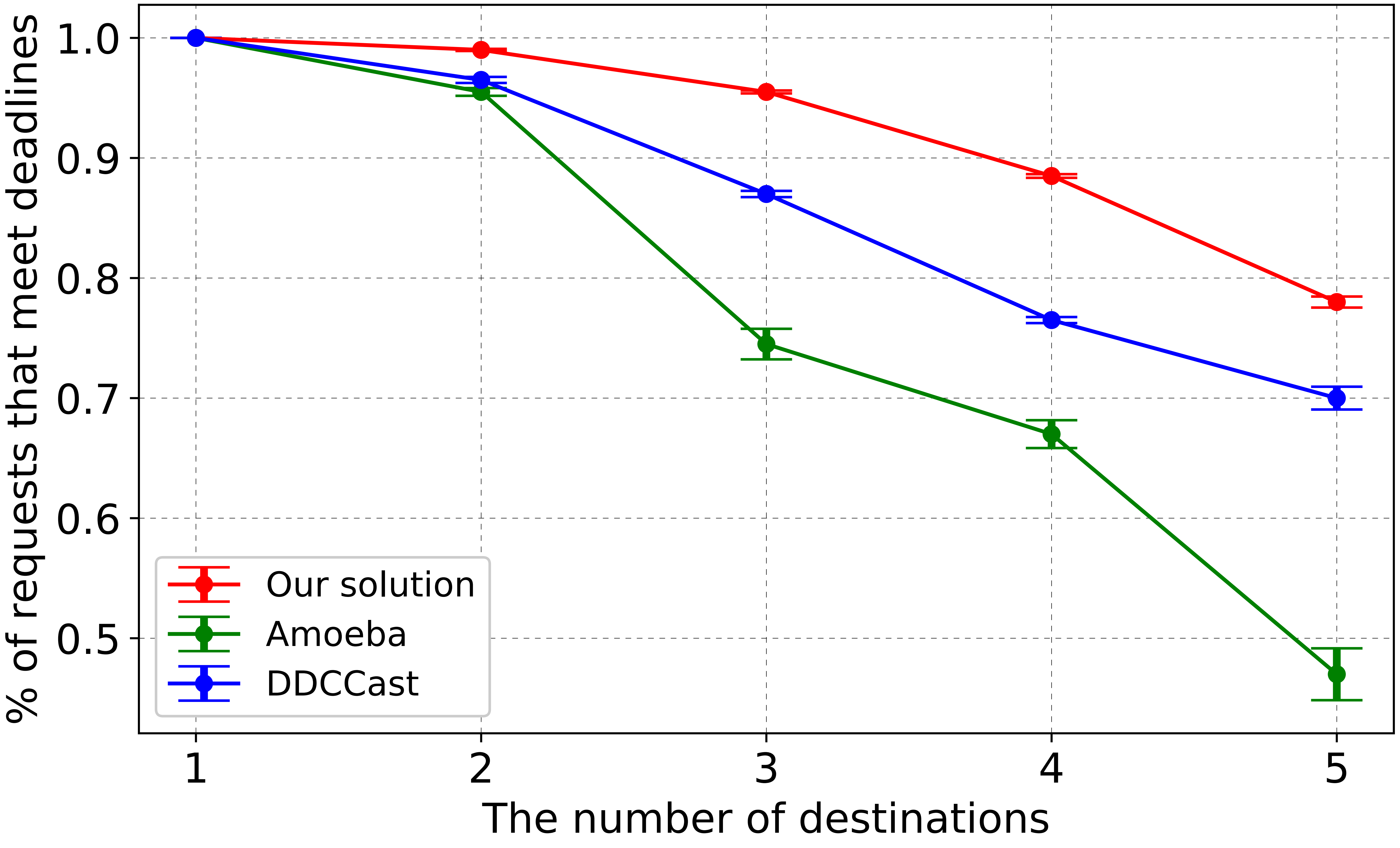
Performance Evaluation

- Comparison of different solutions as the tightness factor increases:



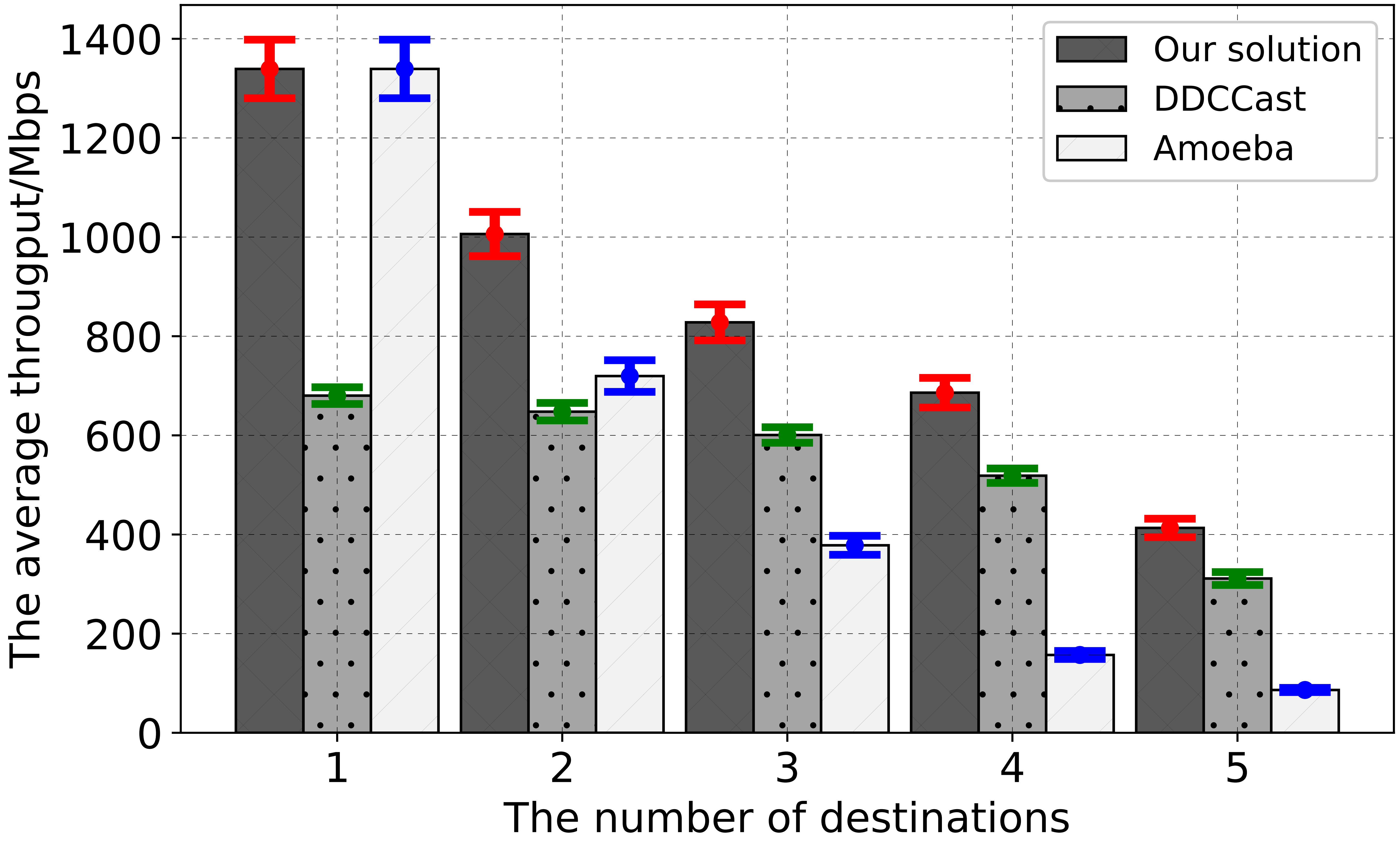
Performance Evaluation

- Comparison of different solutions as the number of destinations increases:



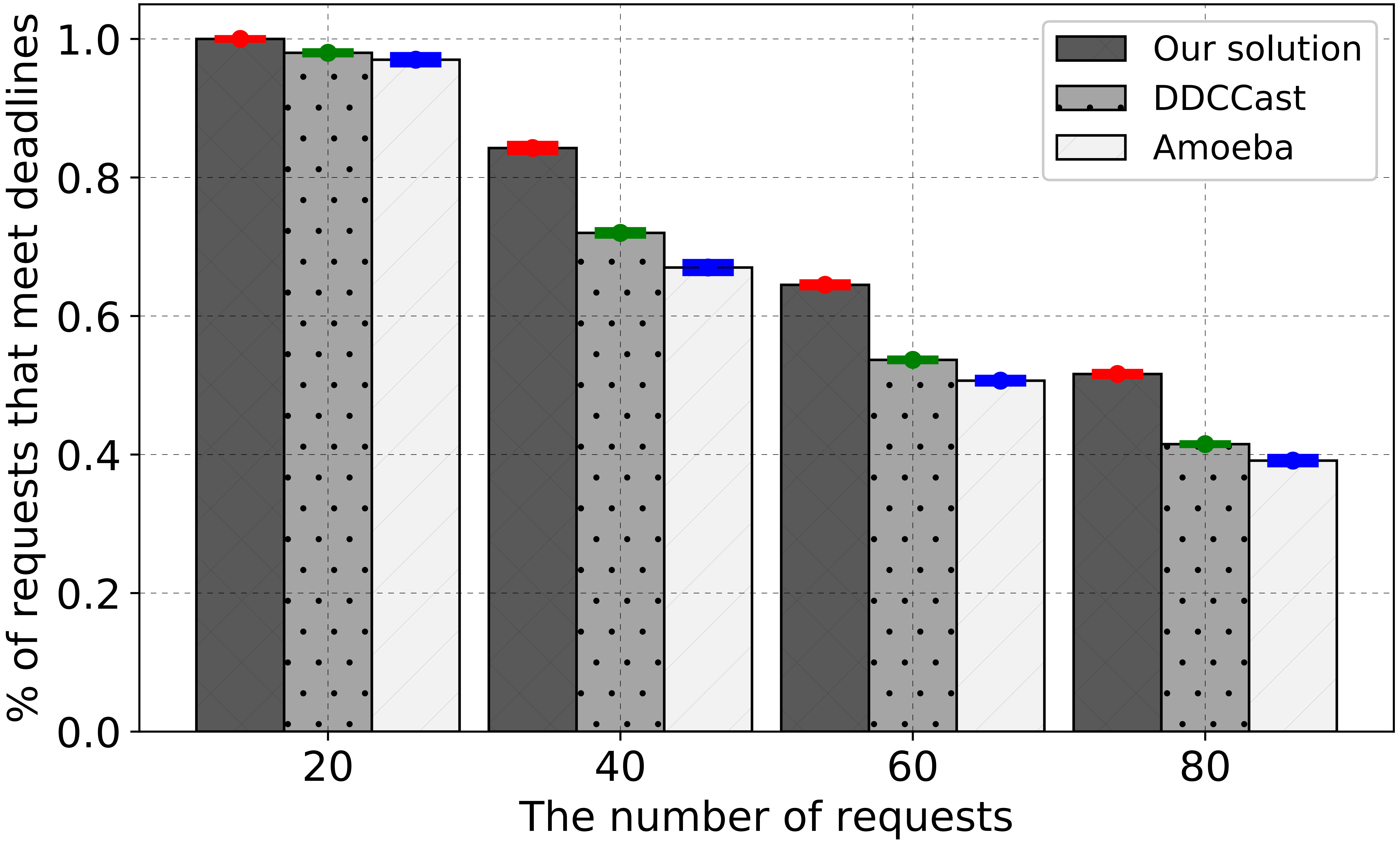
Performance Evaluation

- Throughput comparison of different solutions:



Performance Evaluation

- Comparison of different solutions as the number of requests increases:



Conclusion

- Our solution performs better in maximizing throughput and meeting transfer deadlines than related work.
- Future work:
  - Dynamic resources
  - Different request arrival rates

Thank you!
Q&A