Séminaire STORE

DCoflow: Deadline-Aware Scheduling Algorithm for Coflows in Datacenter Networks

Quang-Trung Luu, Olivier Brun, Rachid El-Azouzi,
Francesco De Pellegrini, Balakrishna J. Prabhu

LAAS–CNRS, Toulouse, France

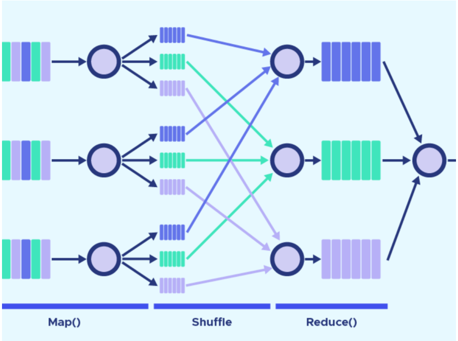
April 6^{th}, 2022

Outline

Introduction

Context

- Distributed computing frameworks such as Hadoop MapReduce or Apache Spark
- Massive data transfers in datacenter networks (e.g, shuffle phase)



- Coflow: set of concurrent flows related to a common task

Coflow scheduling

- Minimization of Coflow Completion Time (CCT)
  - ✔ Maximize the rate at which coflows are dispatched in the network fabric.
  - ✔ NP-hard, inapproximable below a factor 2
  - ✔ Near-optimal algorithms[1]
- Maximization of Coflow Acceptance Rate (CAR)
  - ✔ Strict coflow deadlines for online services and mission critical computing tasks
  - ✔ Joint coflow admission control and scheduling
  - ✔ NP-hard, inapproximable within any constant factor

1

☞ M. Shafiee et *al.*, An improved bound for minimizing the total weighted completion time of coflows in datacenters, IEEE/ACM Trans. Netw., vol. 26, no. 4, 2018.
☞ S. Agarwal et *al.*, Sincronia: Near-optimal network design for coflows. in Proc. ACM SIGCOMM, 2018.
☞ M. Chowdhury et *al.*,. Near optimal coflow scheduling in networks, in Proc. ACM SPAA, 2019.

Contributions

- Lightweight method for coflow scheduling under deadlines
  - ✔ Admission control and coflow priorities.
  - ✔ Based on known results for open-shop scheduling
- Offline and Online versions
- Extensive simulations with synthetic traffics and real traces obtained from a Facebook dataset.

Problem Formulation and Existing Works

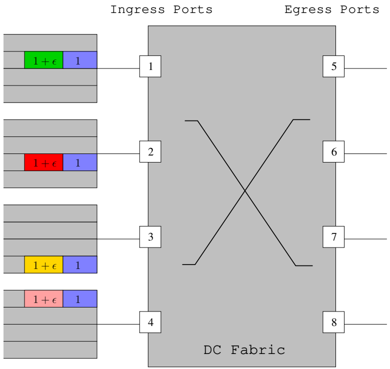
System model and notations

- Big-Switch model
  - ✔ Capacity B_ℓ for port ℓ
- Set $\mathcal{C} = \{1, 2, \ldots, N\}$ of coflows
  - ✔ Coflow k is a set $\mathcal{F}_k$ of flows, where flow $j \in \mathcal{F}_k$ has size $v_{k,j}$
  - ✔ Coflow k arrive at time 0 and has deadline T_k
  - ✔ $\mathcal{F}_{k,\ell}$ is the of flows of coflow k which use port ℓ
  - ✔ The completion time of coflow k at port ℓ in isolation is

$$p_{\ell,k} = \frac{\sum_{j \in \mathcal{F}_{k,l}} v_{k,j}}{B_\ell}$$

System model and notations

- Example
  - ✔ All fabric ports have the same normalized bandwidth of 1
  - ✔ The flows are organised in virtual output queues at the ingress ports. The virtual queue index represents the flow output port



CAR maximization problem

- Decision variables:
  - ✔ $r_{k,j}(t) \geq 0$: rate allocated to flow $j \in \mathcal{F}_k$ at time t
  - ✔ $z_k \in \{0, 1\}$ is 1 if coflow k is accepted, 0 otherwise
- Mathematical formulation:

$$\max \sum_{k \in \mathcal{C}} z_k \tag{P1}$$

$$\text{s.t.} \sum_{k \in \mathcal{C}} \sum_{j \in \mathcal{F}_{k,\ell}} r_{k,j}(t) \leq B_\ell, \quad \forall \ell \in \mathcal{L}, \forall t \in \mathcal{T}, \tag{1}$$

$$\int_0^{T_k} r_{k,j}(t)\, \mathrm{d}t \geq v_{k,j} z_k, \quad \forall j \in \mathcal{F}_k, \forall k \in \mathcal{C}, \tag{2}$$

- MILP formulation[2] assuming that rate allocations are constant over the intervals $[0, T_{i(1)}),\ [T_{i(1)}, T_{i(2)}), \ldots, [T_{i(N-1)}, T_{i(N)})$

2

☞ S.-H. Tseng et *al.*, Coflow deadline scheduling via network-aware optimization, Annu. Allert. Conf. Commun. Control Comput., 2018.

σ-order scheduling

- The transport layer may not be able to enforce the per-flow rate allocation $r_{k,j}(t)$.
- Alternative approach: order the coflows in some appropriate order, and leverage priority forwarding mechanisms
  - ✔ Order σ such that coflow $\sigma(n)$ has priority over coflow $\sigma(n+1)$
  - ✔ A flow is blocked if and only if either its ingress port or its egress port is busy serving a higher-priority flow
  - ✔ Preemption is allowed

CS-MHA algorithm

- Moore-Hogdson algorithm

EDD order	1	2	3	4	5	6	7	8	Rejected
Due date	6	8	9	11	20	25	28	35	Jobs
Proc. time	4	1	6	3	6	8	7	10	
CCT	4	5	11						
CCT	4	5	⋆						3
CCT	4	5	⋆	8	14	22	29		3
CCT	4	5	⋆	8	14	⋆	21		3, 6
CCT	4	5	⋆	8	14	⋆	21	31	3, 6

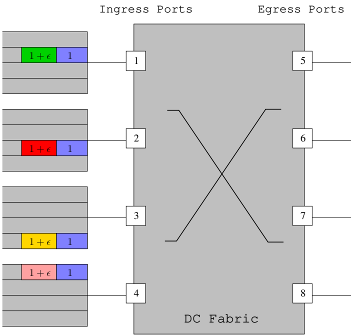
- CS-MHA[3]
  - ✔ **First round**: computes the set of admitted coflows at each port ℓ with Moore-Hogdson. A coflow is admitted if it is admitted at all ports.
  - ✔ **Second round**: order rejected coflows by increasing value of $\frac{1}{T_k} max_\ell p_{\ell,k}$

3

☞ S. Luo et *al.*, Decentralized deadline-aware coflow scheduling for datacenter networks, in Proc. IEEE ICC, 2016.

CS-MHA (2)

- Example



- $T_1 = 1$, $T_2 = T_3 = T_4 = T_5 = 2$
- CS-MHA rejects C_2, C_3, C_4, C_5 (CAR is $\frac{1}{5}$)
- The optimal solution rejects only C_1 (CAR is $\frac{4}{5}$)

- CS-MHA neglects the impact that a coflow may have on other coflows on multiple ports.

DCoflow

Parallel inequalities

- If the set $\mathcal{S} \subseteq \mathcal{C}$ of accepted coflows is feasible, then

$$f_\ell(\mathcal{S}) - \sum_{k \in \mathcal{S}} p_{\ell,k} T_k \leq 0, \qquad \text{for all ports } \ell,$$

where $f_\ell(\mathcal{S}) = \frac{1}{2}\sum_{k \in \mathcal{S}} p_{\ell,k}^2 + \frac{1}{2}\left(\sum_{k \in \mathcal{S}} p_{\ell,k}\right)^2$

- If the subset $\mathcal{S} \subseteq \mathcal{C}$ of coflows is not feasible, we need to reject at least one coflow $k' \in \mathcal{S}$. We choose k' so as to minimize

$$f_\ell(\mathcal{S} \setminus \{k'\}) - \sum_{k \in \mathcal{S} \setminus \{k'\}} p_{\ell,k} T_k = f_\ell(\mathcal{S}) - \sum_{k \in \mathcal{S}} p_{\ell,k} T_k + \Psi_{\ell,k'}$$

where $\Psi_{\ell,k'} := p_{\ell,k'}\left(T_{k'} - \sum_{k \in \mathcal{S}} p_{\ell,k}\right)$

DCoflow

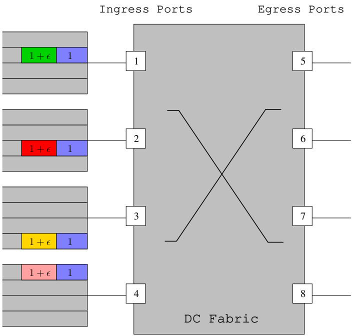
- Input: a set $\mathcal{S} = \{1, \ldots, N\}$ of unsorted coflows
- Output: scheduling order σ of accepted coflows.
- At each round, `DCoflow` either accepts a coflow or it rejects one:
  - Bottleneck link $\ell_b = \underset{\ell}{\operatorname{argmax}} \sum_{k \in \mathcal{S}} p_{\ell,k}$
  - Let k be the coflow with the largest deadline on port ℓ_b. If coflow k meets its deadline when scheduled last on port ℓ_b, then accept k
  - Otherwise, reject the coflow k' which uses port ℓ_b and minimizes

$$\sum_{\ell : \Psi_{\ell,k'} < 0} \Psi_{\ell,k'}$$

- A post-processing is done to accept unduly rejected coflows

DCoflow (2)

- Example



- $T_1 = 1$, $T_2 = T_3 = T_4 = T_5 = 2$
- Round 1: $\ell_b = 1$ with CT $2 + \epsilon$

$$\sum_{\ell:\Psi_{\ell,1}<0} \Psi_{\ell,1} = 8 \times 1 \times (1 - (2 + \epsilon)) \approx -8$$

$$\sum_{\ell:\Psi_{\ell,2}<0} \Psi_{\ell,2} = 2 \times (1 + \epsilon) \times (2 - (2 + \epsilon)) \approx 0$$

- C_1 is rejected an all other coflows are accepted (CAR is $\frac{4}{5}$)

DCoflow – Online Setting

- Coflows arrive sequentially and possibly in batches
- `DCoflow` recomputes a schedule at frequency f:
  - Updates at arrival instants of coflows ($f = \infty$)
  - Periodic updates with period $1/f$
  - Scheduling order for all coflows present in the system (with residual size)
- The scheduler knows everything about coflows that have arrived, and nothing about future coflows

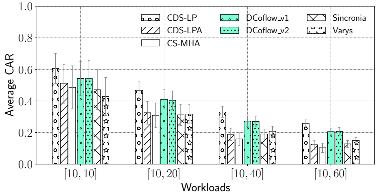
Numerical Results

Simulation setup

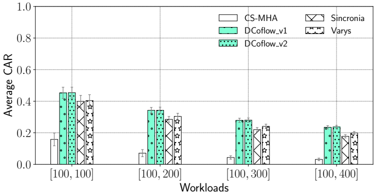
- Algorithms : `DCoflow`, `CS-MHA`, `CDS-LP`, `CDS-LPA`, `Varys`, `Sincronia`
- Instances $[M, N]$ with $2 \times M$ ports and N coflows
  - Greedy rate allocation by the transport network
- Synthetic traffic with 2 types of coflows (type-1 with proba 0.4)
  - Type-1 coflows have a single flow of random volume $\mathcal{N}(1, 0.04)$. The number of flows of type-2 coflows is $\mathcal{U}\left(\frac{2}{3}M, M\right)$ (volume ratio is 0.8). The deadline is chosen randomly in $[CCT^0, 2CCT^0]$.
- Facebook dataset (MapReduce shuffle, 3000-machines cluster)
  - N coflows are randomly sampled from the dataset.

Offline setting

- Synthetic traffic (100 random instances)

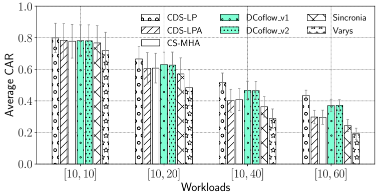


(a) small-scale networks

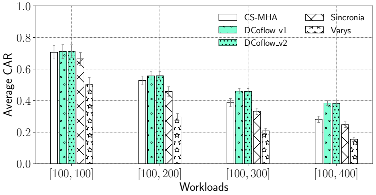


(b) large-scale networks

- Facebook (100 random instances)



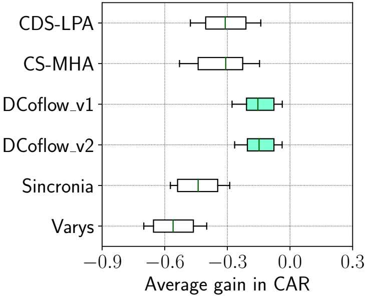
(a) small-scale networks



(b) large-scale networks

Offline setting (2)

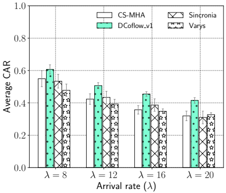
- 1^{st}-10^{th} -50^{th}-90^{th}-99^{th} percentiles of gain in CAR for $[10, 60]$



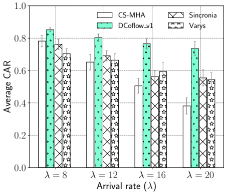
- Prediction error
  - Relative difference between the number of coflows satisfying their deadline before/after `GreedyFlowScheduling`
  - Average prediction error below 3.6% for both traffic traces

Online setting – Impact of arrival rate

- Synthetic traffic (40 instances)

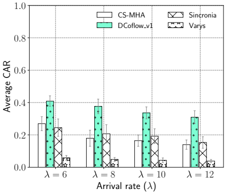


(a) $[10, 4000]$

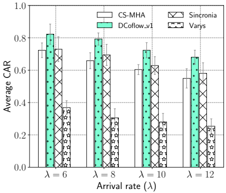


(b) $[50, 4000]$

- Facebook (40 instances)



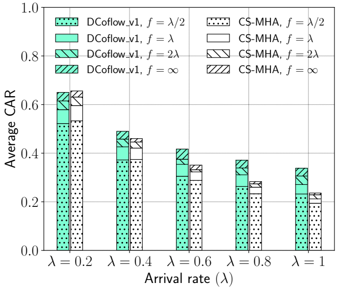
(a) $[10, 4000]$



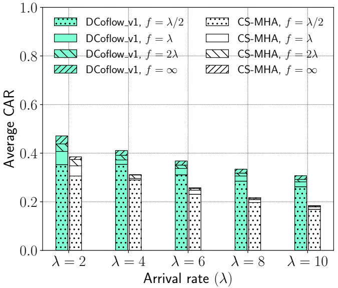
(b) $[100, 4000]$

Online setting – Impact of update frequenccy

- Synthetic traffic $[10, 8000]$ (40 instances)



(a) Without batch arrivals



(b) Batch arrivals

Conclusion

Conclusion

- Joint coflow admission control and scheduling with deadlines
  - ✔ Based on known results for open-shop scheduling
  - ✔ Produces a σ-order of accepted coflows
  - ✔ Significant improvements w.r.t. existing algorithms, in particular for large-scale and congested networks
- Future works
  - ✔ Workload is composed of coflows with deadlines and coflows without deadlines
  - ✔ Weighted coflow admission control
  - ✔ Incomplete information on the flow volume

Questions?