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 6th, 2022
Outline

Introduction

Problem Formulation and Existing Works

DCoflow

Numerical Results

Conclusion
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

- **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

---

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 $B_\ell$ for port $\ell$

- Set $C = \{1, 2, \ldots, N\}$ 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 $\ell$
  - The completion time of coflow $k$ at port $\ell$ in isolation is
    \[ p_{\ell,k} = \frac{\sum_{j \in \mathcal{F}_{k,\ell}} v_{k,j}}{B_\ell} \]
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

![Diagram showing DC Fabric and ingress/egress ports with bandwidth notations]
CAR maximization problem

- **Decision variables:**
  - $r_{k,j}(t) \geq 0$: rate allocated to flow $j \in F_k$ at time $t$
  - $z_k \in \{0, 1\}$ is 1 if coflow $k$ is accepted, 0 otherwise

- **Mathematical formulation:**
  
  \[
  \text{max} \sum_{k \in C} z_k \quad (P1)
  \]

  \[
  \text{s.t.} \quad \sum_{k \in C} \sum_{j \in F_k} r_{k,j}(t) \leq B_\ell, \quad \forall \ell \in \mathcal{L}, \forall t \in \mathcal{T}, \quad (1)
  \]

  \[
  \int_0^{T_k} r_{k,j}(t) \, dt \geq v_{k,j} z_k, \quad \forall j \in F_k, \forall k \in C, \quad (2)
  \]

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

<table>
<thead>
<tr>
<th>EDD order</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>Rejected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Due date</td>
<td>6</td>
<td>8</td>
<td>9</td>
<td>11</td>
<td>20</td>
<td>25</td>
<td>28</td>
<td>35</td>
<td>Jobs</td>
</tr>
<tr>
<td>Proc. time</td>
<td>4</td>
<td>1</td>
<td>6</td>
<td>3</td>
<td>6</td>
<td>8</td>
<td>7</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CCT</th>
<th>4</th>
<th>5</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCT</td>
<td>4</td>
<td>5</td>
<td>*</td>
</tr>
<tr>
<td>CCT</td>
<td>4</td>
<td>5</td>
<td>*</td>
</tr>
<tr>
<td>CCT</td>
<td>4</td>
<td>5</td>
<td>*</td>
</tr>
<tr>
<td>CCT</td>
<td>4</td>
<td>5</td>
<td>*</td>
</tr>
</tbody>
</table>

- CS-MHA³

- **First round**: computes the set of admitted coflows at each port \( \ell \) with Moore-Hogdson. A coflow is admitted if it is admitted at all ports.

- **Second round**: order rejected coflows by increasing value of \( \frac{1}{T_k} \max_\ell p_{\ell,k} \)

---

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
If the set $S \subseteq C$ of accepted coflows is feasible, then

$$f_\ell(S) - \sum_{k \in S} p_{\ell,k} T_k \leq 0,$$

for all ports $\ell$, where

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

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

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

where

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

- **Input:** a set $S = \{1, \ldots, N\}$ of unsorted coflows
- **Output:** scheduling order $\sigma$ of accepted coflows.
- **At each round,** DCoflow either accepts a coflow or it rejects one:
  - Bottleneck link $\ell_b = \arg\max \sum_{k \in S} p_{\ell,k}$
  - Let $k$ be the coflow with the largest deadline on port $\ell_b$. If coflow $k$ meets its deadline when scheduled last on port $\ell_b$, then accept $k$
  - Otherwise, reject the coflow $k'$ which uses port $\ell_b$ and minimizes
    $$\sum_{\ell: \Psi_{\ell,k'} < 0} \Psi_{\ell,k'}$$
- **A post-processing is done to accept unduly rejected coflows**
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 and 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)**

![Graphs showing average CAR for different workloads and network sizes.](a) small-scale networks  
(b) large-scale networks

- **Facebook (100 random instances)**

![Graphs showing average CAR for different workloads and network sizes.](a) small-scale networks  
(b) large-scale networks
Offline setting (2)

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

![Box plot showing average gain in CAR for different algorithms]

- 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 frequency

▶ 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 $\sigma$-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?