Non-preemptive Coflow Scheduling and Routing

Ruozhou Yu, Guoliang Xue, and Xiang Zhang

Arizona State University

Jian Tang

Syracuse University
Outline

- Introduction and Motivation
- System Model and Algorithm Design
- Performance Evaluation
- Conclusion
Flows and Coflows

Cloud Task

OMG where's my last piece???
Need it now!!!
Flows and Coflows

- Traditional network scheduling/routing solution

  - Scheduling/Routing regarding **individual flows**
    - General flow: a subset of packet header fields

  - Fails to account for application-level performance metrics
    - *Flow completion time* vs. *task completion time*
Flows and Coflows

Task 1

Oh you’re so nice!!

Task 2

You faster, you go ahead 😊
Flows and Coflows

- Application-aware scheduling/routing: coflows
  - Flows grouped by application/task
    - A coflow finishes when all its component flows finish

- Advantages:
  - Captures application-level requirement
  - Establishes fairness in application-level

- Want to do it in a centralized way
  - Not to leak app privacy to other apps
  - Or to prevent apps from selfishly congest the network
(Non-)Preemptive Scheduling

- Existing coflow scheduling/routing allows preemption!
  - Pause for the shorter ones!

- Advantages:
  - Better performance and network utilization in theory

- Disadvantages:
  - Large overhead for flow switching: performance issue for short flows
    - Switching delays
    - Switch computations
  - No ready support in commodity hardware
    - Standardization on-going: IEEE 802.1Qbu
    - A long way before commercial-ready

- Our stand: non-preemptive scheduling + routing of coflows
Summary of Problem

Task 1

Now, you go first, this way!

Task 2

You next, that way!

Task 4

Sorry, there’s no place. You fired!

Task 3

You this way, free to go!
Contributions

- A first (preliminary) study for Non-preemptive Coflow Scheduling and Routing (NCSR)
- An offline scheduling framework: Shortest-Coflow First
- A multi-path routing algorithm
- A single-path routing algorithm
- Performance evaluations
Outline

- Introduction and Motivation
- System Model and Algorithm Design
- Performance Evaluation
- Conclusion
System Model

- Network: $G = (V, E)$

- Coflow requests: $S = \{ C_1, \ldots, C_m \}$
  - Each request: $C_i = \{ F_{i,1}, \ldots, F_{i,n_i} \}$
  - $F_{i,j} = (s_{i,j}, t_{i,j}, d_{i,j})$: source, destination, flow size (demand, in bytes)

- Bandwidth allocation
  - $B_{i,j}^p(t)$: bandwidth allocation on path $p$ of flow $i, j$, at time $t$
  - $B_{i,j}(t) = \text{sum of bandwidth over all paths at time } t$
Flow/coflow completion time

Flow completion time (FCT):

\[ T_{i,j} = \arg \min_{\tau} \left\{ \int_{0}^{\tau} B_{i,j}(t) \, dt = d_{i,j} \right\} \]

Coflow completion time (CCT): max. FCT of its component flows

\[ T_i = \max_j \{ T_{i,j} \mid F_{i,j} \in C_i \} \]

Objective: minimize total CCT

\[ \min_{C_i \in S} \sum T_i \]
For each coflow:
- Compute per-coflow completion time (CCT)
  - If multi-path enabled, compute using multi-path routing
  - Otherwise, use single-path routing

Schedule coflows in ascending order of CCT
CCT with Multi-path Routing

- Non-linear programming formulation
  - Sharing among flows within the coflow
  - CCT as the maximum FCT of component flows

\[
\begin{align*}
\min & \quad T_i \\
\text{s.t.} & \quad T_i = \max_j \{d_{i,j}/b_{i,j}\} \\
& \quad \sum_{j=1}^{n_i} f_{i,j} \leq c_e \quad \forall e \in E \\
& \quad \sum_{(u,v) \in E} f_{i,j}^{(u,v)} - \sum_{(v,w) \in E} f_{i,j}^{(v,w)} = \begin{cases} 
0, & v \notin \{s_{i,j}, t_{i,j}\} \\
-b_{i,j}, & v = s_{i,j} \\
b_{i,j}, & v = t_{i,j} \\
\forall F_{i,j} \in C_i, v \in V 
\end{cases}
\end{align*}
\]

- Linearization: let \( f_i = 1 / T_i \)

\[
\begin{align*}
\max & \quad f_i \\
\text{s.t.} & \quad f_i \leq b_{i,j}/d_{i,j} \quad \forall F_{i,j} \in C_i \\
& \quad (6c) \text{ and } (6d)
\end{align*}
\]
CCT with Single-path Routing

- Additional integer variables to the Multi-path Routing model
  - $x^e_{i,j}$: link selection for single-path routing

- Linear relaxation and deterministic rounding
  - Relax $x^e_{i,j}$ to take continuous values, and solve linear program;
  - For each flow, find path with maximum minimum $x$ values, and assign;
  - Re-solve program to obtain bandwidth allocation with fixed path assignments
Outline

- Introduction and Motivation
- System Model and Algorithm Design
- Performance Evaluation
- Conclusion
Simulation Setups

- Waxman random graphs
  - 50 nodes
  - Alpha=0.15, beta=0.2
  - Link capacities: [10, 100] Mbps

- Coflows
  - 25 requests
  - 1 to 10 flows per request
  - Flow sizes: [10, 100] Mbps

- Comparison:
  - sSCF, mSCF: single-path and multi-path SCF algorithm (proposed)
  - sRT, mRT: single-path and multi-path Routing-only algorithm (baseline)
  - sSFF, mSFF: single-path and multi-path Shortest-Flow First (baseline)
Simulation Results: Average CCT

Avr. coflow completion time (s)

- mSCF
- sSCF
- mRT
- sRT
- mSFF
- sSFF

# coflow requests
Simulation Results: Running Time
Outline

- Introduction and Motivation
- System Model and Algorithm Design
- Performance Evaluation
- Conclusion
Conclusions

- A first step study on NCSR
  - Offline optimization model
  - SCF scheduler for scheduling
  - Multi-path and single-path routing algorithms

- Experiment results
  - Scheduling more effective than routing: when network congested
  - Application-awareness brings great advantage

- Future work
  - Enable better sharing/work conservation of resources
    - Remove the non-sharing rule of coflows
Q&A?

THANK YOU VERY MUCH!