

Optimization of a Distributed Storage Architecture under Uncertain Demand

A Study in Fast Flow Algorithms

Paul Beaujean^{*†} Éric Gourdin[†]

^{*}ENSIIE-MPRO

[†]Orange Labs

March 20, 2015

Table of Contents

Context

The rise of IP video

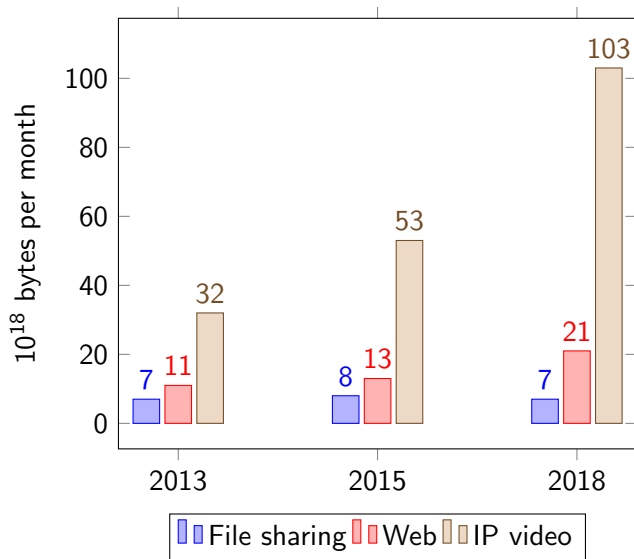
Content Delivery Network

Multicommodity Flow

Column generation

FPTAS

Context: The rise of IP video



- ▶ IP video: 80% of all IP traffic by 2020.

Content Delivery Network

A Content Delivery Network is a distributed storage architecture. Provides content to users by caching it close to where it is requested.
Less upstream bandwidth consumption.

Content Delivery = Multicommodity Flow

Insert a beautiful picture of multiple servers delivering contents to multiple clients.

Fractional Multiflow is in P

No combinatorial algorithm known to solve multiflow.

Polynomial-size LP with edge formulation

$$\begin{aligned} \max \quad & \sum_{s,t \in K} \sum_{a \in \delta^+(s)} f_a^{s,t} \\ \text{s.t.} \quad & \sum_{s,t \in K} f_a^{s,t} \leq c_a, \quad \forall a \in A \\ & \sum_{s,t \in K} \sum_{a \in \delta^-(v)} f_a^{s,t} = \sum_{s,t \in K} \sum_{a \in \delta^+(v)} f_a^{s,t}, \quad \forall v \in V \setminus \{s, t\} \\ & \sum_{a \in \delta^-(s)} f_a^{s,t} = 0, \quad \forall s \in S \\ & f \in \mathbb{R}_+^{|A \times K|} \end{aligned}$$

can be solved in polynomial time with an IPM

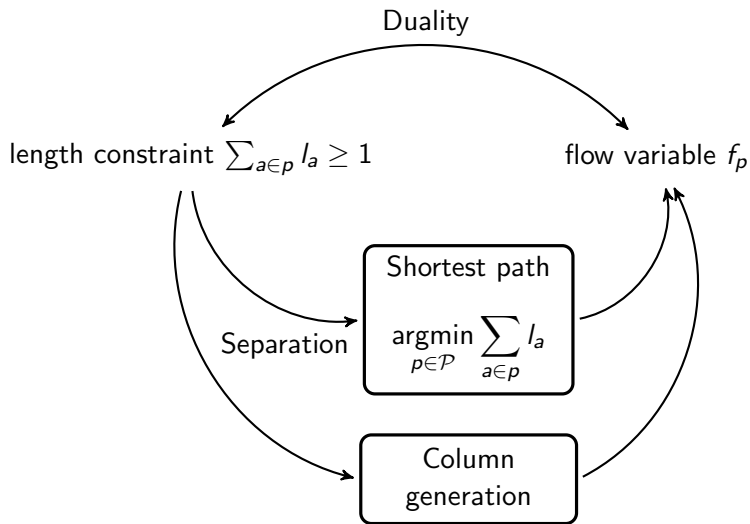
$\implies \text{MULTIFLOW} \in \text{P}$

Can it be solved on large instances?

Implicit exponential-size path formulation

$$\begin{aligned} (\Pi) \quad & \left\{ \begin{array}{ll} \max & \sum_{p \in \mathcal{P}} f_p \\ \text{s.t.} & \sum_{p \ni a} f_p \leq c_a, \quad \forall a \in A \quad (l_a) \\ & f \in \mathbb{R}_+^{|\mathcal{P}|} \end{array} \right. \\ (\Delta) \quad & \left\{ \begin{array}{ll} \min & \sum_{a \in A} c_a l_a \\ \text{s.t.} & \sum_{a \in p} l_a \geq 1, \quad \forall p \in \mathcal{P} \quad (f_p) \\ & l \in \mathbb{R}_+^{|A|} \end{array} \right. \end{aligned}$$

Column generation



Primal-dual approximation algorithm

Fully Polynomial Time Approximation Scheme

An algorithm that returns a solution of value within ϵ of the optimal value:

- ▶ in time bounded by a polynomial in the instance size,
- ▶ in time bounded by a polynomial in ϵ^{-1} .

There exists combinatorial algorithms that achieve such guarantees.

How to design one?

- ▶ based on the multiplicative weights algorithm
- ▶ reuse ideas from column generation

Multiplicative weights

Background

Discovered several times:

- ▶ 1950's: machine learning
- ▶ 1970's: continuous optimization
- ▶ 2000's: design of competitive online algorithms
- ▶ 2010's: approximation algorithms for SDP

Definition

Given a finite set of actions with bounded gain

$$\mathcal{A} = \{a \mid g_a^t \in [-1, 1]\}$$

Associate weights

$$\forall a \in \mathcal{A} \quad \begin{cases} \lambda_a^1 &= 1 \\ \lambda_a^{t+1} &= \lambda_a^t (1 + \epsilon g_a^t) \end{cases}$$

A surprising result

The multiplicative weights algorithm

- At round t , choose action a with probability $\frac{\lambda_a^t}{\sum_a \lambda_a^t}$

has a guaranteed gain of

$$G \geq G_a - \epsilon |G_a| - \frac{\ln |A|}{\epsilon}, \forall a \in \mathcal{A}$$

$$\begin{array}{ll} G = \sum_t \mathbb{E}(g^t) & \text{expected gain of the algorithm} \\ G_a = \sum_t g_a^t & \text{gain of action } a \end{array}$$

Mapping multiplicative weights concepts to multiflow

Mapping concepts

actions	\mathcal{A}	\rightarrow	A	arcs
gain	g_a	\rightarrow	f_a/c_a	arc congestion
compound gain	λ_a	\rightarrow	l_a	length (compound congestion)

Iterative primal-dual algorithm

Successively adds flow to the graph

Choosing arcs by constraint separation

Instead of choosing arcs randomly:

- ▶ compute shortest path w.r.t. the length l_a^t
- ▶ all arcs on that path receive maximum feasible flow
- ▶ probabilities are set a posteriori

Combining gain guarantee and properties of flows

Using the gain guarantee

$$G \geq (1 - \epsilon)G_* - \frac{\log m}{\epsilon}$$

and simple flow considerations

- ▶ G_* $\begin{cases} \text{gain of the best action in hindsight} \\ \text{congestion rate of the most congested arc} \end{cases}$
- ▶ $F/F^* \geq G$: approximation ratio bounds the expected gain

results in

- ▶ If $G_* = \log m \epsilon^{-2}$ then $F/G_* \geq (1 - 2\epsilon)F^* \leftarrow \text{AS}$
- ▶ Runtime: $O(km^2 \log m \epsilon^{-2}) \leftarrow \text{FPT}$

Numerical experiments

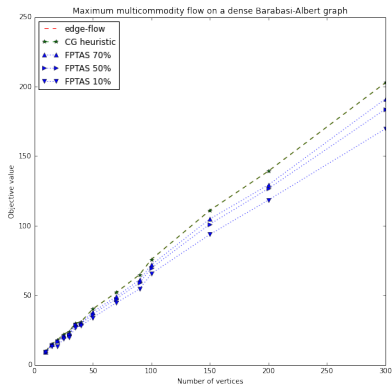
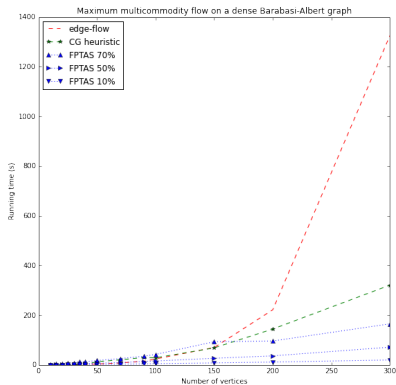


Figure : Performance of LP, column generation, and FPTAS

Conclusion

Highlights

- ▶ CDN design naturally involves multiflows
- ▶ Approximation: an interesting alternative to exact algorithms

Achievements

- ▶ Studied in depth a theoretical problem
- ▶ Implemented a state-of-the-art algorithm
- ▶ Integrated it in an Operations Research library

Perspectives

- ▶ Studying the matrix multiplicative weights algorithm
- ▶ Application to solving SDP approximatively

during a PhD at Orange Labs and Université Paris-Dauphine