On Polynomial-Time Combinatorial Algorithms for Maximum $L$-Bounded Flow

Kateřina Altmanová
Petr Kolman
Jan Voborník

Charles University
Prague

WADS, August 2019
Length Bounded Flow Problem

$L$-bounded flow
- a flow decomposable into flow paths of length at most $L$

Input
- graph $G = (V, E)$
- source-sink pair $s, t \in V$
- integer parameter $L$

Output and Objective
- find an $L$-bounded flow of maximum size
Example of a Length Bounded Flow

$L = 4$

\[ \begin{array}{cccccc}
    & a & b & c & \text{s} & \text{t} \\
\text{s} & 1 & 1 & 1 & \frac{1}{2} & \frac{1}{2} \\
\text{d} & 1 & 1 & 1 & \frac{1}{2} & \frac{1}{2} \\
\text{e} & 1 & 1 & 1 & \frac{1}{2} & \frac{1}{2} \\
\text{t} & 1 & 1 & 1 & \frac{1}{2} & \frac{1}{2} \\
\end{array} \]
Example of a Length Bounded Flow

$L = 4$

\[ f(p_1) = \frac{1}{2} \]

\[ f(p_2) = \frac{1}{2} \]

\[ f(p_3) = \frac{1}{2} \]
Example of a Maximum Length Bounded Flow

Observe
- every $L$-bounded path has to use at least two bottom edges
- three bottom edges $\Rightarrow$ max $L$-bounded flow at most $\frac{3}{2}$

Takeaway
- The maximum $L$-bounded flow need not be integral, even on graphs with unit capacities.
Example of a Maximum Length Bounded Flow

\[ L = 4 \]

\[ s \quad 1 \quad d \quad 1 \quad e \quad 1 \quad t \]

Observe
- every \( L \)-bounded path has to use at least two bottom edges
- three bottom edges \( \Rightarrow \) max \( L \)-bounded flow at most \( \frac{3}{2} \)

Takeaway
- The maximum \( L \)-bounded flow need not be integral, even on graphs with unit capacities.
Non-Example of a Length Bounded Flow

$L = 4$

4-bounded flow?

- No - it’s bigger than the maximum 4-bounded flow.

5-bounded flow?

- Yes - decompose into two paths of length 4 and 5.

On Combinatorial Algorithms for $L$-Bounded Flow
Non-Example of a Length Bounded Flow

$L = 4$

4-bounded flow?
- No - it’s bigger than the maximum 4-bounded flow.

5-bounded flow?
- Yes - decompose into two paths of length 4 and 5.
Length Bounded Cut Problem

Input
- graph \( G = (V, E) \)
- source-sink pair \( s, t \in V \)
- integer parameter \( L \)

Output and Objective
- a subset of edges \( F \subseteq E \) such that in \( G \setminus F \),
  the distance between \( s \) and \( t \) is at least \( L + 1 \)
- find an \( L \)-bounded cut of minimum size

Also known as
- Short paths interdiction problem, and
- Most vital edges for shortest paths problem
Related Results

Length Bounded Flows and Cuts

- 1971 - Adámek and Koubek - introduction of $L$-bounded flows and cuts; duality does not hold
  - 1981 - Koubek and Říha - combinatorial algorithm for the maximum length bounded flow flawed
  - 1995 - Bar-Noy et al. - $L$-bounded cut NP-hard
  - 2002 - K. and Scheideler - $L$-bounded flow in P, by poly-size LP
  - 2003 - Baier - FPTAS for $L$-bounded flow with edge lengths, using the ellipsoid algorithm
  - 2010 - Baier et al. - $L$-bounded flow with edge lengths NP hard
  - 2010 - Baier et al. - $\Theta(n^{2/3})$-gap between maximum $L$-bounded flow and minimum $L$-bounded cut
## Related Results

### Length Bounded Flows and Cuts

1. **1971** - Adámek and Koubek - *introduction* of $L$-bounded flows and cuts; duality does not hold
2. **1981** - Koubek and Říha - *combinatorial algorithm* for the maximum length bounded flow *flawed*
3. **1995** - Bar-Noy et al. - $L$-bounded cut NP-hard
4. **2002** - K. and Scheideler - $L$-bounded flow in P, by poly-size LP
5. **2003** - Baier - FPTAS for $L$-bounded flow with edge lengths, using the ellipsoid algorithm
6. **2010** - Baier et al. - $L$-bounded flow with edge lengths NP hard
7. **2010** - Baier et al. - $\Theta(n^{2/3})$-gap between maximum $L$-bounded flow and minimum $L$-bounded cut
## Related Results

### Length Bounded Flows and Cuts

- **1971** - Adámek and Koubek - introduction of $L$-bounded flows and cuts; duality does not hold
- **1981** - Koubek and Říha - combinatorial algorithm for the maximum length bounded flow *flawed*
- **1995** - Bar-Noy et al. - $L$-bounded cut NP-hard
  - 2002 - K. and Scheideler - $L$-bounded flow in P, by poly-size LP
  - 2003 - Baier - FPTAS for $L$-bounded flow with edge lengths, using the ellipsoid algorithm
  - 2010 - Baier et al. - $L$-bounded flow with edge lengths NP hard
  - 2010 - Baier et al. - $\Theta(n^{2/3})$-gap between maximum $L$-bounded flow and minimum $L$-bounded cut
### Length Bounded Flows and Cuts

- **1971** - Adámek and Koubek - *introduction* of $L$-bounded flows and cuts; duality does not hold
- **1981** - Koubek and Říha - *combinatorial algorithm* for the maximum length bounded flow
  - *flawed*
- **1995** - Bar-Noy et al. - $L$-bounded cut NP-hard
- **2002** - K. and Scheideler - $L$-bounded flow in P, by poly-size LP
  - **2003** - Baier - FPTAS for $L$-bounded flow with edge lengths, using the ellipsoid algorithm
  - **2010** - Baier et al. - $L$-bounded flow with edge lengths NP hard
  - **2010** - Baier et al. - $\Theta(n^{2/3})$-gap between maximum $L$- bounded flow and minimum $L$-bounded cut
Related Results

Length Bounded Flows and Cuts

- 1971 - Adámek and Koubek - *introduction* of $L$-bounded flows and cuts; duality does not hold
- 1981 - Koubek and Říha - *combinatorial algorithm* for the maximum length bounded flow *flawed*
- 1995 - Bar-Noy et al. - $L$-bounded cut NP-hard
- 2002 - K. and Scheideler - $L$-bounded flow in P, by poly-size LP
- 2003 - Baier - FPTAS for $L$-bounded flow with edge lengths, using the ellipsoid algorithm
- 2010 - Baier et al. - $L$-bounded flow with edge lengths NP hard
- 2010 - Baier et al. - $\Theta(n^{2/3})$-gap between maximum $L$-bounded flow and minimum $L$-bounded cut
Related Results

Length Bounded Flows and Cuts

- 1971 - Adámek and Koubek - *introduction* of $L$-bounded flows and cuts; duality does not hold
- 1981 - Koubek and Říha - *combinatorial algorithm* for the maximum length bounded flow *flawed*
- 1995 - Bar-Noy et al. - $L$-bounded cut NP-hard
- 2002 - K. and Scheideler - $L$-bounded flow in P, by poly-size LP
- 2003 - Baier - FPTAS for $L$-bounded flow with edge lengths, using the ellipsoid algorithm
- 2010 - Baier et al. - $L$-bounded flow with edge lengths NP hard
- 2010 - Baier et al. - $\Theta(n^{2/3})$-gap between maximum $L$-bounded flow and minimum $L$-bounded cut
Related Results

Length Bounded Flows and Cuts

- 1971 - Adámek and Koubek - introduction of $L$-bounded flows and cuts; duality does not hold
- 1981 - Koubek and Říha - combinatorial algorithm for the maximum length bounded flow flawed
- 1995 - Bar-Noy et al. - $L$-bounded cut NP-hard
- 2002 - K. and Scheideler - $L$-bounded flow in P, by poly-size LP
- 2003 - Baier - FPTAS for $L$-bounded flow with edge lengths, using the ellipsoid algorithm
- 2010 - Baier et al. - $L$-bounded flow with edge lengths NP hard
- 2010 - Baier et al. - $\Theta(n^{2/3})$-gap between maximum $L$-bounded flow and minimum $L$-bounded cut
Related Problems

Shortest Hop Constrained Path
- find the shortest path between two vertices, wrt edge lengths, with a bounded number of edges (hops)

$L$-Bounded Disjoint Paths
- find the maximum number of disjoint paths between two vertices, each of a bounded length

Most Vital Edges for Shortest Paths Problem
- given an integer $k$, find a subset of $k$ edges whose removal maximizes the distance between $s$ and $t$
### Related Problems

| Problem                                      | Description                                                                 |
|----------------------------------------------|-----------------------------------------------------------------------------|
| **Shortest Hop Constrained Path**            | find the shortest path between two vertices, wrt edge lengths, with a bounded number of edges (hops) |
| **L-Bounded Disjoint Paths**                 | find the maximum number of disjoint paths between two vertices, each of a bounded length |
| **Most Vital Edges for Shortest Paths Problem** | given an integer $k$, find a subset of $k$ edges whose removal maximizes the distance between $s$ and $t$ |
Related Problems

**Shortest Hop Constrained Path**
- find the shortest path between two vertices, wrt edge lengths, with a bounded number of edges (hops)

**L-Bounded Disjoint Paths**
- find the maximum number of disjoint paths between two vertices, each of a bounded length

**Most Vital Edges for Shortest Paths Problem**
- given an integer $k$, find a subset of $k$ edges whose removal maximizes the distance between $s$ and $t$
Our Contribution

Length Bounded Flow

- The algorithm of Koubek and Říha is not correct
- A combinatorial FPTAS for the maximum $L$-bounded flow, i.e., $(1 + \varepsilon)$ approximation of $OPT$ in time $\varepsilon^{-2} |E|^2 L \log L$.
- A combinatorial FPTAS for the NP-hard maximum $L$-bounded flow with edge lengths.

Open Problem

- Design a poly-time combinatorial algorithm for the maximum $L$-bounded flow.

Combinatorial = the algorithm does not explicitly use LP and linear algebra methods.
Example

- Not a maximum $L$-bounded flow.
- No space for adding a new $L$-bounded $s - t$ path.
- By diverting the flow on $c - t$ along $c - b - t$, we obtain space for a new $L$-bounded $s - t$ path $s - a - c - t$. 
The Algorithm of Koubek and Říha, cont’d.

| Main Idea |
|-----------|
| Given an $L$-bounded flow $f$ and its decomposition, describe by a tree structure how to combine segments of paths from the flow $f$ with segments of empty edges into a larger $L$-bounded flow. |

| Technical Details |
|--------------------|
| Many ... |
| Define a tree called *increasing L-system* - generalization of an augmenting flow. |
| Various types of nodes: for diverting flow, for shortening flow, pointers to other nodes, etc. |
| Each node has a plenty of attributes to take care about the length bounds and flow conservation at each vertex. |
The Algorithm of Koubek and Říha, cont’d.

Proof Structure
1. If \( f \) is not maximal \( L \)-bounded flow, then there exists an increasing \( L \)-system.
2. If there exists an increasing \( L \)-system, then it is possible to obtain a larger \( L \)-bounded flow.
   \( \Rightarrow \) iterative improvements possible

Cf. Ford-Fulkerson alg. for classical flow: if there is an augmenting path in the residual network, increase the flow along it

Difficulty
- The second claim does not hold: the existence of an increasing \( L \)-system does not imply the possibility to increase the flow!
The Algorithm of Koubek and Říha, cont’d.

Proof Structure

1. If $f$ is not maximal $L$-bounded flow, then there exists an increasing $L$-system.
2. If there exists an increasing $L$-system, then it is possible to obtain a larger $L$-bounded flow.

⇒ iterative improvements possible

Cf. Ford-Fulkerson alg. for classical flow: if there is an augmenting path in the residual network, increase the flow along it

Difficulty

- The second claim does not hold: the existence of an increasing $L$-system does not imply the possibility to increase the flow!
Example:

For the following graph and the maximum $L$-bounded flow, ...
... an increasing $L$-system exists.

\[
\begin{align*}
  u_0 : & \text{ 1-son} & \overline{q}(u_0) = \{s, a, t\} & \text{saturated edge} = \{sa\} \\
  q(u_0) = & \emptyset \\
  u_1 : & \text{ 3-son} & \text{saturated edge} = \{sb\} \\
  q(u_1) = & \emptyset & h(u_1) = \{sa\} & j(u_1) = 1 \\
  \overline{q}(u_1) = & \{s, b, t\} \\
  u_2 : & \text{ 3-son} & \text{saturated edge} = \{sa\} \\
  q(u_2) = & \emptyset & h(u_2) = \{sb\} & j(u_2) = 2 \\
  \overline{q}(u_0) = & \{s, a, t\} \\
  u_3 : & \text{ 4-son} & h(u_3) = \{sa\} & o(u_3) = u_1 \\
  j(u_1) = & 1 \\
  j(u_2) = & 2
\end{align*}
\]

Informally, the pointer nodes in the tree may create a deadlock cycle: every node is expecting from some other node to do the job (of diverting some flow) but nobody does it.
Part II
Consider the path based LP formulation of the maximum $L$-bounded flow, and its dual:

\[
\begin{align*}
\text{max} & \quad \sum_{P \in \mathcal{P}_L} x(P) \\
\text{s.t.} & \quad \sum_{P \in \mathcal{P}_L: e \in P} x(P) \leq c(e) \quad \forall e \in E \\
\quad & \quad x \geq 0
\end{align*}
\]

\[
\begin{align*}
\text{min} & \quad \sum_{e \in E} c(e) y(e) \\
\text{s.t.} & \quad \sum_{e \in P} y(e) \geq 1 \quad \forall P \in \mathcal{P}_L \\
\quad & \quad y \geq 0
\end{align*}
\]
FPTAS for Maximum $L$-bounded Flow, cont’d.

**Algorithm**
- iteratively construct (to-be) solutions for both primal and dual:
  - an $L$-bounded flow $x$ (may violate the capacities, initially $x = 0$),
  - a length $y$ on the edges (initially $y(e) = \delta(\varepsilon)$ for each $e$)

**In each iteration**
- find a $y$-shortest $L$-bounded path $P \in \mathcal{P}_L$
- route $c$ units of flow on $P$, where $c = \min_{e \in P} c(e)$
- for $e \in P$, update the lengths: $y(e) := y(e)(1 + \varepsilon \frac{c}{c(e)})$

**Termination**
- stop when the $y$-shortest path $P$ is longer than 1
- down-scale $x$ to satisfy all capacity constraints
FPTAS for Maximum $L$-bounded Flow, cont’d.

### $\text{APPROX}(\varepsilon)$

1. $y(e) \leftarrow \delta(\varepsilon) \quad \forall e \in E, \quad x(P) \leftarrow 0 \quad \forall P \in \mathcal{P}_L$
2. while the $y$-shortest $L$-bounded s-t path has length $< 1$ do
3. $P \leftarrow$ the $y$-shortest $L$-bounded s-t path
4. $c \leftarrow \min_{e \in P} c(e)$
5. $x(P) \leftarrow x(P) + c$
6. $y(e) \leftarrow y(e)(1 + \varepsilon c/c(e)) \quad \forall e \in P$
7. end while
8. return $x$

### Intuition
- make edges with large flow long
- send flow along short paths, i.e., avoid heavily loaded edges

### Note:
The $y$-shortest $L$-bounded s-t path can be computed by a modification of Dijkstra’s algorithm.
Remarks

- The same structure as in the algorithm for maximum multicommodity flow by Garg and Könemann (2007).
- Example of the Multiplicative Weights Update Method.
Lemma

The flow $x$ scaled down by a factor of $\log_{1+\varepsilon} \frac{1+\varepsilon}{\delta}$ is feasible.

Proof.

Consider an edge $e \in E$ and let $f(e)$ be the final flow on $e$. Iterations $i_1, \ldots, i_r$ contributed to $f(e)$ by $c_1, \ldots, c_r$, i.e., $f(e) = \sum_{j=1}^{r} c_j$. At the end: $1 + \varepsilon > y(e)$.

Thus,

$$1 + \varepsilon > y(e) = \delta \prod_{j=1}^{r} \left(1 + \varepsilon \frac{c_j}{c(e)} \right) \geq \delta \prod_{j=1}^{r} \left(1 + \varepsilon \right) \frac{c_j}{c(e)} = \delta \left(1 + \varepsilon \right) \frac{f(e)}{c(e)},$$

which implies

$$\log_{1+\varepsilon} \frac{1+\varepsilon}{\delta} \geq \frac{f(e)}{c(e)}.$$
Theorem

For every $0 < \varepsilon < 1$, the algorithm computes an $(1 + \varepsilon)$-approximation to the maximum $L$-bounded flow in time $\varepsilon^{-2} m^2 L \log L$. 
Generalized Setting

- With some adjustments, the FPTAS works even in the NP-hard setting with edge lengths.

Differences

- **difficulty**: finding the $y$-shortest $L$-bounded path (a procedure of the FPTAS) is NP-hard if edges have lengths
- use an approximately $y$-shortest $L$-bounded path instead
**Conclusion**

**L-bounded Flow - State of Art**

- LP algorithm - *OPT* in poly time
- Combinatorial algorithm - \((1 + \varepsilon)\) approx. in time \((\varepsilon^{-2}|E|^2L \log L)\)

**Open Problem**

- Design an exact poly-time combinatorial algorithm for the maximum \(L\)-bounded flow.
Thank you!