Transshipments over time, submodular functions, and discrete Newton

Miriam Schlöter (ETH Zürich)  Martin Skutella (TU Berlin)  Khai Van Tran (TU Berlin)

ICERM Workshop on Combinatorics and Optimization
Flows Over Time: Example

- flow $\leftrightarrow$ fluid / packets
- arcs $\leftrightarrow$ pipes
- transit time $\leftrightarrow$ length of pipe
- capacity $\leftrightarrow$ width of pipe
Flows Over Time: History
## The Complexity Landscape of Flows Over Time

<table>
<thead>
<tr>
<th></th>
<th>s-t-flow</th>
<th>trans-shipment</th>
<th>min-cost</th>
<th>multi-commodity</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>static flow</strong></td>
<td></td>
<td>polynomial</td>
<td>polynomial</td>
<td>polyn. (LP)</td>
</tr>
<tr>
<td></td>
<td>polynomial minimize submodular functions [2]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### References.

1. Ford, Fulkerson (1958)
# Computing Quickest Transshipments Efficiently

<table>
<thead>
<tr>
<th>Problem</th>
<th>Time Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Quickest s-t-Flow Problem</strong></td>
<td>( \tilde{O}(m^2 n) )</td>
</tr>
<tr>
<td>[Saho, Shigeno 2017]</td>
<td>( \tilde{O}(m^2 n) )</td>
</tr>
<tr>
<td><strong>Evacuation Problem (single source or single sink)</strong></td>
<td>( \tilde{O}(m^2 k^5 + m^2 nk) )</td>
</tr>
<tr>
<td>[Schlöter 2018; Kamiyama 2019]</td>
<td>( \tilde{O}(m^2 k^5 + m^2 nk) )</td>
</tr>
<tr>
<td><strong>Quickest Transshipment Problem (multiple sources and sinks)</strong></td>
<td>( \tilde{O}(m^4 k^{15}) )</td>
</tr>
<tr>
<td>[Hoppe, Tardos 2000]</td>
<td>( \tilde{O}(m^4 k^{15}) )</td>
</tr>
<tr>
<td>[Schlöter, Sk. 2017]</td>
<td>( \tilde{O}(m^4 k^{14}) ) (^1)</td>
</tr>
<tr>
<td>[Schlöter, Sk., Tran 2022]</td>
<td>( \tilde{O}(m^2 k^5 + m^3 k^3 + m^3 n) ) (^1)</td>
</tr>
<tr>
<td>[Schlöter, Sk., Tran 2023+]</td>
<td>( \tilde{O}(m^2 k^6 + m^3 k^4 + m^3 n) )</td>
</tr>
</tbody>
</table>

\(^1\) fractional solutions only

\( m := \# \text{ arcs} \quad n := \# \text{ nodes} \quad k := \# \text{ terminals} \)
Maximum $s$-$t$-Flows Over Time

Algorithm. [Ford, Fulkerson 1958]

**Input:** $D = (V, A)$, $s, t \in V$, capacities $u_a$, transit times $\tau_a$, time $\theta \geq 0$

**Output:** maximum $s$-$t$-flow over time with time horizon $\theta$

1. **compute static $s$-$t$-flow $x$ in $D$**

   $$\text{maximizing} \quad \theta |x| - \sum_{a \in A} \tau_a x_a$$

2. **determine path-decomposition**

   $$x_a = \sum_{P \in \mathcal{P}: a \in P} x_P \quad \text{for all } a \in A$$

3. **send flow at rate $x_P$ into $s$-$t$-paths $P \in \mathcal{P}$, as long as there is enough time left to arrive at the sink before time $\theta$**
Maximum $s$-$t$-Flow Over Time: Example

‘temporally repeated’ flow
Transshipment Over Time Problem

Given: \( D = (V, A), u_a, \tau_a \) for \( a \in A \), sources/sinks \( S^+, S^- \subset V \) with supplies/demands \( b : S^+ \cup S^- \to \mathbb{R} \), time horizon \( \theta \)

Task: find flow over time satisfying supplies/demands in time \( \theta \)

Example:

\[
\begin{align*}
    u & \equiv 1 \\
    \tau & \equiv 1 \\
    \theta & = 4
\end{align*}
\]

Observations:
- not clear how to use super-source / super-sink
- no temporally repeated solution
Transshipment Over Time Problem

Given: \( D = (V, A), u_a, \tau_a \) for \( a \in A \), sources/sinks \( S^+, S^- \subset V \) with supplies/demands \( b : S^+ \cup S^- \to \mathbb{R} \), time horizon \( \theta \)

Task: find flow over time satisfying supplies/demands in time \( \theta \)

Definition. Let \( o : 2^{S^+ \cup S^-} \to \mathbb{R} \) be defined as follows: for \( X \subseteq S^+ \cup S^- \)

\[ o(X) := \text{max flow over time value from } S^+ \cap X \text{ to } S^- \setminus X \]

Lemma. [Klinz 1994] The problem is feasible if and only if

\[ b(X) \leq o(X) \quad \text{for all } X \subseteq S^+ \cup S^- . \]
Sufficiency of Criterion: \( b(X) \leq o(X) \) for all \( X \subseteq S^+ \cup S^- \)

Base polytope of submodular function \( o \):
\[ B(o) := \{ y \in \mathbb{R}^{S^+ \cup S^-} \mid y(X) \leq o(X) \ \forall X \subseteq S^+ \cup S^-, \ y(S^+ \cup S^-) = 0 \} \]

[Edmonds ’70]: vertices of \( B(o) \) \( \leftrightarrow \) linear orders \( \prec \) of \( S^+ \cup S^- \)

That is, each vertex is greedy solution \( y \prec \) for some order \( r_1 \prec \cdots \prec r_k \):
\[ y_{r_i} := o(\{r_1, \ldots, r_i\}) - o(\{r_1, \ldots, r_{i-1}\}), \ \text{for} \ i = 1, \ldots, k. \]

Note: \( y \prec \) is supply/demand vector satisfied by lex-max flow over time \( f \prec \); if \( b \in B(o) \), then a convex combination of lex-max flows satisfies \( b \). \( \square \)
Convex Combination of Lex-Max Flows Over Time

$u \equiv 1 \quad \tau \equiv 1 \quad \theta = 4$

$t \prec s_1 \prec s_2$

$t \prec s_2 \prec s_1$

no flow

two units of flow from $s_1$ to $t$

two units of flow from $s_2$ to $t$
Computing Lex-Max Flows Over Time

Given order:

\[ s_1 ≺ t_1 ≺ s_2 ≺ t_2 \]

\[ u ≡ 1 \]
\[ \tau ≡ 1 \]
\[ \theta = 4 \]

Outline of algorithm:

- start with zero flow
- always maintain static min-cost flow with path decomposition
- consider terminals in reverse order
  - for sink \( t_i \), add arc \( t_i t \), find min-cost circulation in residual graph
  - for source \( s_i \), delete arc \( ss_i \), find min-cost maximum \( s-s_i \)-flow

Observation. If \( u \in \mathbb{Z}^A \), the computed lex-max flow over time is integral.
Finding Convex Combination of Lex-Max Flows Over Time

Consider submodular function given by

\[ g(X) := o(X) - b(X) \quad \text{for } X \subseteq S^+ \cup S^- . \]

Then,

\[ \mathcal{B}(g) = \mathcal{B}(o) - b , \]

and thus

\[ b \in \mathcal{B}(o) \iff 0 \in \mathcal{B}(g) \iff \min_{X \subseteq S^+ \cup S^-} g(X) = 0 . \]

Idea of SFM algorithms:

Output: \( 0 = \arg\max \{ y^-(U) \mid y \in \mathcal{B}(g) \} \)

as convex combination of vertices

\[ \implies b \text{ as convex combination of vertices of } \mathcal{B}(o) \]
Finding Integral Flow Over Time similar to [Hoppe, Tardos 2000]

\[ \mathcal{B}(o) := \{ y \in \mathbb{R}^{S^+ \cup S^-} \mid y(X) \leq o(X) \ \forall X \subseteq S^+ \cup S^-, \ y(S^+ \cup S^-) = 0 \} \]

**Idea:** Carefully tighten constraints on \( \mathcal{B}(o) \) until \( b \) is a vertex of \( \mathcal{B}(o) \).

**Combinatorial implementation:** Split one terminal at a time and add delay

\[ b_i' \quad b_i - b_i' \]

\[ s_i' \quad s_i \quad \tau_{s_i's_i} = \gamma \]

How to choose terminal?

Maintain chain of tight subsets

\[ \emptyset \subset X_1 \subset \cdots \subset S^+ \cup S^- \]

with \( o(X_j) = b(X_j) \) for all \( j \). Set

\[ b_i' \!(\gamma) := o^{\gamma}(X_j \cup \{s_i'\}) - o(X_j). \]
Finding Integral Flow Over Time similar to [Hoppe, Tardos 2000]

\[ \mathcal{B}(o) := \{ y \in \mathbb{R}^{S^+ \cup S^-} \mid y(X) \leq o(X) \ \forall X \subseteq S^+ \cup S^-, \ y(S^+ \cup S^-) = 0 \} \]

**Idea:** Carefully tighten constraints on \( \mathcal{B}(o) \) until \( b \) is a vertex of \( \mathcal{B}(o) \).

**Combinatorial implementation:** Split one terminal at a time and add delay

\[ b'_i \quad \text{to} \quad b_i - b'_i \]

How to choose terminal?

Maintain chain of tight subsets

\[ \emptyset \subset X_1 \subset \cdots \subset S^+ \cup S^- \]

with \( o(X_j) = b(X_j) \) for all \( j \). Set

\[ b'_i(\gamma) := o''(X_j \cup \{s'_i\}) - o(X_j). \]

How to choose \( \gamma \)?

- \( \gamma = 0 \) yields infeasible problem
- \( \gamma = \theta \) is feasible

choose minimum feasible \( \gamma \) (parametric SFM problem!)

This yields another tight subset \( Q \) with \( X_j \cup \{s'_i\} \subset Q \subset X_{j+1} \).
Computing Integral Transshipment Over Time: Example

\[ u \equiv 1 \quad \tau \equiv 1 \quad \theta = 4 \]

\[ b \left( o \right) \]

\[ s_1 \prec s_2 \prec t \]

\[ y_{s_1} \]

\[ y_{s_2} \]

\[ y_t \]

\[ t \prec s_1 \prec s_2 \]
Computing Integral Transshipment Over Time: Example

\[ u \equiv 1 \quad \tau \equiv 1 \quad \theta = 4 \]

\[ \tau_{s'_2 s_2} = 1 \]

\[ s'_2 \sim t \sim s_1 \sim t \]

\[ s'_2 \sim t \sim s_1 \sim t \]

\[ B(o) \]

\[ y_{s_1}, y_{s_2'}, y_t \]

\[ t \sim s_1 \sim s_2' \]

\[ s_2' \sim s_1 \sim t \]

\[ s'_2 \sim t \sim s_1 \sim t \]
How to find the minimum feasible time horizon $\theta^*$?

$$\theta^* = \min\{\theta \mid o^\theta(X) - b(X) \geq 0 \text{ for all } X \subseteq S^+ \cup S^-\}$$

Related recent work: [McCormick, Oriolo, Peis 2014] [Goemans, Gupta, Jaillet 2017] [Schlöter 2018] [Kamiyama 2019] [Dadush, Koh, Natura, Végh 2021]
## Conclusion

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Quickest s-t-Flow Problem</strong></td>
<td>[Saho, Shigeno 2017] (\tilde{O}(m^2 n))</td>
</tr>
<tr>
<td><strong>Evacuation Problem (single source or single sink)</strong></td>
<td>[Schlöter 2018; Kamiyama 2019] (\tilde{O}(m^2 k^5 + m^2 nk))</td>
</tr>
<tr>
<td><strong>Quickest Transshipment Problem (multiple sources and sinks)</strong></td>
<td>[Hoppe, Tardos 2000] (\tilde{O}(m^4 k^{15}))</td>
</tr>
<tr>
<td></td>
<td>[Schlöter, Sk. 2017] (\tilde{O}(m^4 k^{14})^*)</td>
</tr>
<tr>
<td></td>
<td>[Schlöter, Sk., Tran 2022] (\tilde{O}(m^2 k^5 + m^3 k^3 + m^3 n)^*)</td>
</tr>
<tr>
<td></td>
<td>[Schlöter, Sk., Tran 2023+] (\tilde{O}(m^2 k^6 + m^3 k^4 + m^3 n))</td>
</tr>
</tbody>
</table>

**Main Open Problem:**

How to minimize specific submodular functions more efficiently?