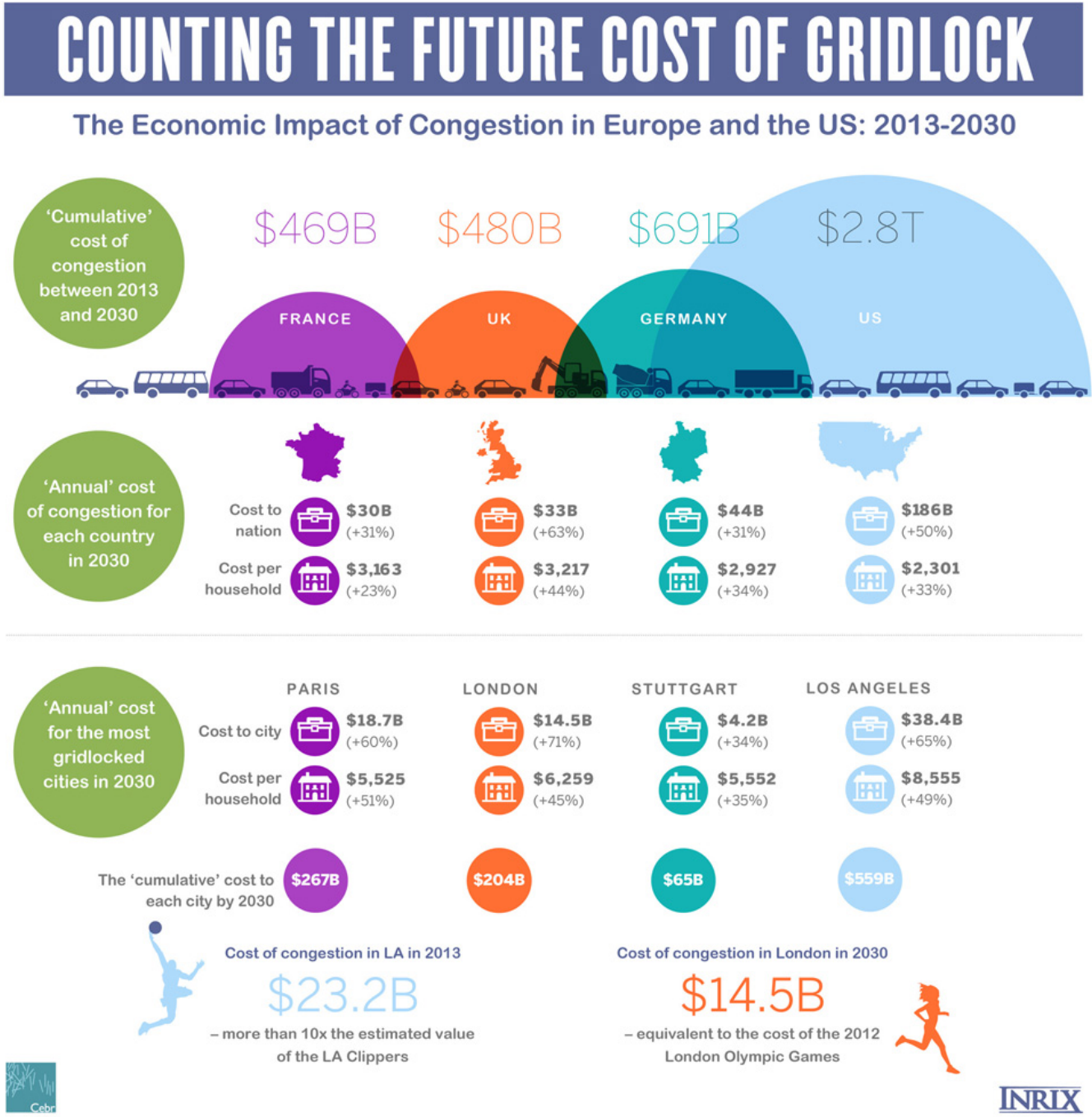
New Complexity Results and Algorithms for Minimum Tollbooth Problem

Soumya Basu
Thanasis Lianeas Evdokia Nikolova

Department of ECE
The University of Texas at Austin

WINE 2015, Amsterdam


COUNTING THE FUTURE COST OF GRIDLOCK
The Economic Impact of Congestion in Europe and the US: 2013-2030
'Cumulative' cost of congestion between 2013 and 2030
$469B
$480B
$691B
$2.8T
FRANCE
UK
GERMANY
US
'Annual' cost of congestion for each country in 2030
Cost to nation
$30B (+31%)
$33B (+63%)
$44B (+31%)
$186B (+50%)
Cost per household
$3,163 (+23%)
$3,217 (+44%)
$2,927 (+34%)
$2,301 (+33%)
'Annual' cost for the most gridlocked cities in 2030
PARIS
LONDON
STUTTGART
LOS ANGELES
Cost to city
$18.7B (+60%)
$14.5B (+71%)
$4.2B (+34%)
$38.4B (+65%)
Cost per household
$5,525 (+51%)
$6,259 (+45%)
$5,552 (+35%)
$8,555 (+49%)
The 'cumulative' cost to each city by 2030
$267B
$204B
$65B
$559B
Cost of congestion in LA in 2013
$23.2B
– more than 10x the estimated value of the LA Clippers
Cost of congestion in London in 2030
$14.5B
– equivalent to the cost of the 2012 London Olympic Games
Cebr
INRIX

Congestion Gridlock

“ The combined annual cost of gridlock to these (U.S., U.K., France and Germany) countries is expected to soar to $ 293.1 billion by 2030. . . ” –INRIX and the CEBR

How to tackle congestion?

Introduction

Congestion Gridlock

“ The combined annual cost of gridlock to these (U.S., U.K., France and Germany) countries is expected to soar to $ 293.1 billion by 2030. . . ” –INRIX and the CEBR

How to tackle congestion?

- Infrastructure growth is good.

Introduction

Congestion Gridlock

“ The combined annual cost of gridlock to these (U.S., U.K., France and Germany) countries is expected to soar to $ 293.1 billion by 2030. . . ” –INRIX and the CEBR

How to tackle congestion?

- Infrastructure growth is good.
- But Not Always: **Selfish users** lead us to **Braess' Paradox**
- Solution: Influence the users by placing appropriate Tolls

Introduction

Congestion Gridlock

“ The combined annual cost of gridlock to these (U.S., U.K., France and Germany) countries is expected to soar to $ 293.1 billion by 2030. . . ” –INRIX and the CEBR

How to tackle congestion?

- Infrastructure growth is good.
- But Not Always: **Selfish users** lead us to **Braess' Paradox**
- Solution: Influence the users by placing appropriate Tolls
- No Free Lunch: Each toll comes with **large overhead**

Outline

System Model

- Directed graph $G(V, E)$ with single commodity (s, t)
- Flow dependent latency, $\mathcal{L} = \{\ell_e(\cdot) : \forall e \in E\}$
- Traffic network, $\mathcal{G} = \{G, (s, t), \mathcal{L}\}$

Social Optimum (SO)

SO flow o is a flow that minimizes social cost.

Nash Equilibrium (NE)

A flow is said to be in NE iff property (1) and (2) holds:
(1) All used paths cost **L** (2) All unused paths cost $\geq$ **L**

Problem Definition

- **Tolled edge cost**: Under toll θ and flow f, tolled edge cost $c_e = \ell_e(f_e) + \theta_e$
- **Induce flow** f: Under the **tolled edge cost** f is in NE
- **Induced length**: Under NE the **tolled cost** of used paths

Minimum Tollbooth Problem (MINTB)

- Given traffic network $\mathcal{G}$ and an optimal flow **o**
- Find toll with **minimum support** which **induces** o

MINTB: Where do we stand?

- Formulated as Mixed integer linear program (**MILP**). *Hearn et al., 1998*
- Heuristics developed: **Genetic Algorithm, LP relaxation**. *Hardwood et al., '08; Bai et al., '09*
- In Multi-commodity networks it is **NP-hard**. *Bai et al.,'08*
- Design toll for inducing **general flow**. *Harks et al.,'08*

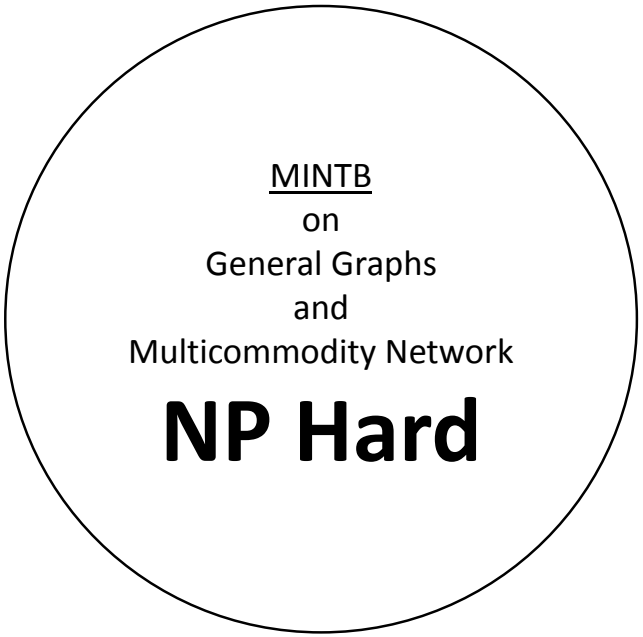


Figure: Complexity Diagram

Contributions

- **First APX-hardness:**
  Single commodity networks with **linear latencies**.
- **MINTB with used edges only**:
  Is *MINTB* efficiently solvable? **No still NP hard.**
- **First Exact Poly-time Algorithm**:
  Algorithm for **Series-Parallel** graphs.

First APX hardness result

Theorem

*For instances with **linear latencies** and **single commodity**, it is **NP-hard to approximate** the solution of MINTB by a factor of less than* **1.1377**.

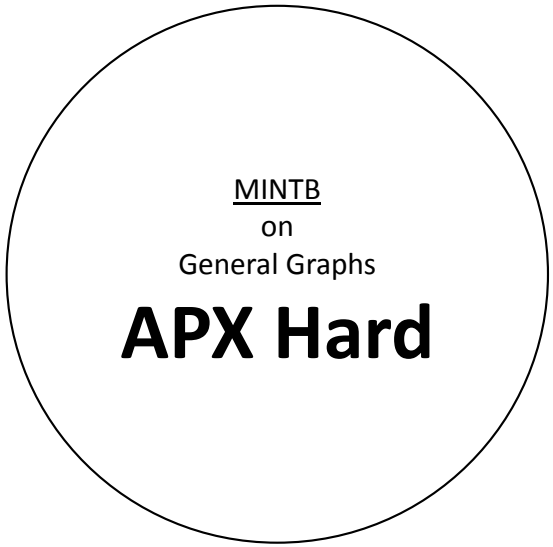


Figure: Complexity Diagram

MINTB Reduction: Vertex Cover

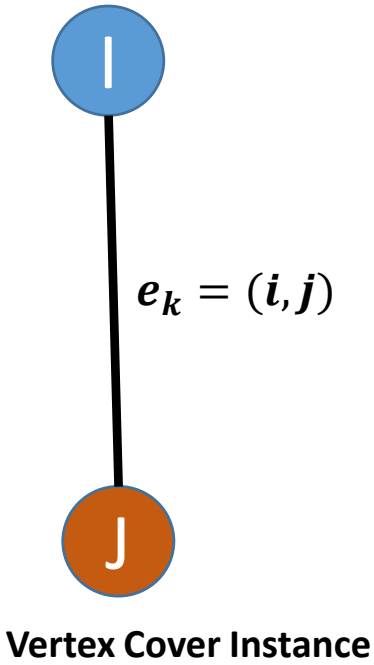
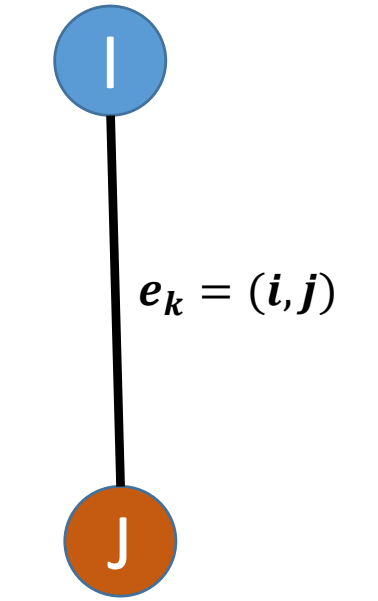


Figure: MINTB Reduction

MINTB Reduction: Vertex Cover



Vertex Cover Instance

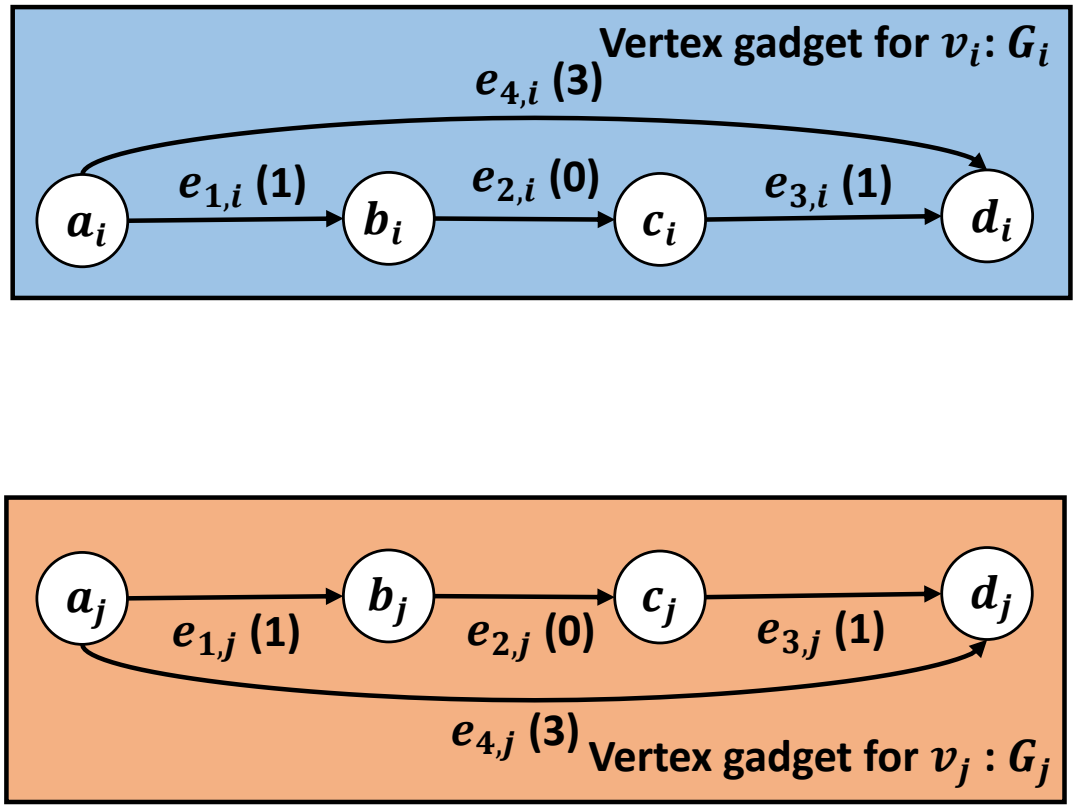
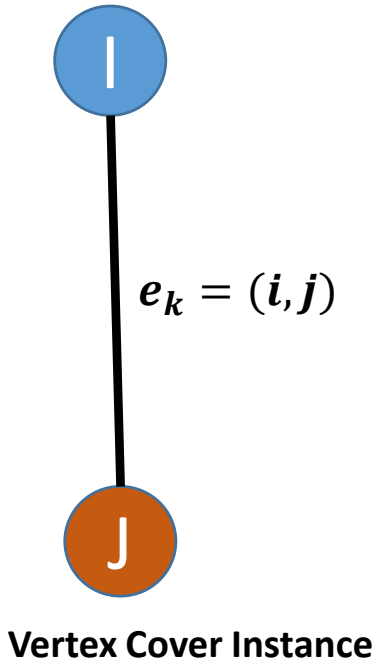


Figure: MINTB Reduction

MINTB Reduction: Vertex Cover



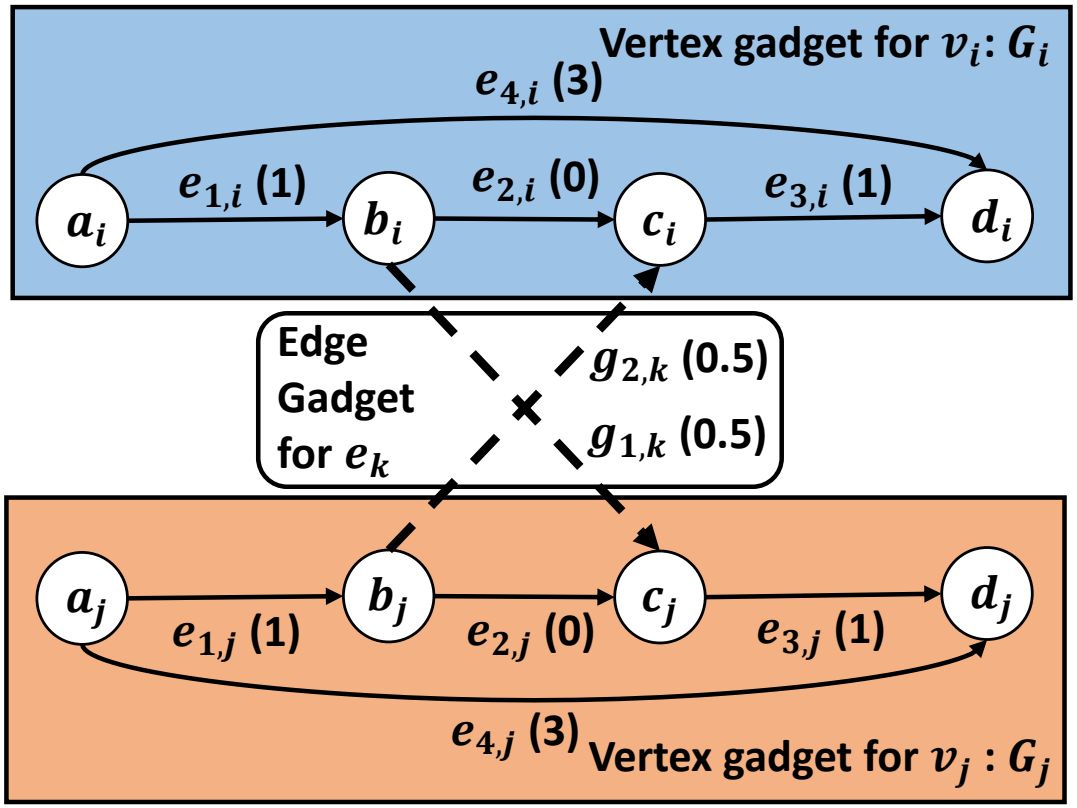


Figure: MINTB Reduction

MINTB Reduction: Vertex Cover

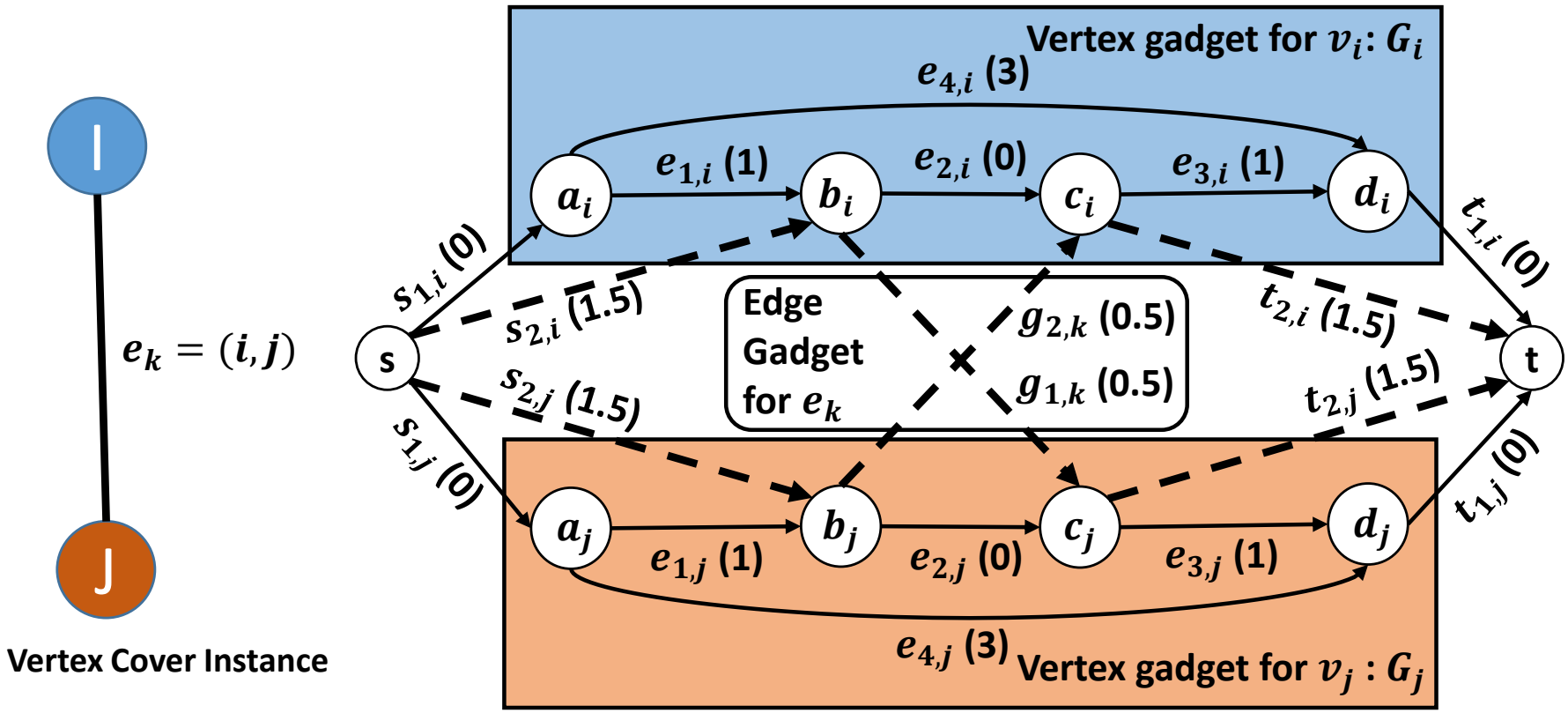


Figure: MINTB Reduction

MINTB Reduction: Vertex Cover

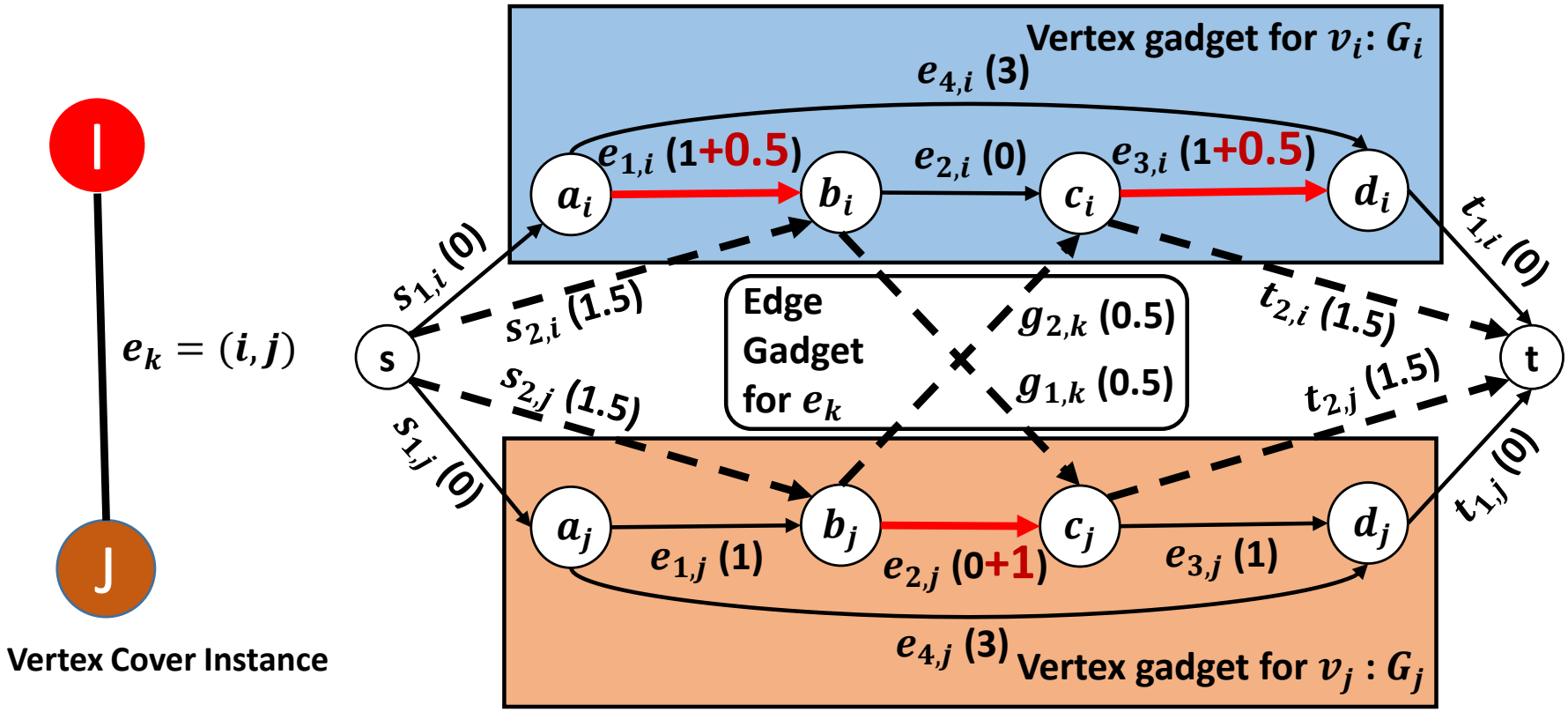


Figure: MINTB Reduction

MINTB Reduction: Vertex Cover

- Given Vertex Cover instance G_{vc} with n_{vc} vertices
- Create a Single Commodity network $\mathcal{G}$

Lemma 1

$\exists$ A Vertex Cover of size $\mathbf{x}$ in G_{vc} $\iff$
$\exists$ Opt-inducing toll with support $\mathbf{n_{vc}} + \mathbf{x}$ in $\mathcal{G}$.

Theorem

It is NP hard to approximate Minimum Vertex Cover to within a factor of **1.3606**. *Dinur et al 2005*

Are Unused Edges the Troublemaker?

- **Motivation:** Less overhead present in edge removal.
- **Model:** Unused edges in social optimum flow are removed.

Theorem

*For instances with **linear latencies**, it is **NP-hard** to solve MINTB even if **all edges are used** by the optimal solution.*

Reduction follows from *PARTITION* problem.

Complexity Diagram

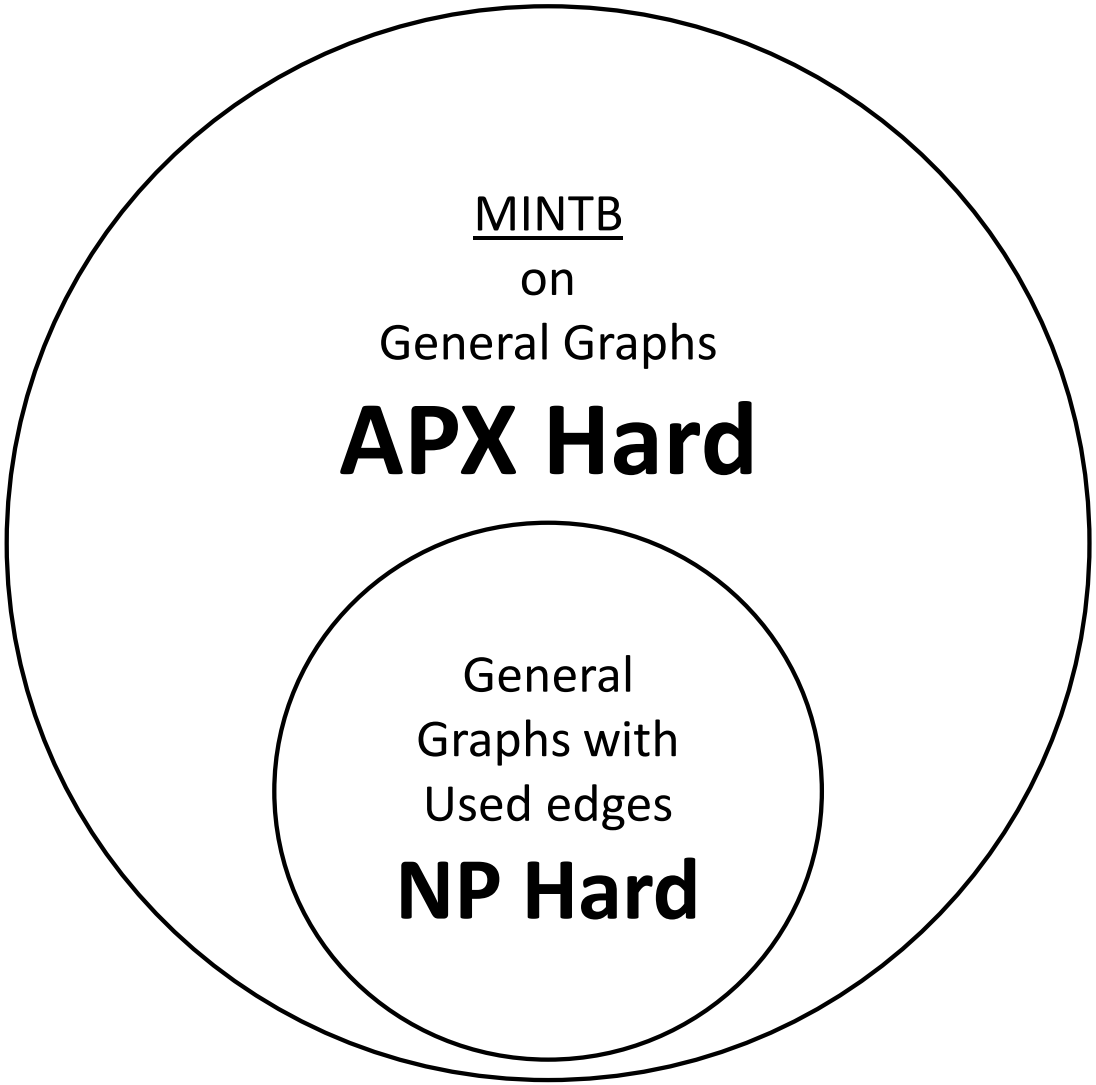


Figure: Complexity Diagram

Series Parallel Graphs

Series Parallel Graphs (*SP*)

An *SP* graph is created by starting from a *directed edge* and *inductively connecting* two graphs in *series or in parallel*.

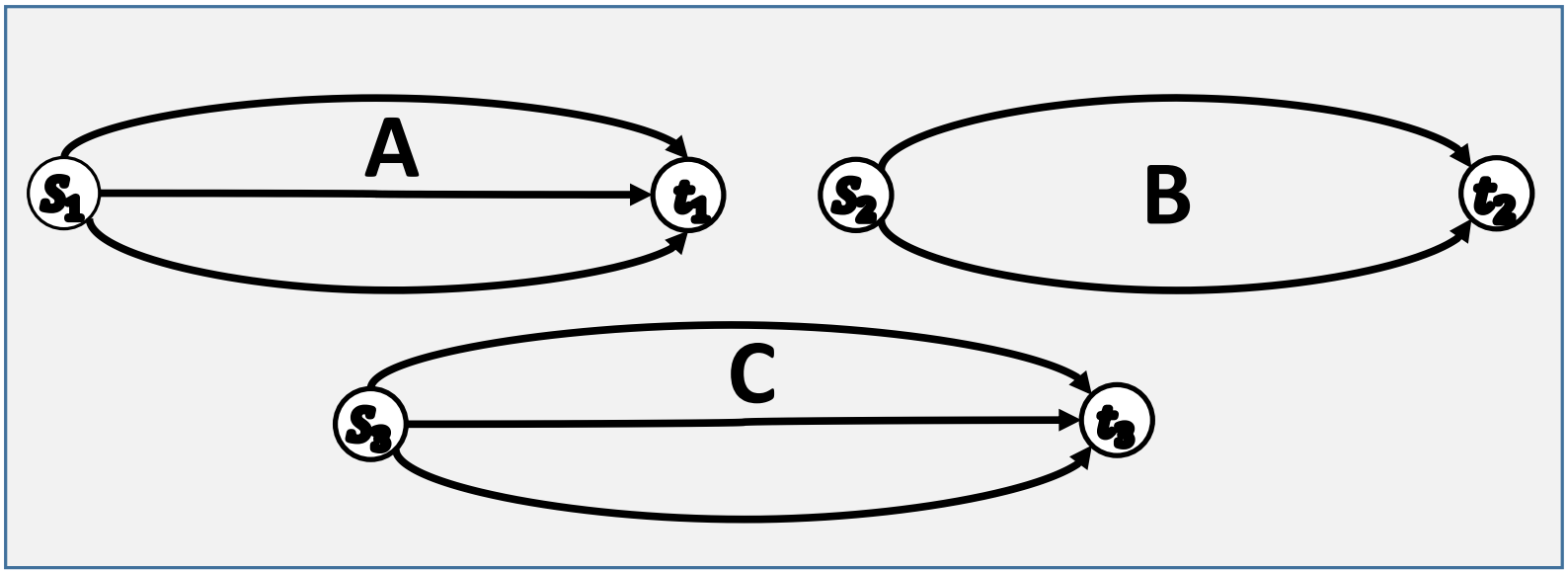


Figure: *SP* Graph

Series Parallel Graphs

Series Parallel Graphs (*SP*)

An *SP* graph is created by starting from a *directed edge* and *inductively connecting* two graphs in *series or in parallel*.

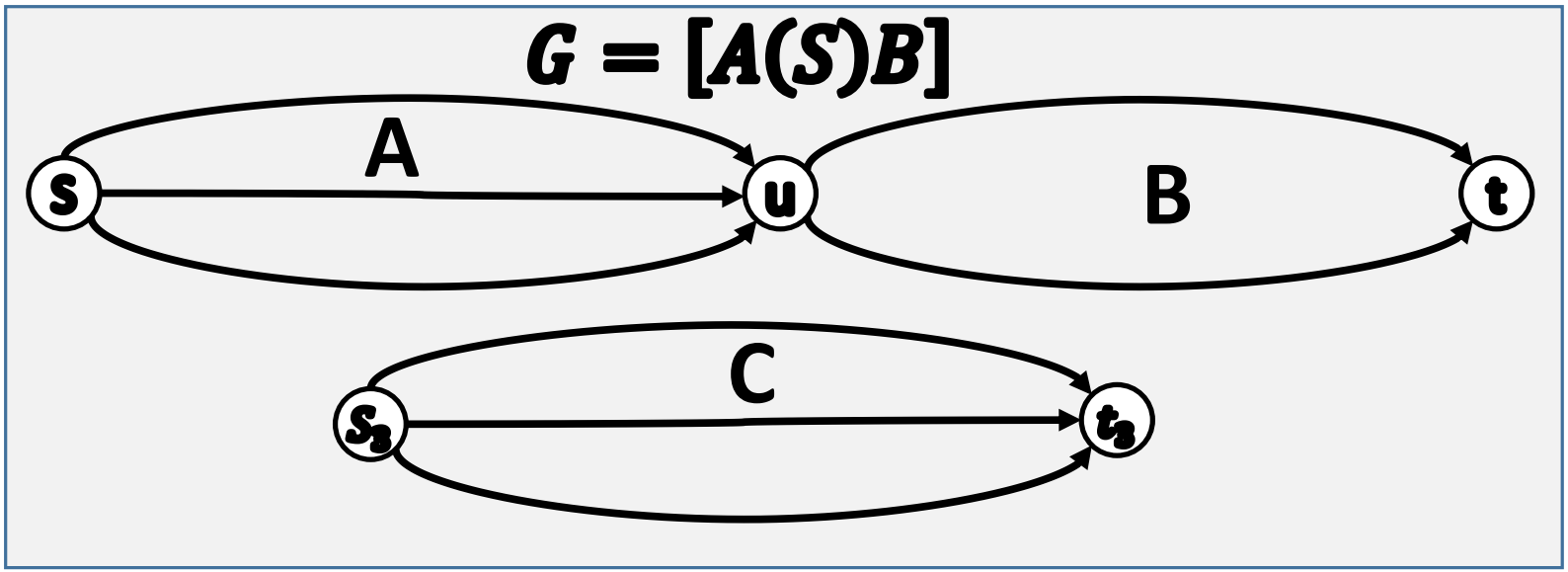


Figure: *SP* Graph and Parse Tree

Series Parallel Graphs

Series Parallel Graphs (*SP*)

An *SP* graph is created by starting from a *directed edge* and *inductively connecting* two graphs in *series or in parallel*.

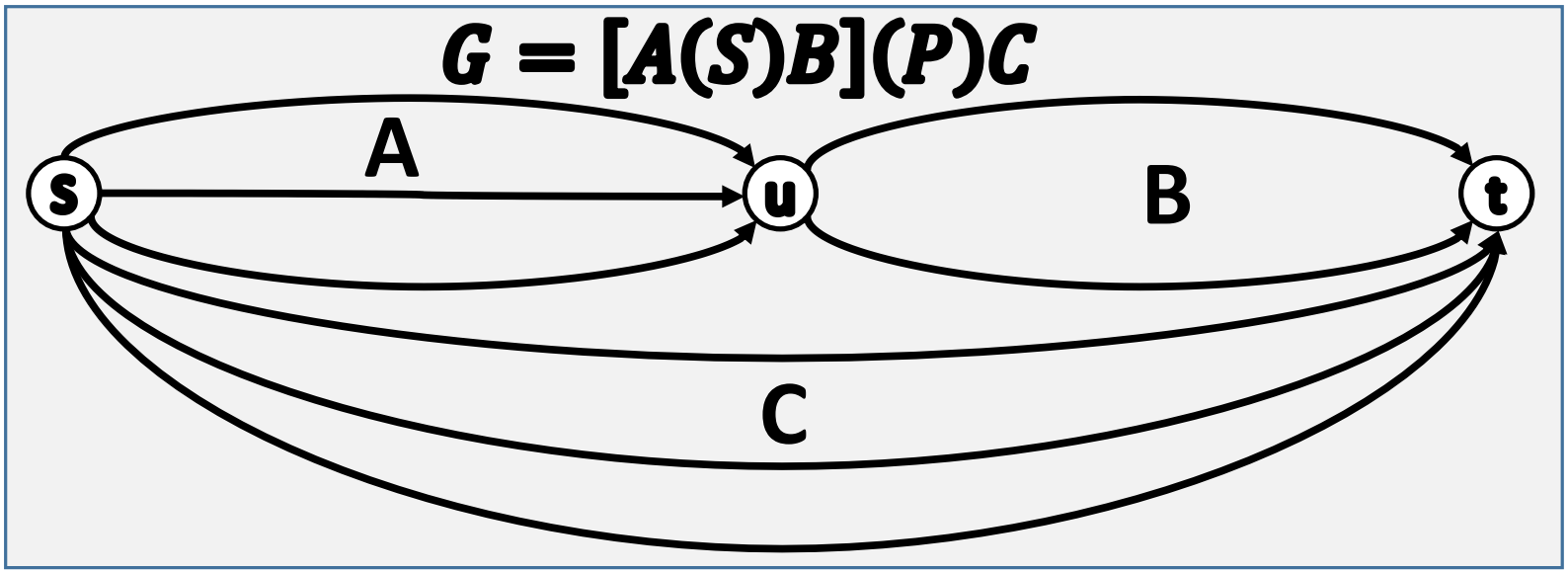


Figure: *SP* Graph

Algorithm for Parallel Link (PL) Graph

Algorithm 1

Input: A parallel link network
Output: Induce L with min support

1. Sort edges
2. Append length $\ell_{end} = \infty$

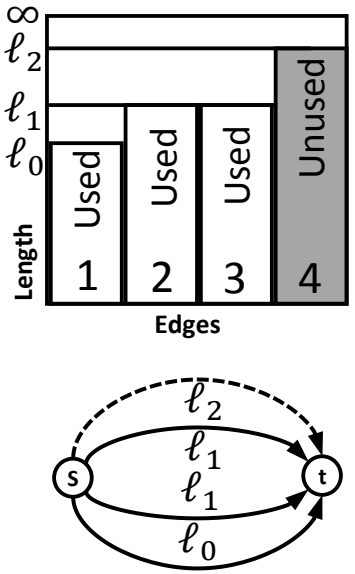


Figure: Example

Algorithm for Parallel Link (PL) Graph

Algorithm 1

Input: A parallel link network
Output: Induce L with min support

1. Sort edges
2. Append length $\ell_{end} = \infty$
3. Max used edge cost ℓ_{max}

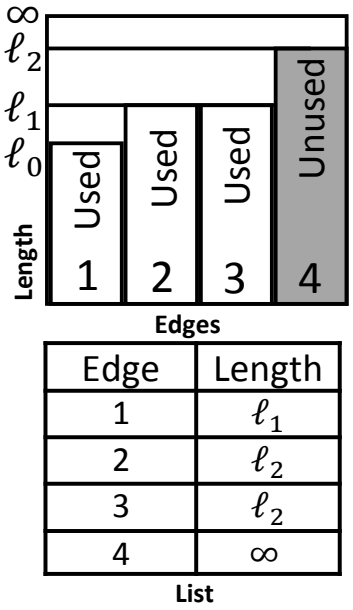


Figure: Example: $\ell_{max} = \ell_1$, $i_0 = 1$

Algorithm for Parallel Link (PL) Graph

Algorithm 1

Input: A parallel link network
Output: Induce L with min support

1. Sort edges
2. Append length $\ell_{end} = \infty$
3. Max used edge cost ℓ_{max}
4. Create list $\{(\eta, \ell)\}$
   - Toll on edges $1, \ldots, \eta$
   - Maximally induce ℓ

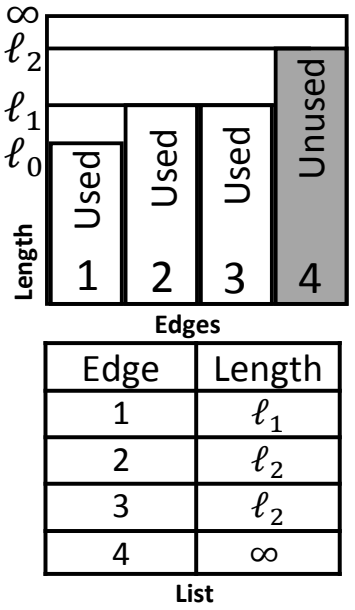


Figure: Example: $\ell_{max} = \ell_1$, $i_0 = 1$

Algorithm for Parallel Link (PL) Graph

Algorithm 1

Input: A parallel link network
Output: Induce L with min support

1. Sort edges
2. Append length $\ell_{end} = \infty$
3. Max used edge cost ℓ_{max}
4. Create list $\{(\eta, \ell)\}$
   - Toll on edges $1, \ldots, \eta$
   - Maximally induce ℓ
5. Place toll to induce L

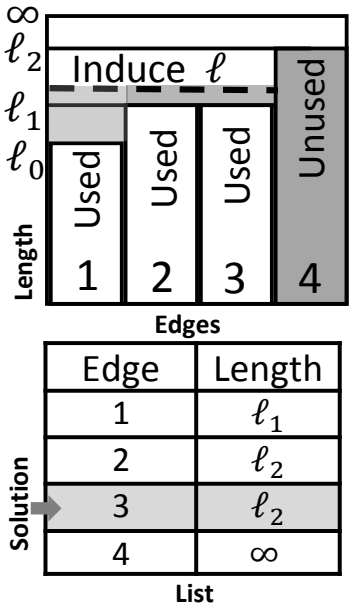


Figure: Example: Placing tolls

Series Parallel in P: Extension to SP

Induce length L

- Series comb. $\mathbf{G_1(S)G_2}$: Induce $\mathbf{L_1}(\leq L)$ in $\mathbf{G_1}$ and $\mathbf{L - L_1}$ in $\mathbf{G_2}$.
- Parallel comb. $\mathbf{G_1(P)G_2}$: Induce $\mathbf{L}$ in $\mathbf{G_1}$ and $\mathbf{G_2}$.

Monotonicity Lemma

- In a SP graph G with maximum used path length $\ell_{\mathbf{max}}$,
  We can induce length $\mathbf{L} \iff \mathbf{L} \geq \ell_{\mathbf{max}}$
- $\mathbf{L}$ is **induced optimally** with support $\mathbf{T}$
  $\Rightarrow \ell \leq \mathbf{L}$ can be **induced optimally** with support $\mathbf{t} \leq \mathbf{T}$

Series Parallel in *P*: Extension to *SP*

MakeList: Bottom-up List creation: Dynamic Programming

1) **Start** from PL 2) **Combine** list in series/parallel 3) **Recur** till root

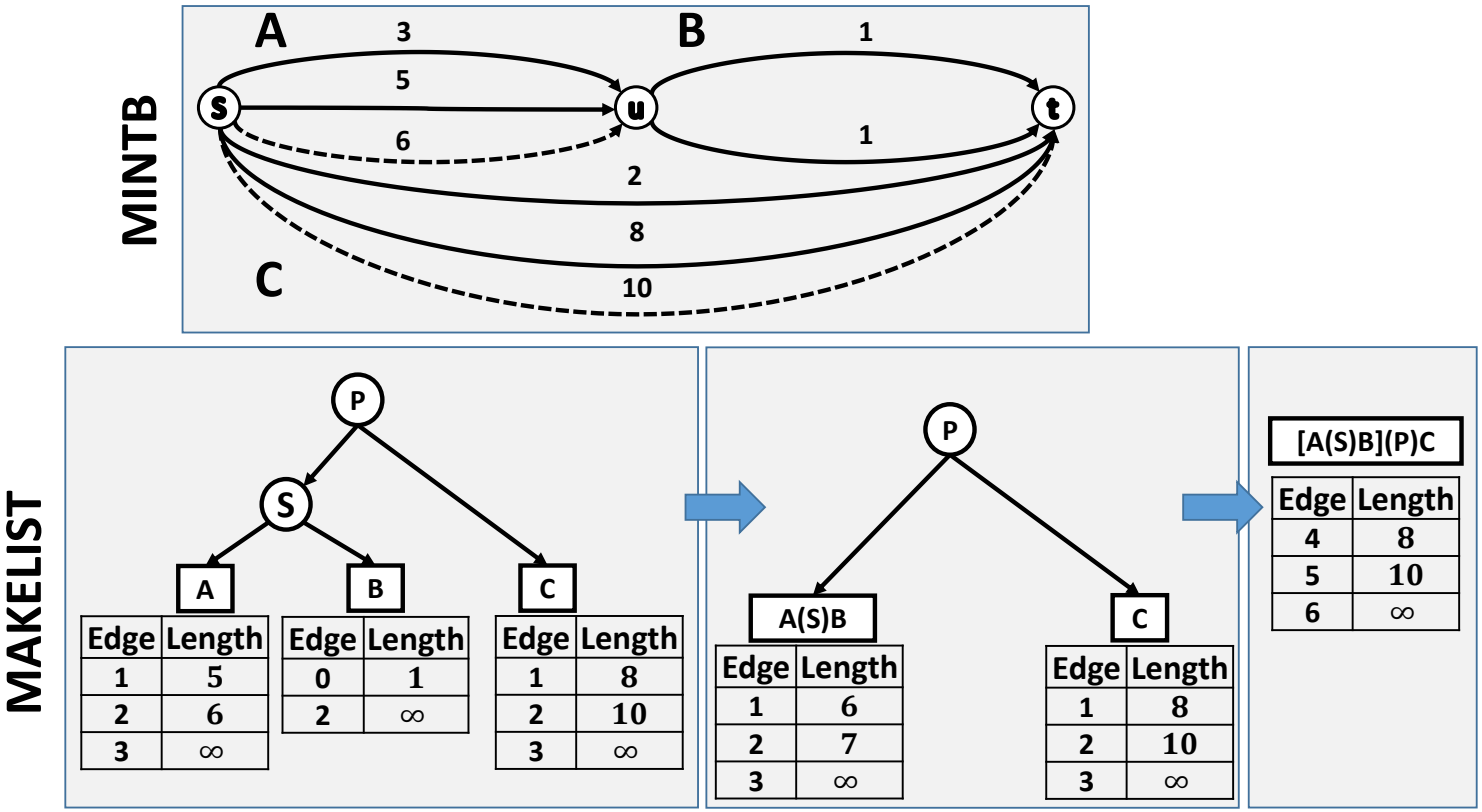


Figure: Step1: Example Run of MINTB Algorithm

Series Parallel in *P*: Extension to *SP*

Placetoll: Top-down Induction of length: Traceback

1) **Start** from root. 2) **Branch** in series/parallel. 3) **Recur** till PL.

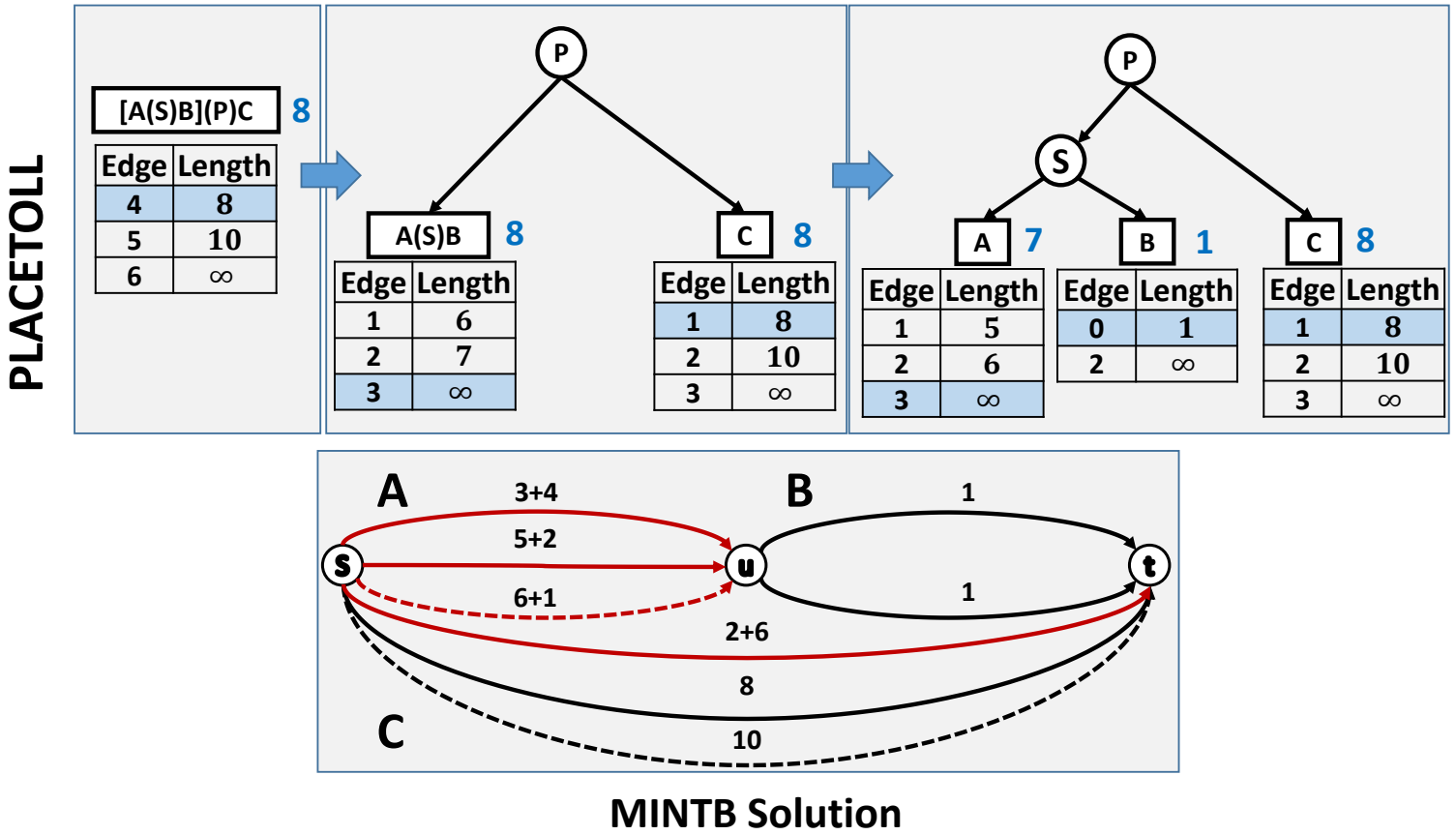


Figure: Step2: Example Run of MINTB Algorithm

Series Parallel in P: Extension to SP

Theorem

The MINTB on a SP graph with $|E| = m$, is solved optimally in time $\mathcal{O}\left(m^3\right)$.

Series Parallel in P: Extension to SP

Theorem

The MINTB on a SP graph with $|E| = m$, is solved optimally in time $\mathcal{O}\left(m^3\right)$.

Presence of Braess Structure:

- The **Monotonicity Lemma** breaks.
- Edges to induce length $3 = 3$.
- Edges to induce length $4 = 2$.

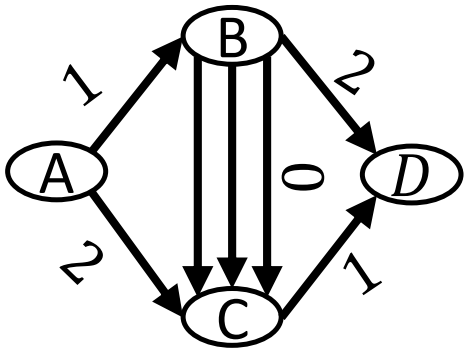


Figure: Counter

Summary

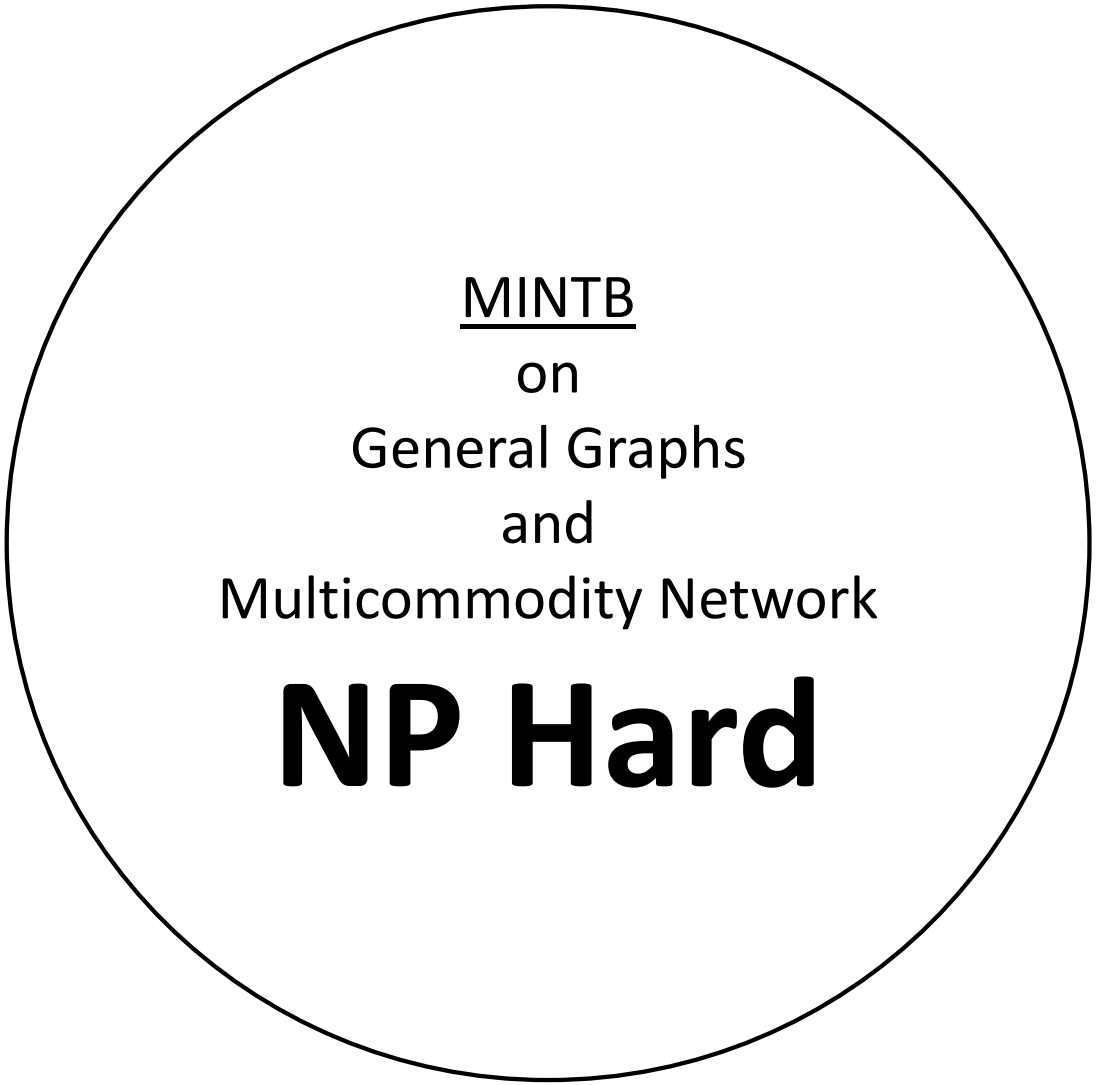


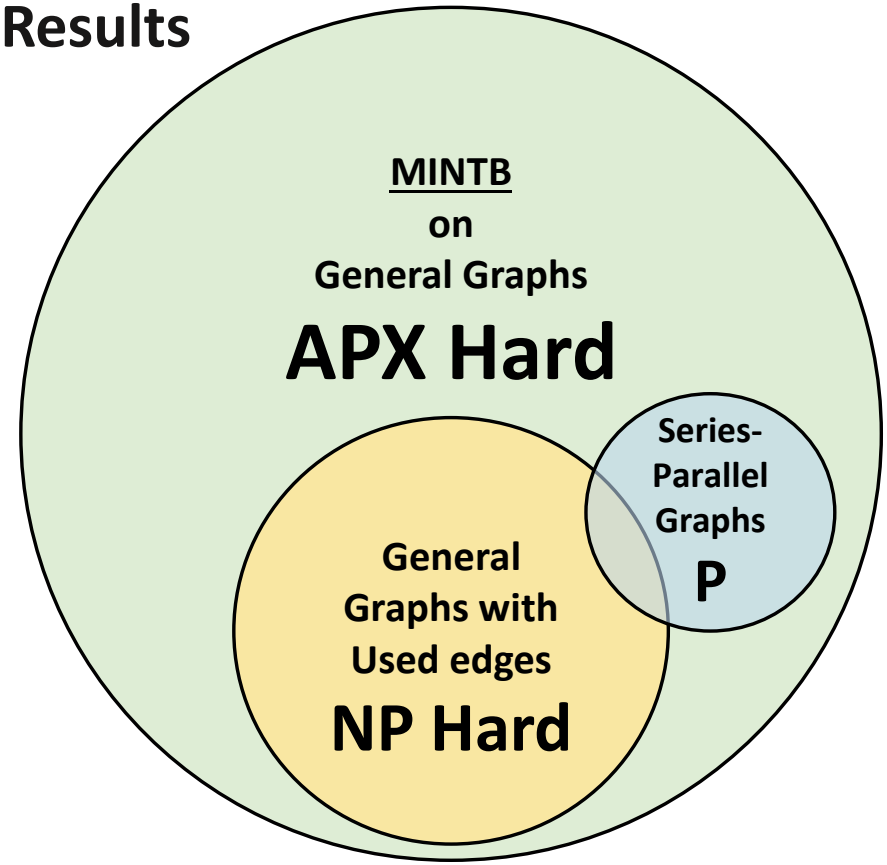
Figure: Complexity Diagram **Prior to the Work**

Summary

- Reduction From Vertex Cover Problem
- APX hardness of 1.1377
- Exploits Unused Edges in an Optimal Flow

- Reduction from Partition Problem
- Exploits Braess Structure

Complexity Results



Algorithm

- Absence of Braess Structure
- Monotonicity Property
- Tree Decomposition

- Dynamic Programming
- Bottom Up List Creation
- Top Down Toll Placement

Future Directions

- Is the APX hardness result **tight**?
  - YES Find matching approximation algorithms.
  - NO Give tighter APX hardness results.
- Design Practical Algorithms:
  - Improved heuristics with **performance guarantee**.
  - Faster algorithms for **large-scale traffic networks**.
- What happens while **Taxing Sub-networks**?

Future Directions

- Is the APX hardness result **tight**?
  - YES Find matching approximation algorithms.
  - NO Give tighter APX hardness results.
- Design Practical Algorithms:
  - Improved heuristics with **performance guarantee**.
  - Faster algorithms for **large-scale traffic networks**.
- What happens while **Taxing Sub-networks**?

Questions?