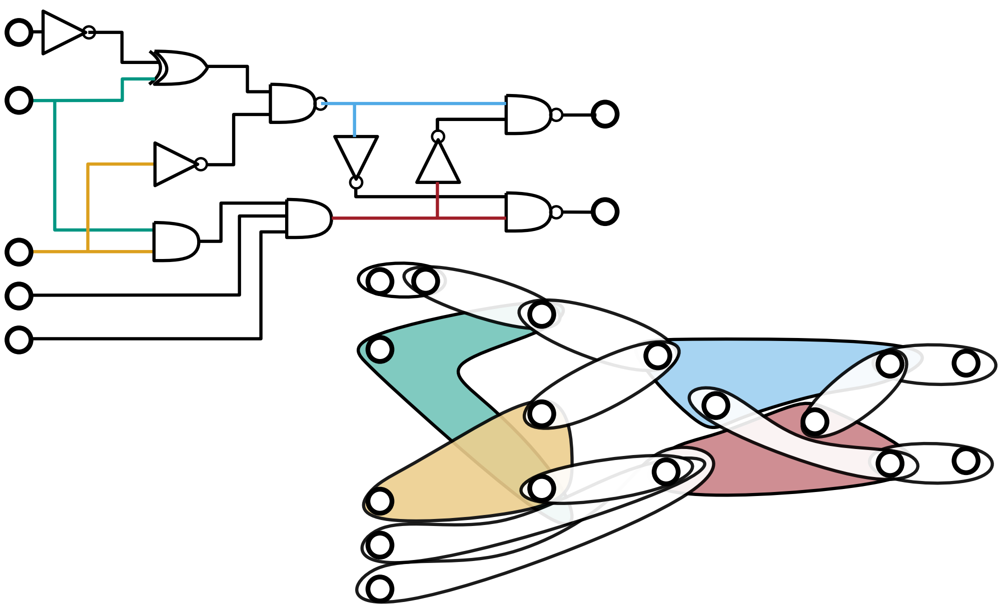
Karlsruhe Institute of Technology

High Quality Hypergraph Partitioning via Max-Flow-Min-Cut Computations

Master Thesis · February 16, 2018

Tobias Heuer

INSTITUTE OF THEORETICAL INFORMATICS · ALGORITHMICS GROUP



Outline

Task

Developing a **local search** algorithm based on **Max-Flow-Min-Cut** computations for the *n*-level hypergraph partitioner **KaHyPar**.

Outline

Karlsruhe Institute of Technology

Task

Developing a **local search** algorithm based on **Max-Flow-Min-Cut** computations for the *n*-level hypergraph partitioner **KaHyPar**.

Contributions

- Outperforms 5 different systems on 73% of 3216 benchmark instances
- Improve quality of *KaHyPar* by 2.5%, while only incurring a slowdon by a factor of 2
- Comparable running time to *hMetis* and outperforms it on 84% of the instances

Institute of Theoretical Informatics
Algorithmics Group

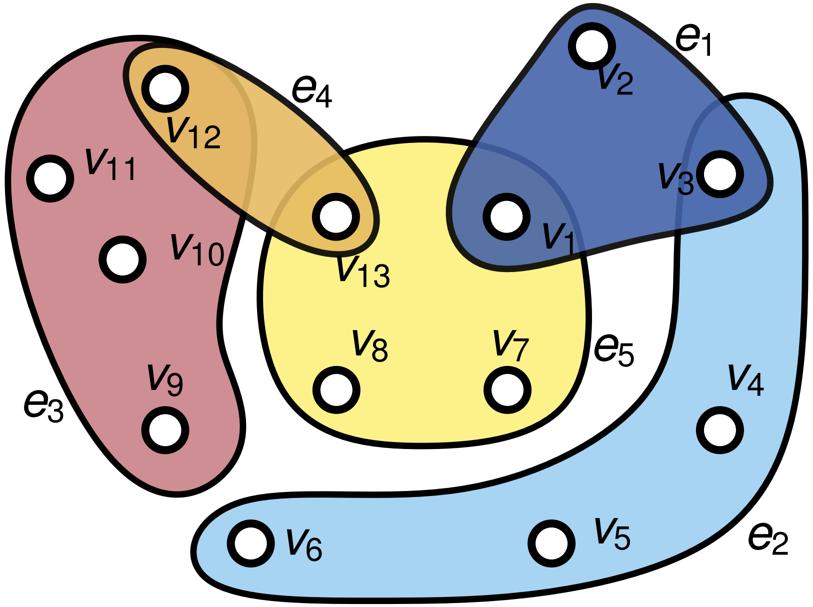
Hypergraphs

[from SEA'17]

- Generalization of graphs
  $\Rightarrow$ hyperedges connect $\geq$ 2 nodes

- Graphs $\Rightarrow$ dyadic ($\mathbf{2}$-**ary**) relationships
- Hypergraphs $\Rightarrow$ ($\mathbf{d}$-**ary**) relationships

- Hypergraph $H = (V, E, c, \omega)$
  - Vertex set $V = \{1, ..., n\}$
  - Edge set $E \subseteq \mathcal{P}(V) \setminus \emptyset$
  - Node weights $c : V \to \mathbb{R}_{\geq 1}$
  - Edge weights $\omega : E \to \mathbb{R}_{\geq 1}$



Institute of Theoretical Informatics
Algorithmics Group

Hypergraphs

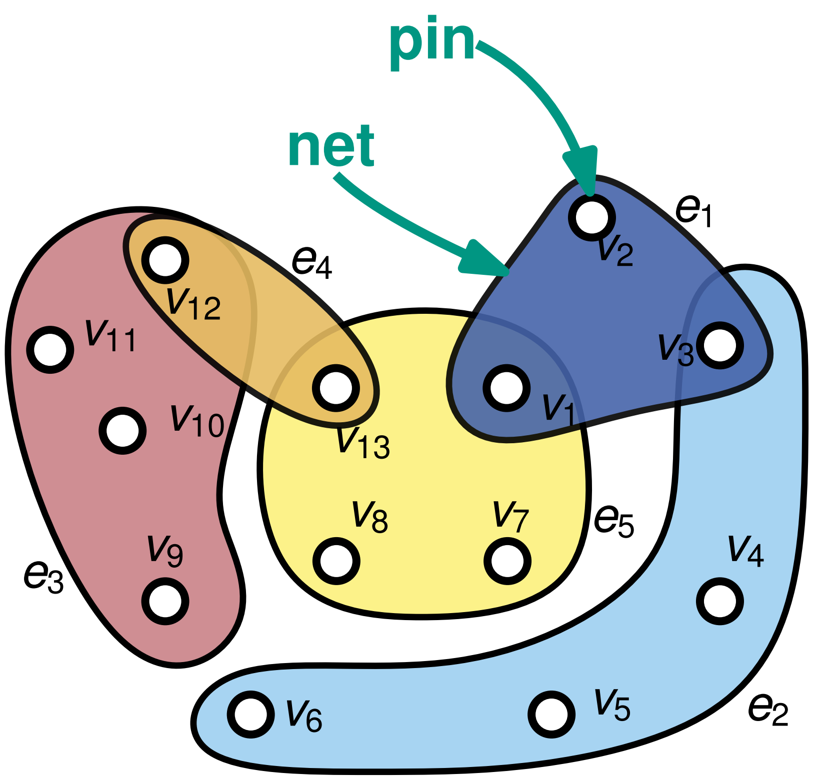
[from SEA'17]

- Generalization of graphs
  $\Rightarrow$ hyperedges connect ≥ 2 nodes

- Graphs $\Rightarrow$ dyadic ($\mathbf{2}$**-ary**) relationships
- Hypergraphs $\Rightarrow$ ($\mathbf{d}$**-ary**) relationships

- Hypergraph $H = (V, E, c, \omega)$
  - Vertex set $V = \{1, ..., n\}$
  - Edge set $E \subseteq \mathcal{P}(V) \setminus \emptyset$
  - Node weights $c : V \rightarrow \mathbb{R}_{\geq 1}$
  - Edge weights $\omega : E \rightarrow \mathbb{R}_{\geq 1}$

- $|P| = \sum_{e \in E} |e| = \sum_{v \in V} d(v)$



Institute of Theoretical Informatics
Algorithmics Group

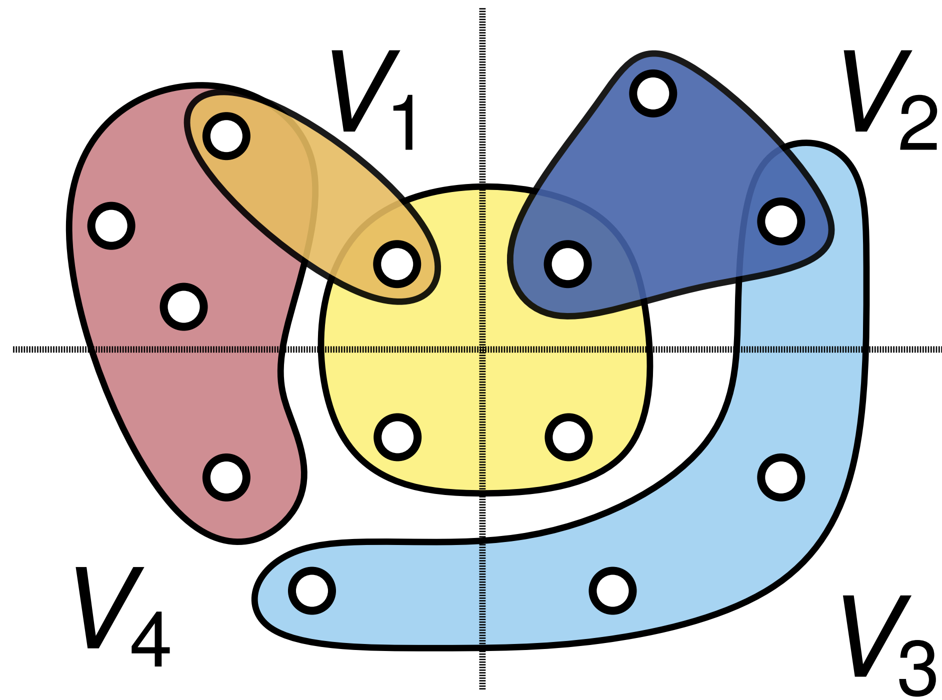
Hypergraph Partitioning Problem

[from SEA'17]

Partition hypergraph $H = (V, E, c, \omega)$ into $\mathbf{k}$ non-empty disjoint blocks $\Pi = \{V_1, \ldots, V_k\}$ such that:

- blocks V_i are **roughly equal-sized**:

$$c(V_i) \leq (1 + \varepsilon) \left\lceil \frac{c(V)}{k} \right\rceil$$



Hypergraph Partitioning Problem

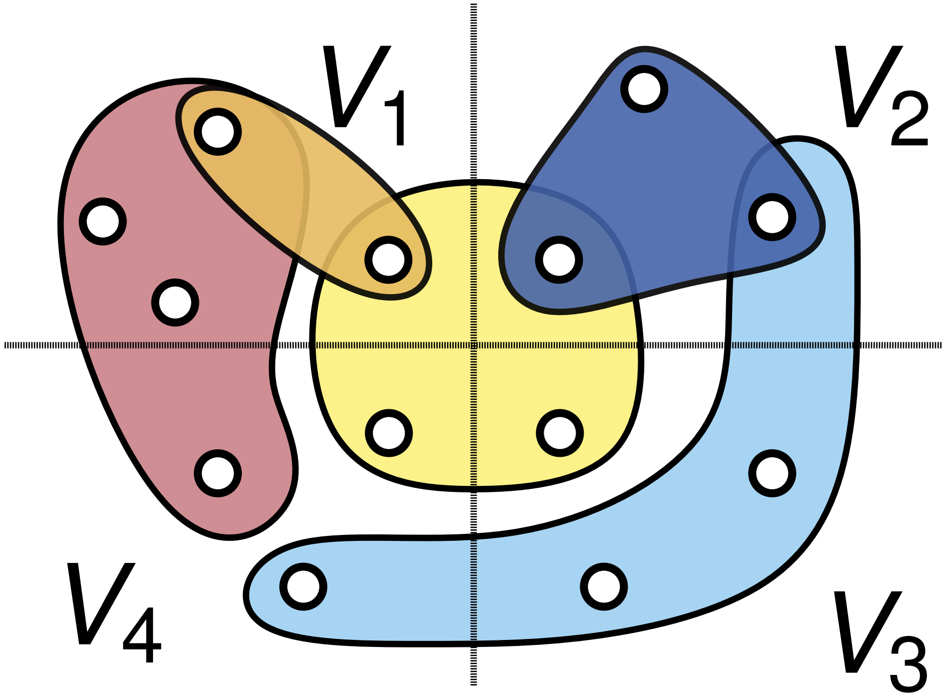
[from SEA'17]

Partition hypergraph $H = (V, E, c, \omega)$ into $\mathbf{k}$ non-empty disjoint blocks $\Pi = \{V_1, \ldots, V_k\}$ such that:

- blocks V_i are **roughly equal-sized**:

imbalance parameter

$$c(V_i) \leq (1 + \varepsilon) \left\lceil \frac{c(V)}{k} \right\rceil$$



Hypergraph Partitioning Problem

[from SEA'17]

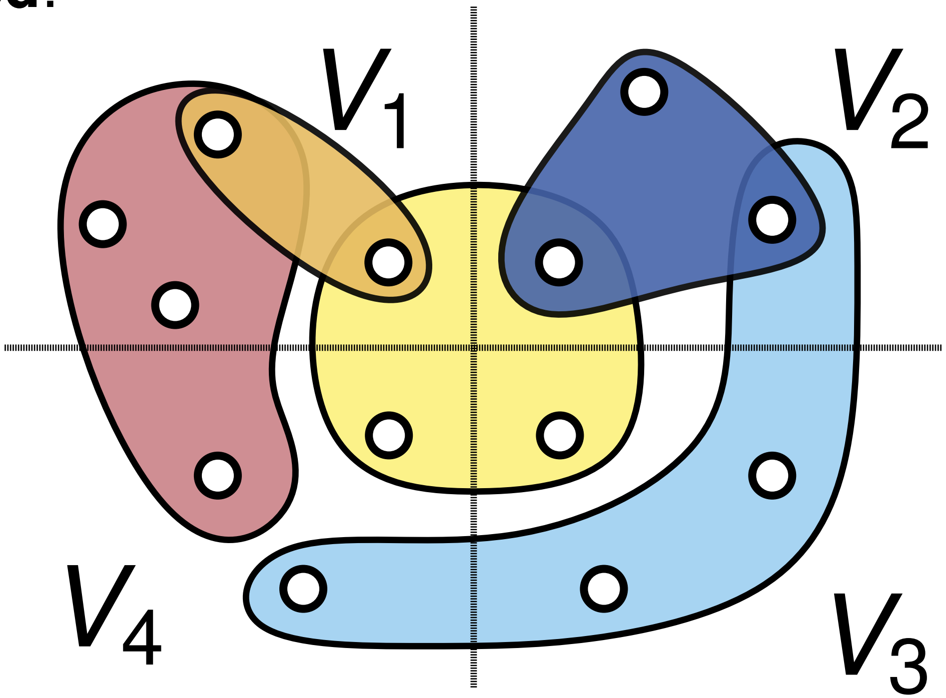
Partition hypergraph $H = (V, E, c, \omega)$ into $\mathbf{k}$ non-empty disjoint blocks $\Pi = \{V_1, \ldots, V_k\}$ such that:

- blocks V_i are **roughly equal-sized**:

imbalance parameter

$$c(V_i) \leq (1 + \varepsilon) \left\lceil \frac{c(V)}{k} \right\rceil$$

- **connectivity** objective is **minimized**:



Institute of Theoretical Informatics
Algorithms Group

Hypergraph Partitioning Problem

[from SEA'17]

Partition hypergraph $H = (V, E, c, \omega)$ into $\mathbf{k}$ non-empty disjoint blocks $\Pi = \{V_1, \ldots, V_k\}$ such that:

- blocks V_i are **roughly equal-sized**:

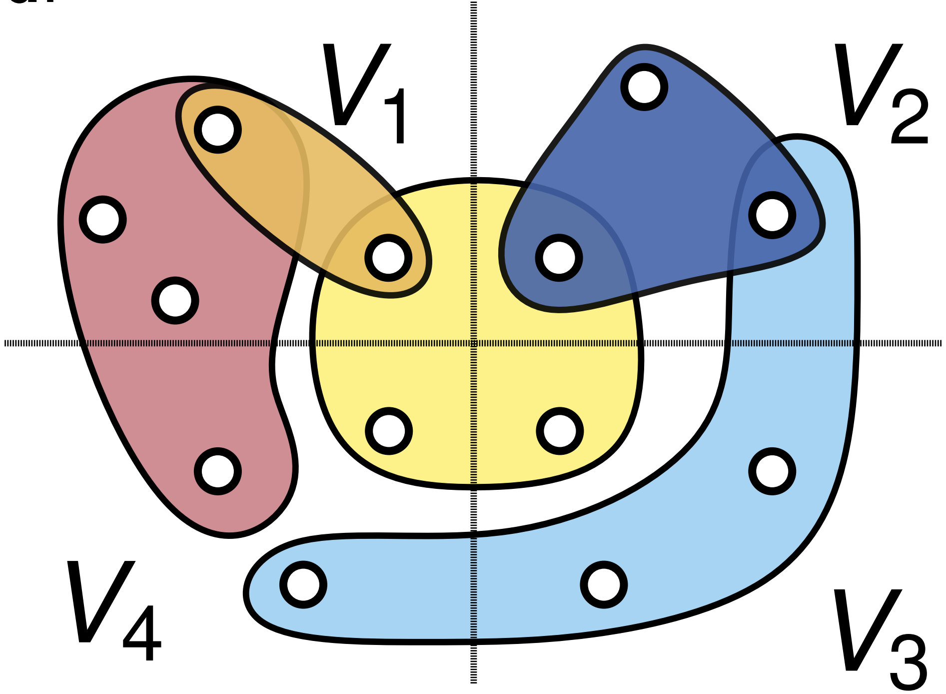
imbalance parameter

$$c(V_i) \leq (1 + \varepsilon) \left\lceil \frac{c(V)}{k} \right\rceil$$

- **connectivity** objective is **minimized**:

$$\sum_{e \in \text{cut}} (\lambda - 1)\, \omega(e)$$

connectivity:
 # blocks connected by net *e*



Hypergraph Partitioning Problem

[from SEA'17]

Partition hypergraph $H = (V, E, c, \omega)$ into $\mathbf{k}$ non-empty disjoint blocks $\Pi = \{V_1, \ldots, V_k\}$ such that:

- blocks V_i are **roughly equal-sized**:

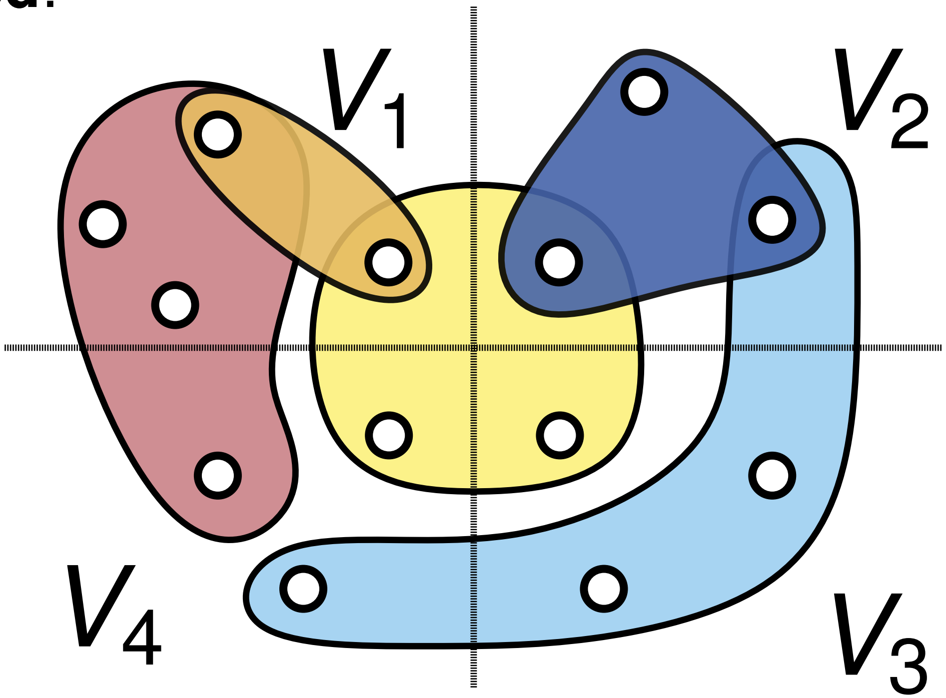
imbalance
parameter

$$c(V_i) \leq (1 + \varepsilon) \left\lceil \frac{c(V)}{k} \right\rceil$$

- **connectivity** objective is **minimized**:

$$\sum_{e \in \text{cut}} (\lambda - 1)\, \omega(e) = 6$$

connectivity:
 # blocks connected by net e

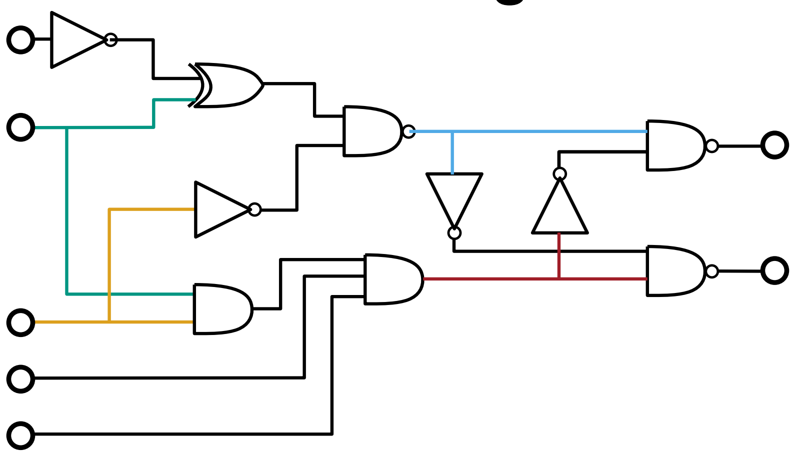


Institute of Theoretical Informatics
Algorithmics Group

Applications

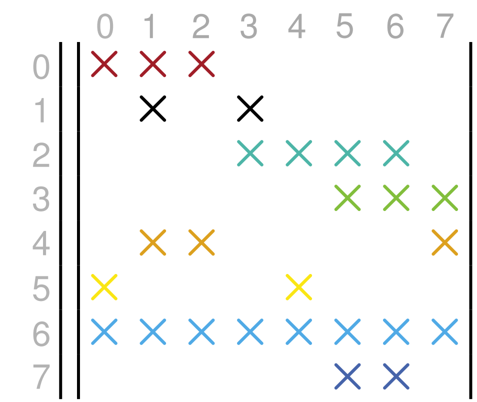
[from SEA'17]

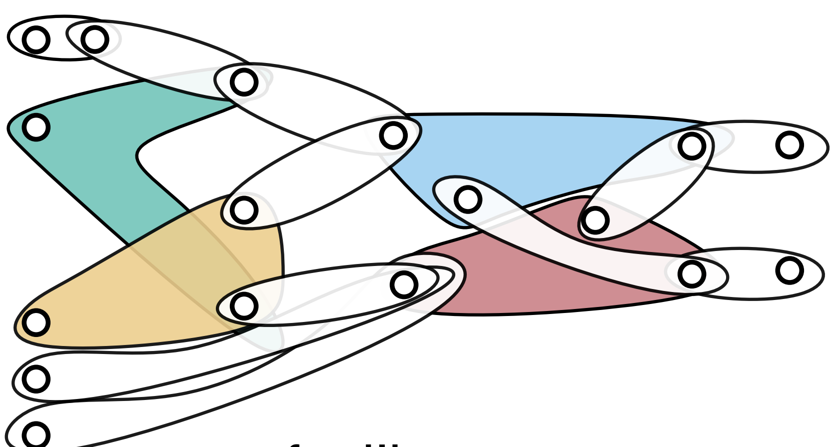
VLSI Design



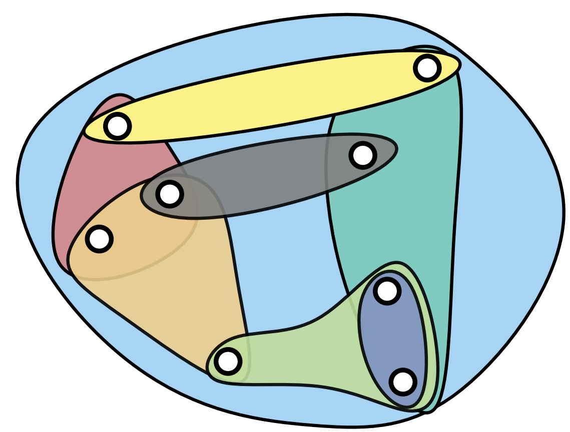
Application Domain

Scientific Computing





Hypergraph Model



facilitate floorplanning & placement

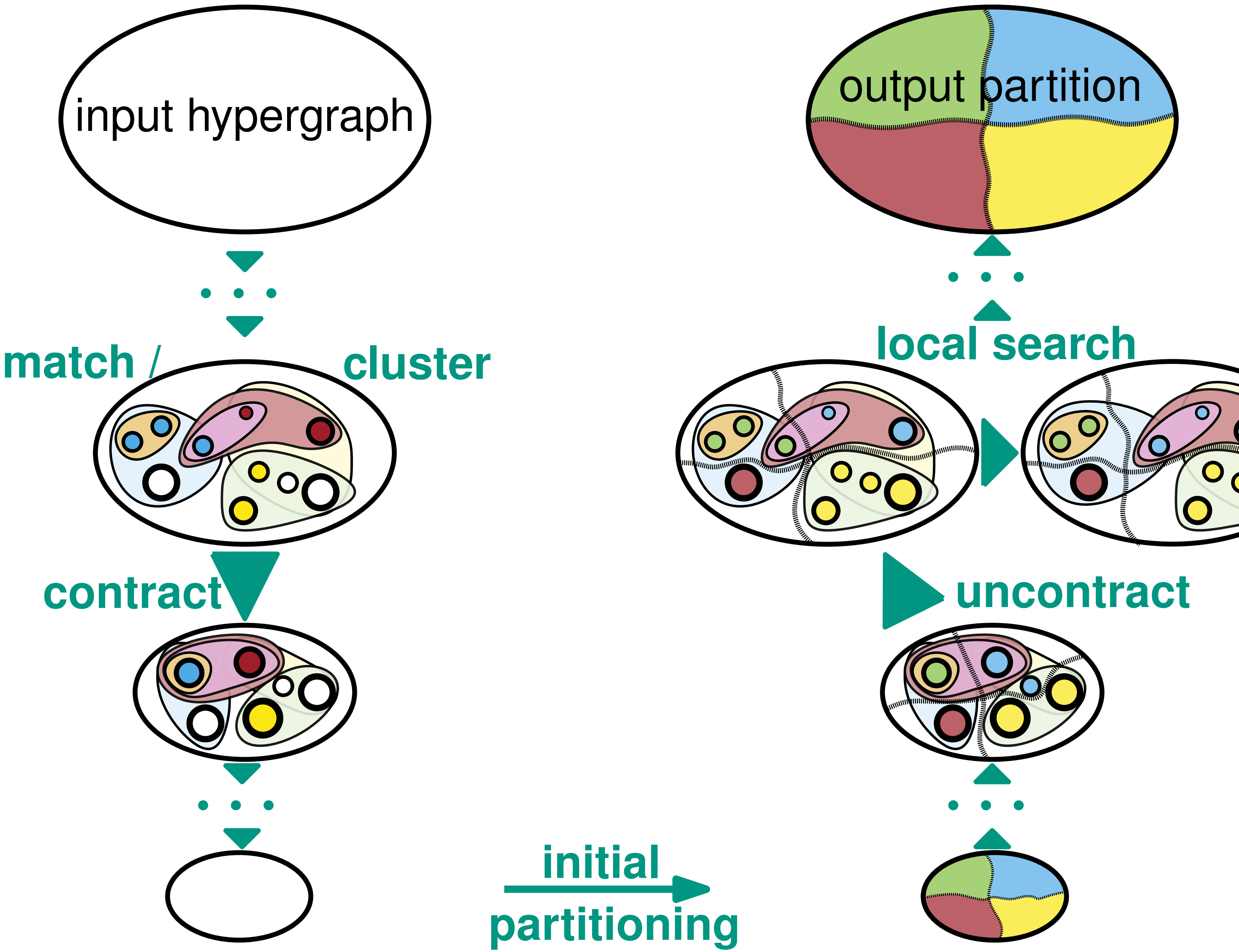
Goal

minimize communication

Institute of Theoretical Informatics
Algorithmics Group

The Multilevel Framework

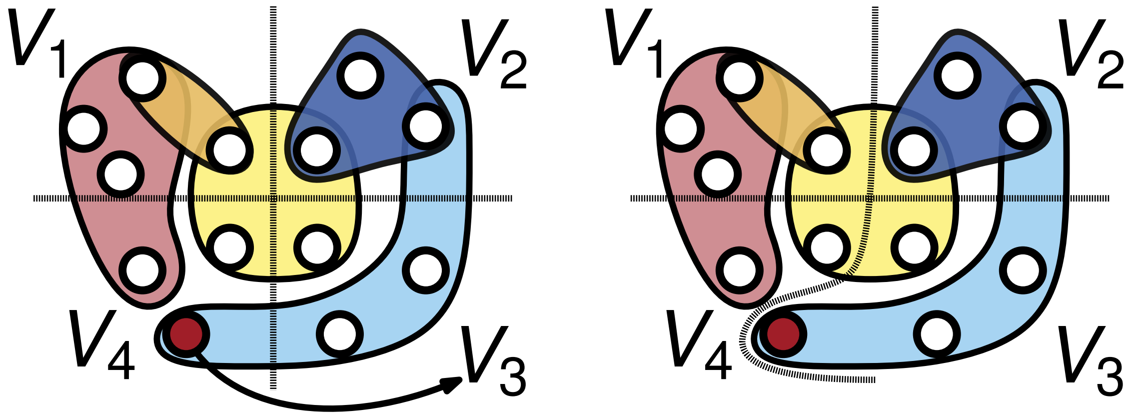
[from SEA'17]



Institute of Theoretical Informatics
Algorithmics Group

FM Algorithm

- **Move**-based heuristic that **greedily** move vertices between blocks based on **local** informations of incident nets

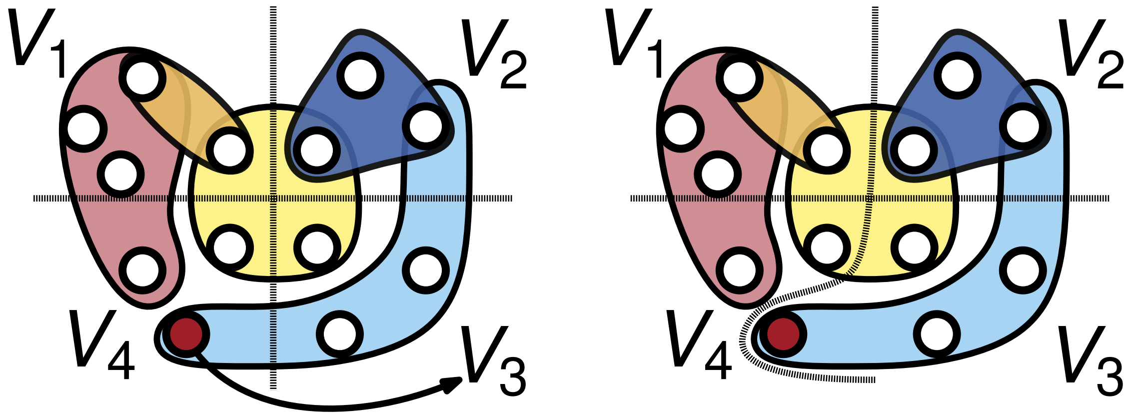


Moving ● from V_4 to V_3 reduces cut by 1

Institute of Theoretical Informatics
Algorithmics Group

FM Algorithm

- **Move**-based heuristic that **greedily** move vertices between blocks based on **local** informations of incident nets



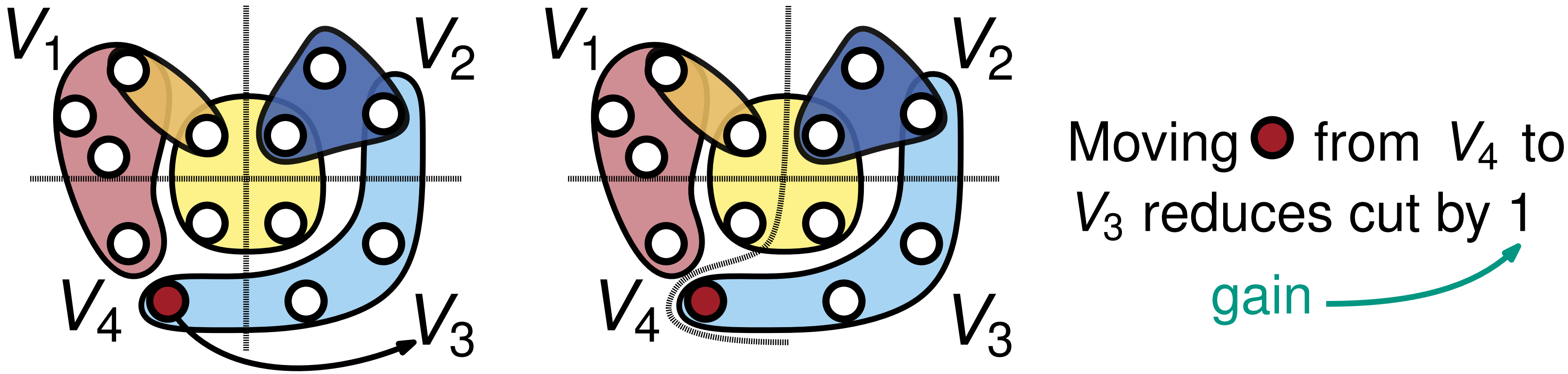
Moving  from V_4 to V_3 reduces cut by 1

gain

Institute of Theoretical Informatics
Algorithmics Group

FM Algorithm

- **Move**-based heuristic that **greedily** move vertices between blocks based on **local** informations of incident nets

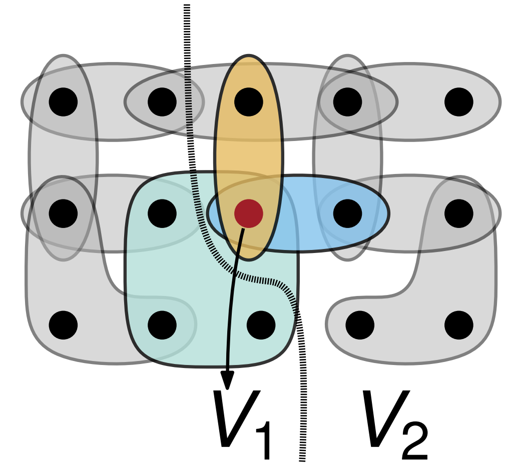


- Performs moves of vertices with **maximum gain** in each step
- All modern hypergraph partitioners implements variations of the *FM* algorithm

Institute of Theoretical Informatics
Algorithmics Group

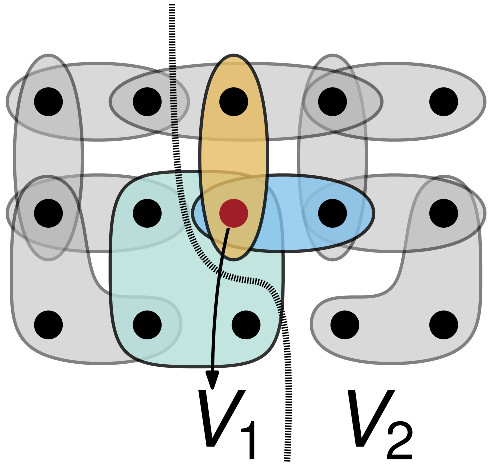
FM Algorithm - Disadvantages

- Only incorparates **local** informations about the problem structure
  - Heavily depends on *initial partition*
  - In multilevel context: Depends on quality of *coarsening*

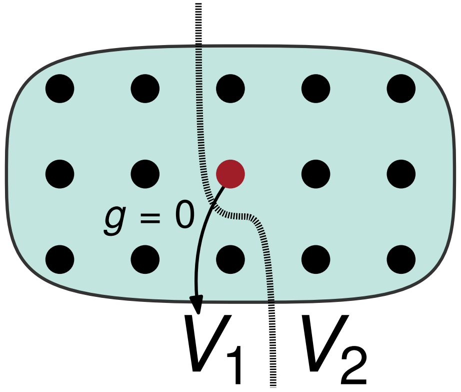


FM Algorithm - Disadvantages

- Only incorparates **local** informations about the problem structure
  - Heavily depends on *initial partition*
  - In multilevel context: Depends on quality of *coarsening*



- Large hyperedges induce **Zero-Gain** moves
  - Quality mainly depends on random decisions made within the algorithm



Institute of Theoretical Informatics
Algorithmics Group

Flow-based Approaches

Institute of Theoretical Informatics
Algorithmics Group

Flows

Given a graph $G = (V, E, u)$ and two nodes $s, t \in V$

- $u : E \to \mathbb{R}_+$ is the **capacity** function
- s and t are called **source** and **sink**

Institute of Theoretical Informatics
Algorithmics Group

Flows

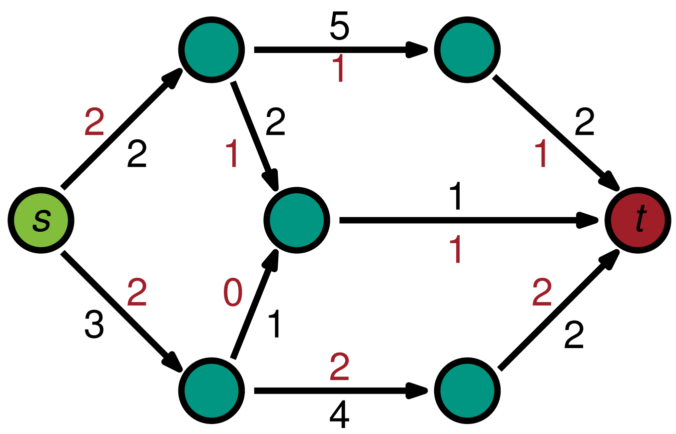
Given a graph $G = (V, E, u)$ and two nodes $s, t \in V$

- $u : E \to \mathbb{R}_+$ is the **capacity** function
- s and t are called **source** and **sink**

A valid **flow** is a function $f : E \to \mathbb{R}_+$ with the constraints:

- $\forall (v, w) \in E : f(v, w) \leq u(v, w)$
- $\forall v \in V \setminus \{s, t\} : \sum_{(w,v) \in E} f(w, v) = \sum_{(v,w) \in E} f(v, w)$

The value of the flow is $|f| = \sum_{(s,v) \in E} f(s, v)$

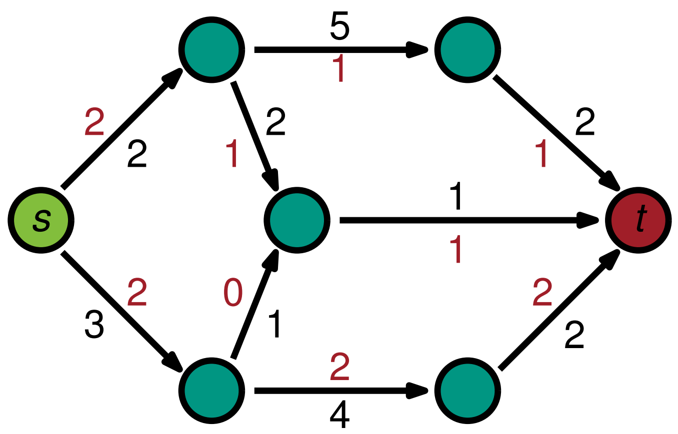


Institute of Theoretical Informatics
Algorithmics Group

Flows

The **residual capacity** $r_f : V \times V \to \mathbb{R}_+$ is defined as follows:

- $\forall (v, w) \in E : r_f(v, w) = u(v, w) - f(v, w)$
- $\forall (v, w) \in E :$ If $f(v, w) > 0$ and $u(w, v) = 0$, then $r_f(w, v) = f(v, w)$



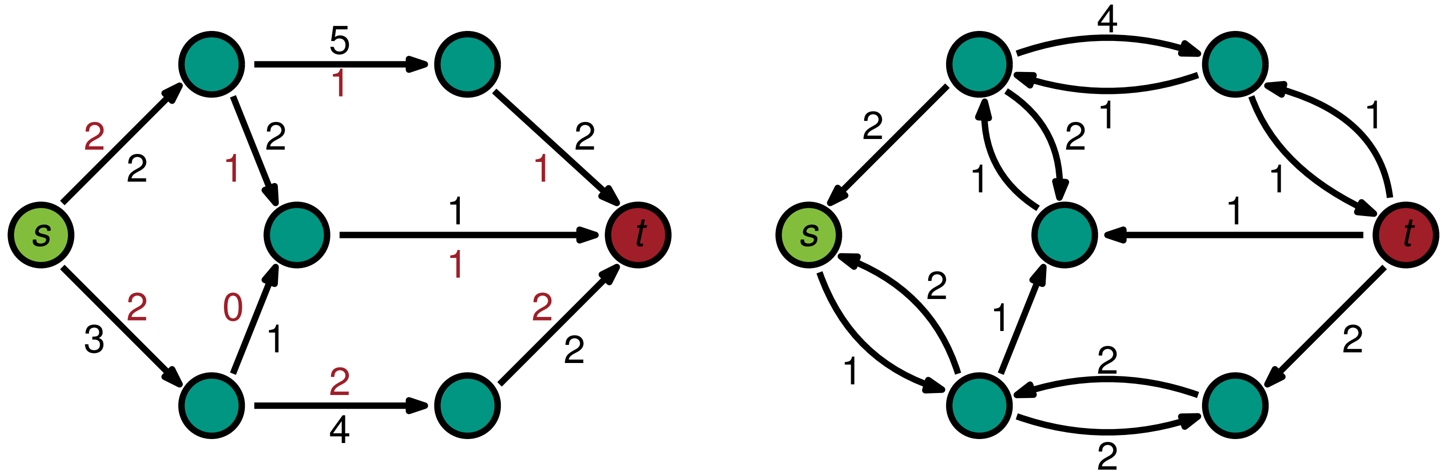
Institute of Theoretical Informatics
Algorithmics Group

Flows

The **residual capacity** $r_f : V \times V \to \mathbb{R}_+$ is defined as follows:

- $\forall (v, w) \in E : r_f(v, w) = u(v, w) - f(v, w)$
- $\forall (v, w) \in E :$ If $f(v, w) > 0$ and $u(w, v) = 0$, then $r_f(w, v) = f(v, w)$

The **residual graph** $G_f = (V, E_f, r_f)$ contains all edges $(v, w) \in V \times V$ with $r_f(v, w) > 0$



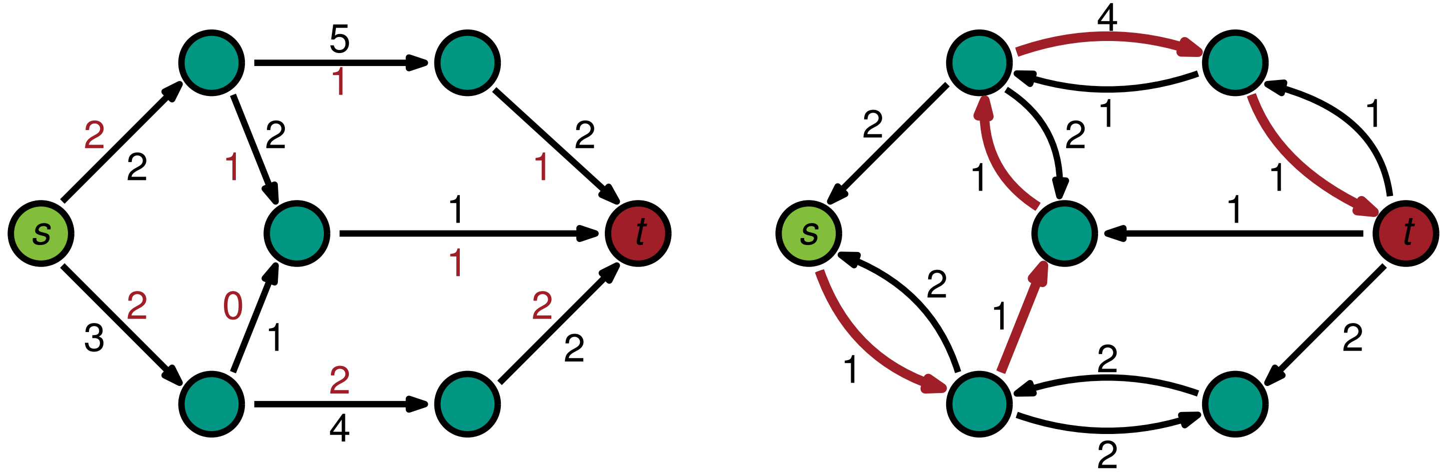
Flows

The **residual capacity** $r_f : V \times V \rightarrow \mathbb{R}_+$ is defined as follows:

- $\forall (v, w) \in E : r_f(v, w) = u(v, w) - f(v, w)$
- $\forall (v, w) \in E$: If $f(v, w) > 0$ and $u(w, v) = 0$, then $r_f(w, v) = f(v, w)$

The **residual graph** $G_f = (V, E_f, r_f)$ contains all edges $(v, w) \in V \times V$ with $r_f(v, w) > 0$

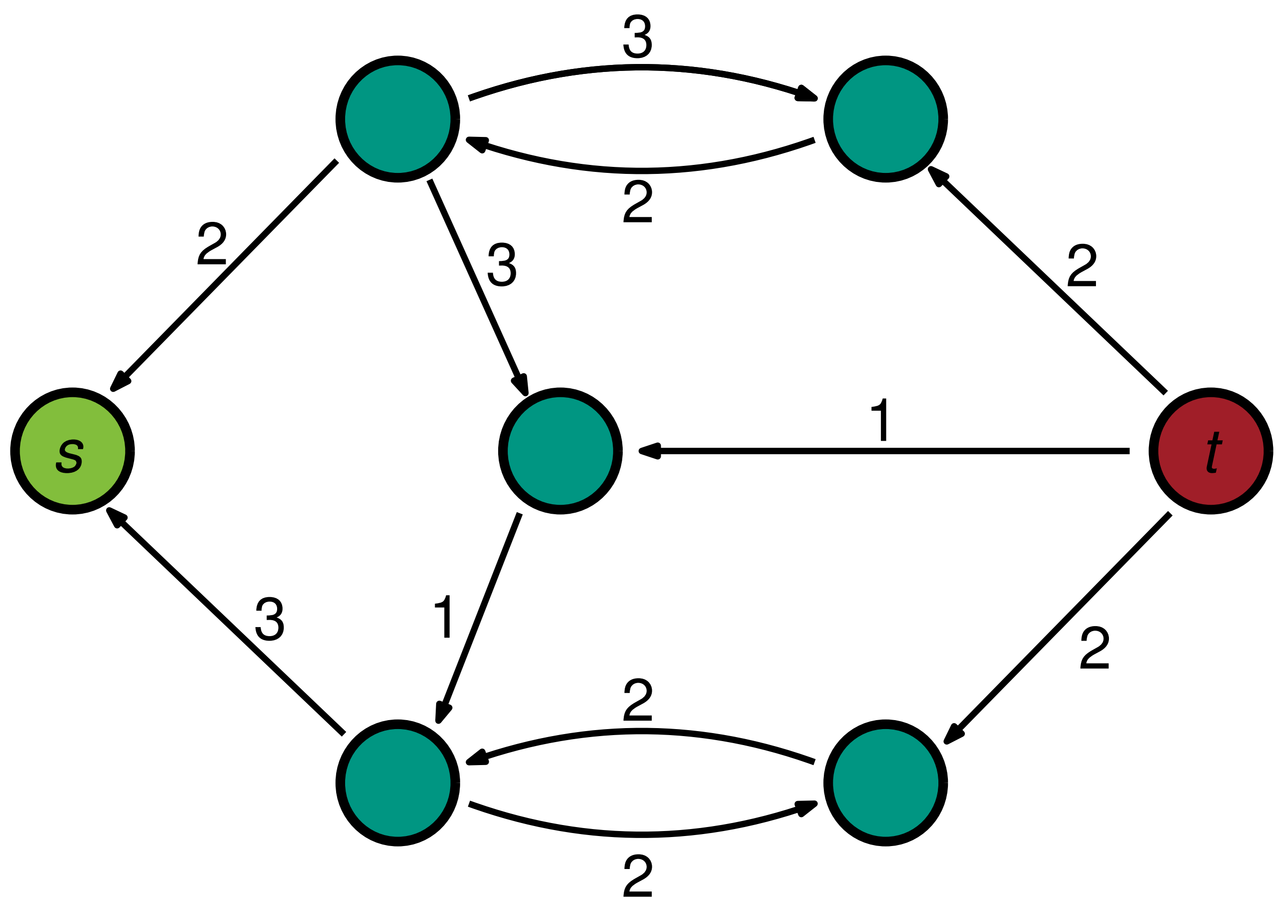
- An **augmenting path** is a path in G_f from s to t
- f is a **maximum flow**, if there is no augmenting path from s to t in G_f



Institute of Theoretical Informatics
Algorithmics Group

Minimum (s, t)-Bipartition

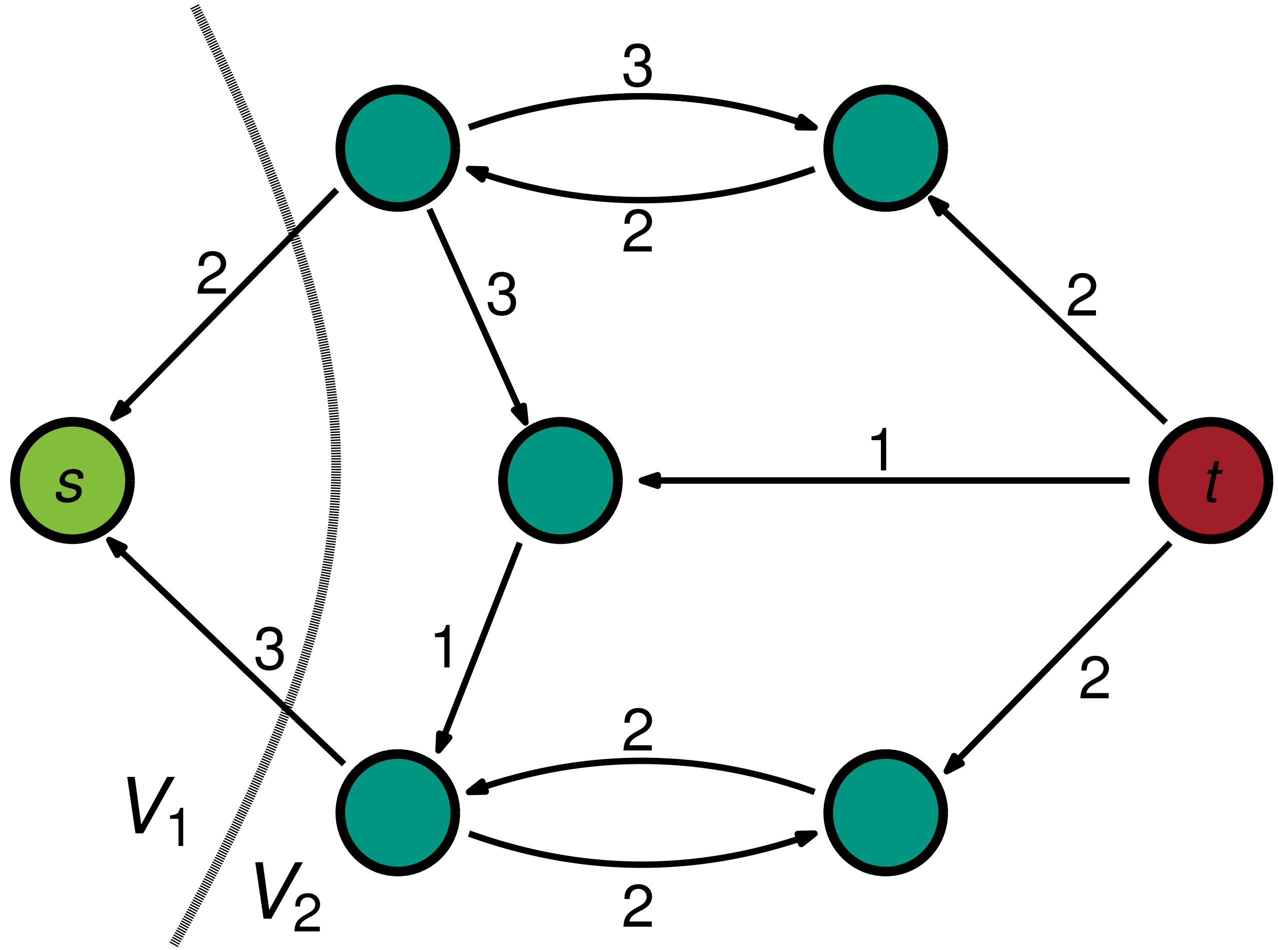
All nodes *reachable* from s are part of V_1 and $V_2 = V \setminus V_1$



Residual Graph G_f of a maximum flow f

Minimum (s, t)-Bipartition

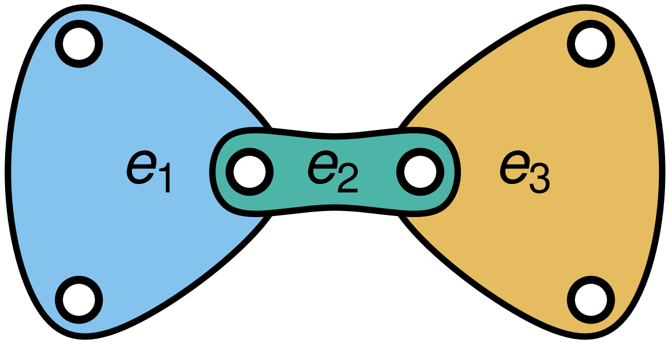
All nodes *reachable* from s are part of V_1 and $V_2 = V \setminus V_1$



Residual Graph G_f of a maximum flow f

Hypergraph Flow Network

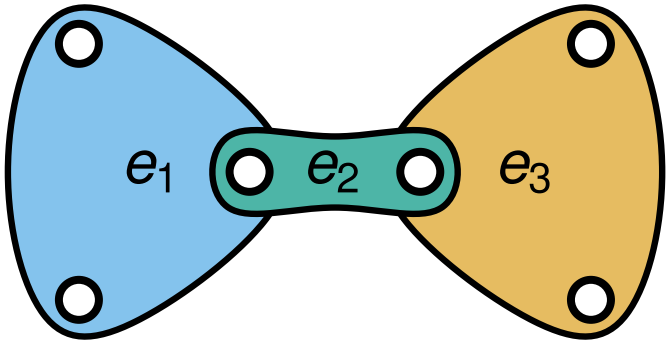
Hypergraph H



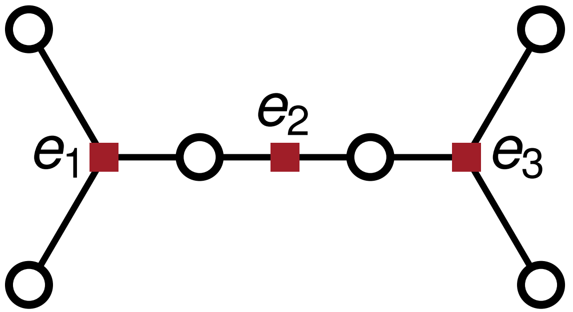
Institute of Theoretical Informatics
Algorithmics Group

Hypergraph Flow Network

Hypergraph H



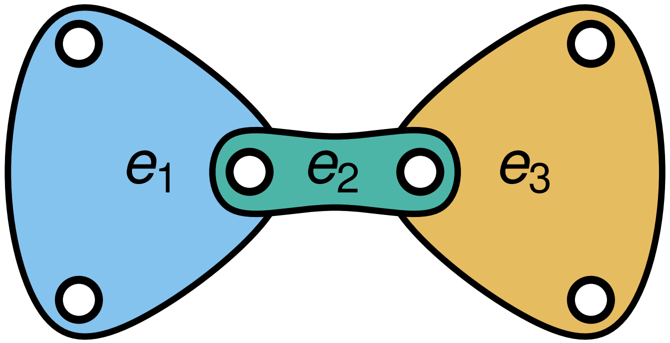
Bipartite Graph $G_*(H)$



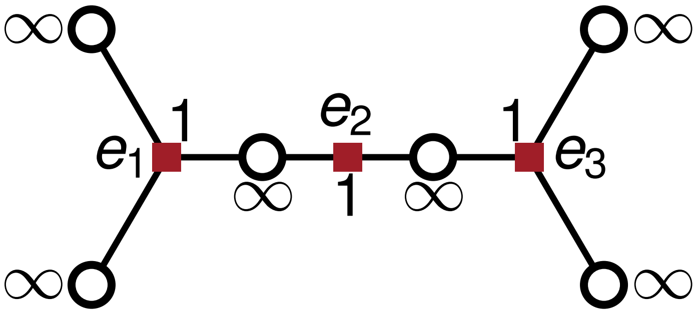
Institute of Theoretical Informatics
Algorithmics Group

Hypergraph Flow Network

Hypergraph H



Bipartite Graph $G_*(H)$

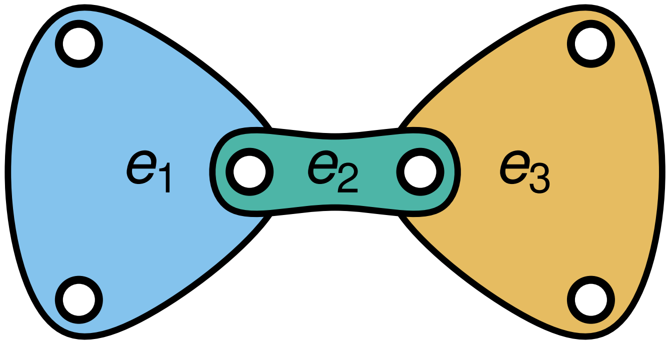


Vertex Separator Problem

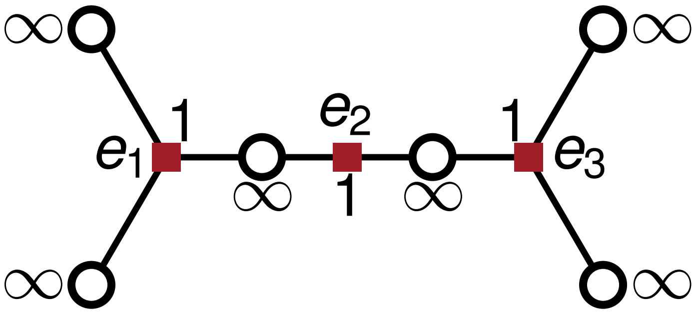
Institute of Theoretical Informatics
Algorithmics Group

Hypergraph Flow Network

Hypergraph H

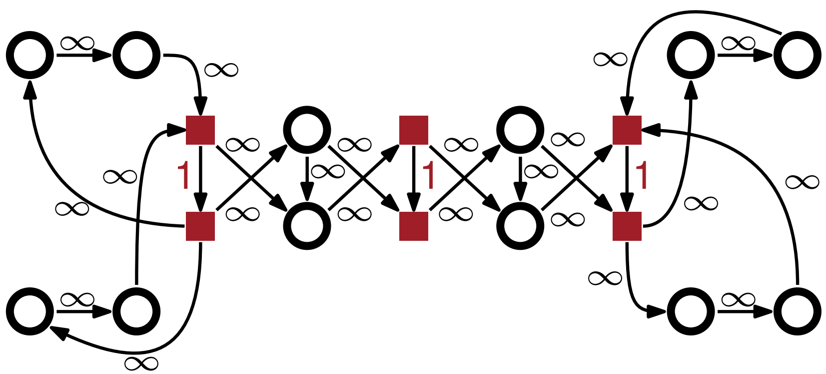


Bipartite Graph $G_*(H)$



Vertex Separator Problem

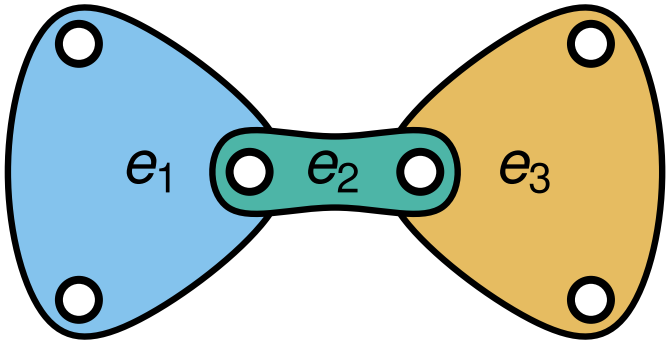
Vertex Separator Transformation



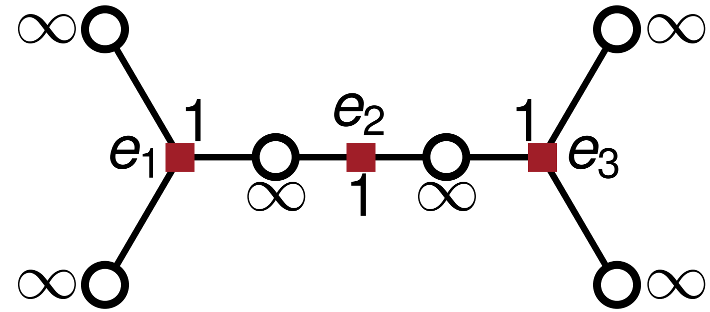
Institute of Theoretical Informatics
Algorithmics Group

Hypergraph Flow Network

Hypergraph H

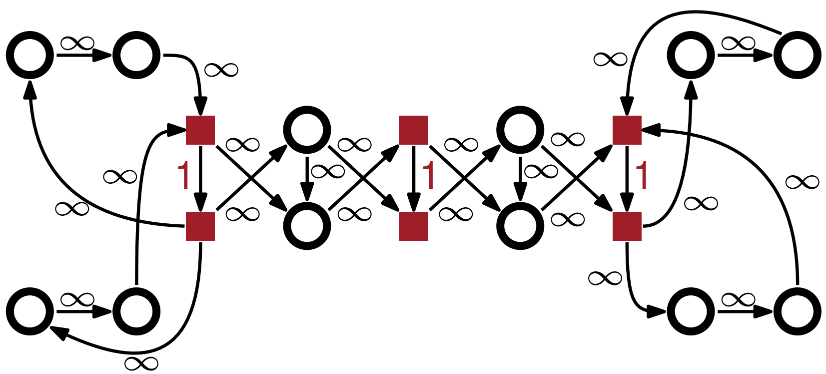


Bipartite Graph $G_*(H)$

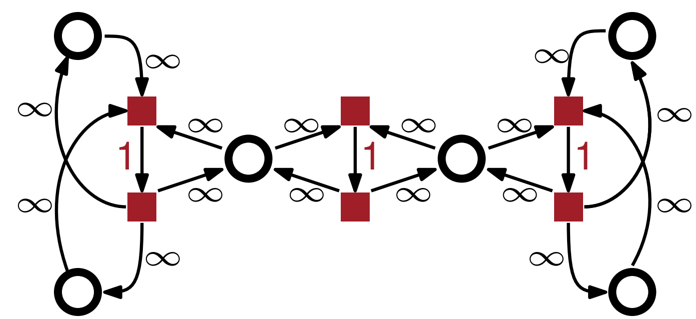


Vertex Separator Problem

Vertex Separator Transformation

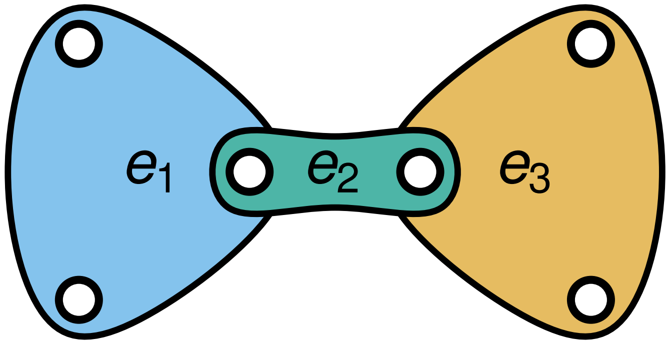


Lawler Network

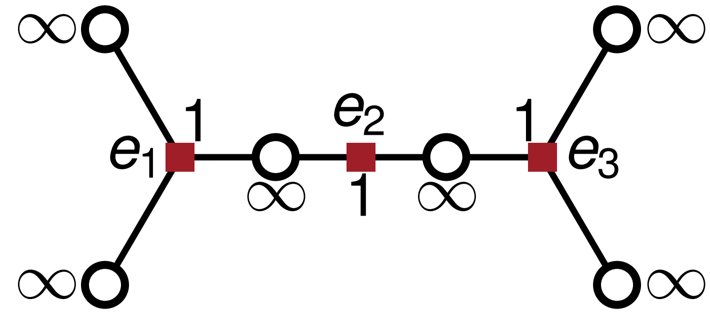


Hypergraph Flow Network

Hypergraph H

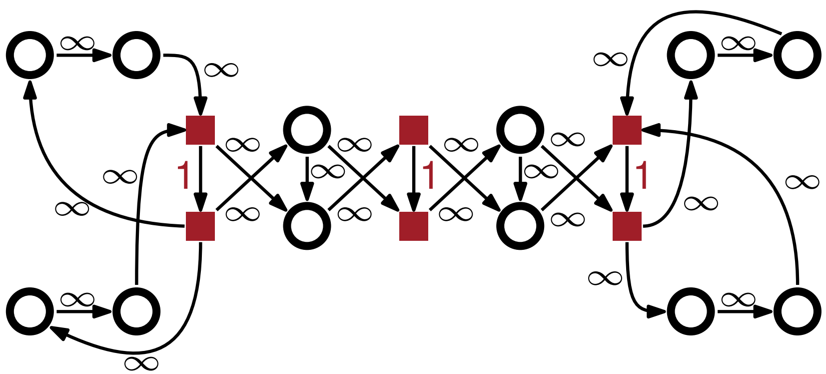


Bipartite Graph $G_*(H)$

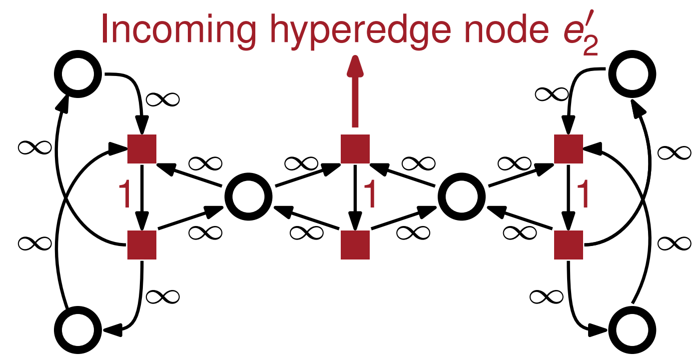


Vertex Separator Problem

Vertex Separator Transformation

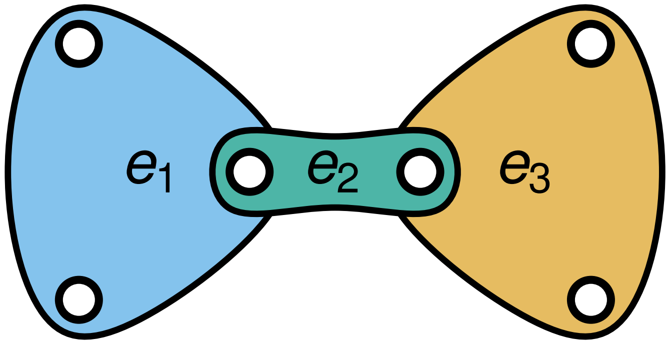


Lawler Network

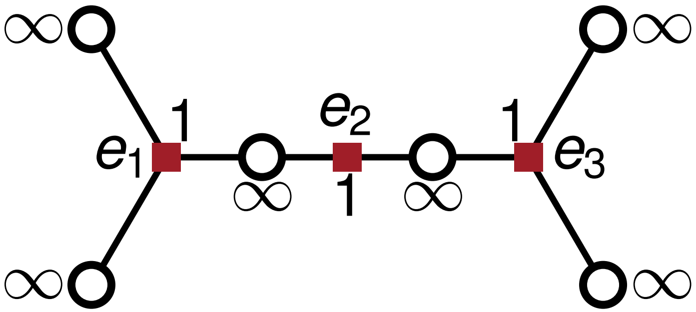


Hypergraph Flow Network

Hypergraph H

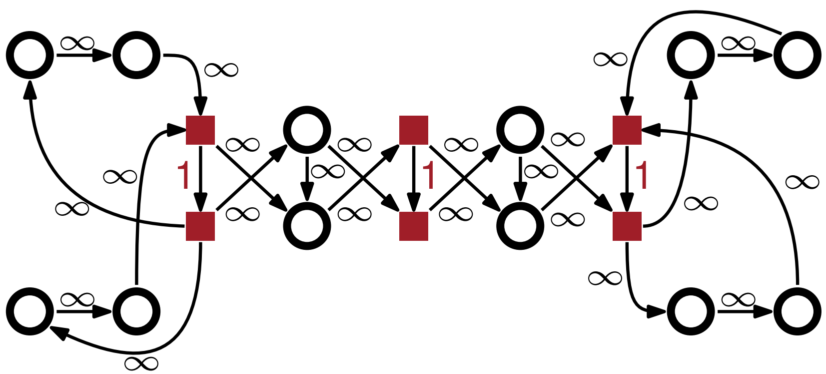


Bipartite Graph $G_*(H)$



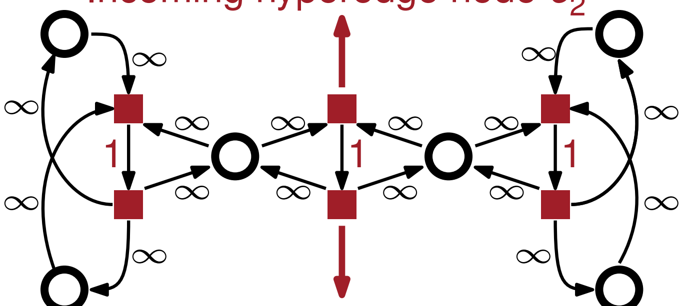
Vertex Separator Problem

Vertex Separator Transformation



Lawler Network

Incoming hyperedge node e_2'

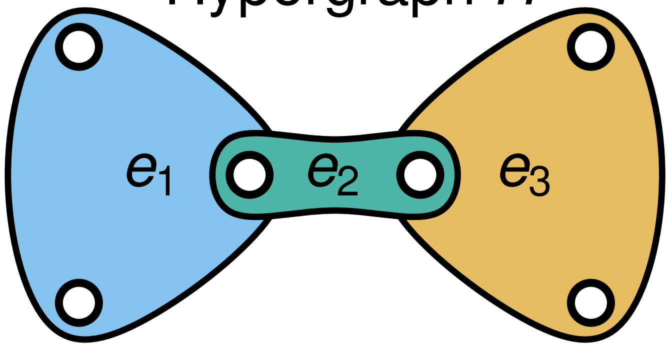


Outgoing hyperedge node e_2''

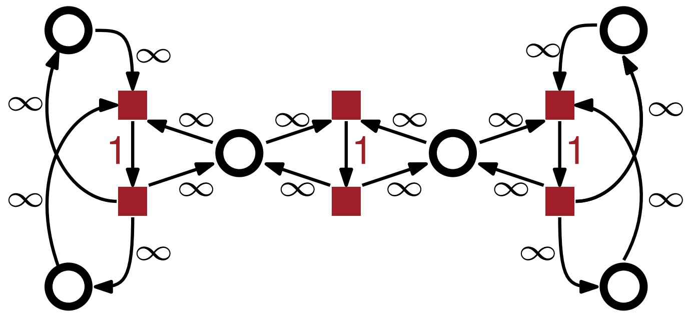
Institute of Theoretical Informatics
Algorithmics Group

Hypergraph Flow Network - Graph Edges

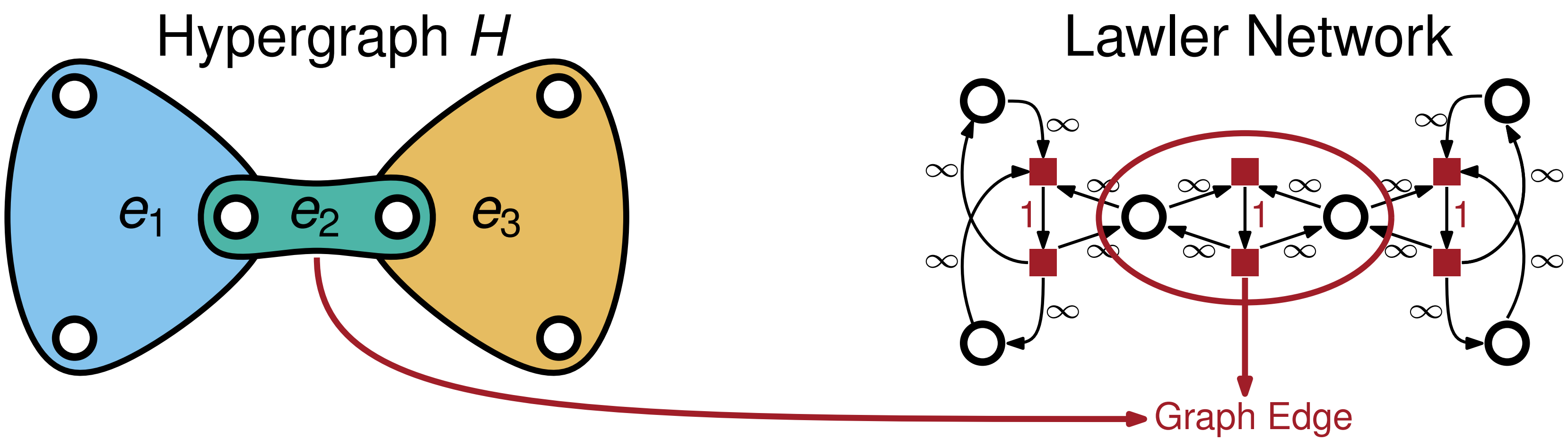
Hypergraph H



Lawler Network

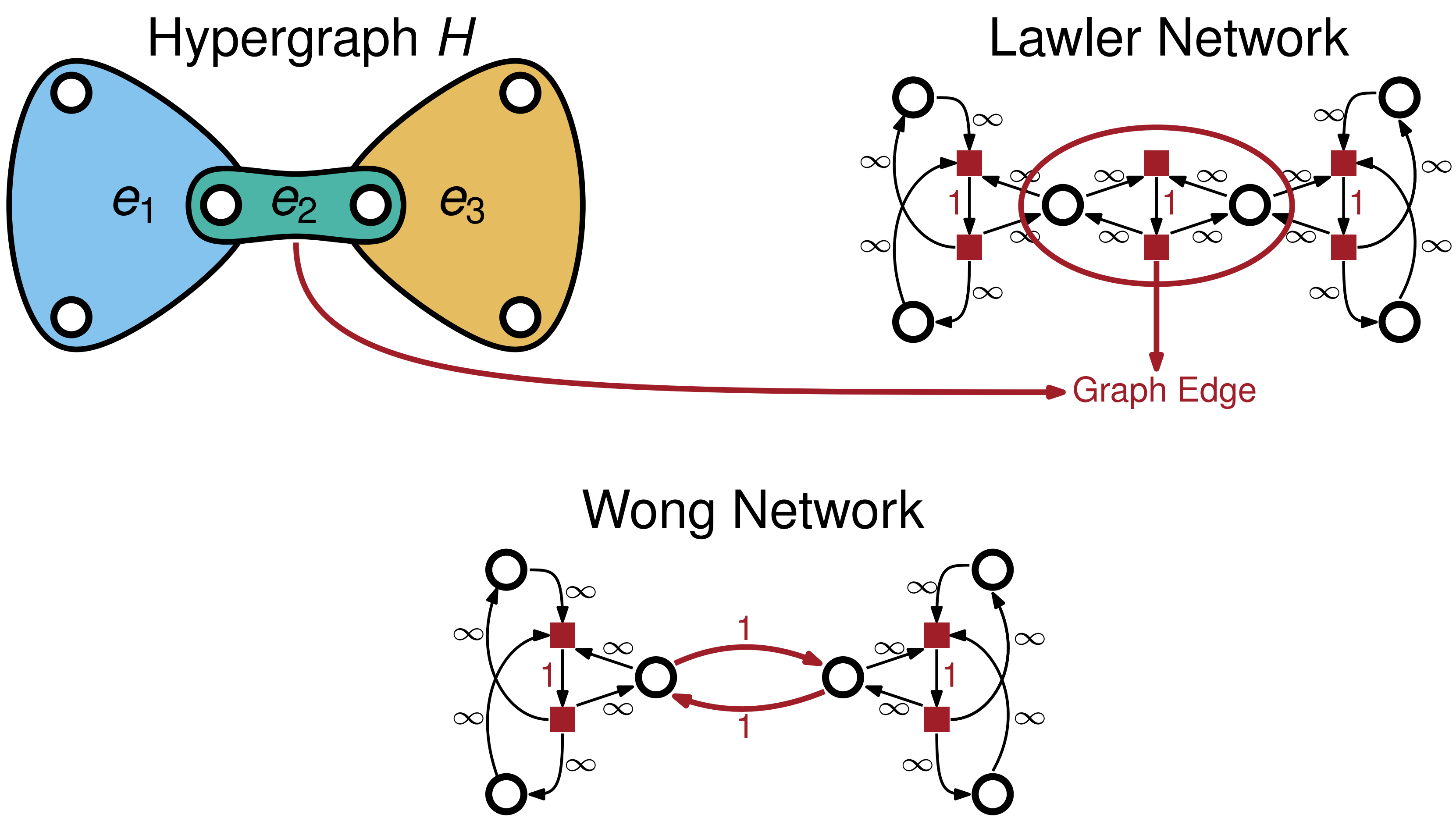


Hypergraph Flow Network - Graph Edges



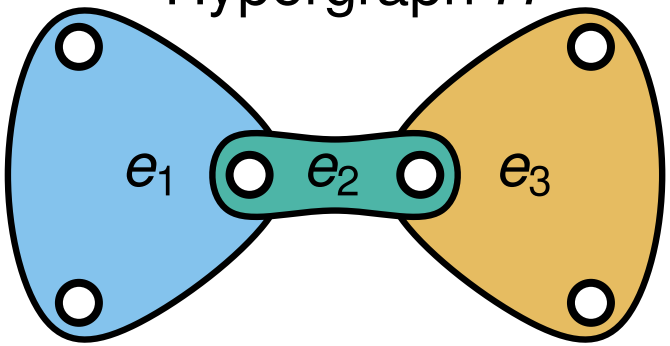
Institute of Theoretical Informatics
Algorithmics Group

Hypergraph Flow Network - Graph Edges

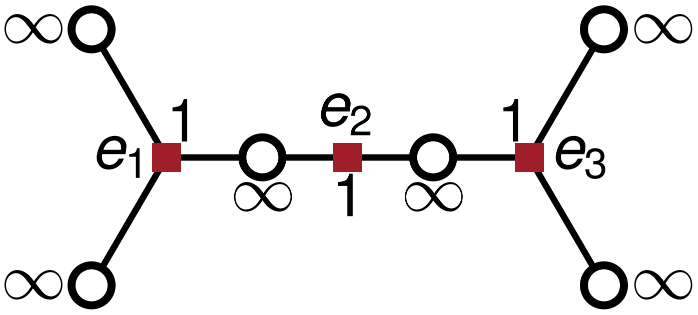


Hypergraph Flow Network - Low Degree Vertices

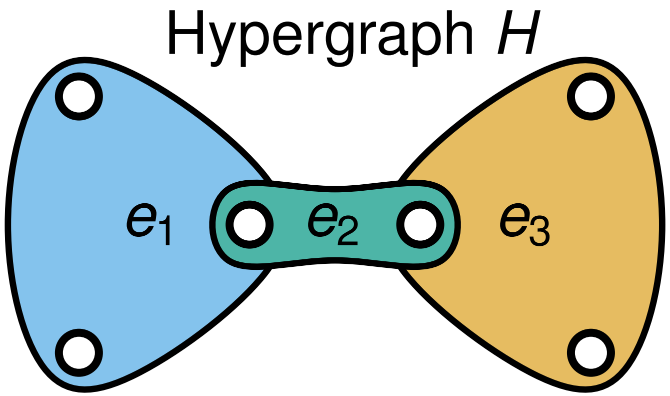
Hypergraph H

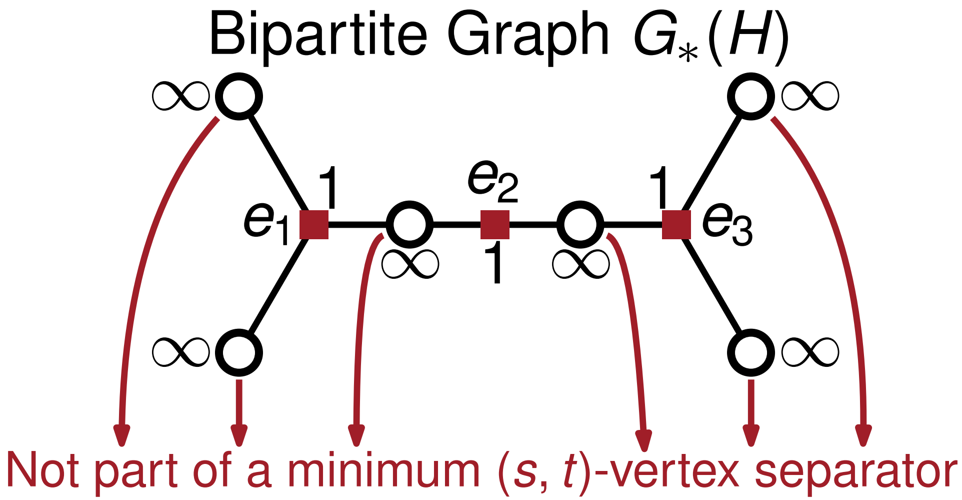


Bipartite Graph $G_*(H)$

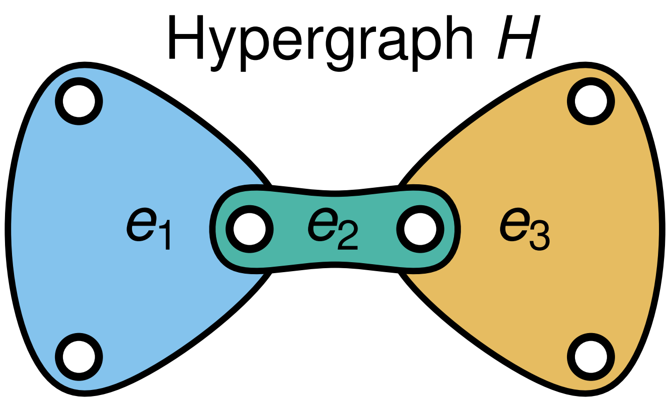


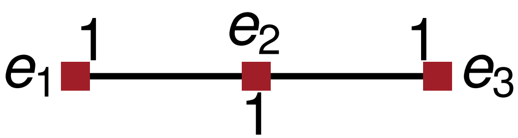
Hypergraph Flow Network - Low Degree Vertices





Hypergraph Flow Network - Low Degree Vertices

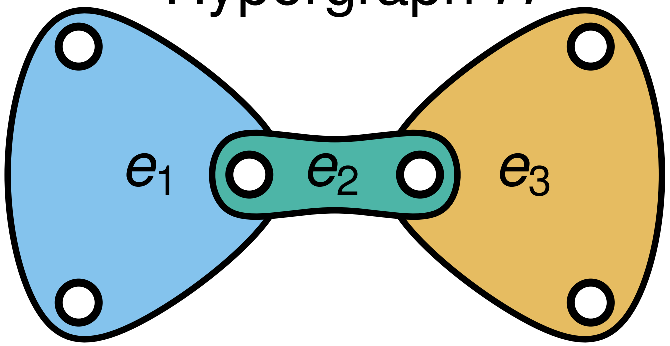


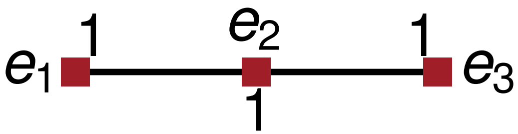


Remove all vertices by adding a clique

Institute of Theoretical Informatics
Algorithmics Group

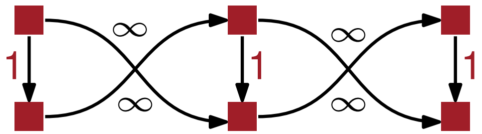
Hypergraph Flow Network - Low Degree Vertices





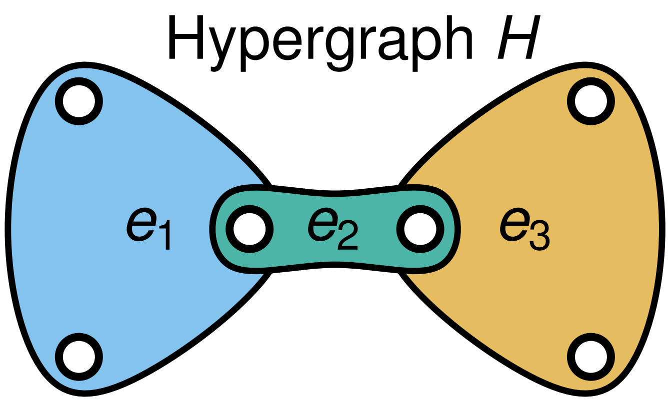
Remove all vertices by adding a clique

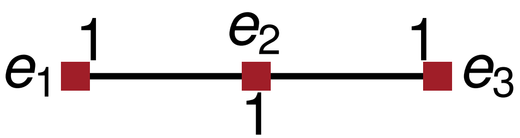
Our Network



Institute of Theoretical Informatics
Algorithmics Group

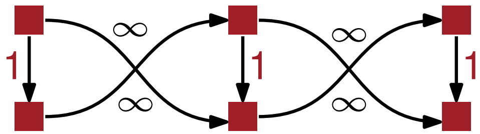
Hypergraph Flow Network - Low Degree Vertices





Remove all vertices by adding a clique

Our Network



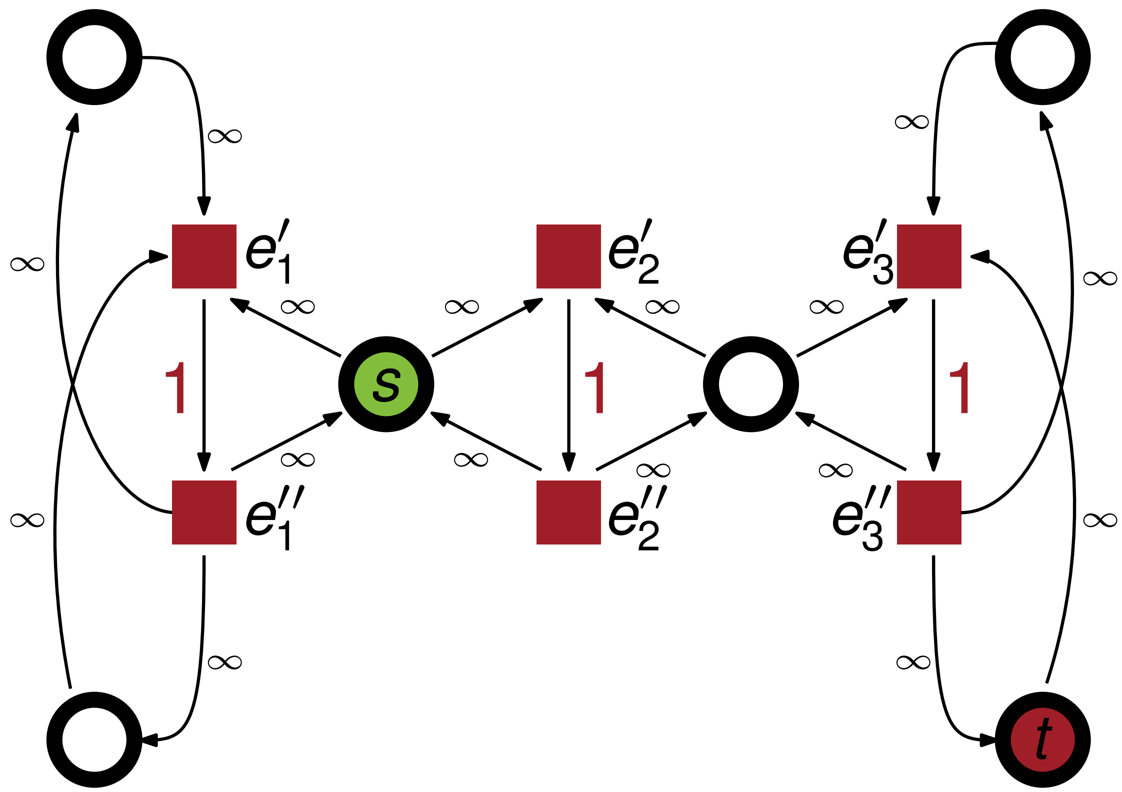
A hypernode v induces ...

- ... $2d(v)$ edges in the Lawler Network
- ... $d(v)(d(v) - 1)$ edges in our network

If $d(v) \leq 3$, then $d(v)(d(v) - 1) \leq 2d(v)$

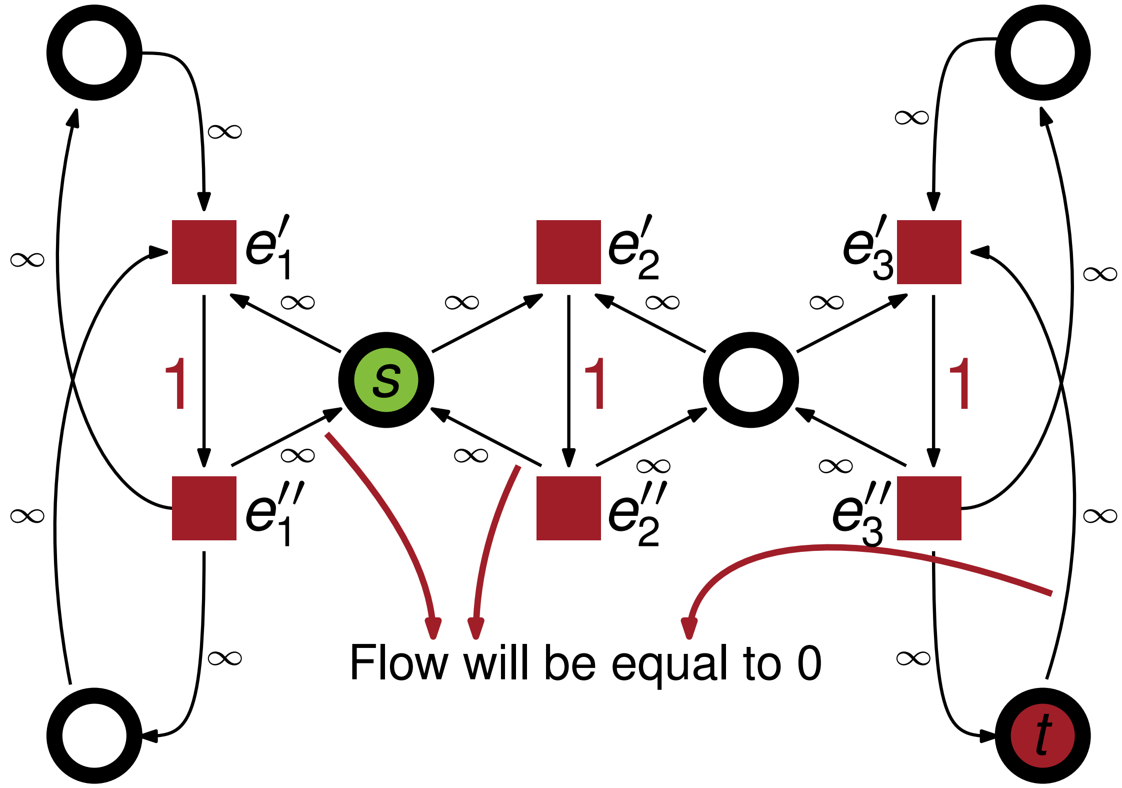
Institute of Theoretical Informatics
Algorithmics Group

Removing Source and Sink Vertices



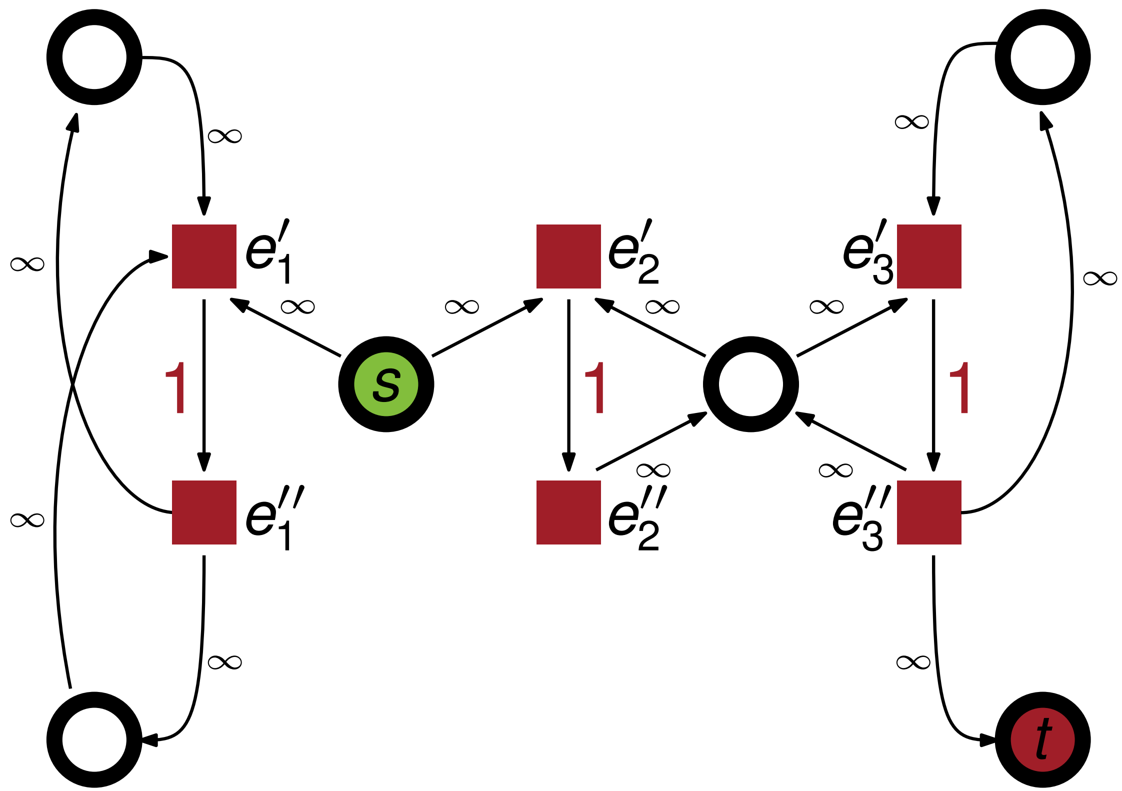
Lawler Network

Removing Source and Sink Vertices



Lawler Network

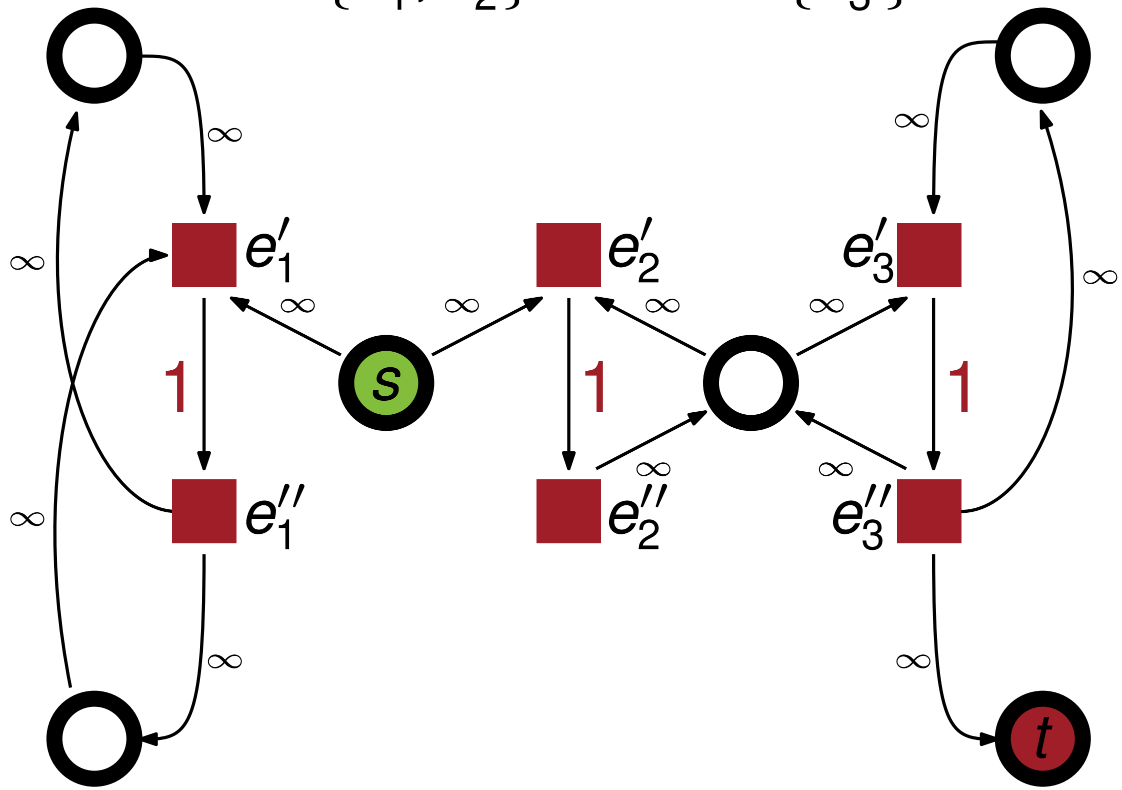
Removing Source and Sink Vertices



Lawler Network

Removing Source and Sink Vertices

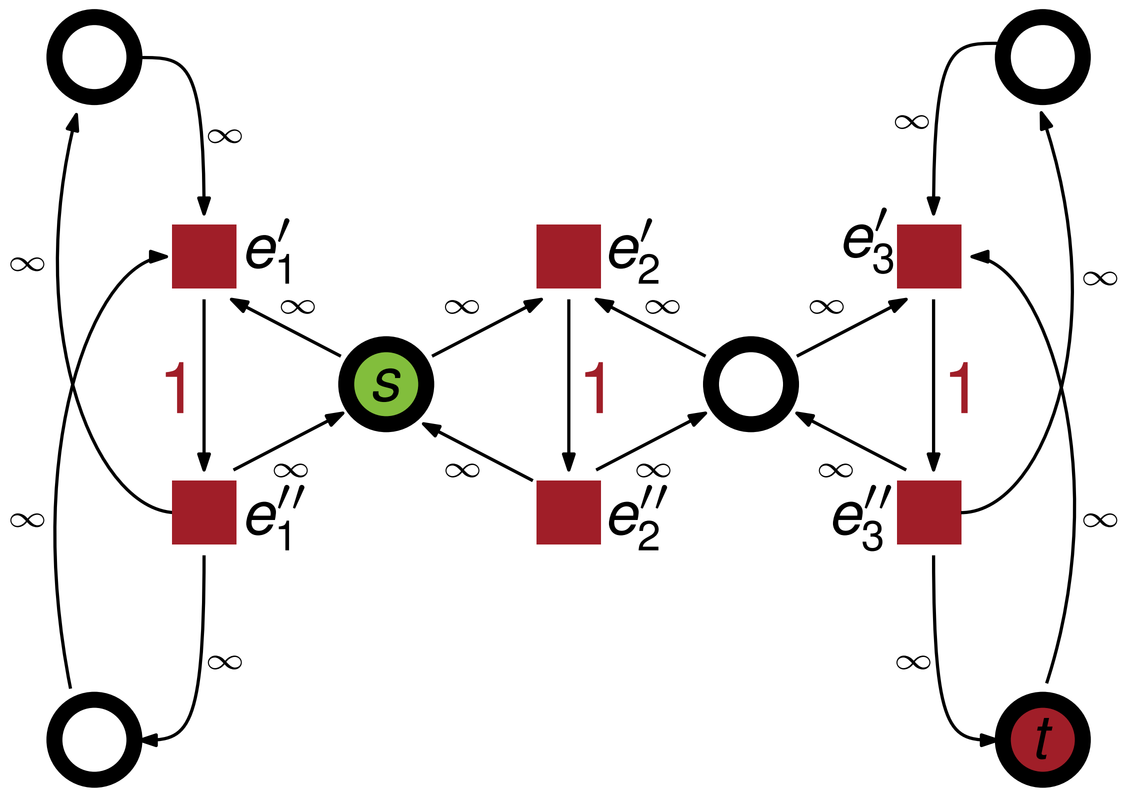
Corresponds to *Multi-Source Multi-Sink* problem with
$S = \{e'_1, e'_2\}$ and $T = \{e''_3\}$



Lawler Network

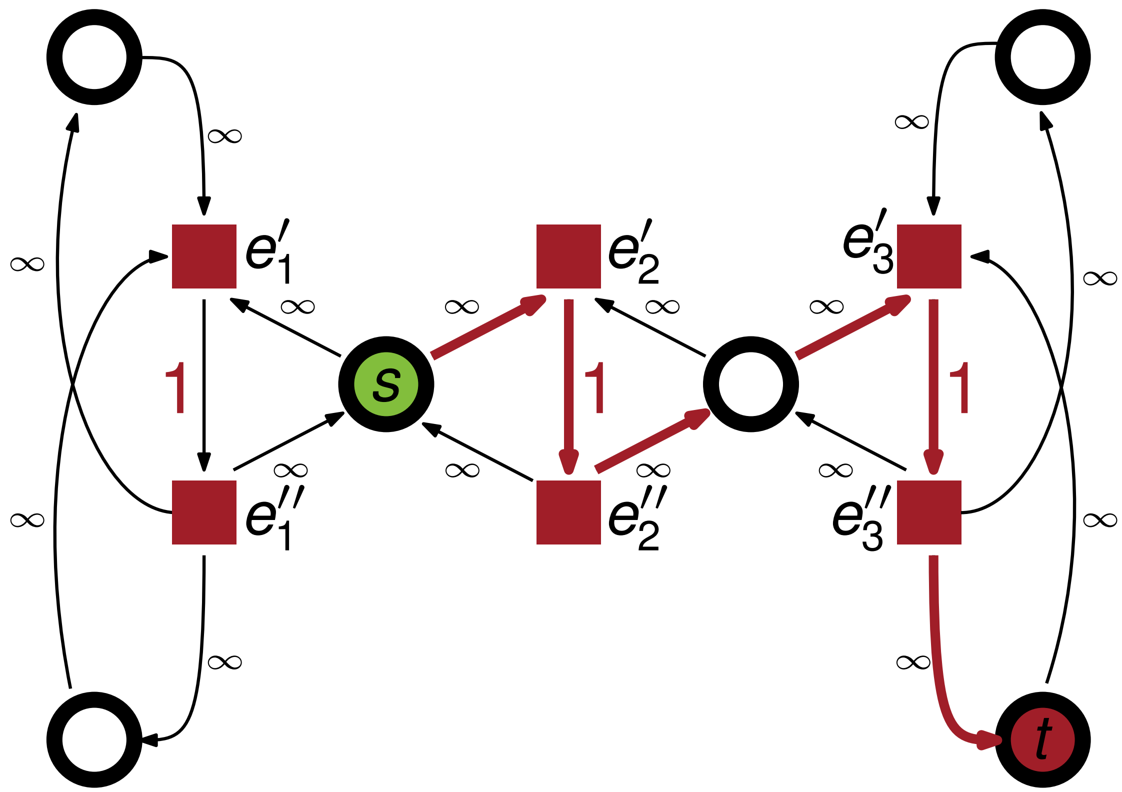
Institute of Theoretical Informatics
Algorithmics Group

Reconstruction of Minimum (s, t)-Bipartition



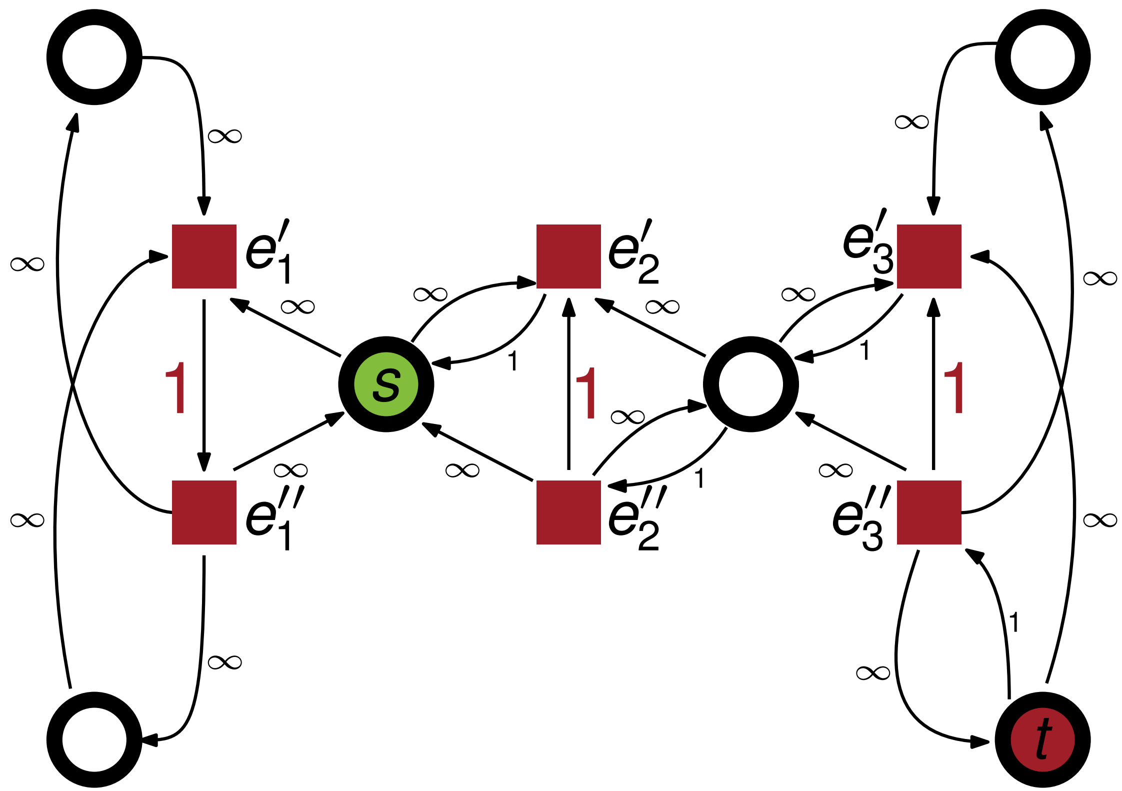
Institute of Theoretical Informatics
Algorithmics Group

Reconstruction of Minimum (s, t)-Bipartition

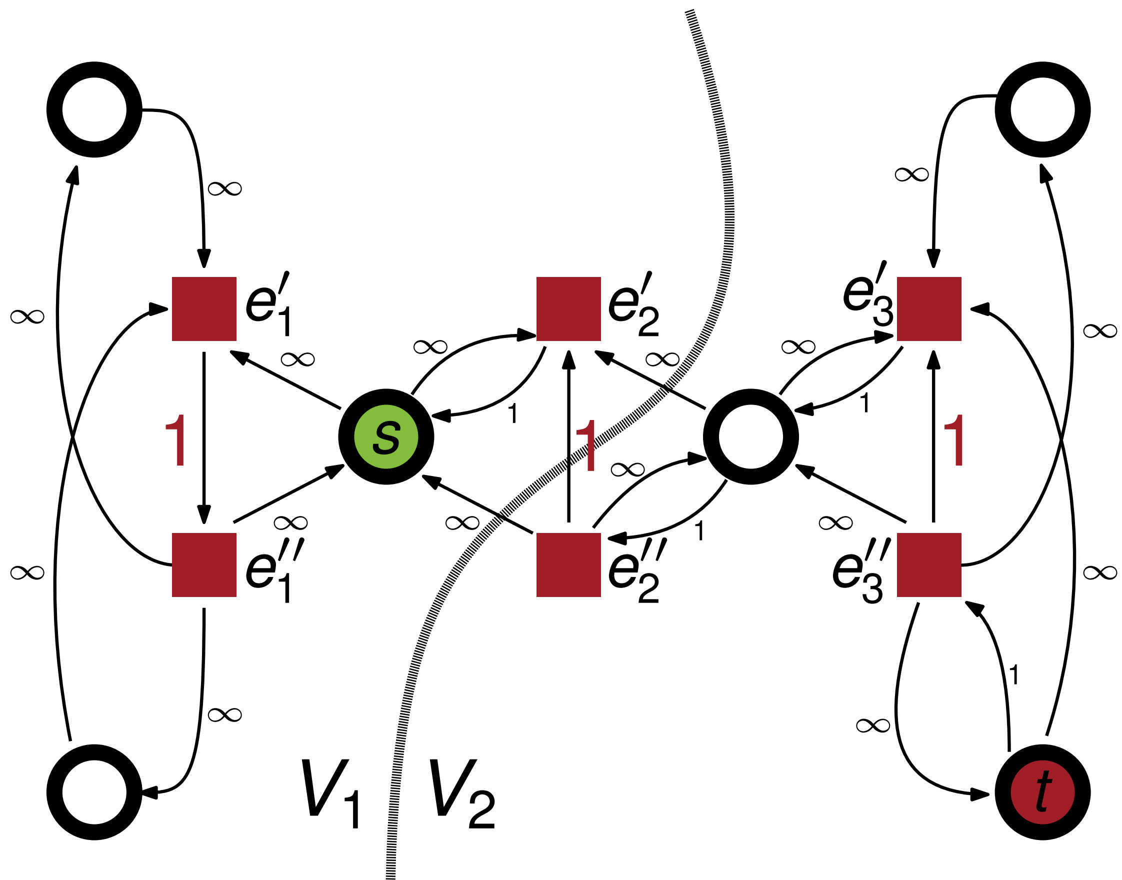


Institute of Theoretical Informatics
Algorithmics Group

Reconstruction of Minimum (s, t)-Bipartition



Reconstruction of Minimum (s, t)-Bipartition

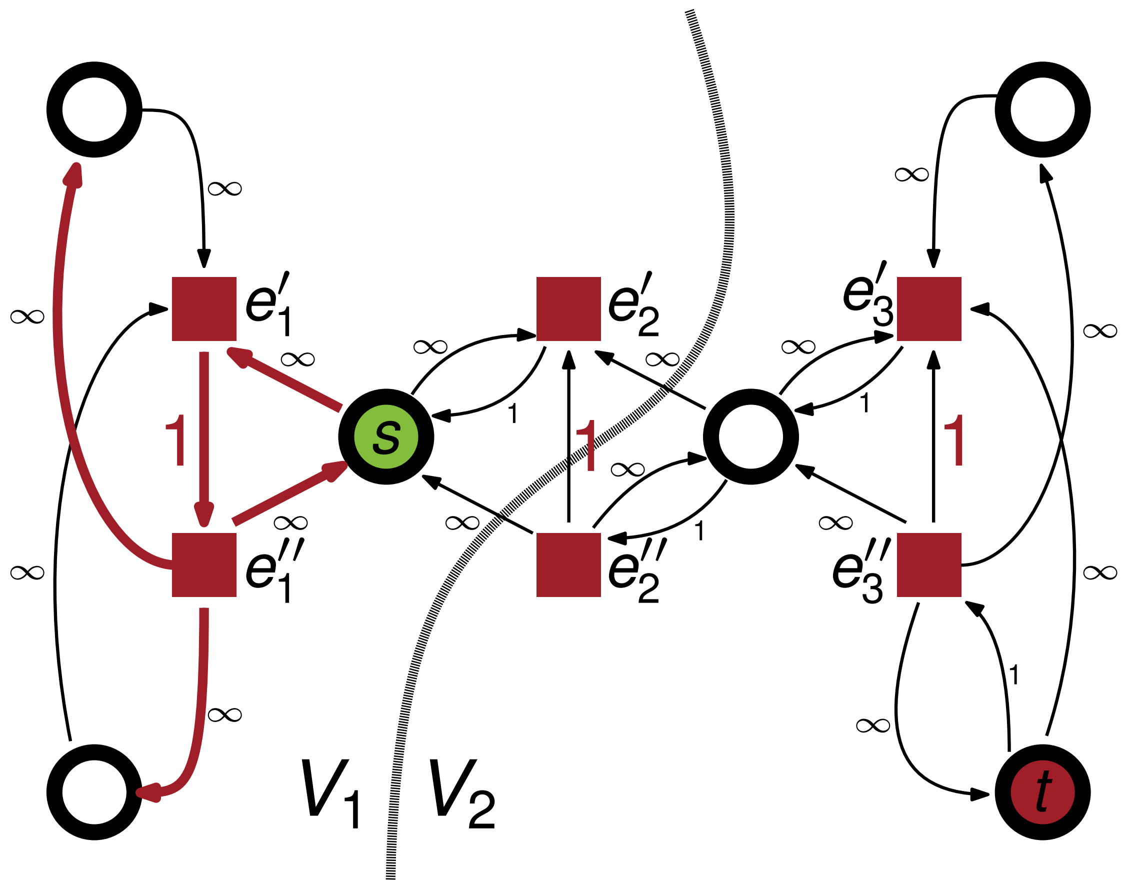


All nodes *reachable* from s are part of V_1 and $V_2 = V \setminus V_1$

Institute of Theoretical Informatics
Algorithmics Group

Reconstruction of Minimum (s, t)-Bipartition

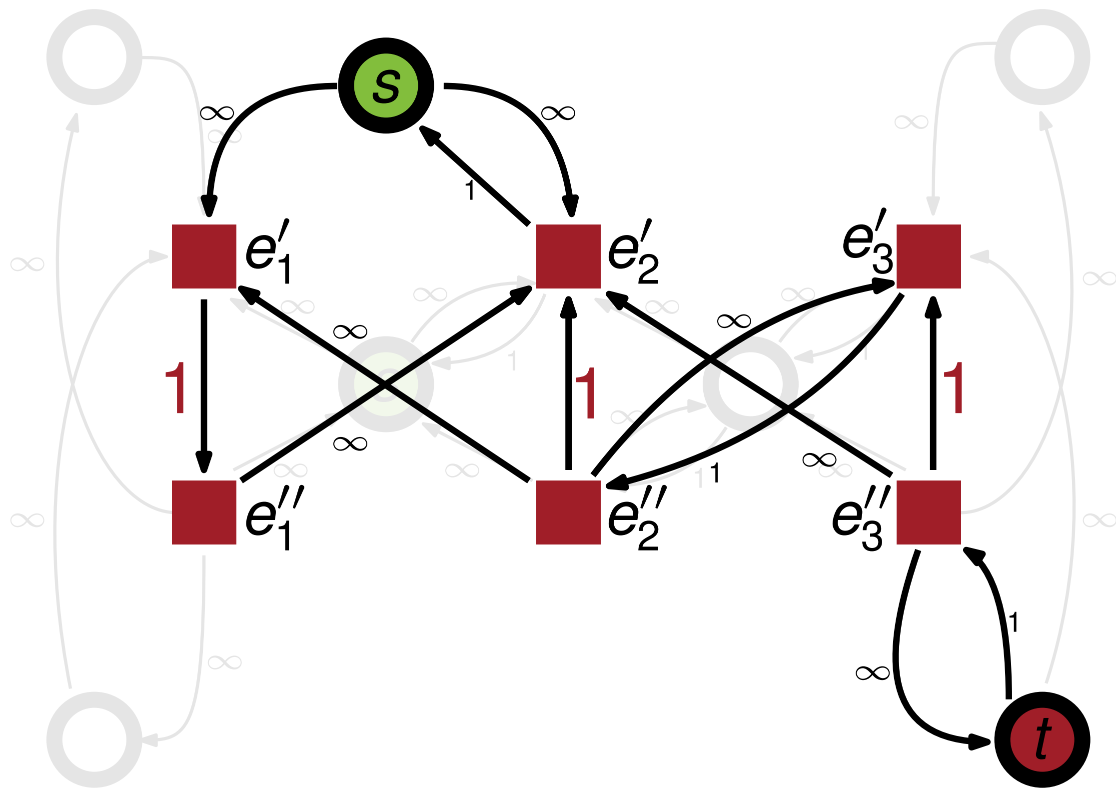
For each hypernode $v \in V_1$, there exists at least one $e \in I(v)$ with $e'' \in V_1$



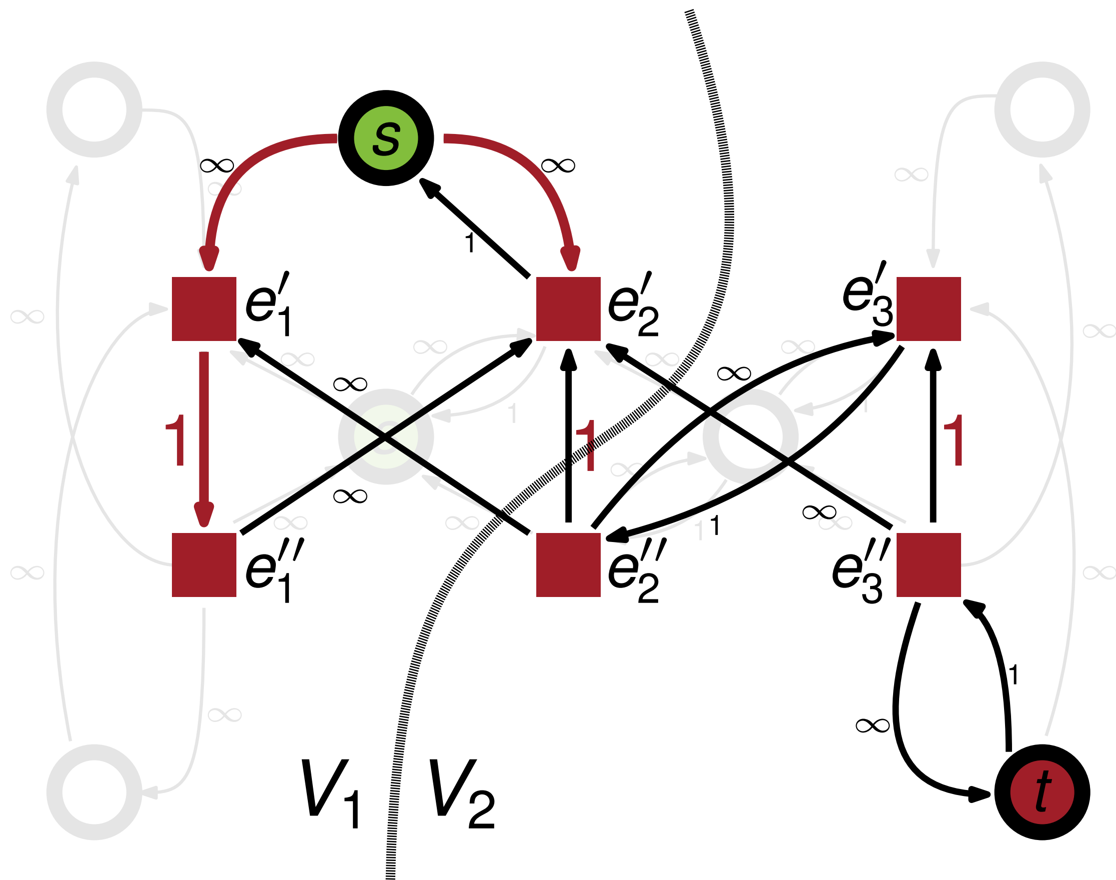
All nodes *reachable* from s are part of V_1 and $V_2 = V \setminus V_1$

Institute of Theoretical Informatics
Algorithmics Group

Reconstruction of Minimum (s, t)-Bipartition

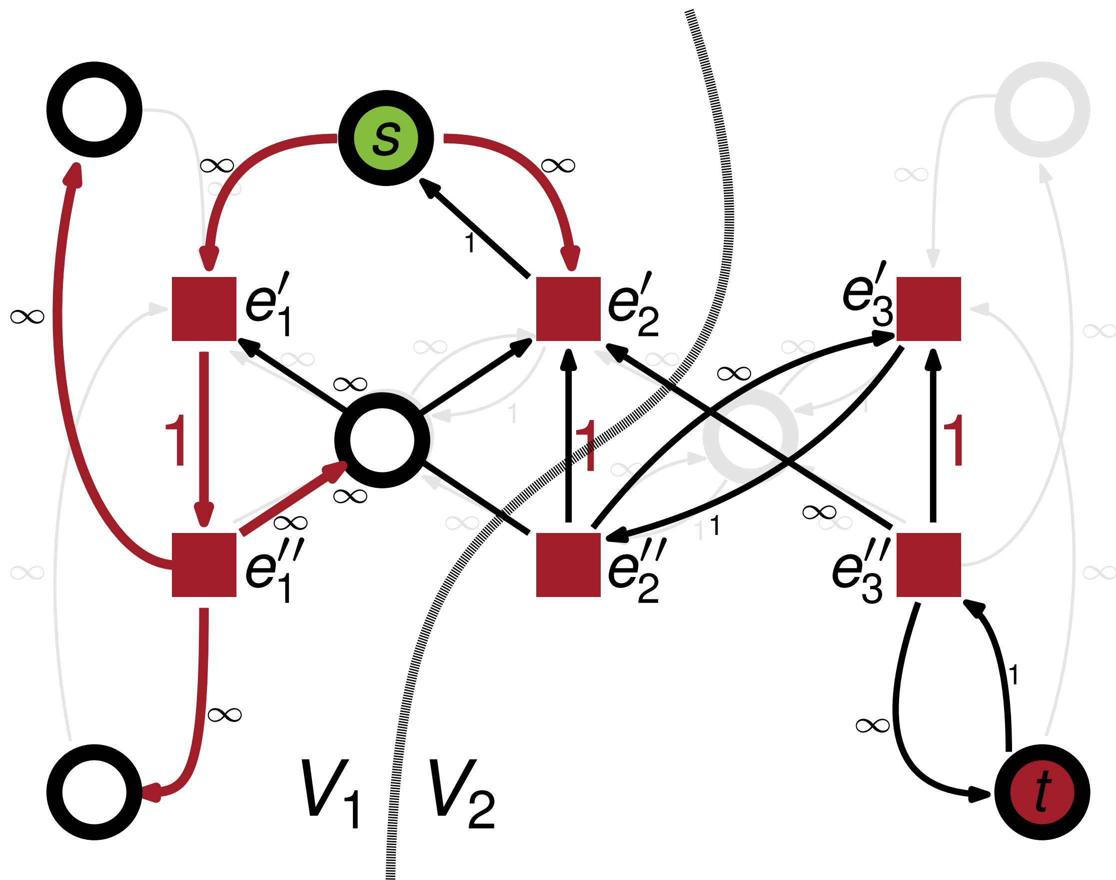


Reconstruction of Minimum (s, t)-Bipartition



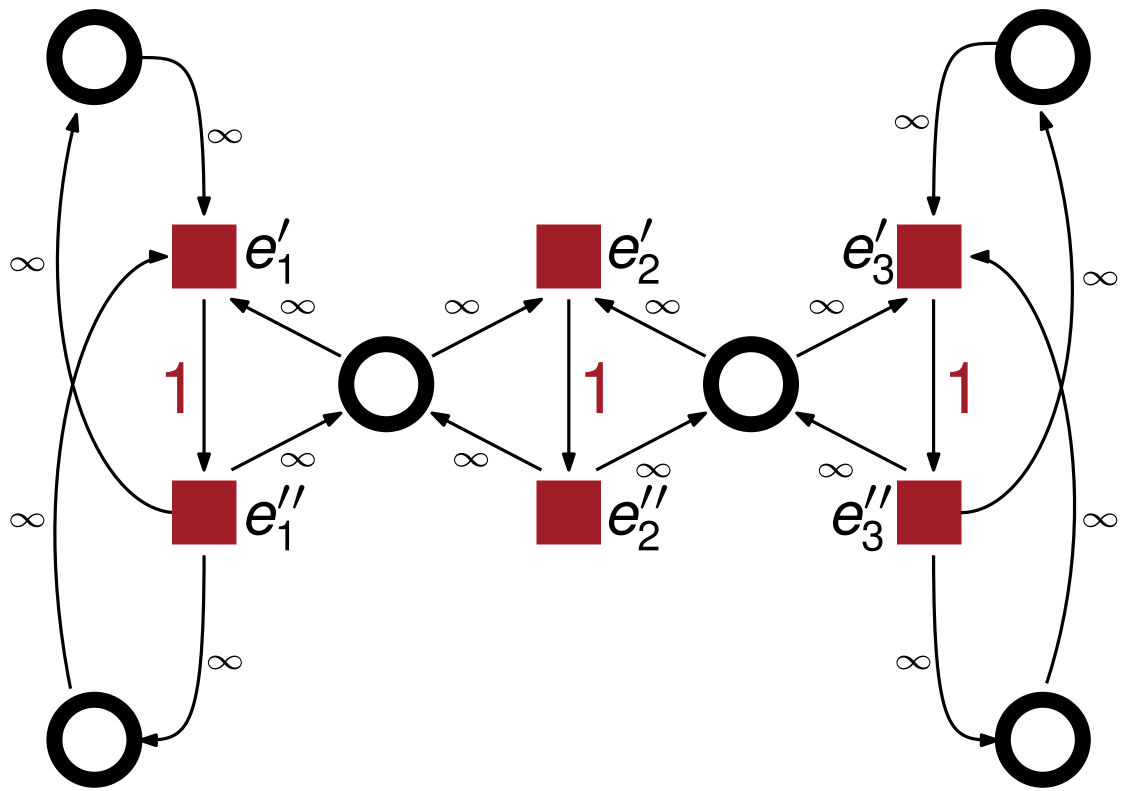
Institute of Theoretical Informatics
Algorithmics Group

Reconstruction of Minimum (s, t)-Bipartition



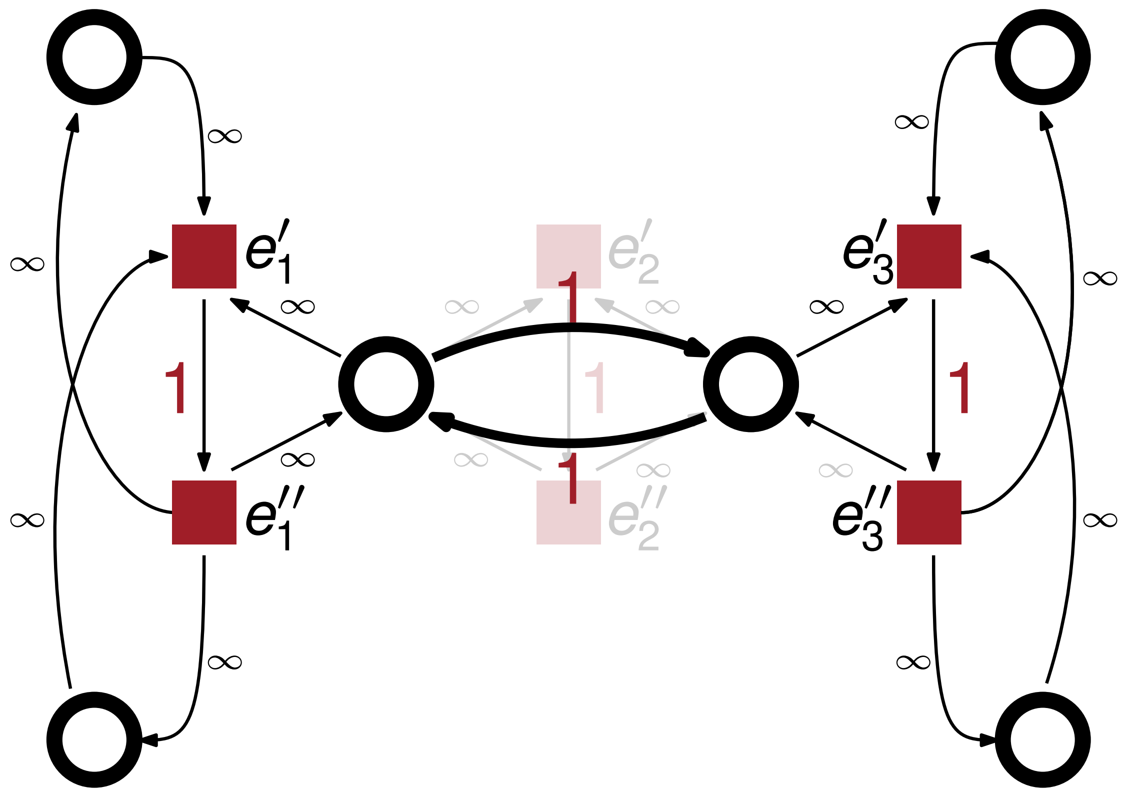
Institute of Theoretical Informatics
Algorithmics Group

Hypergraph Flow Network - Summary



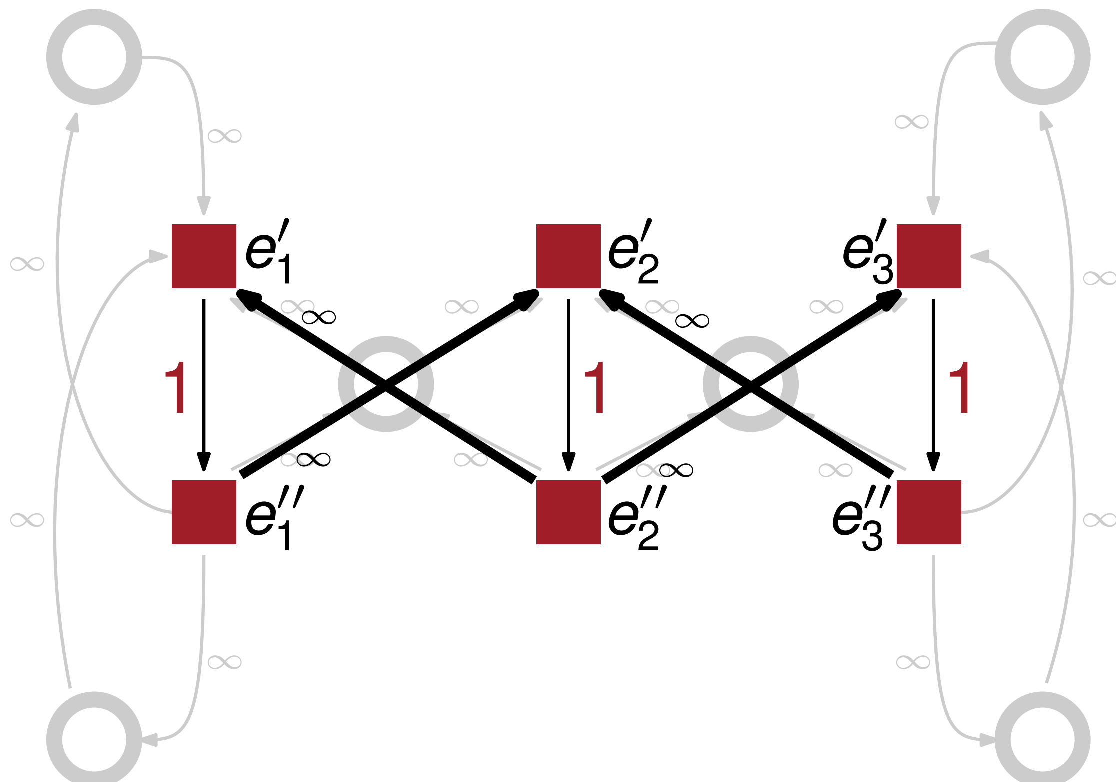
Lawler Network

Hypergraph Flow Network - Summary



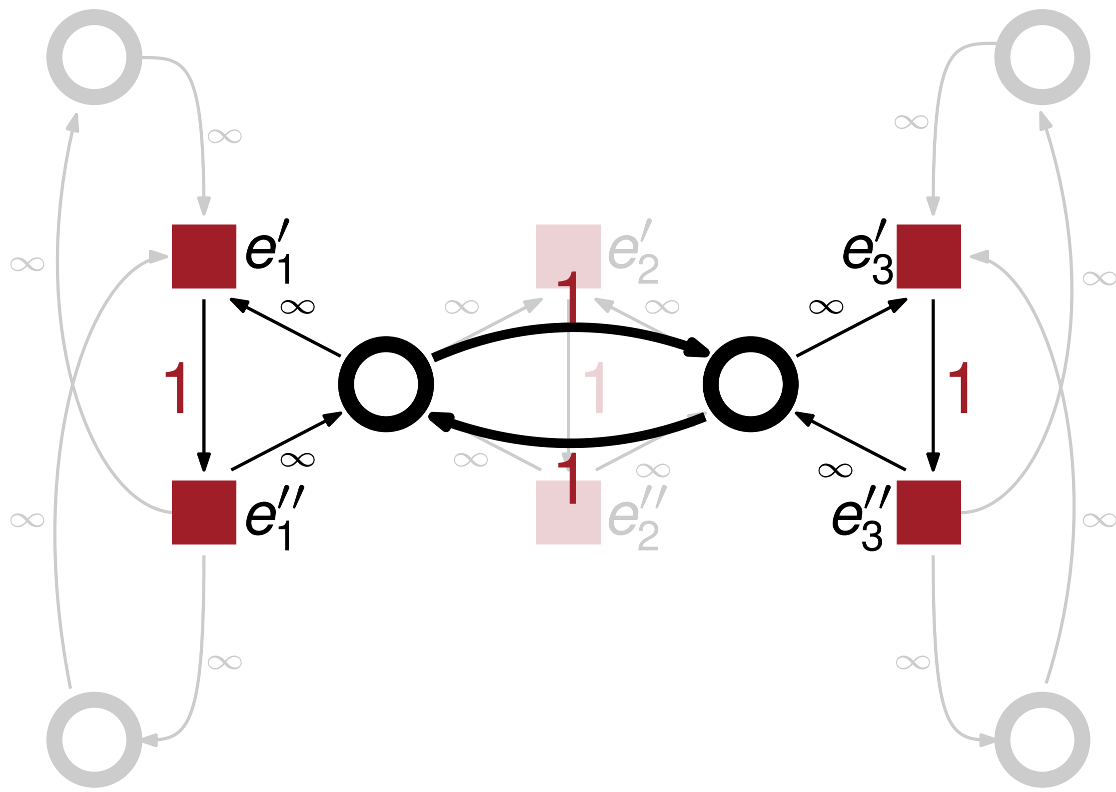
Wong Network

Hypergraph Flow Network - Summary



Our Network

Hypergraph Flow Network - Summary

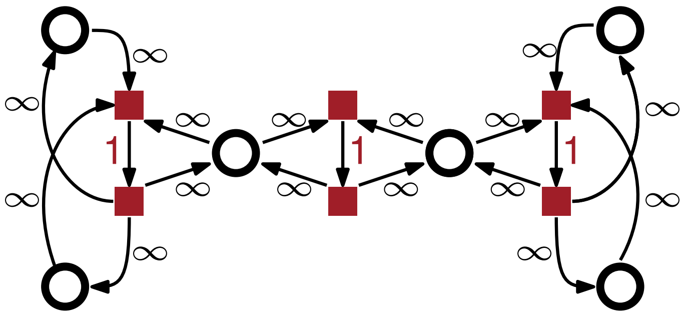


Hybrid Network

Institute of Theoretical Informatics
Algorithmics Group

Hypergraph Flow Network - Summary

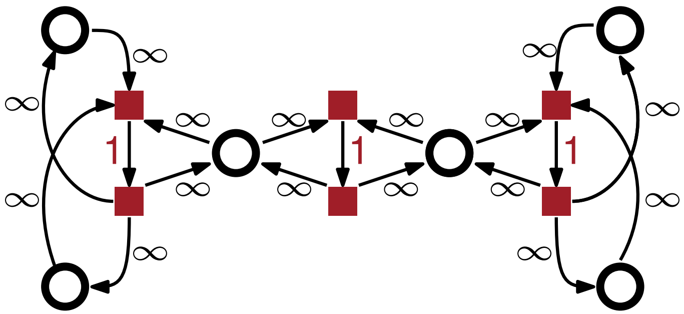
Lawler Network



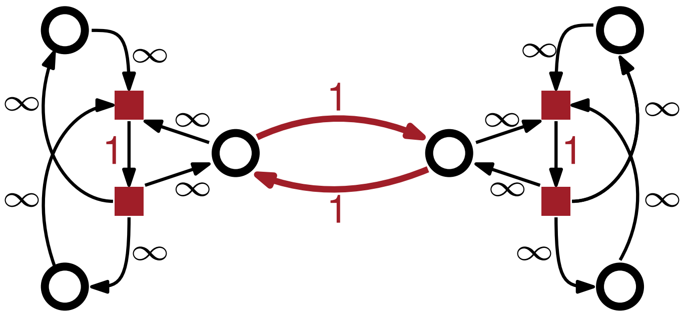
Institute of Theoretical Informatics
Algorithmics Group

Hypergraph Flow Network - Summary

Lawler Network



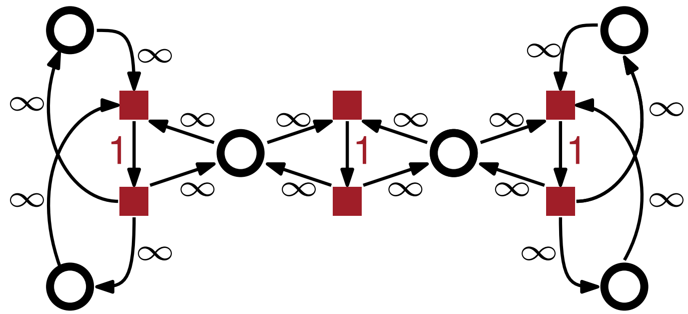
Wong Network



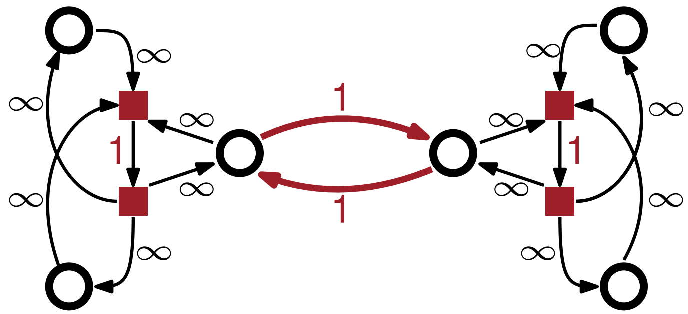
Institute of Theoretical Informatics
Algorithmics Group

Hypergraph Flow Network - Summary

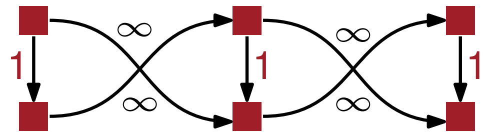
Lawler Network



Wong Network



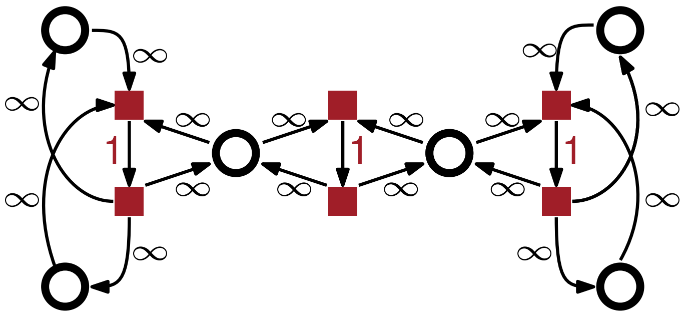
Our Network



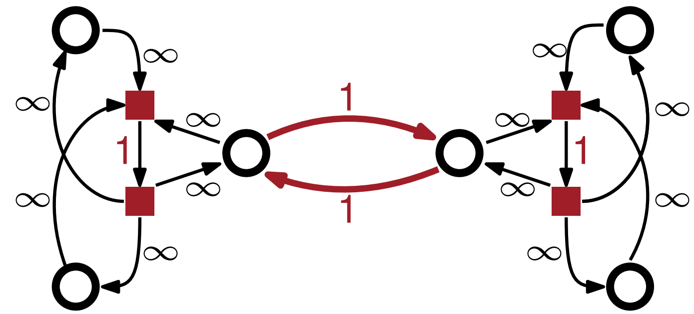
Institute of Theoretical Informatics
Algorithmics Group

Hypergraph Flow Network - Summary

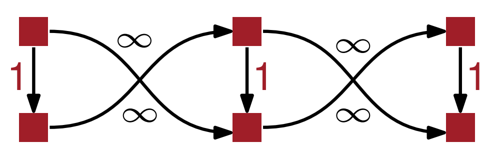
Lawler Network



Wong Network



Our Network



Hybrid Network