カットとフロー: Cuts and Flows

離散数学・オートマトン
2024 年後期
佐賀大学理工学部　只木進一

ネットワークとフロー: Networks and Flows

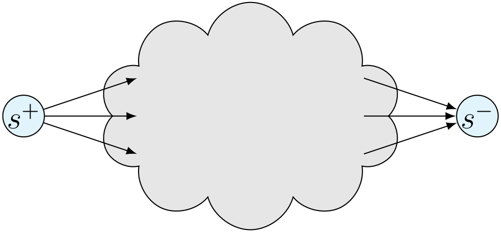
- **交通網中の流れ**: Flows in transportation networks
  - **都市間を流れる車両の数、及びその上限**: Traffic flow between cities and its capacity
  - **都市間を結ぶ航空路線が輸送する人数、及びその上限**: Number of passengers on an airline route between cities and its capacity
- **物流**: Logistics
  - **倉庫間を移動している商品数とその上限**: Number of goods moving between warehouses and its capacity
- **作業**: Operations
  - **各工程における処理数とその上限**: Number of items processed at each stage and its capacity

容量と流れ: Capacities and Flows

- 各辺に容量 (上限) がある: Capacity on each edge
  - 交通機関の輸送能力: Transportation capacity
  - 通信速度: Communication speed
- 各辺に実際に流れる流量: Flow on each edge
  - 容量以下: Less than or equal to the capacity
- ネットワークの二点に最大流量を実現する方法: How to achieve the maximum flow between two points in a network

2端子フロー: Two-terminal Flows

- 有向ネットワーク: In a directed network
- 入口 s^+ と出口 s^-: Source s^+ and sink s^-
- 入口から出口までの有向道が存在する: Directed paths from the source to the sink exist.
- 各辺に容量が定義され、それ以下の流量を割り当てる。: Assign flow less than or equal to the capacity on each edge



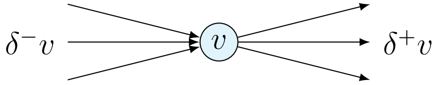
2端子フローの定義 Definition of Two-terminal Flows

- グラフ (Graph) $G = (V, E)$
- 入口 s^+ と出口 s^- の間に有向道がある: Directed paths from the source to the sink
- $\forall e \in E$ に、流量上限 (capacity) $c(e) \geq 0$ を定義する: Capacity $c(e) \geq 0$ defined on each edge
- $\forall e \in E$ に、流量 (flow) $c(e) \geq \phi(e) \geq 0$ を設定する: Assign flow $\phi(e) \geq 0$ on each edge
- 入口 s^+ から出口 s^- への**最大流量**を求める: Find the **maximum flow** from the source to the sink

流量に対する制約: Constraints on Flows

- $\forall v \in V \setminus \{s^+, s^-\}$
- **容量による制約**: $0 \le \phi(e) \le c(e)$: Capacity constraints
- **流量保存則**: **頂点** v **で「湧き出し** (source)**」と「吸い込み** (sink)**」がない**: Flow conservation at each vertex

$$\partial\phi(v) \equiv \sum_{e \in \delta^+ v} \phi(e) - \sum_{e \in \delta^- v} \phi(e) = 0$$



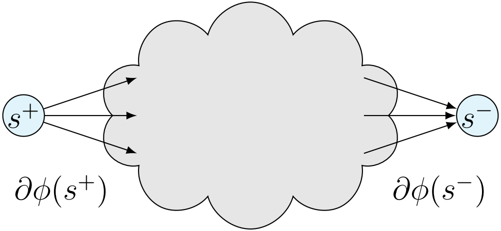
$$\sum_{e \in \delta^- v} \phi(e) \qquad \sum_{e \in \delta^+ v} \phi(e)$$

ネットワークフローのイメージ
Image of Network Flows

- **ネットワークの流量**: Flow in the network

$$Q(\phi) = \partial\phi(s^+) = -\partial\phi(s^-) \tag{1.1}$$

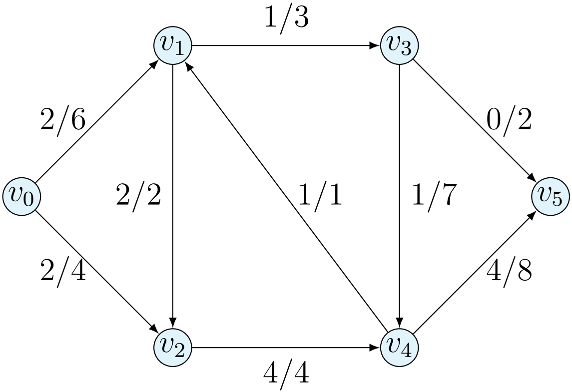
- s^+ **から入った流れは全て** s^- **へ至る**: All flows entering s^+ reach s^-



最大フローを見つける考え方
How to find the maximum flow

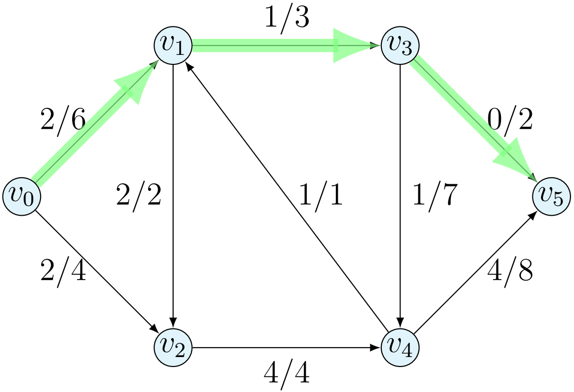
数字: 流量/容量

Numbers: flow/capacity



最大フローを見つける考え方 0
How to find the maximum flow 0

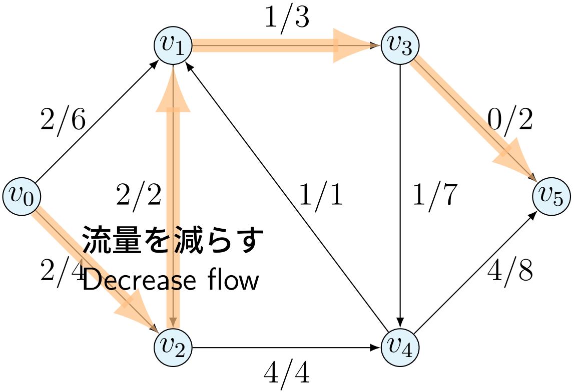
数字: 流量/容量
Numbers: flow/capacity



流量を増やせる道を見つける 1
Find a path to increase the flow 1

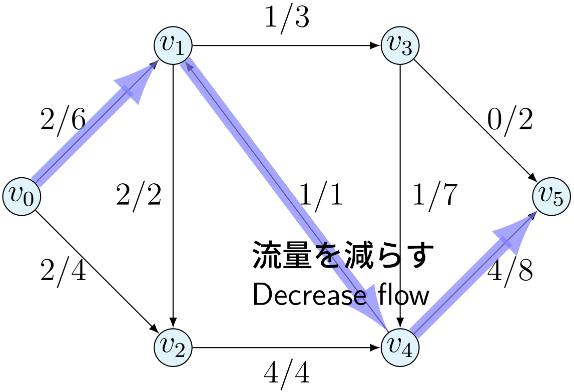
数字: 流量/容量

Numbers: flow/capacity



流量を増やせる道を見つける2
Find a path to increase the flow 2

数字: 流量/容量
Numbers: flow/capacity



わかりやすいアルゴリズムへ: Towards a more understandable algorithm

- わかりにくい点: Difficulties
  - 辺の向きとは逆方向に「流す」: Flow in the opposite direction
  - 逆向きの辺の流量を減らす: Decrease the flow on the edge with the opposite direction
  - 容量と流量の二つの量が出てくる: Two quantities, capacity and flow on each edge
- 改善: 補助ネットワークの導入: Improvement by introducing an auxiliary network
  - 各辺の一つの量に注目: Focus on one quantity on each edge

補助ネットワーク: Auxiliary Network

$$N_A = \left(G_\phi\left(V, E_\phi\right), s^+, s^-, c_\phi\right) \tag{2.1}$$

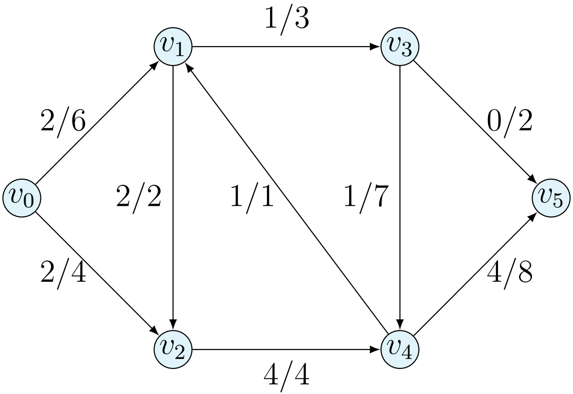
$$E_\phi = E_\phi^+ \cup E_\phi^- \tag{2.2}$$

- E_ϕ^+: **元のネットワークと順方向の辺**: Edges with the same direction as the original network
  - **容量として、容量の残り余裕を設定**: Set the capacity as the remaining capacity
- E_ϕ^-: **元のネットワークと逆方向の辺**: Edges with the opposite direction as the original network
  - **容量として、流量を設定**: Set the capacity as the flow

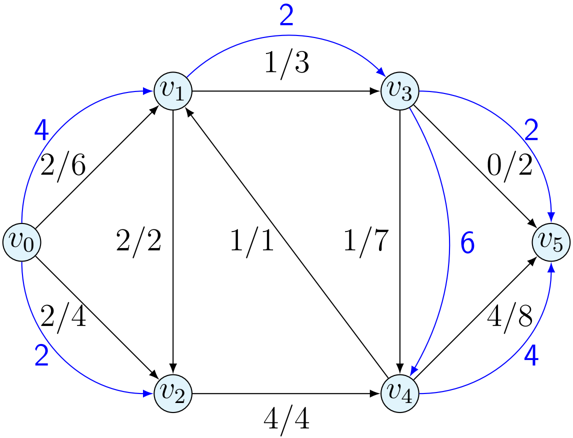
E_ϕ^+ の構成: Construction of E_ϕ^+

- 元のネットワーク G の $\forall e \in E$ に対して、
  For $\forall e \in E$ in the original network G
  - 辺の追加 (Add an edge) : $E_\phi^+ \ \{\forall e \in E\}$
  - 容量の設定: 残りの容量 (Set the remaining capacity as the capacity): $c(e) \leftarrow c(e) - \phi(e)$

E_ϕ^+ の構成: Construction of E_ϕ^+



E_ϕ^+ の構成: Construction of E_ϕ^+



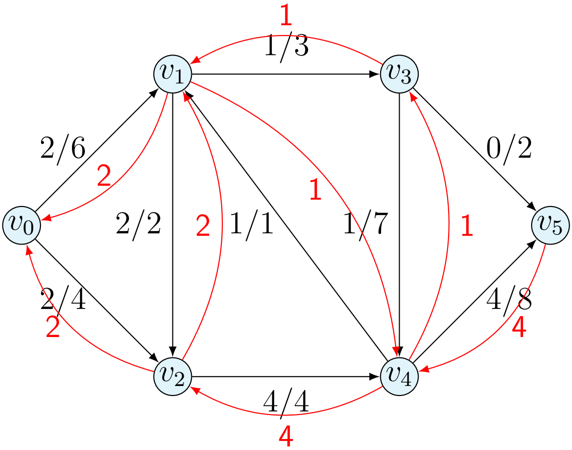
$c(e) = 0$ である $e \in E_\phi^+$ は表示しない

Edges with $c(e) = 0$ ($e \in E_\phi^+$) are not shown

E_ϕ^- の構成: Construction of E_ϕ^-

- **元のネットワーク G の $\forall e \in E$ に対して**
  For $\forall e \in E$ in the original network G
  - **e と逆向きの辺 $e^\dagger$ の追加**: Add an edge with the opposite direction
    $E_\phi^- = \left\{ e^\dagger \mid \forall e \in E \right\}$
  - **容量の設定**: **削減可能流量**: Set the capacity as the reducible flow
    $c(e^\dagger) \leftarrow \phi(e)$
  - **元のネットワークと辺の向きが逆である**
    Note that the edges are in the opposite direction of the original network

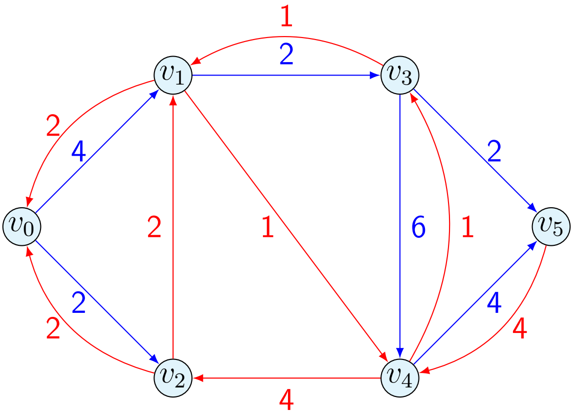
E_ϕ^- の構成: Construction of E_ϕ^-



$c(e) = 0$ である $e \in E_\phi^+$ は表示しない

Edges with $c(e) = 0$ in E_ϕ^+ are not shown

補助ネットワーク: Auxiliary Network



増加道を見つける
Finding an augmenting path

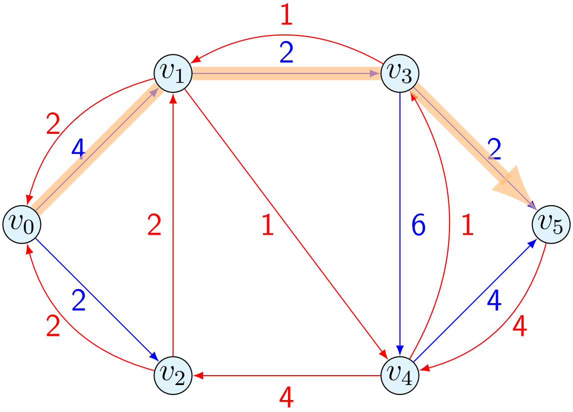
- 補助ネットワーク中に、s^+ から s^- への有向道 P (有向道) があれば、
  If there is a directed path P from s^+ to s^- in the auxiliary network
  $$d = \min_{e \in P} c_\phi(e) \tag{2.3}$$
  だけ流量を増やすことができる:We can increase the flow by d
  - $e \in E_\phi^+$ ならば、容量に余裕がある: There is room in the capacity
  - $e \in E_\phi^-$ ならば、逆方向の辺の流量を減らすことができる: We can reduce the flow on the edge in the opposite direction

- **このときの、ネットワーク中の新しい流量**: New flow in the network

$$\phi'(e) = \begin{cases} \phi(e) + d & \text{for } e \in E_\phi^+ \wedge e \in P \\ \phi(e) - d & \text{for } e \in E_\phi^- \wedge e \in P \\ \phi(e) & \text{otherwise} \end{cases} \tag{2.4}$$

- **容量が整数ならば流量は整数**: If the capacity is an integer, the flow is an integer

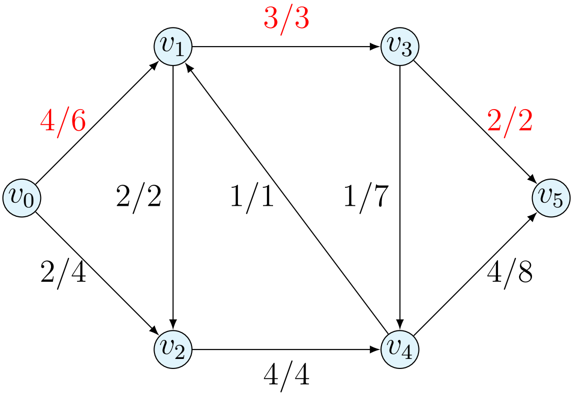
増加道を見つける
Finding an Augmenting Path



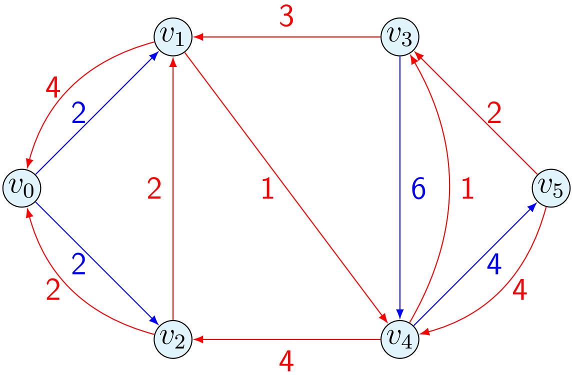
容量 0 の辺は対象外とする

Ignore edges with capacity 0

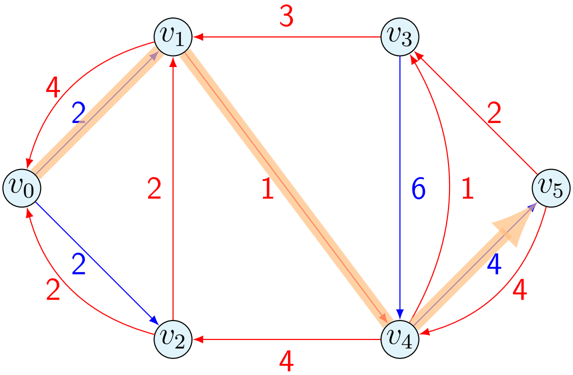
流量増加 $d = 2$: Augmenting the Flow



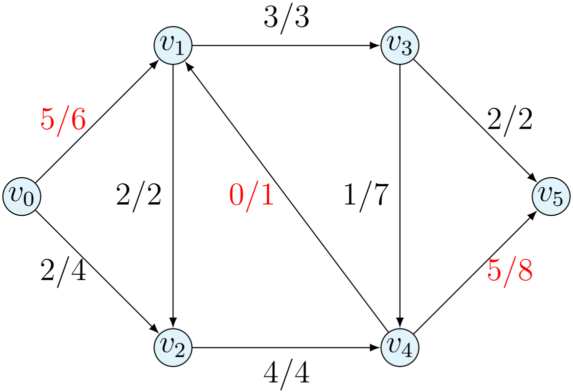
補助ネットワーク構成
Construct the auxiliary network



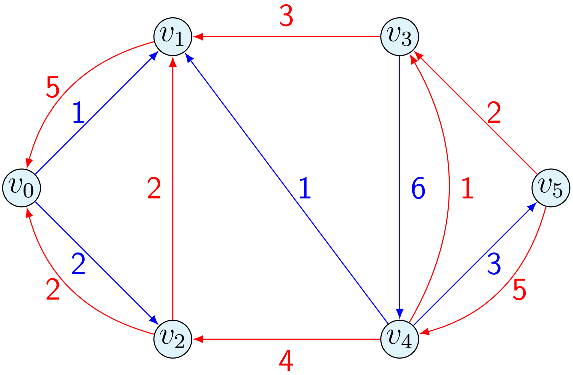
別の増加道を見つける
Finding another Augmenting Path



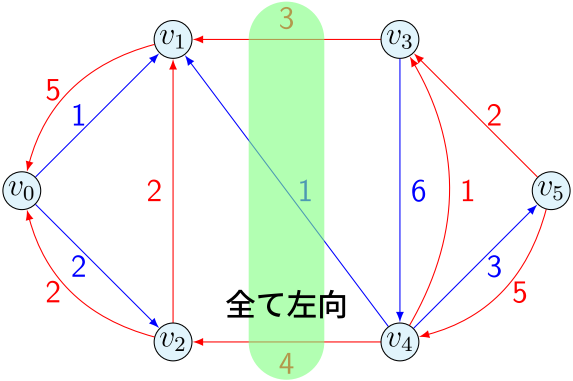
流量増加 $d = 1$: Augmenting the Flow



補助ネットワーク構成
Construct the auxiliary network



補助ネットワークに有向道はない
No Augmenting Path in the Auxiliary Network



いちいち元のネットワーク戻す必要があるか

Do we need to return to the original network every step?

- **補助ネットワークの各辺の容量を更新するアルゴリズム**
  Algorithm to update the capacity of each edge in the auxiliary network
- $e \in E_\phi^\pm$ **の容量を更新したら、** $e \in E_\phi^\mp$ **も更新**
  Update $e \in E_\phi^\mp$ after updating the capacity of $e \in E_\phi^\pm$

アルゴリズムとして整理

Algorithm 1 最大フローのアルゴリズム

```
補助ネットワーク N_A を構成する
s^+ から s^- への有向道 P を得る
while P が存在 do
    d = min_{e∈P} c_φ(e)
    update(N_A, P, D)                ▷ N_A を更新
    s^+ から s^- への有向道 P を得る
end while
deploy(N_A)                          ▷ 元のネットワークへ反映
```

Organizing as an algorithm

Algorithm 2 Algorithm for Maximum Flow

Construct the auxiliary network N_A
Find a directed path P from s^+ to s^-
while P exists **do**
  $d = \min_{e \in P} c_\phi(e)$
  update(N_A, P, D) ▷ Update N_A
  Find a directed path P from s^+ to s^-
end while
deploy(N_A) ▷ Reflect the state of N_A in the original network

Algorithm 3 補助ネットワーク更新: Update the auxiliary network

procedure $\textsc{Update}(N_A, P, d)$
  for $e \in P$ **do**
    $c_\phi(e) = c_\phi(e) - d$
    $e \in E_\phi^\pm$ **ならば、対応する辺** $e^\dagger \in E_\phi^\mp$ **を選択**
    (Select the corresponding edge $e^\dagger \in E_\phi^\mp$ if $e \in E_\phi^\pm$)
    $c_\phi(e^\dagger) = c_\phi(e^\dagger) + d$
  end for
end procedure

Algorithm 4 元のネットワークへの反映: Reflect to the original network

procedure DEPLOY(N_A)
  for $e \in E$ **do**
    $f \in E_\phi^-$ **は** a **に対応する辺**
    (The edge $f \in E_\phi^-$ corresponds to the edge a)
    $\phi(e) = c_\phi(f)$
  end for
end procedure

カット: Cut

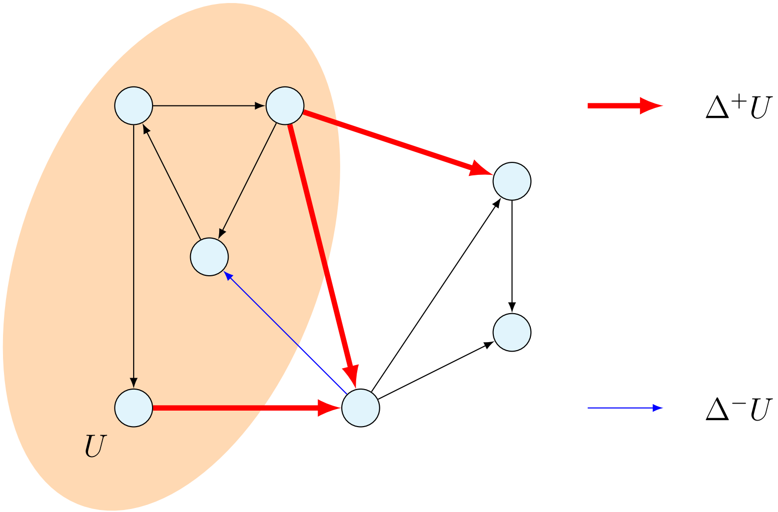
- ネットワークのカット:Cuts in a network: $U \subset V$
  - s^+ を含み、s^- を含まない頂点集合
    A set of vertices that includes s^+ and does not include s^-

$$\left(s^+ \in U\right) \wedge (s^- \notin U) \tag{4.1}$$

- カットの容量: Capacity of a cut: $\kappa_{\mathrm{C}}(U)$
  - $\Delta^+ U$: U から出て、$U \setminus V$ へ入る辺全体の容量
    Capacity of all edges leaving U and entering $U \setminus V$

$$\kappa_{\mathrm{C}}\left(U\right) = \sum_{e \in \Delta^+ U} c(e) \tag{4.2}$$

カットとその境界: Cut and its boundary



流量とカット:Flow and Cut

- N 中の任意のフロー ϕ と任意のカット U: For any flow ϕ in N and any cut U
$$Q(\phi) \leq \kappa_{\mathrm{C}}(U) \tag{4.3}$$
- 直感的には: s^+ と s^- の途中にあるボトルネック部分で流量上限が定まる: Intuitively, the flow limit is determined at the bottleneck part between s^+ and s^-

$$\begin{aligned} Q(\phi) &= \sum_{e \in \Delta^+ U} \phi(e) - \sum_{e \in \Delta^- U} \phi(e) \\ &\leq \sum_{e \in \Delta^+ U} c(e) - 0 = \kappa_{\mathrm{C}}(U) \end{aligned} \tag{4.4}$$

最大流量と最小カット Maximum Flow and Minimum Cut

$$\max Q(\phi) \le \min \kappa_{\mathrm{C}}(U) \tag{4.5}$$

- 実際には等号がなりたつ: In fact, the equality holds
- つまり、ボトルネック容量で最大流量が定まる: The maximum flow is determined by the bottleneck capacity

最大流量と最小カットが等しいこと Equality of Maximum Flow and Minimum Cut

- ネットワーク N の最大流量 ϕ が実現しているならば: If the maximum flow ϕ in the network N is realized
  - 補助ネットワーク N_A には、s^+ から s^- への有向道は存在しない: There is no directed path from s^+ to s^- in the auxiliary network N_A
  - 注意: 容量 0 の辺は無視する: Note that edges with capacity 0 are omitted.
- 補助ネットワーク中の s^+ から到達可能な頂点集合: $W \subset V$: Set of vertices reachable from s^+ in the auxiliary network
  - N_A には、W から外向きの辺は存在しない。: There are no edges leaving out of W in N_A

W への内向きの辺 e は、以下のいずれかである Inward edges e to W are one of the followings

- $e^{\dagger} \in E_{\phi}^{-}$: N の対応する辺 e の容量を使い切っている: The corresponding edge e in N has used up its capacity

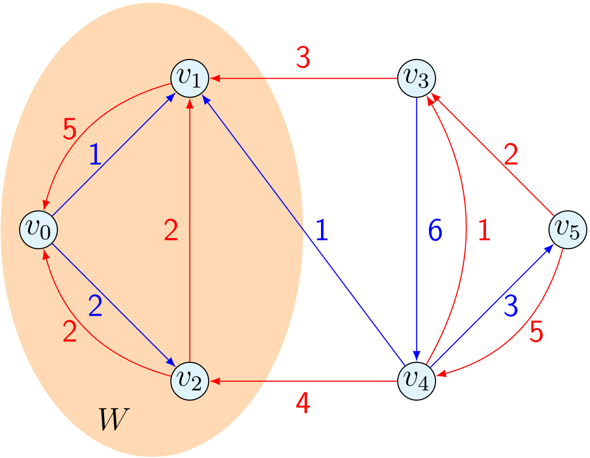
$$\phi(e) = c(e) \tag{4.6}$$

- $e \in E_{\phi}^{+}$: N の辺の流量は 0: The flow on the edge in N is 0

$$\phi(e) = 0 \tag{4.7}$$

$$\begin{aligned} Q(\phi) &= \sum_{e \in \Delta^{+} W} \phi(e) - \sum_{e \in \Delta^{-} W} \phi(e) \\ &= \sum_{e \in \Delta^{+} W} c(e) - 0 = \kappa_{\mathrm{C}}(W) \end{aligned} \tag{4.8}$$

補助ネットワークにおけるカット
The cut in the auxiliary network



元のネットワークにおけるカット: The cut in the original network