数理計画法 (数理最適化) 第7回

ネットワーク最適化
最大流問題と増加路アルゴリズム

担当： 塩浦昭義
(情報科学研究科　准教授)
shioura@dais.is.tohoku.ac.jp

中間試験について

- 日時：11月27日（木）13:00～14:30
- 11/20までにレポートを一度も出していない場合，受験不可
- 手書きのA4用紙一枚のみ持ち込み可（印刷やコピーは不可）
  - これも採点の対象，試験終了後に回収します
- 教科書，ノート等の持ち込みは不可
- 座席はこちらで指定
- 試験内容：11/20（第7回目，来週）までの講義内容で，指定したところ
  - 様々な数理計画モデル
  - 線形計画：標準形，単体法，各種定理
  - 組合せ最適化
- 50点満点，29点以下は不合格

組合せ最適化問題

- 組合せ最適化問題とは:
  - 有限個の「もの」の組合せの中から,
    目的関数を最小または最大にする組合せを見つける問題
  - 例1:整数計画問題全般(整数の組合せ)
  - 例2:グラフの最小木問題，最短路問題，(グラフの枝の組合せ)
  - 例3:巡回セールスマン問題(都市の順列)

- 解きやすい問題と解きにくい問題
  - 解きやすい問題≒多項式時間で解ける問題
  - 解きにくい問題≒NP困難な問題
    (多項式時間で解けないと予想されている問題)

※注意:組合せ最適化問題の解は有限個➔有限時間で必ず解ける!

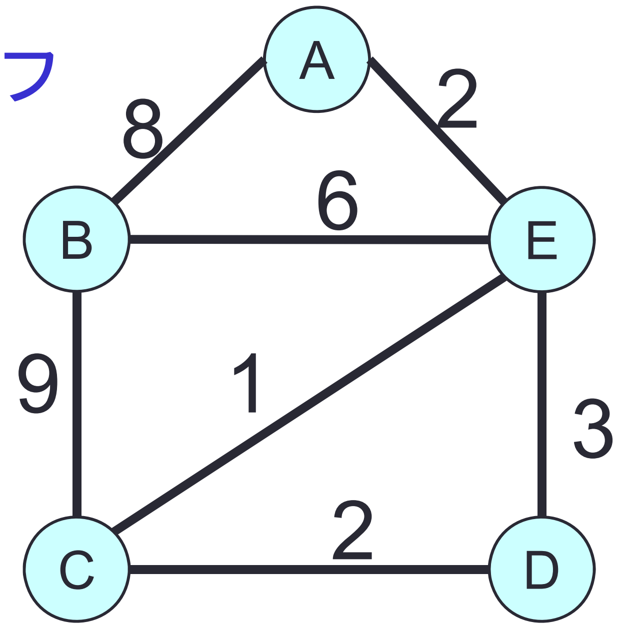
組合せ最適化問題に対するアプローチ

- 組合せ最適化問題をどのように解くか？
- 解きやすい問題の場合
  - 多項式時間アルゴリズムを構築➔より高速な解法へ
- 解きにくい問題の場合
  - 絶対に最適解が必要な場合➔厳密解法
    - 分枝限定法（授業で説明）🡐現在の主流
    - 動的計画法（「アルゴリズムとデータ構造」の講義）
  - ある程度良い解であれば十分という場合
    - 精度保証付き近似アルゴリズム
      （解の良さに対する理論保証あり）
    - ヒューリスティックス（解の良さは実験的に証明）

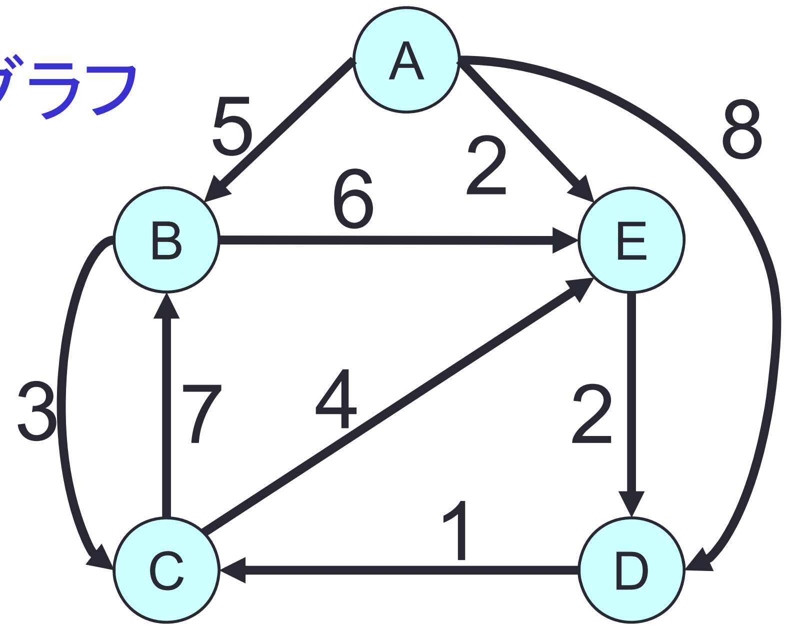
ネットワーク最適化問題

- （無向，有向）グラフ
  - 頂点（vertex，接点，点）が枝（edge，辺，線）で結ばれたもの
- ネットワーク
  - 頂点や枝に数値データ（距離，コストなど）が付加されたもの
- ネットワーク最適化問題
  - ネットワークを使って表現される組合せ最適化問題

無向グラフ



有向グラフ



ネットワーク最適化問題の例

「ネットワーク」に関する数理計画問題

最小木問題
(minimum spanning tree prob.)
最短路問題
(shortest path prob.)

他の講義で扱う
「アルゴリズムとデータ構造」
「情報数学」

最大流問題
(maximum flow prob.)
最小費用流問題
(minimum cost flow prob.)
割当問題
(assignment prob.)

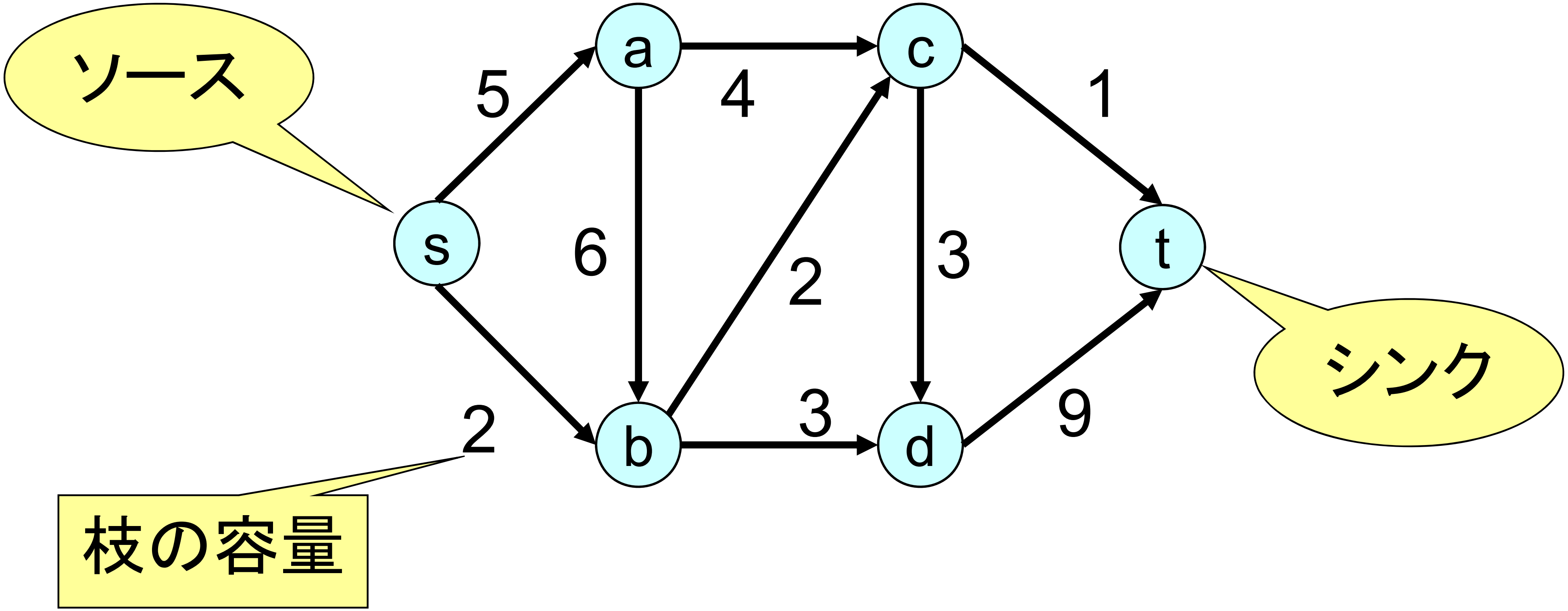
この授業で扱う

最大流問題の定義(その1)

入力：有向グラフ $G = (V, E)$

ソース（供給点） $s \in V$, シンク（需要点） $t \in V$

各枝 $(i, j) \in V$ の容量 $u_{ij} \geqq 0$



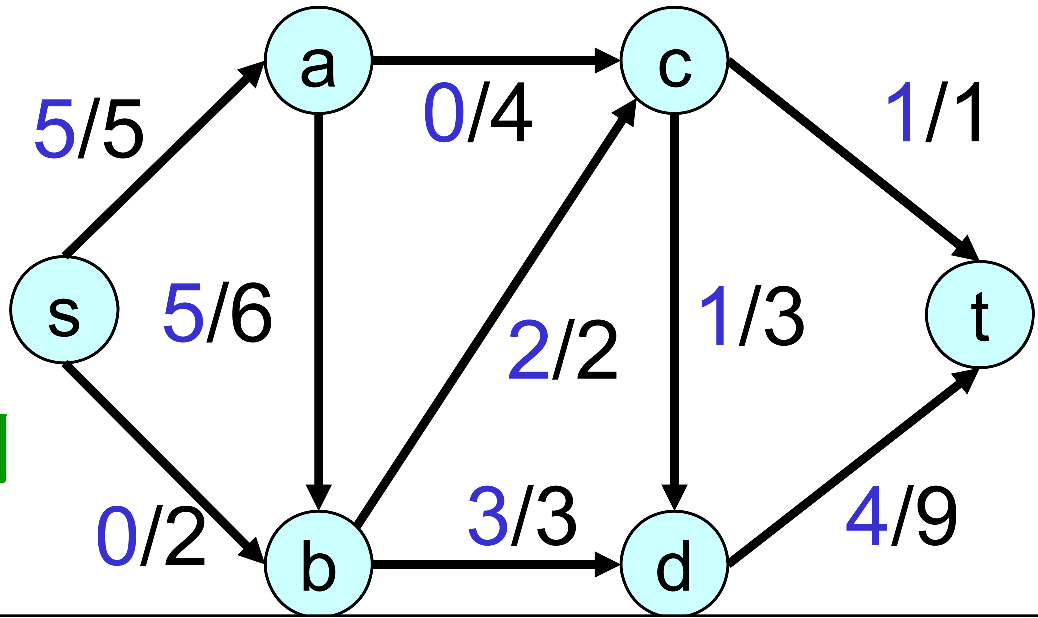
最大流問題の定義(その2)

目的:ソースからシンクに向けて，枝と頂点を経由して
「もの」を出来るだけたくさん流す

条件1(容量条件):
0 ≦ 各枝を流れる「もの」の量 ≦ 枝の容量

条件2(流量保存条件):
頂点から流れ出す「もの」の量＝　流れ込む「もの」の量

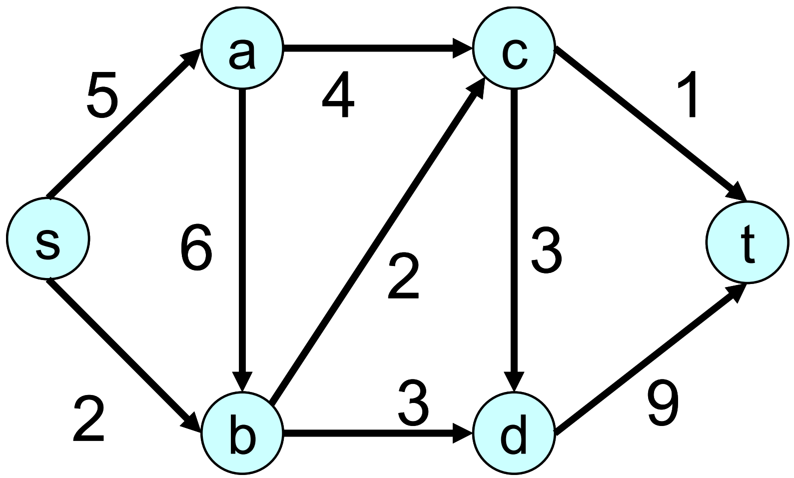
実行可能解の例



最大流問題の定式化:
変数, 目的関数と容量条件

変数 x_{ij}: フロー＝枝 (i, j) を流れる「もの」の量
変数 f: 総流量　＝ シンクに流れ込む「もの」の総量
　　　　　　　　＝ ソースから流れ出す「もの」の総量

目的:ソースからシンクに「もの」を出来るだけ多く流したい
⇒ 最大化 f
容量条件:0 ≦ 各枝を流れる「もの」の量 ≦ 枝の容量
⇒ $0 \leqq x_{ij} \leqq u_{ij}$ 　($(i,j) \in E$)



最大化 f
容量条件:
$0 \leqq x_{sa} \leqq 5$, $0 \leqq x_{sb} \leqq 2$, $0 \leqq x_{ab} \leqq 6$,
$0 \leqq x_{ac} \leqq 4$, $0 \leqq x_{bc} \leqq 2$,
…

最大流問題の定式化:流量保存条件

流量保存条件:

(頂点から流れ出す「もの」の量) − (流れ込む「もの」の量) = 0

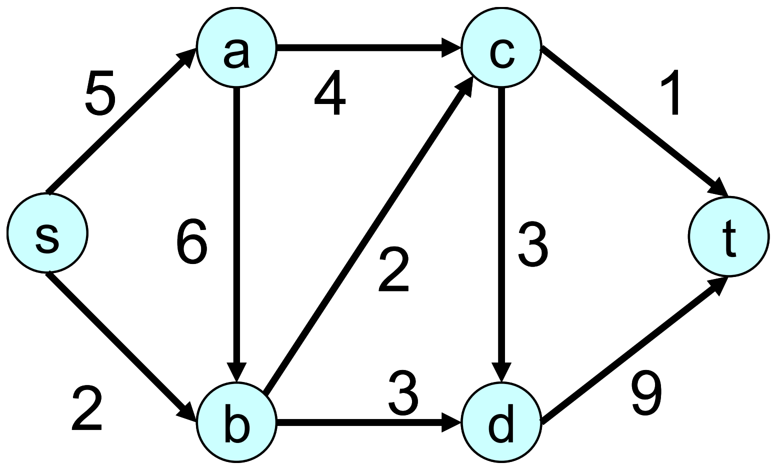
⇒ $\sum\{x_{kj}$ | 枝 (k,j) は 頂点 k から出る}

　　− $\sum\{x_{ik}$ | 枝 (i,k) は 頂点 k に入る} = 0 (k ∈ V − {s, t})

ソースとシンクに関する条件:

$\sum\{x_{sj}$ | (s,j) は s から出る} - $\sum\{x_{is}$ | (i,s) は s に入る} = f

$\sum\{x_{tj}$ | (t,j) は t から出る} - $\sum\{x_{it}$ | (i,t) は t に入る} = - f



流量保存条件の例:

$x_{ac} + x_{ab} - x_{sa} = 0$

$x_{bc} + x_{bd} - x_{ab} - x_{sb} = 0$

$x_{ct} + x_{cd} - x_{ac} - x_{cb} = 0$

$x_{dt} - x_{cd} - x_{bd} = 0$

$x_{sa} + x_{sb} = f$,　　$- x_{ct} - x_{dt} = - f$

最大流問題の定式化:まとめ

最大化　f

条件　$0 \leqq x_{ij} \leqq u_{ij}$　　$((i,j) \in E)$

$\sum\{x_{kj} \mid (k,j)$ は k から出る$\} - \sum\{x_{ik} \mid (i,k)$ は k に入る$\} = 0$

(k : s, t 以外の頂点)

$\sum\{x_{sj} \mid (s,j)$ は s から出る$\} - \sum\{x_{is} \mid (i,s)$ は s に入る$\} = f$

$\sum\{x_{tj} \mid (t,j)$ は t から出る$\} - \sum\{x_{it} \mid (i,t)$ は t に入る$\} = -f$

この問題の実行可能解 x_{ij} --- 実行可能フロー
総流量が最大の実行可能フロー --- 最大フロー

最大流問題の応用例

- 物流
- シーズン途中でのプロ野球チームの優勝可能性判定
  - 残り試合全勝しても優勝の可能性がないかどうか？
- 画像処理における物体の切り出し
  - 画像内の物体のみ取り出す
- その他多数

最大流問題の解法

最大流問題は線形計画問題の特殊ケース

⇒　単体法で解くことが可能

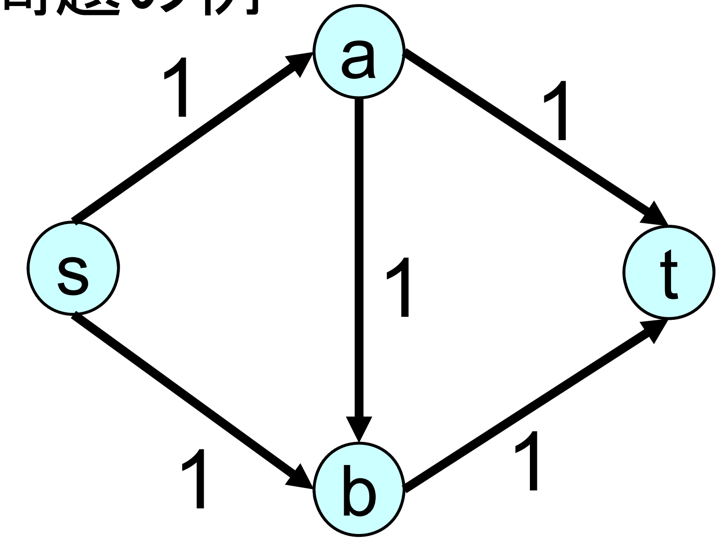
最大流問題は良い（数学的な）構造をもつ

⇒　この問題専用の解法（増加路アルゴリズムなど）

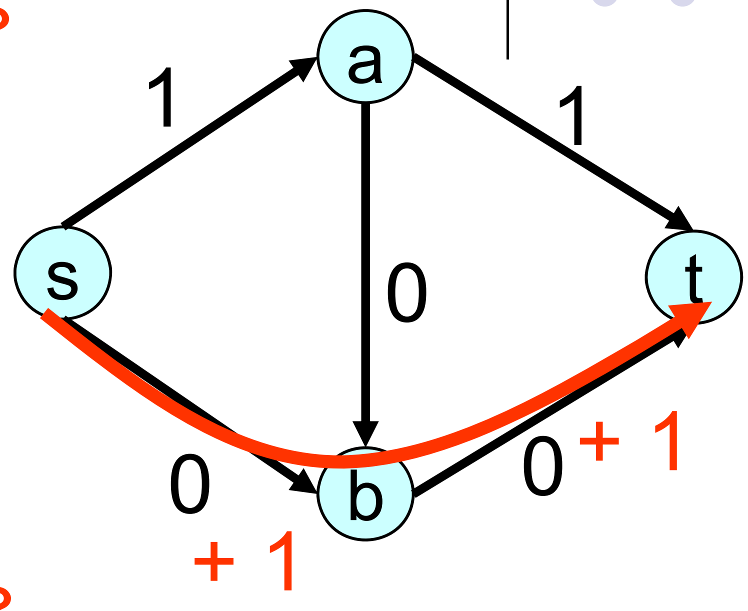
を使うと，より簡単＆より高速に解くことが可能

最大フローの判定

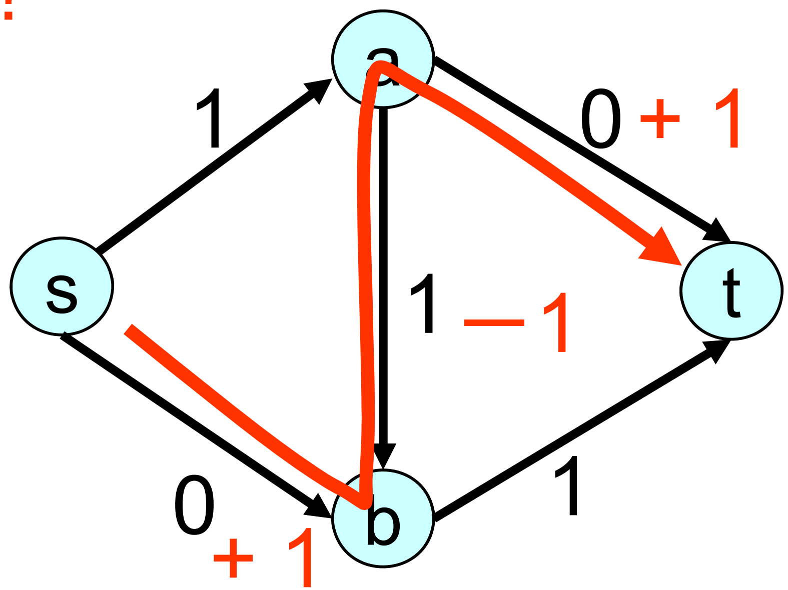
問題の例



フロー例1：最大？
最大ではない



フロー例2：最大？
最大ではない



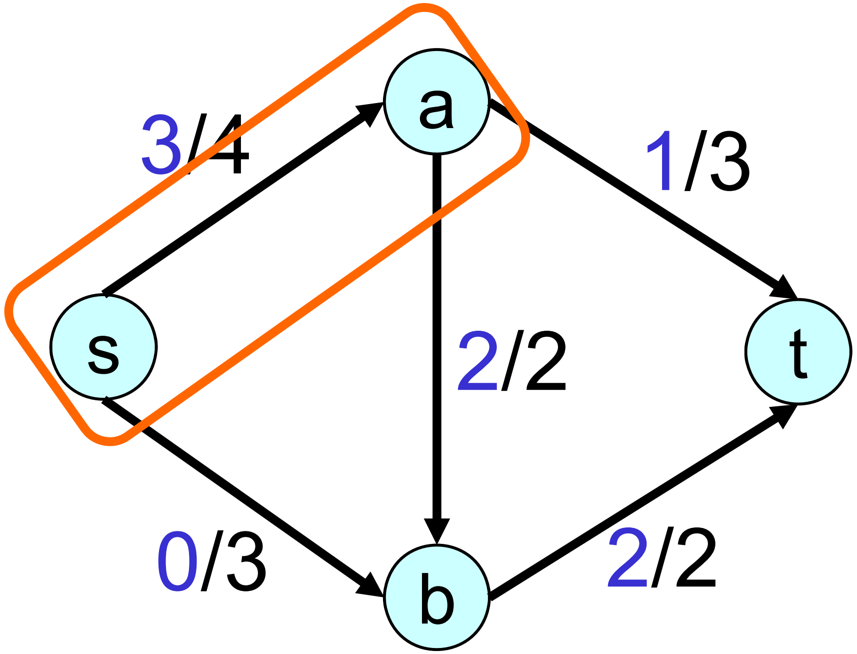
最大フローであることの判定を

効率よく行うには？

⇒　残余ネットワークを利用

残余ネットワークの定義

残余ネットワークの作り方

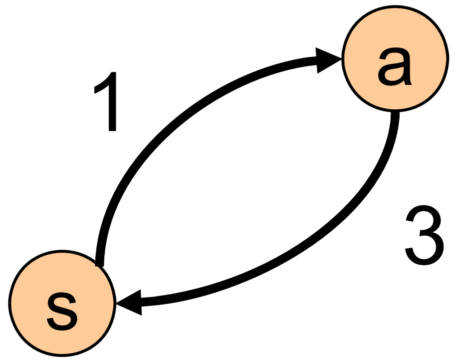


問題例とフロー

各枝のデータは

（フロー量/容量）

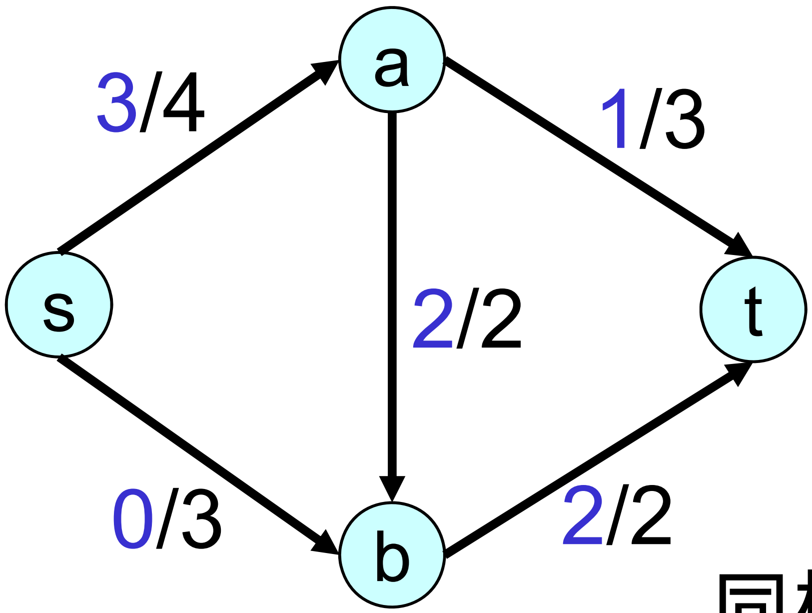
枝(s,a)において
☆さらに4－3＝1だけフロー
を流せる
⇒　残余ネットワークに
容量1の枝(s,a)を加える



☆現在のフロー3を逆流させて
0にすることが出来る
⇒　容量3の枝(a,s)を加える

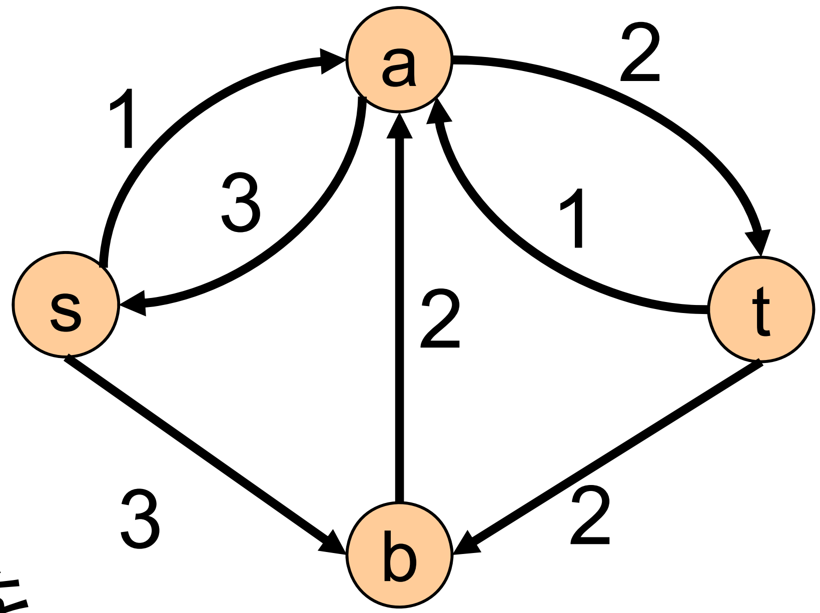
残余ネットワークの定義

残余ネットワークの作り方



問題例とフロー

同様の操作を
各枝に行う



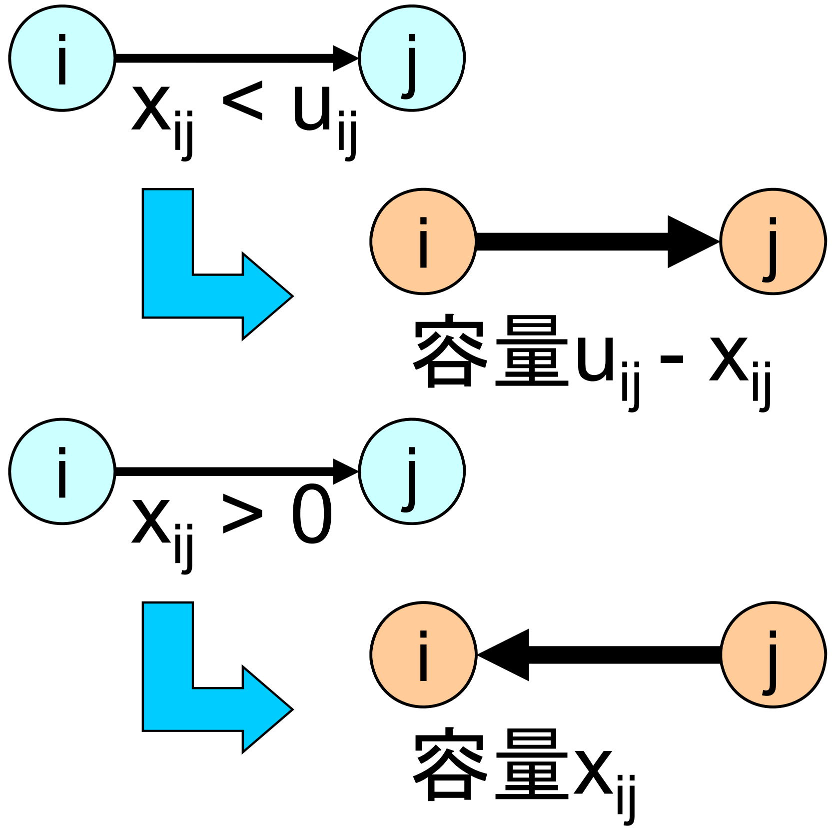
残余ネットワーク
の完成

残余ネットワークの定義(まとめ)

$x = (x_{ij} \mid (i,j) \in E)$: 現在のフロー

フロー x に関する残余ネットワーク $G^x = (V, E^x)$

$E^x = F^x \cup R^x$



順向きの枝集合

$F^x = \{ (i, j) \mid (i, j) \in E, x_{ij} < u_{ij} \}$

各枝の容量 $u^x_{ij} = u_{ij} - x_{ij}$

逆向きの枝集合

$R^x = \{ (j, i) \mid (i, j) \in E, x_{ij} > 0 \}$

各枝の容量 $u^x_{ji} = x_{ij}$

注意! : 現在のフローが変わると残余ネットワークも変わる

残余ネットワークに関する定理

増加路：残余ネットワークでの
　　　　ソース s からシンク t へのパス（路・みち）

定理１：残余ネットワークに 増加路が存在する
　　　　➔ 現在のフローの総流量は増加可能

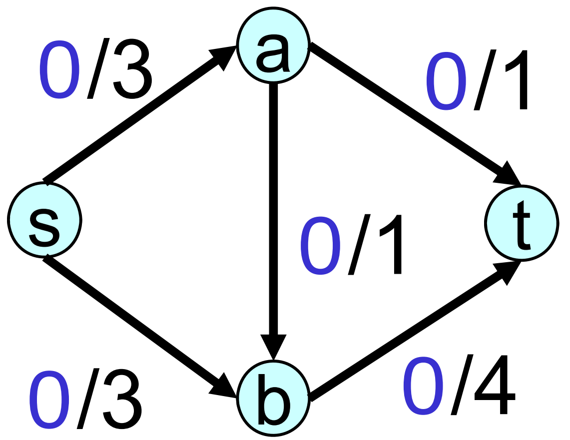
定理２：残余ネットワークに 増加路が存在しない
　　　　➔ 現在のフローは最大フロー

定理1の例

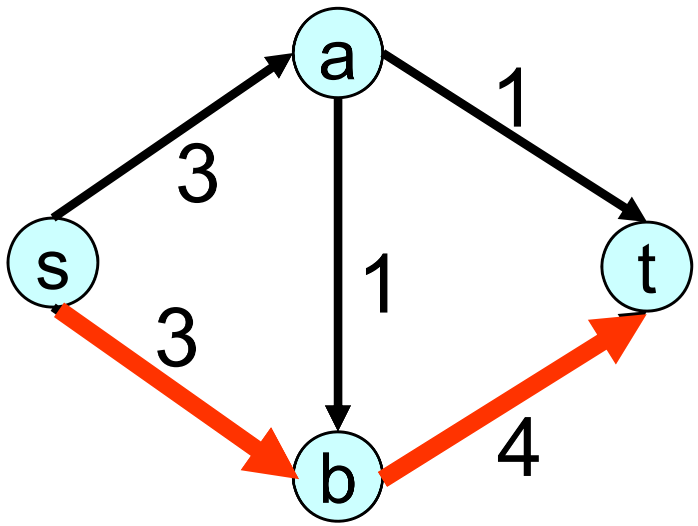
定理１：残余ネットワークに増加路が存在する
➔ 現在のフローの総流量は増加可能

証明: 増加路(s-tパス)を使うと，本当に総流量を増加できる

現在のフローx

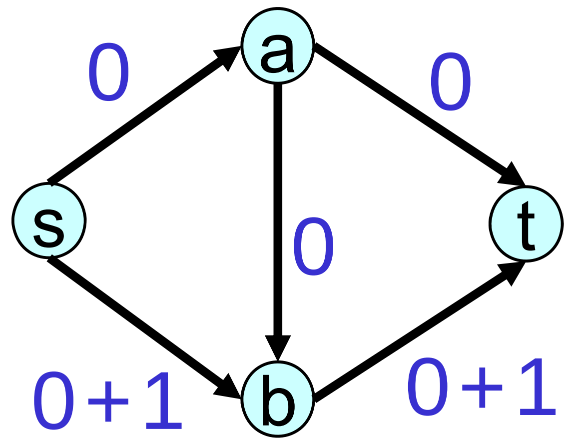


残余ネットワーク



増加路が存在

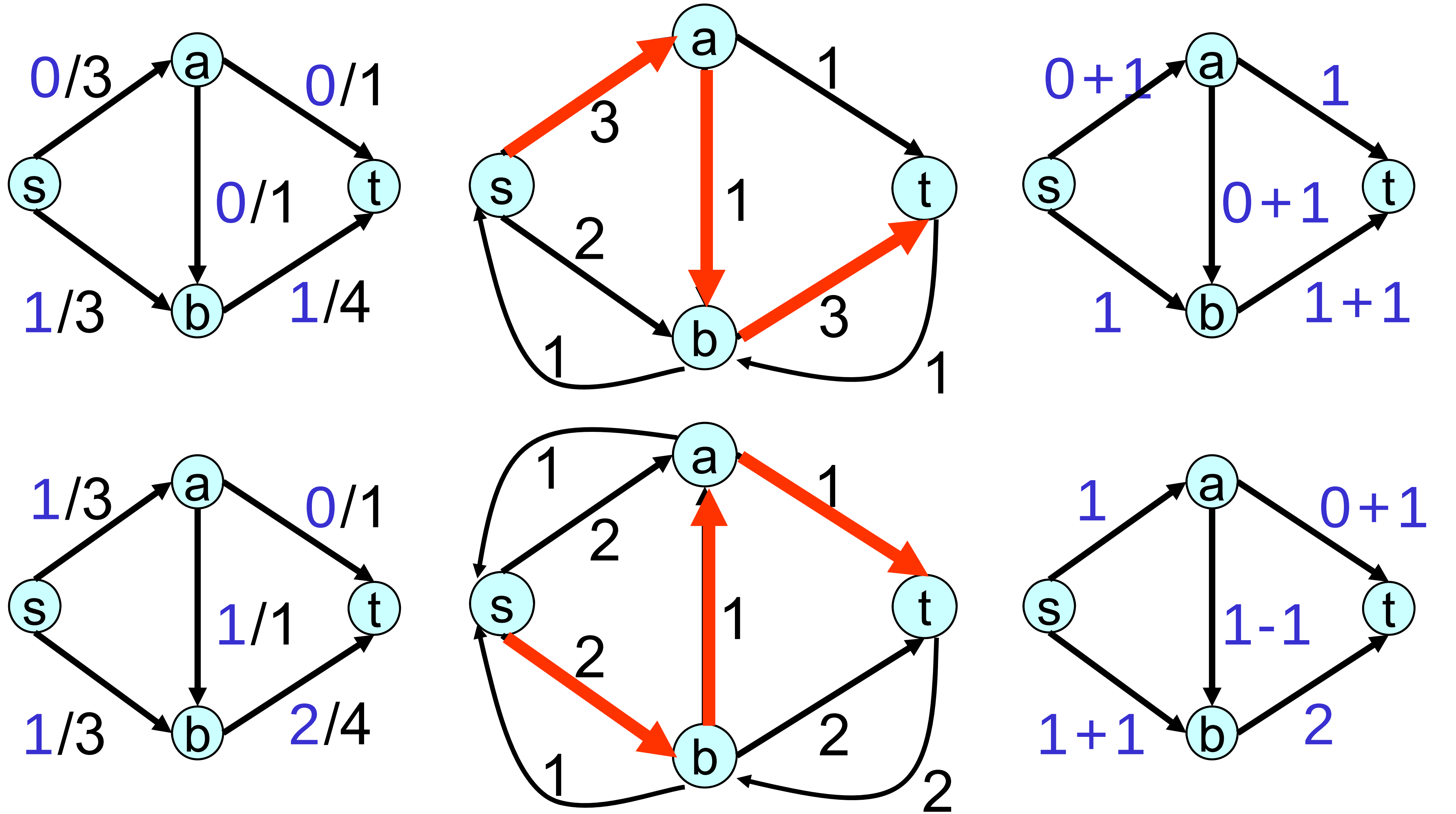
新しいフローx’



総流量が
１増えた

定理1の例

現在のフローx　　残余ネットワーク　　新しいフローx’

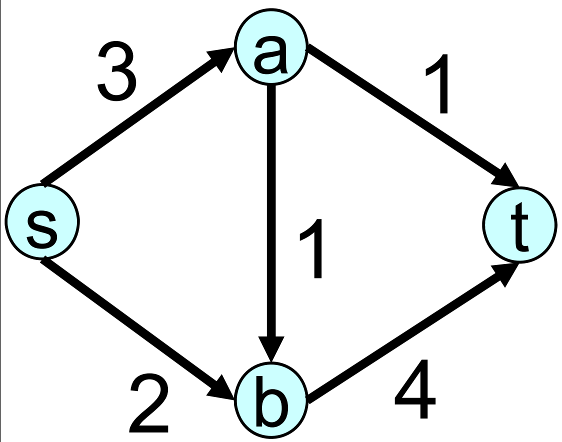


定理2の例

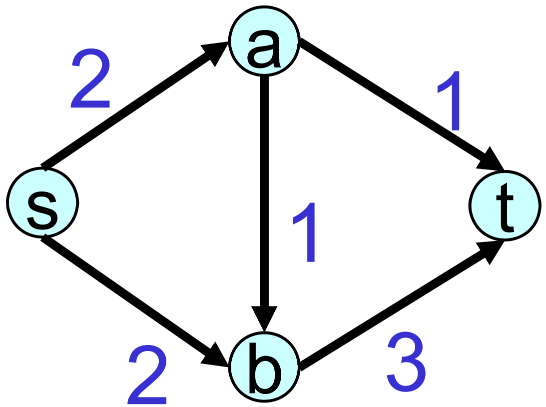
定理2：残余ネットワークに s-t パスが存在しない
➔ 現在のフローは最大フロー

証明は次回

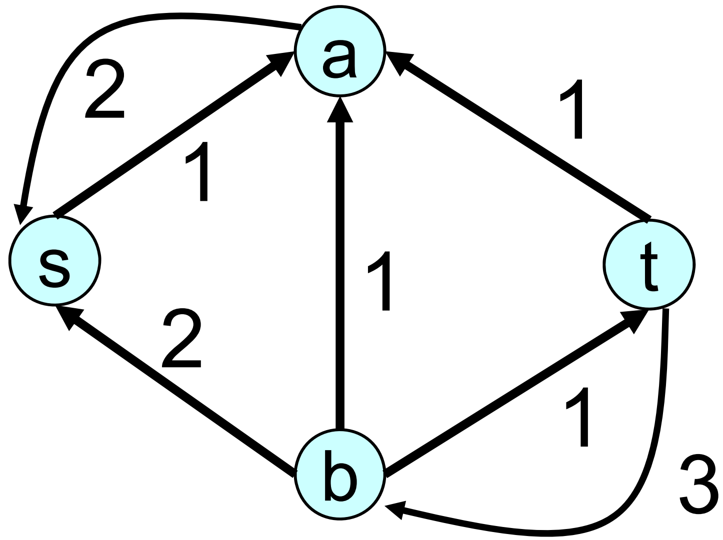
与えられた問題



現在のフロー



残余ネットワーク



s-t パスがない
➔現在のフローは最適！

増加路アルゴリズム

最大フローを求めるアルゴリズム

ステップ0：初期の実行可能フローとして，
全ての枝のフロー量を0とする

ステップ1：現在のフローに関する残余ネットワークを作る

ステップ2：残余ネットワークに 増加路が存在しない ⇒ 終了

ステップ3：残余ネットワークの増加路をひとつ求め，
それを用いて現在のフローを更新する

ステップ4：ステップ1へ戻る

増加路アルゴリズムの計算時間

※各枝の容量は整数と仮定
　U = 容量の最大値
　m = 枝の数，n = 頂点の数

各反復において総流量が1以上増加
➔反復回数 ≦ 総流量の最大値 ≦ m U

各反復での計算時間
＝ 残余ネットワークの増加路を求める時間
➔深さ優先探索，幅優先探索などを使うと O(m + n) 時間

∴ 計算時間は O((m+n) m U)
（入力サイズは m + n + log U なので，指数時間）

増加路アルゴリズムの改良

反復回数を少なくしたい

➔ 各反復での増加路の選び方を工夫する

（改良法１） 各反復での総流量の増加量を大きくする

➔各反復で**容量最大の増加路**を選ぶ

➔反復回数 O(m log (n U)), 計算時間 $O(m^2 \log (n U))$

（改良法2） 各反復で**最短（枝数最小）の増加路**を選ぶ

➔反復回数 O(m n), 計算時間 $O(m^2 n)$

※この他にも，増加路アルゴリズムの計算時間を短縮するための様々なテクニックが存在

全く違うアイディアのアルゴリズム：「プリフロー」を利用

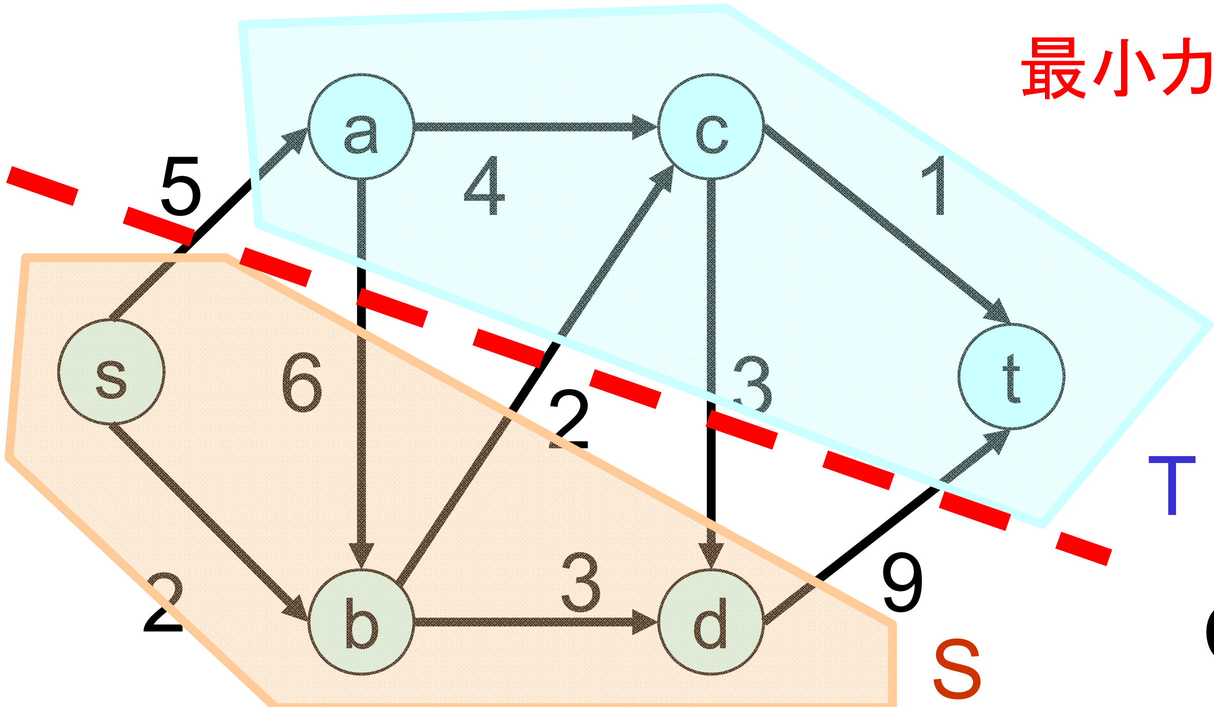
カット

フローを流すとき，ネットワークのボトルネックはどこ？

カット (S, T): S, T は頂点集合Vの分割（$S \cap T = \emptyset, S \cup T = V$）
S はソース s を含む，Tはシンク t を含む

カット (S, T) の容量C(S,T) ＝ SからTへ向かう枝の容量の和

最小カット：容量が最小のカット



C(S,T)=5+2+9=16

カットの性質(その1)

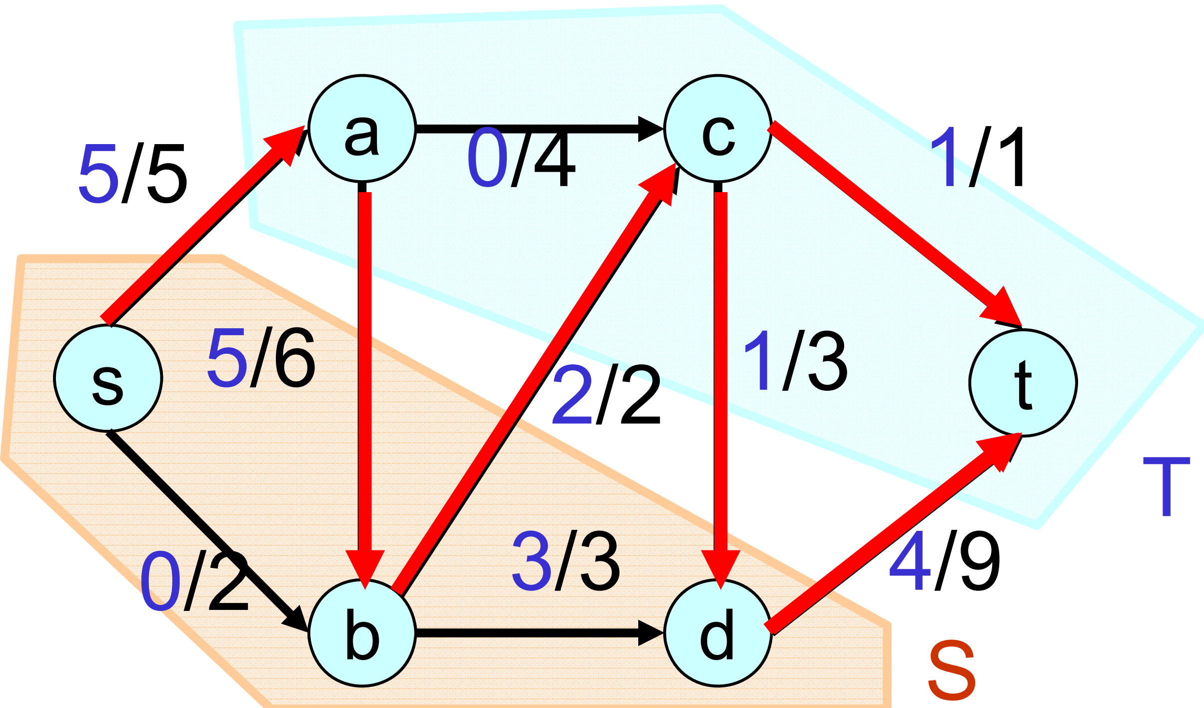
性質1：

任意のカット(S, T) と任意の実行可能フロー (x_{ij} | (i,j) ∈ E) に対し

SからTへの枝のフローの和 x(S,T)

ー TからSへの枝のフローの和 x(T,S)

＝ フローの総流量 f



$f = 1 + 4 = 5$

$x(S, T) = 5 + 2 + 4 = 11$

$x(T, S) = 5 + 1 = 6$

$f = 11 – 6 = 5$

レポート問題

問1：次の2つの最大流問題に対する定式化を書きなさい

問2：次の2つの最大流問題に対して，増加路アルゴリズムで最大流を求めよ（各反復での残余ネットワークやフローを省略せずに書くこと）

問3：2つのグラフの最小カット（と思われるカット）を求めよ（頑張って探してみてください）

提出締切：次回講義