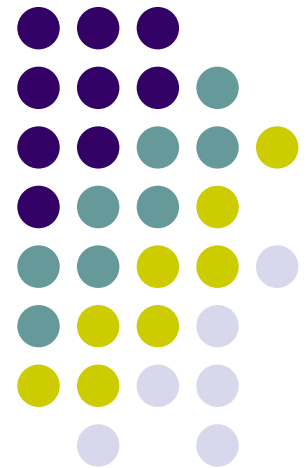


数理計画法 第8回

ネットワーク最適化

ネットワーク最適化問題とは？
最大フロー問題

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





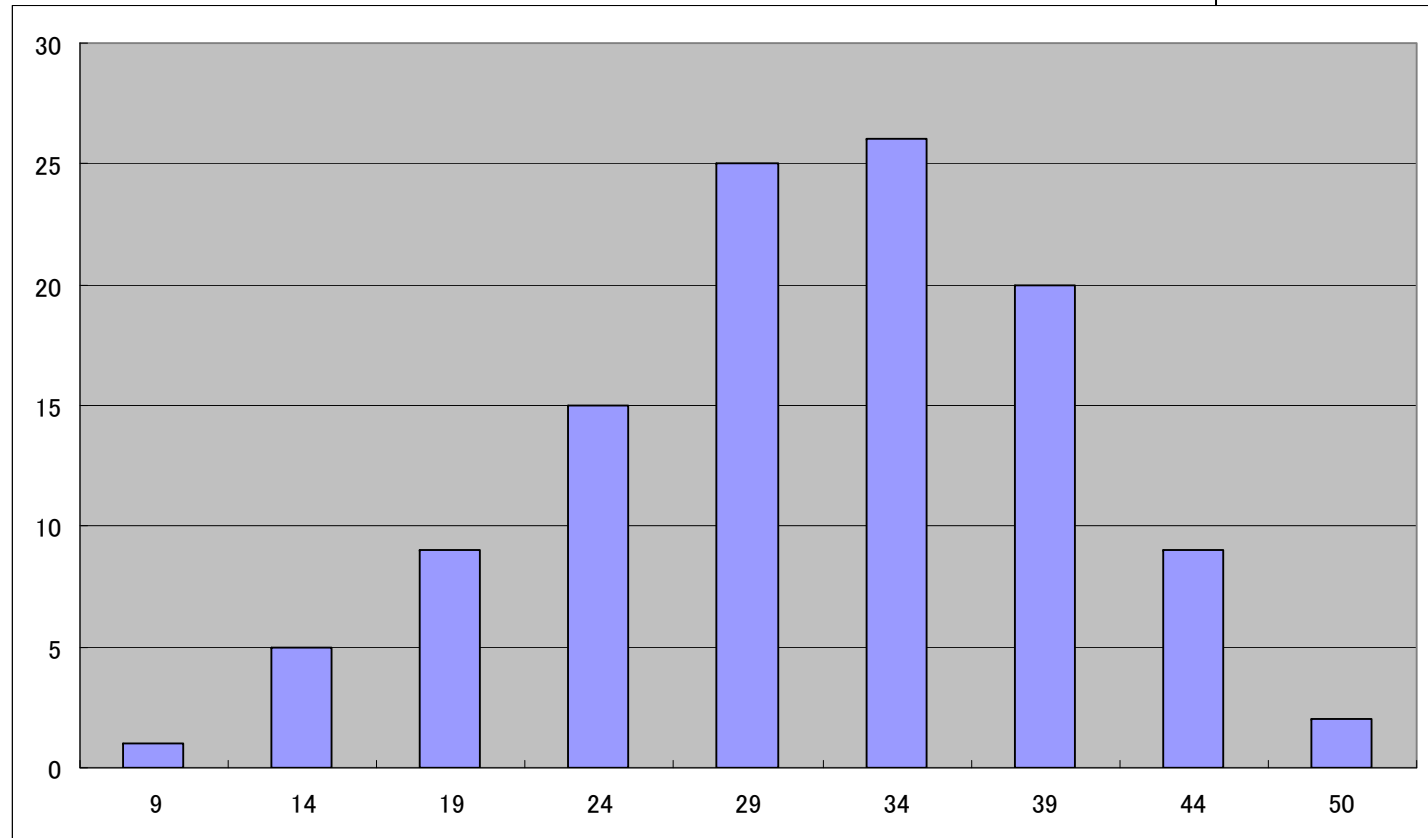
中間試験の結果

平均29.0点

最高47.5点

合格:80人

不合格:32人



試験結果に関する問い合わせ、質問はメールにて。

もしくは研究室に直接来てください(ただし、4時15分以降)

グラフとネットワーク

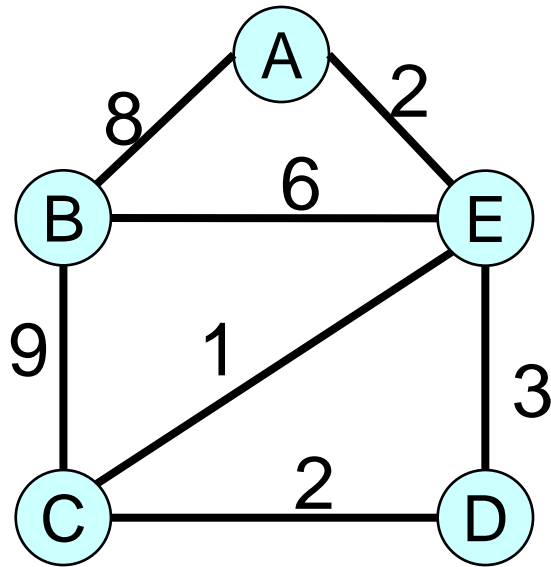


★(無向、有向)グラフ

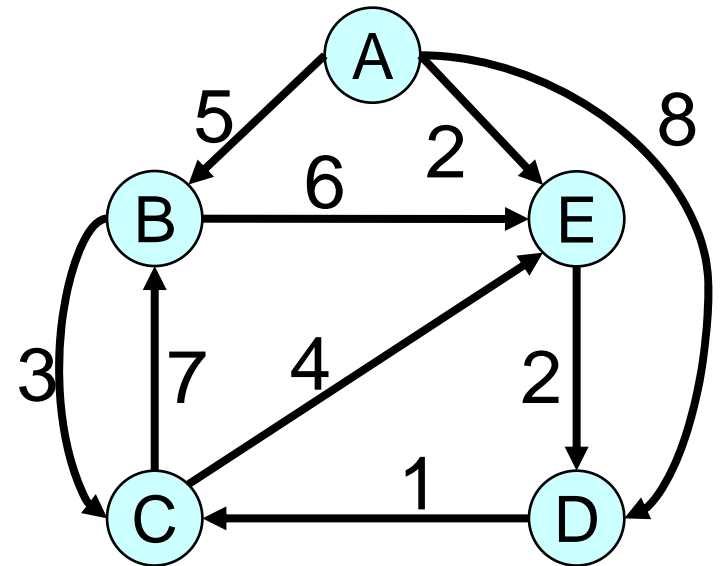
頂点(接点、点)が枝(辺、弧, 線)で結ばれたもの

★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



無向グラフ



有向グラフ

ネットワーク最適化問題



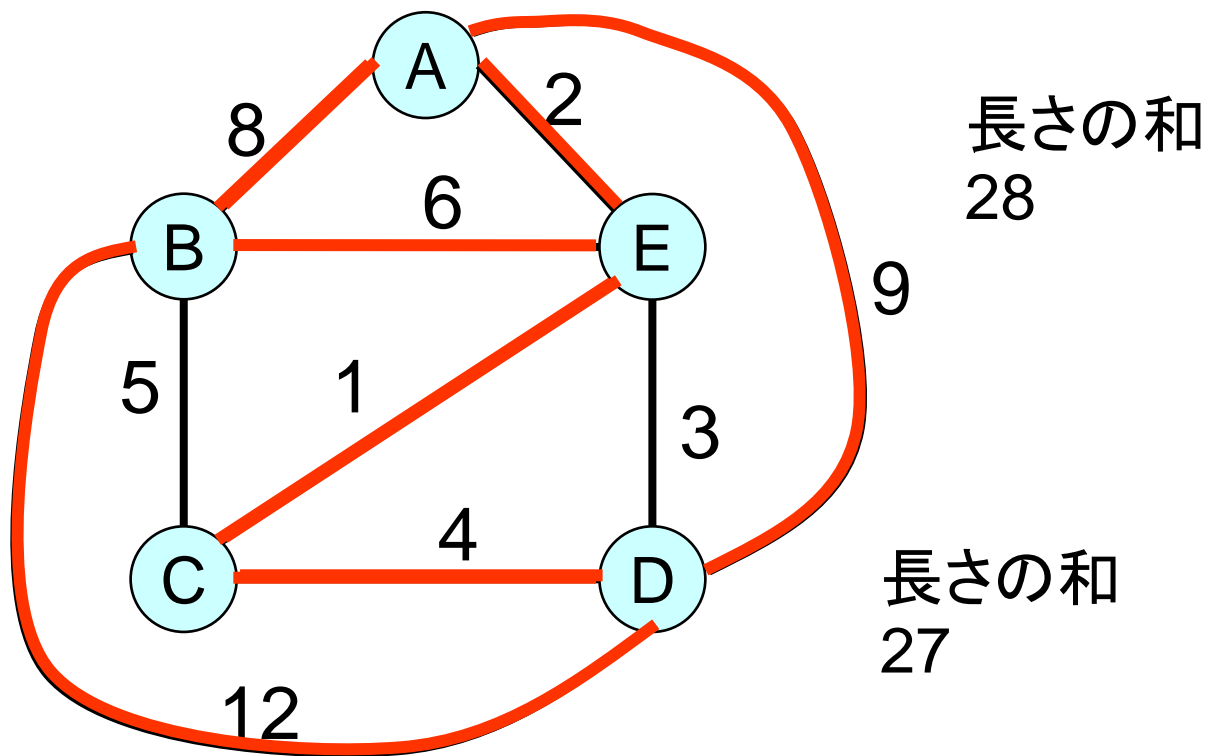
ネットワークに関する数理計画問題(最適化問題)

例: 最小木問題	}	他の講義で扱う 「アルゴリズムとデータ構造」 「情報数学」
最短路問題		
最大フロー問題	}	この授業で扱う
最小費用フロー問題		
巡回セールスマン問題	}	問題のみ紹介

巡回セールスマン問題

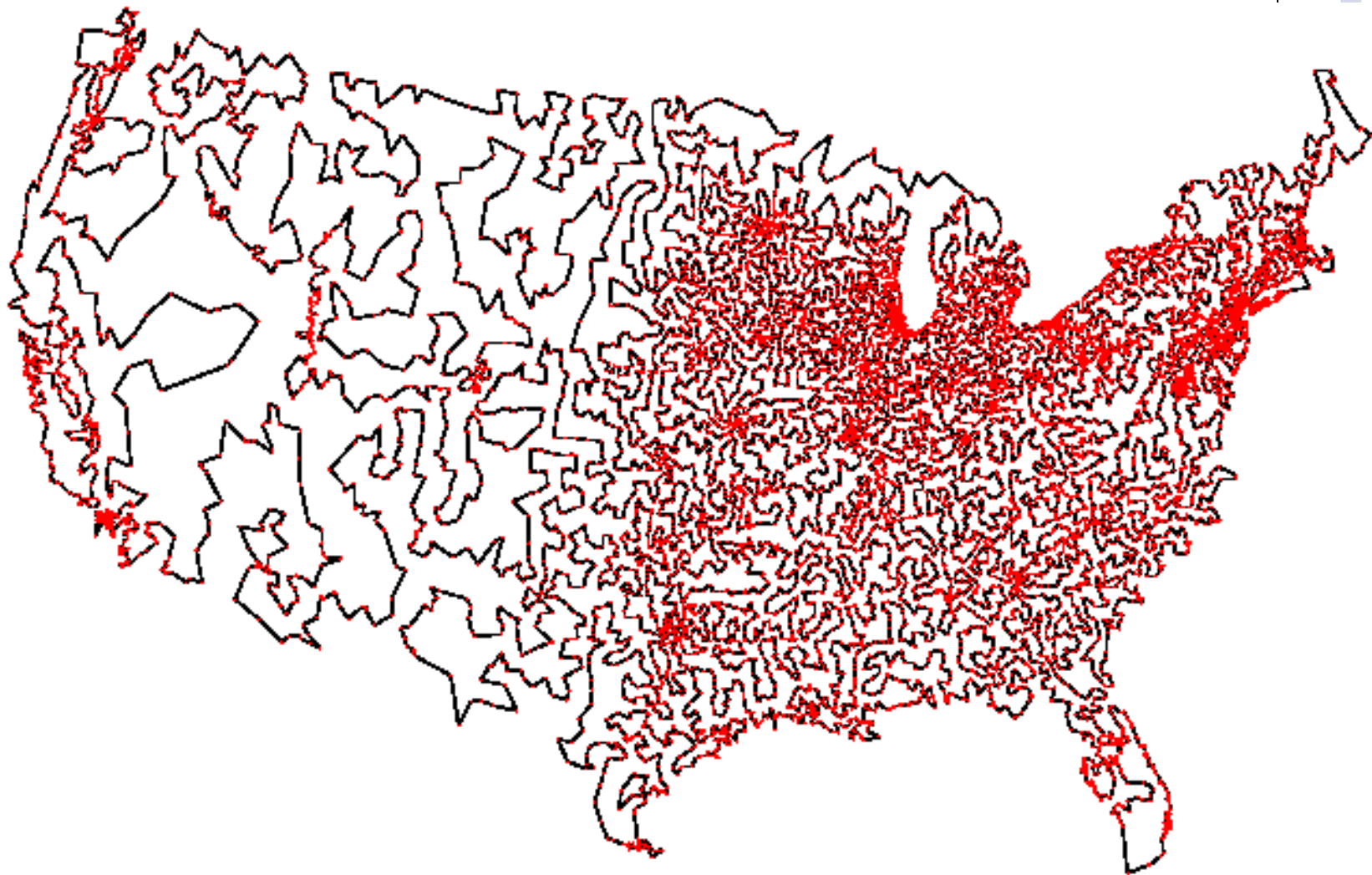


枝の長さの和が最小の巡回路を求める

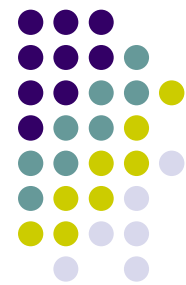
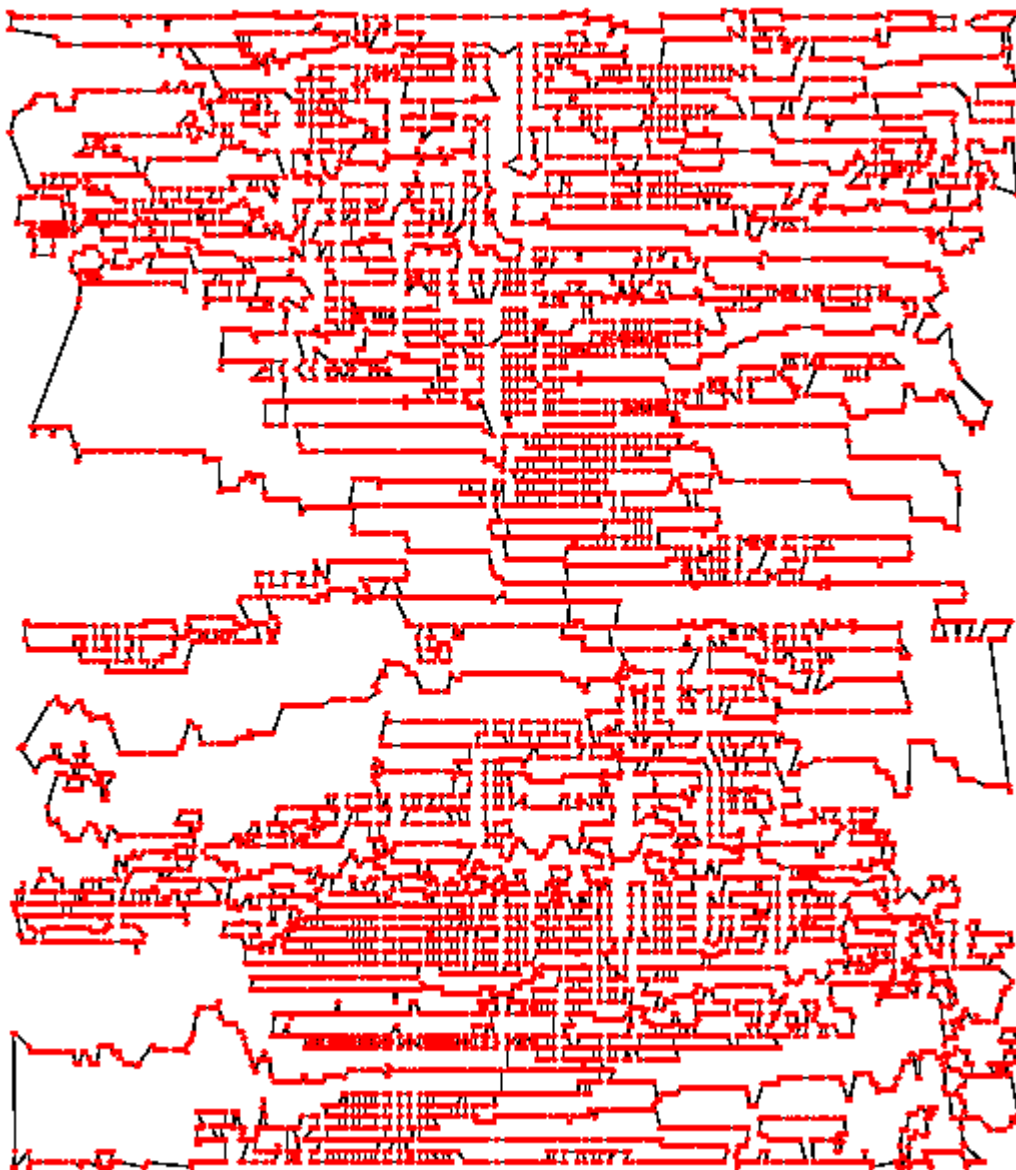


具体例： 運搬経路問題，VLSI設計，基盤配線
ドリルでの穴あけ，など

巡回セールスマン問題



巡回セールスマン問題



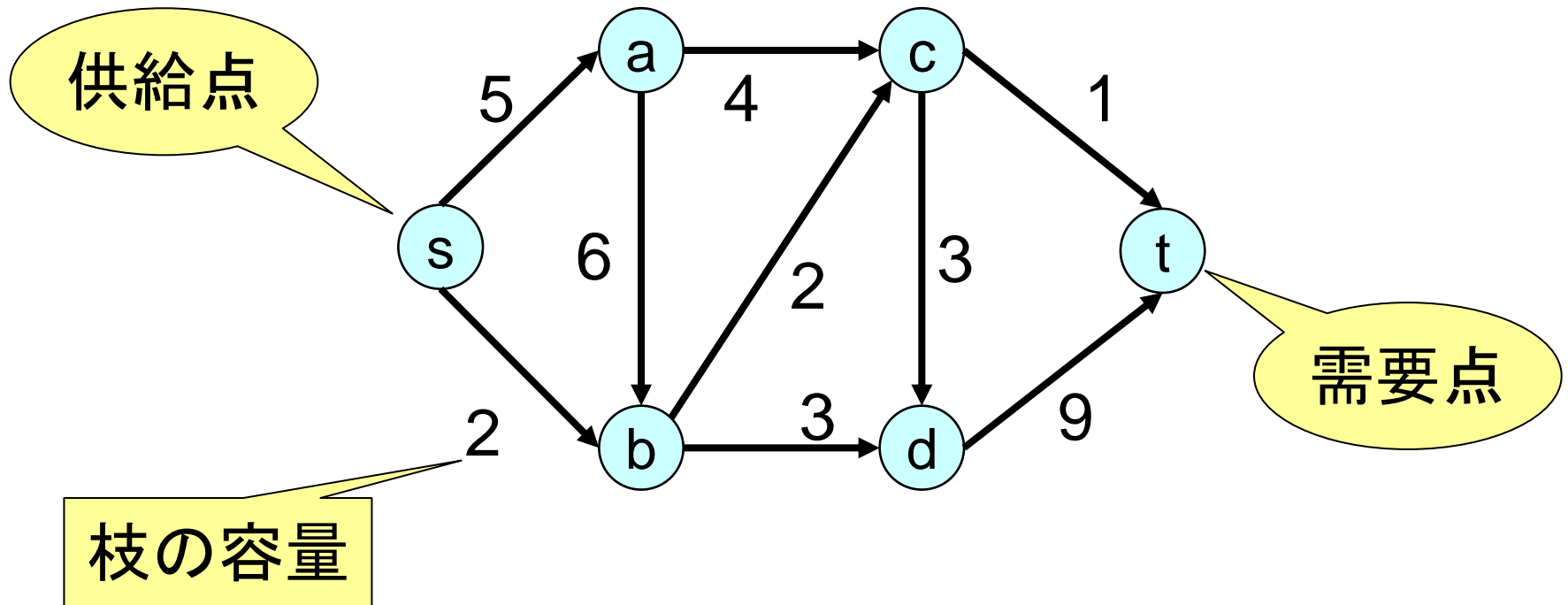
最大フロー問題の定義(その1)



入力: 有向グラフ $G = (V, E)$

供給点 $s \in V$, 需要点 $t \in V$

各枝 $(i, j) \in E$ の容量 $u_{ij} \geq 0$



最大フロー問題の定義(その2)



目的: 供給点から需要点に、

枝と頂点を経由して「もの」をたくさん流したい

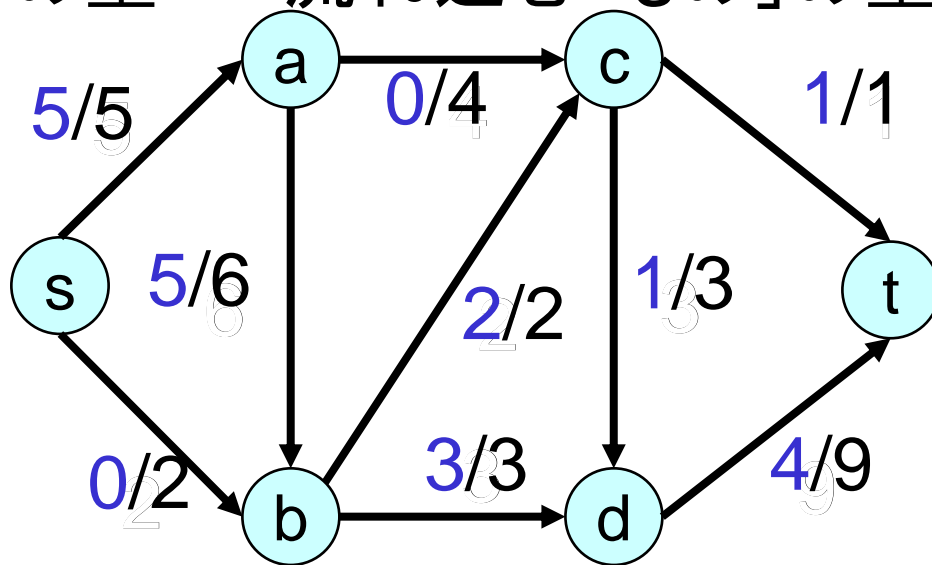
条件1 (容量条件):

$0 \leq$ 各枝を流れる「もの」の量 \leq 枝の容量

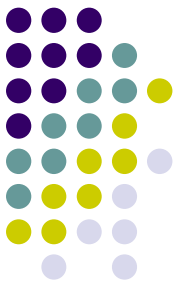
条件2 (流量保存条件):

頂点から流れ出す「もの」の量 = 流れ込む「もの」の量

与えられたネットワーク
と解の一例



最大フロー問題の定式化



変数 x_{ij} : フロー = 枝 (i, j) を流れる「もの」の量

変数 f : フロー量 = 需要点に流れ込む「もの」の量
(= 供給点から流れ出す「もの」の量)

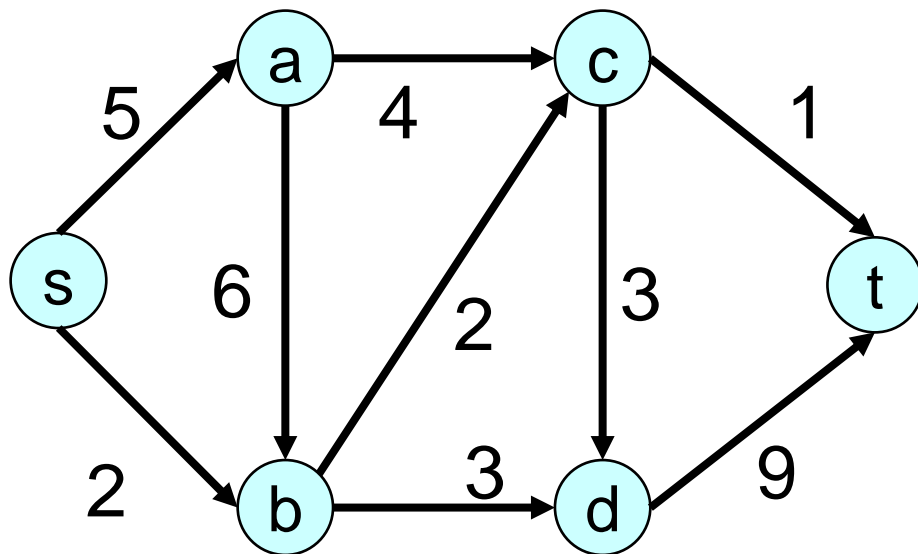
目的: 供給点から需要点に「もの」をたくさん流したい

⇒ 最大化 f

容量条件: $0 \leq$ 各枝を流れる「もの」の量 \leq 枝の容量

⇒ $0 \leq x_{ij} \leq u_{ij} \quad ((i, j) \in E)$

最大フロー問題の定式化: 例



目的: 最大化 f

容量条件:

$$0 \leq x_{sa} \leq 5, 0 \leq x_{sb} \leq 2, 0 \leq x_{ab} \leq 6, 0 \leq x_{ac} \leq 4, 0 \leq x_{bc} \leq 2,$$

...

最大フロー問題の定式化(その2)



流量保存条件:

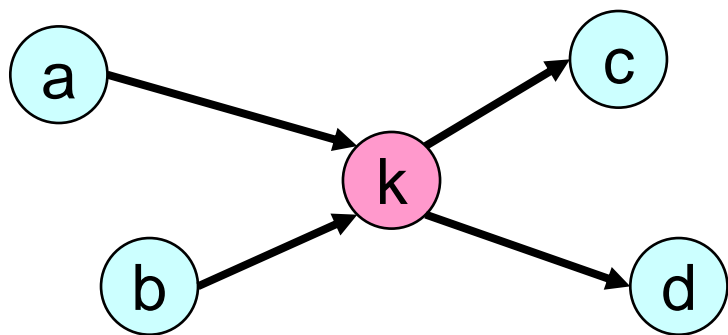
(頂点から流れ出す「もの」の量)

$$- (\text{流れ込む「もの」の量}) = 0$$

⇒ $\sum \{x_{kj} \mid \text{枝 } (k,j) \text{ は頂点 } k \text{ から出る}\}$

$$- \sum \{x_{ik} \mid \text{枝 } (i,k) \text{ は頂点 } k \text{ に入る}\} = 0$$

$$(k \in V - \{s, t\})$$



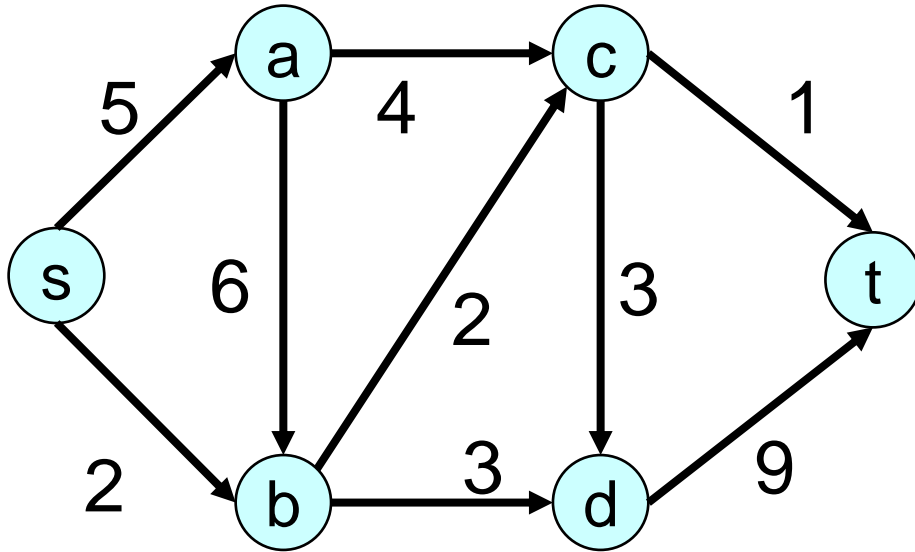
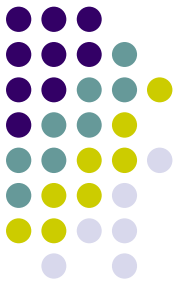
$$(x_{kc} + x_{kd}) - (x_{ak} + x_{bk}) = 0$$

供給点と需要点に関する条件:

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

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

最大フロー問題の定式化: 例



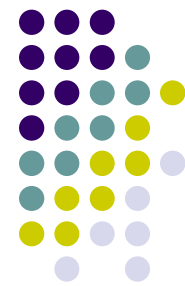
流量保存条件:

$$X_{sa} + X_{sb} = f, \quad -X_{ct} - X_{dt} = -f,$$

$$X_{ac} + X_{ab} - X_{as} = 0, \quad X_{bc} + X_{bd} - X_{ab} - X_{sb} = 0,$$

$$X_{ct} + X_{cd} - X_{ac} - X_{cb} = 0, \quad X_{dt} - X_{cd} - X_{bd} = 0$$

最大フロー問題の定式化(その3)



定式化をまとめると

最大化 f

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

$$\sum \{x_{kj} \mid (k,j) \text{ は } k \text{ から出る}\}$$

$$- \sum \{x_{ik} \mid (i,k) \text{ は } k \text{ に入る}\} = 0 \quad (k \in V - \{s, t\})$$

$$\sum \{x_{sj} \mid (s,j) \text{ は } s \text{ から出る}\}$$

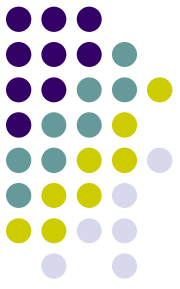
$$- \sum \{x_{is} \mid (i,s) \text{ は } s \text{ に入る}\} = f$$

$$\sum \{x_{tj} \mid (t,j) \text{ は } t \text{ から出る}\}$$

$$- \sum \{x_{it} \mid (i,t) \text{ は } t \text{ に入る}\} = -f$$

この問題の許容解 x_{ij} — フロー
フローの目的関数値 f — フロー値

最大フロー問題の解法



最大フロー問題は線形計画問題の特殊ケース

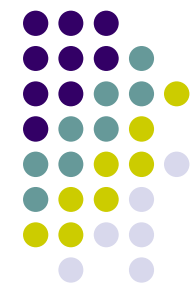
⇒ 単体法で解くことが可能！

最大フロー問題は良い(数学的な)構造をもつ

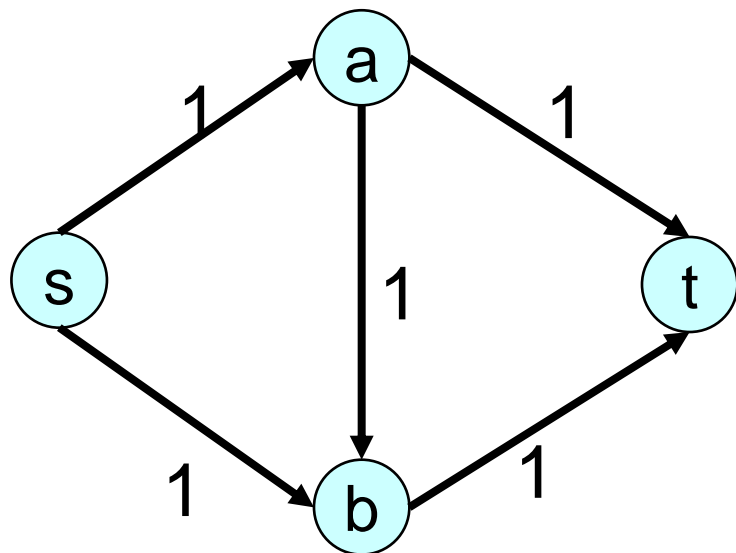
⇒ この問題専用の解法(フロー増加法など)

により、より簡単に、より高速に解くことが可能

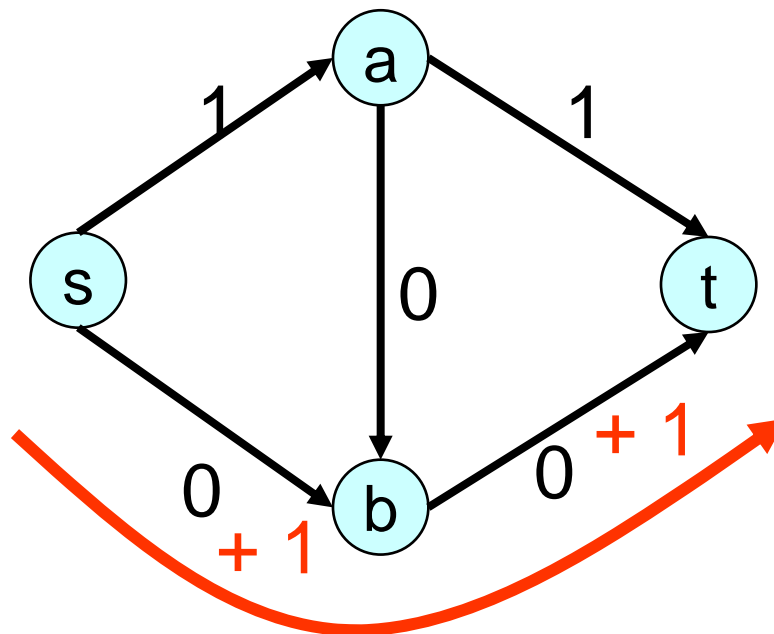
最大フローの判定



問題の例



フローの例1: **最大?**

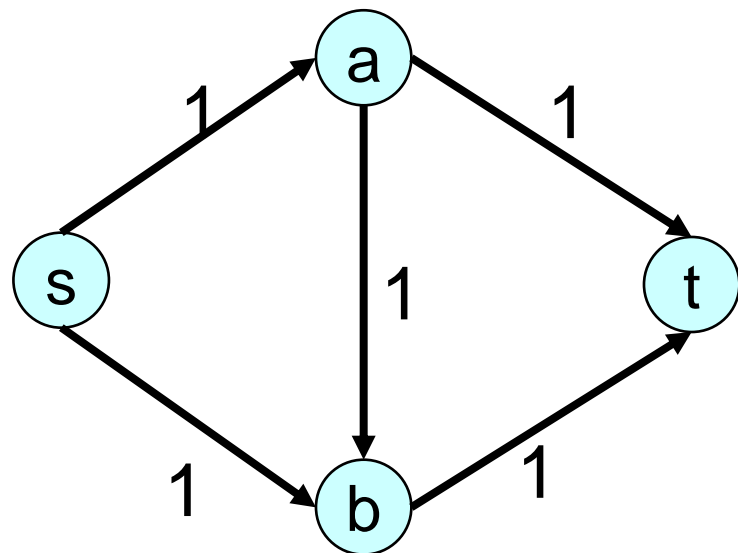


最大フローではない

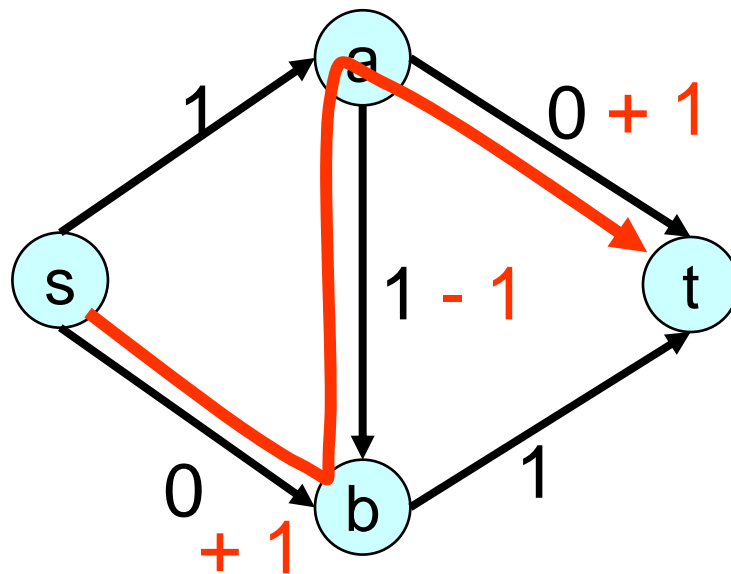
最大フローの判定



問題の例



フローの例2: 最大?



最大フローではない

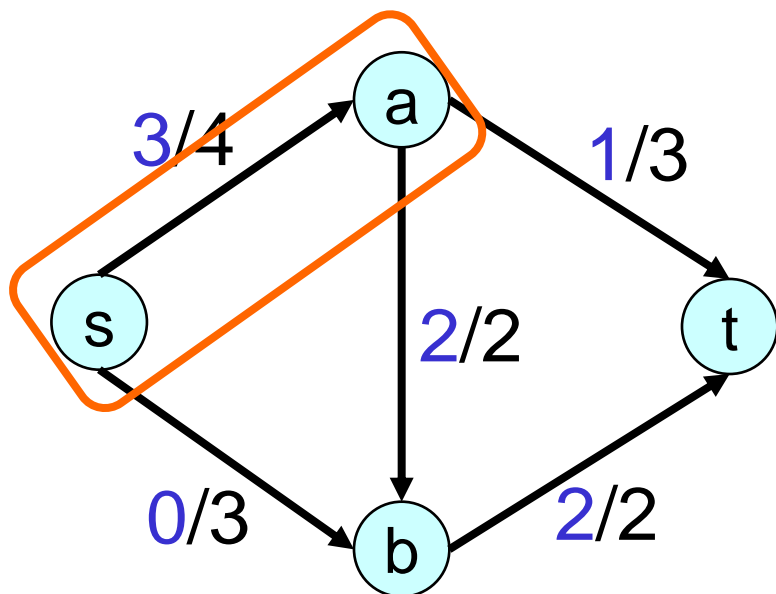
最大フローであることの判定を効率よく行うには？

⇒ 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

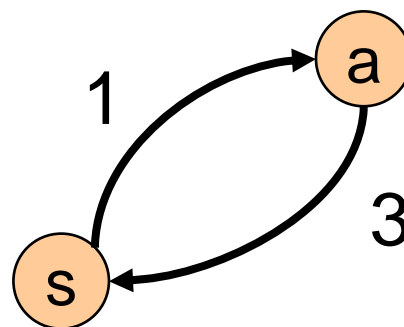


問題例とフロー
各枝のデータは
(**フロー量**/容量)

枝 (s,a) において

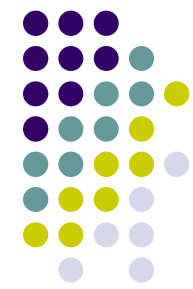
☆さらに $4 - 3 = 1$ だけフロー
を流せる

⇒ 残余ネットワークに
容量1の枝 (s,a) を加える

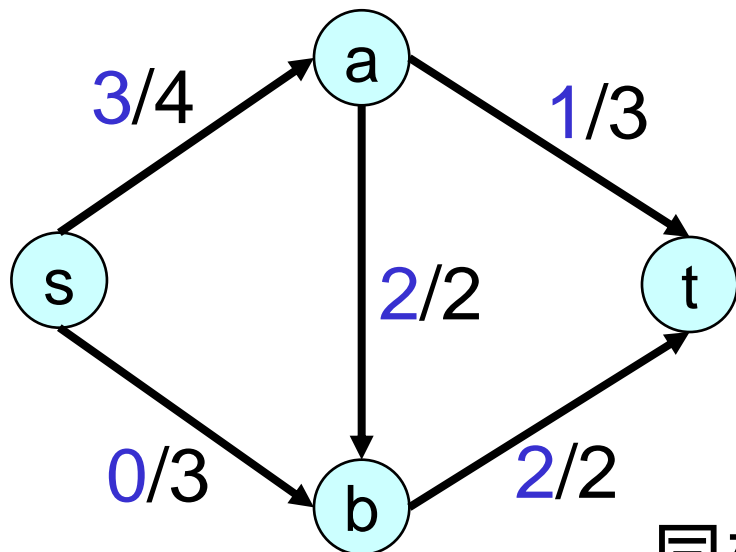


☆現在のフロー3を逆流させて
0にすることが出来る
⇒ 容量3の枝 (a,s) を加える

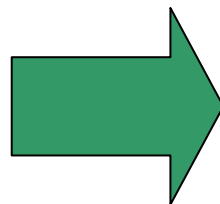
残余ネットワークの定義



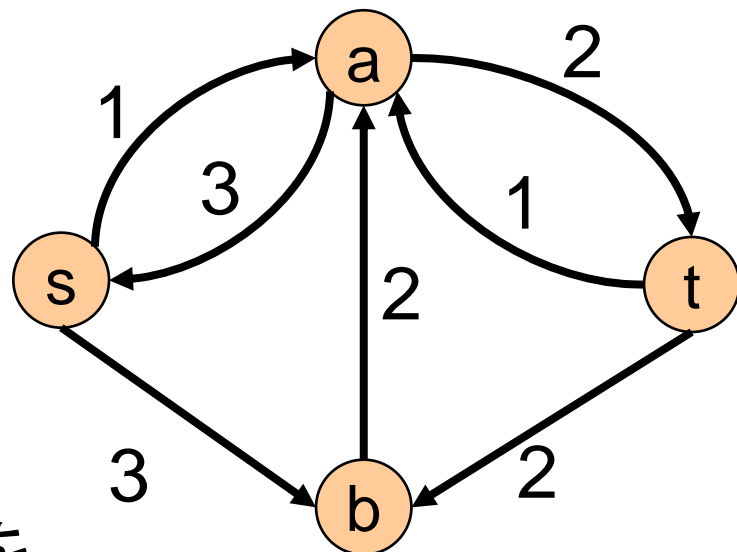
残余ネットワークの作り方



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

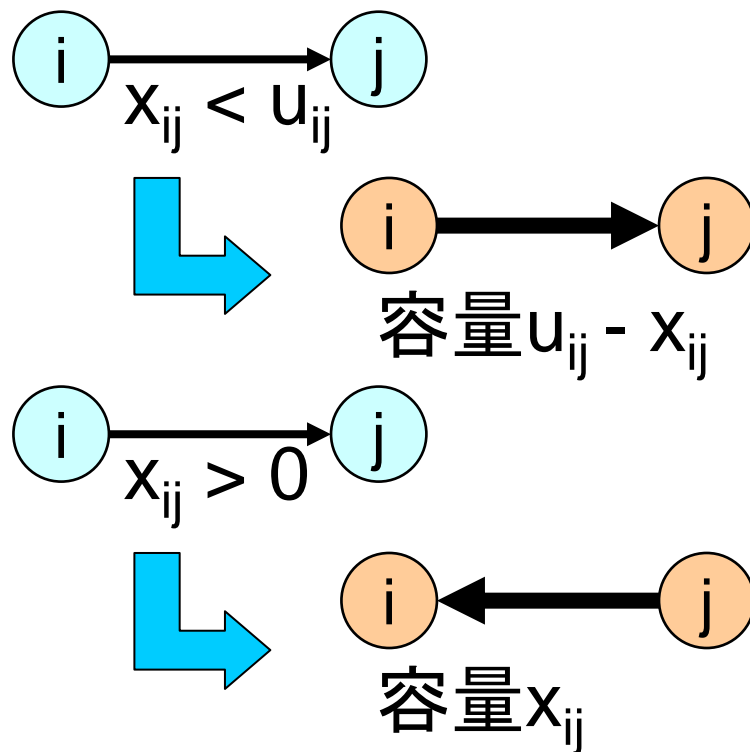
$$F^x = \{ (i, j) \mid (i, j) \in E, x_{ij} < u_{ij} \}$$

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

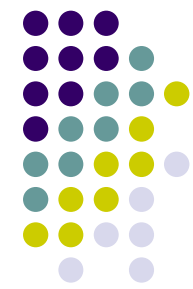


定理 1 : 残余ネットワークに s - t パスが存在する
→ 現在のフローは増加可能

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

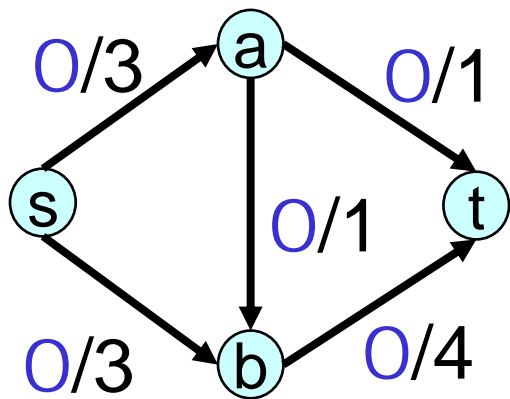
証明は次回

定理1の例

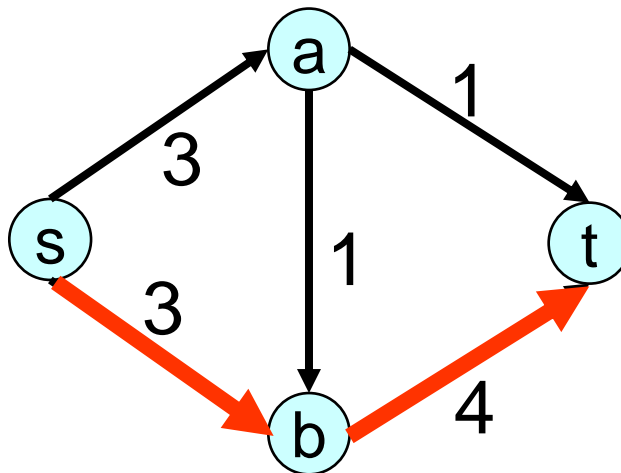


定理 1 : 残余ネットワークに s-t パスが存在する
→ 現在のフローは増加可能

与えられた問題と
現在のフロー x

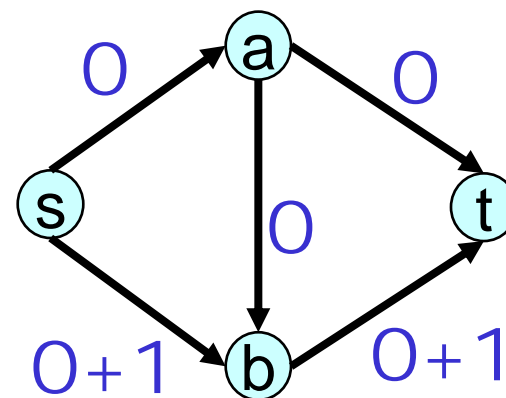


残余ネットワーク



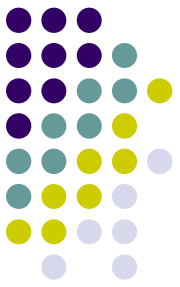
s-t パスが存在

新しいフロー x'



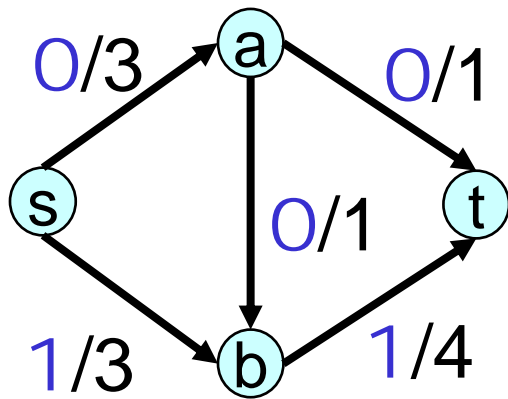
フロー値が
1 増えた

定理1の例

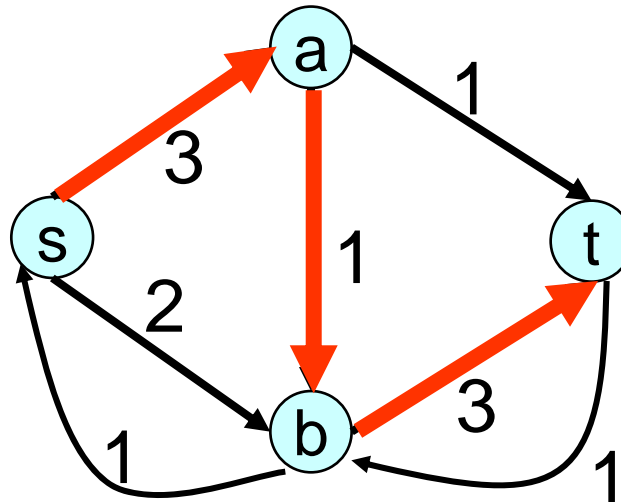


定理 1 : 残余ネットワークに s-t パスが存在する
→ 現在のフローは増加可能

与えられた問題と
現在のフロー x

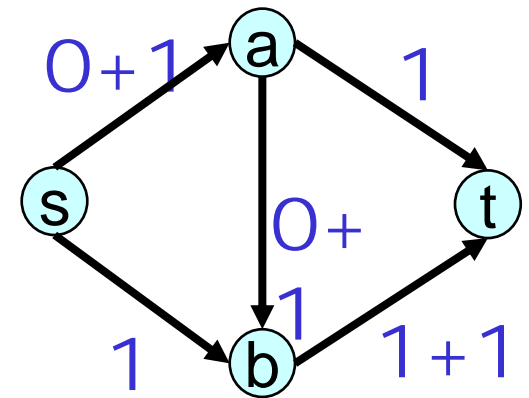


残余ネットワーク



s-t パスが存在

新しいフロー x'



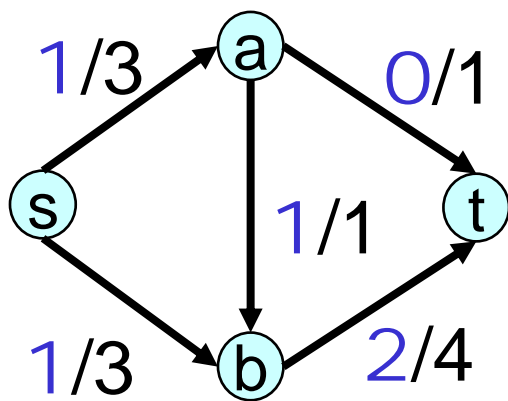
フロー値が
1 増えた

定理1の例

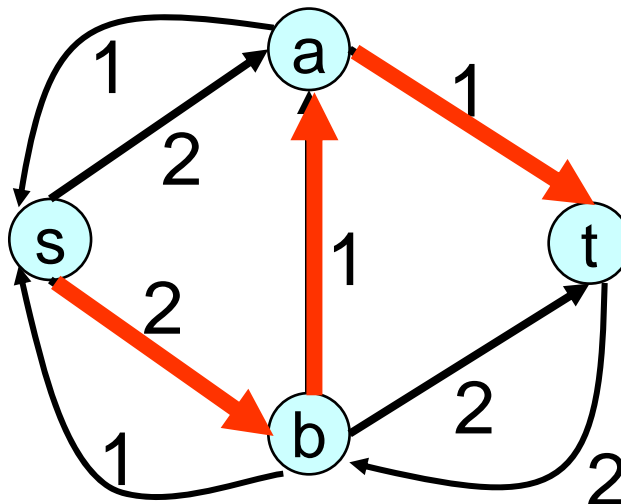


定理 1 : 残余ネットワークに s-t パスが存在する
→ 現在のフローは増加可能

与えられた問題と
現在のフロー x

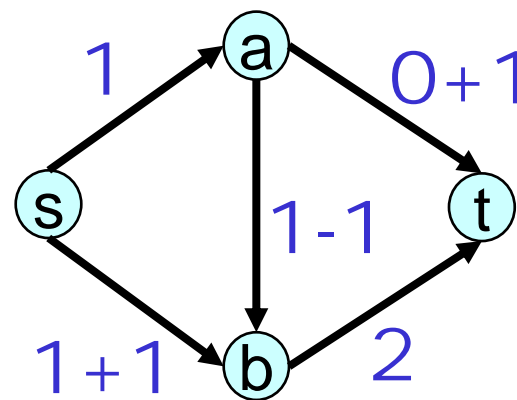


残余ネットワーク



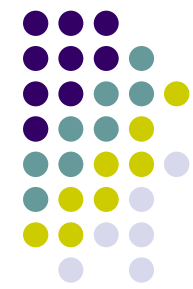
s-t パスが存在

新しいフロー x'



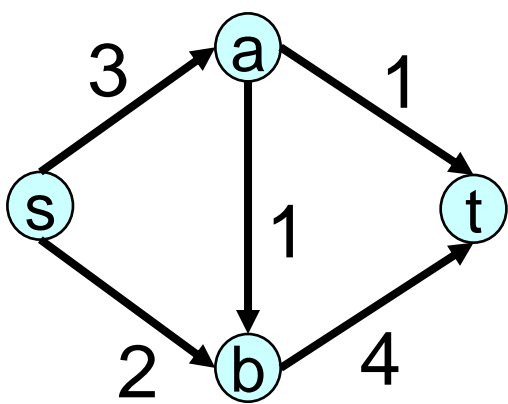
フロー値が
1増えた

定理2の例

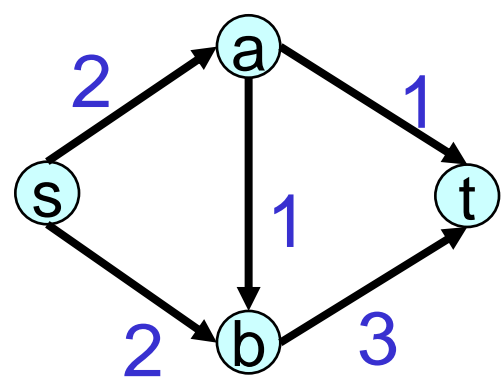


定理2: 残余ネットワークに s-t パスが存在しない
→ 現在のフローは最大フロー

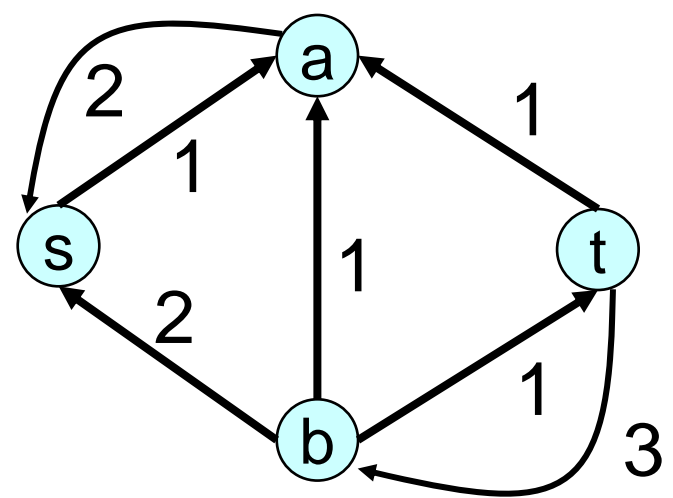
与えられた問題



現在のフロー

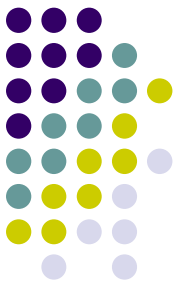


残余ネットワーク



s-t パスがない
→ 現在のフローは最適！

フロー増加法



最大フローを求めるためのアルゴリズム

ステップ0: 初期フローとして、全ての枝のフロー量を
0とする

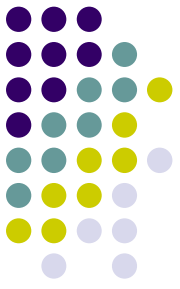
ステップ1: 現在のフローに関する残余ネットワークを作る

ステップ2: 残余ネットワークに s-t パスが存在しない
⇒ 終了

ステップ3: 残余ネットワークの s-t パスをひとつ求め、
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

フロー増加法の計算時間



※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

各反復においてフローが1以上増加

→ 反復回数 \leq 最大フロー量 $\leq m U$

各反復での計算時間

= 残余ネットワークのs-tパスを求める時間

→ 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**)

フロー増加法の改良



フロー増加法の反復回数を少なくしたい

→ 各反復での $s-t$ パスの選び方を工夫する

(改良法1) 各反復でのフロー増加量を大きくする

→ 各反復で**容量最大の $s-t$ パス**を選ぶ

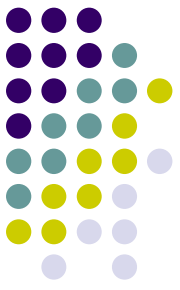
→ 反復回数 $O(m \log (n U))$, 計算時間 $O(m^2 \log (n U))$

(改良法2) 各反復で**最短の $s-t$ パス**を選ぶ

→ 反復回数 $O(m n)$, 計算時間 $O(m^2 n)$

※この他にも、フロー増加法の計算時間を短縮するための様々なテクニックが存在する

レポート問題



問1: 次の2つの最大フロー問題に対する定式化を書きなさい

問2: 次の2つの最大フロー問題に対して、フロー増加法で最大フローを求めよ(各反復での残余ネットワークやフローも省略せずに書くこと)

締切: 次回講義の開始10分後まで

