数理計画法　第7回

ネットワーク計画

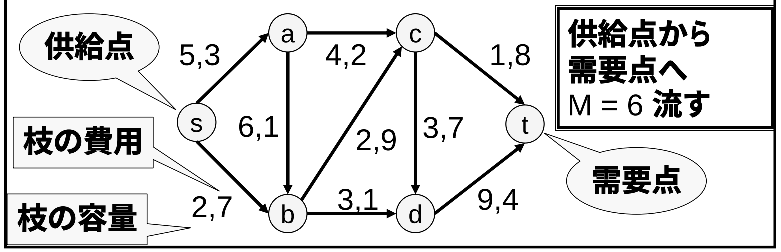
3．最小費用フロー問題

非線形計画

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

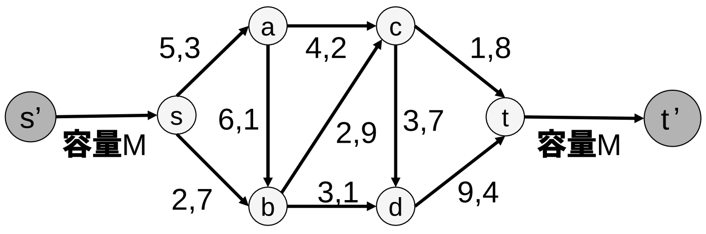
復習：最小費用フロー問題の定義

入力：有向グラフ G = (V, E)
　　供給点 s ∈ V, 需要点 t ∈ V,
　　需要 (供給) 量M > 0
　　各枝 (i, j) ∈ V の容量 u_{ij} ≧ 0, 費用 c_{ij}
出力：需要供給を満たすフローで総費用が最小のもの



復習：需要供給を満たすフローの求め方

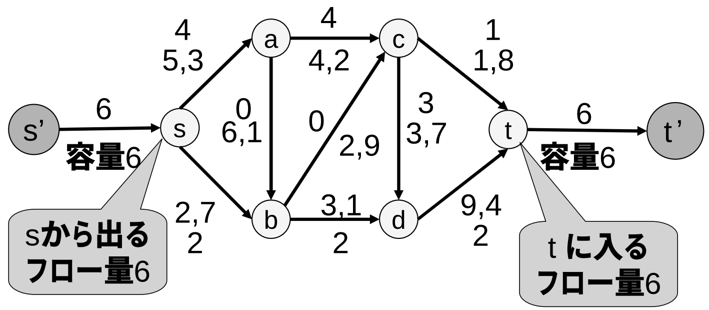
(1) 人工問題として最大フロー問題を作る
(2) 人工問題の最大フローにおいて
　f = M ⇒ 現在のフローは需要供給を満たす
　f < M ⇒ 需要供給を満たすフローは存在しない



各枝の費用は無視

復習：需要供給を満たすフローの求め方

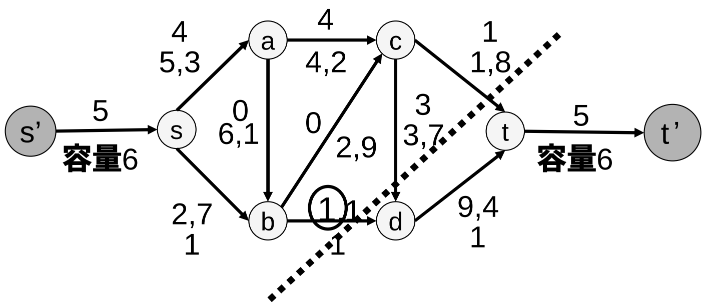
人工問題の最大フローを求める



t’ への供給量 ＝M ⇒ 需要供給を満たすフローが得られた

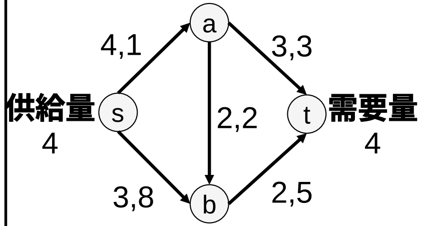
復習：需要供給を満たすフローの求め方

人工問題の最大フローを求める



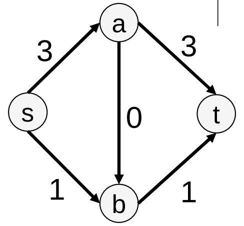
t' **への供給量** <M
⇒ **需要供給を満たすフローは存在しない (証拠：最小カット)**

フローの最適性判定



問題例

フローの例



このフローの費用は25
最小か？

どうやって最小費用フローであることを判定するか？
――― 残余ネットワークの利用

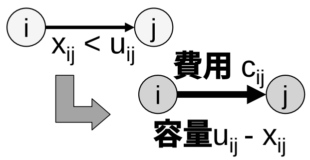
残余ネットワークの作り方 (その1)

最大フローのときとほとんど同じ作り方

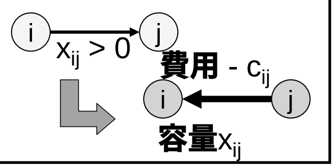
$x = (x_{ij} \mid (i,j) \in E)$: **現在のフロー**

フロー x **に関する残余ネットワーク** $G^x = (V, E^x)$
$E^x = F^x \cup R^x$

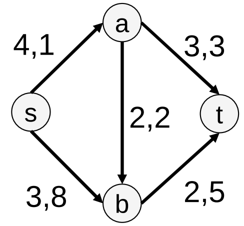
順向きの枝集合
$F^x = \{ (i, j) \mid (i, j) \in E, x_{ij} < u_{ij} \}$
容量 $u^x_{ij} = u_{ij} - x_{ij}$, **費用** c_{ij}



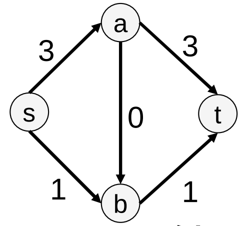
逆向きの枝集合
$R^x = \{ (j, i) \mid (i, j) \in E, x_{ij} > 0 \}$
容量 $u^x_{ji} = x_{ij}$, **費用** $- c_{ij}$



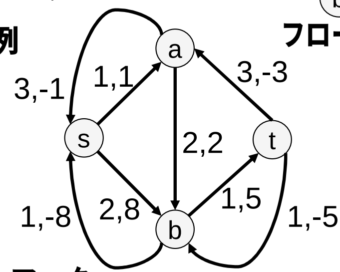
残余ネットワークの作り方 (その2)



問題例



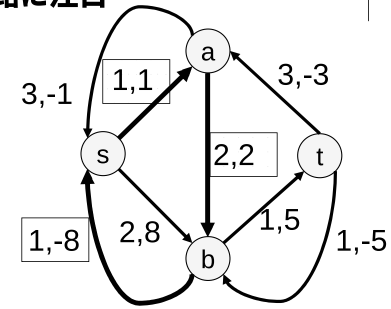
フローの例



残余ネットワーク

残余ネットワークの性質（その1）

残余ネットワークの閉路に注目

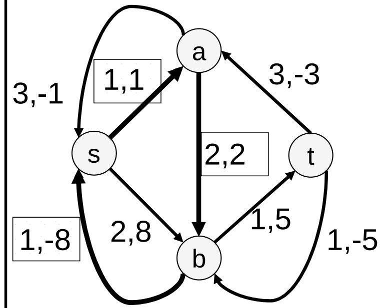


閉路の容量α
＝閉路に含まれる枝の容量の最小値=1
閉路の費用γ
＝閉路に含まれる枝の費用の和=-5

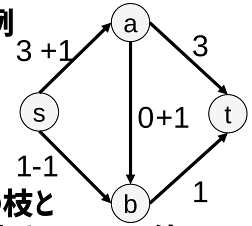
残余ネットワークの性質（その2）

残余ネットワークの閉路を用いてフローを更新

残余ネットワーク



フローの例



閉路の枝と
同じ向き⇒フロー値に＋α
逆の向き⇒フロー値にーα
無関係⇒フロー値は不変

閉路の容量α=1
閉路の費用γ=-5

この更新により、フローの費用はαγ（＝ー5）増加

残余ネットワークの性質（その3）

以上の議論より、以下が成り立つ

性質：残余ネットワークに費用が負の閉路が存在
⇒　フローの費用を減少させることが可能
⇒　現在のフローは費用最小でない

実は、逆も成り立つ（証明は後で）

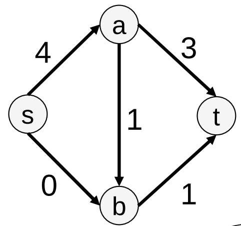
性質：現在のフローは費用最小でない
⇒　残余ネットワークに費用が負の閉路が存在

2**つの性質をまとめると**

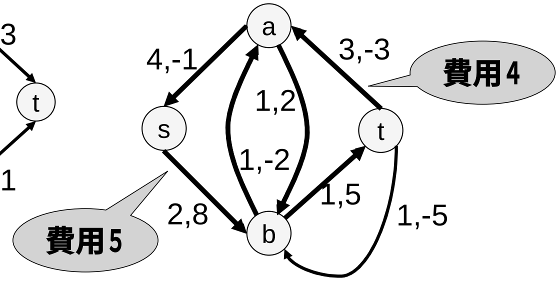
定理：現在のフローは費用最小
⇔　残余ネットワークに費用が負の閉路が存在しない

残余ネットワークの性質（その4）

修正後のフロー



残余ネットワーク



費用が負の閉路がない
⇒　現在のフローは費用最小

負閉路消去法

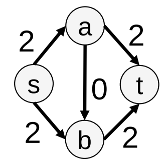
最小費用フローを求めるためのアルゴリズム

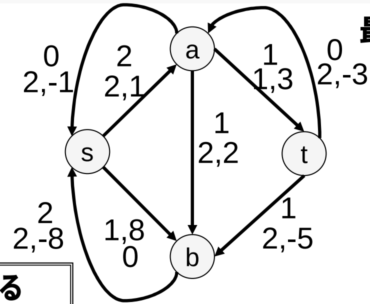
ステップ0：人工問題を解いて、需要供給量を満たす フローを求める
ステップ1：現在のフローに関する残余ネットワークを作る
ステップ2：残余ネットワークに費用が負の閉路が 存在しない⇒ 現在のフローは費用最小（終了）
ステップ3：残余ネットワークの費用が負の閉路を求め、それを用いて現在のフローを更新する
ステップ4：ステップ1へ戻る

性質の証明（その1）

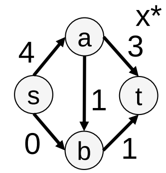
性質：現在のフローは費用最小でない
⇒ 残余ネットワークに費用が負の閉路が存在

現在のフロー x





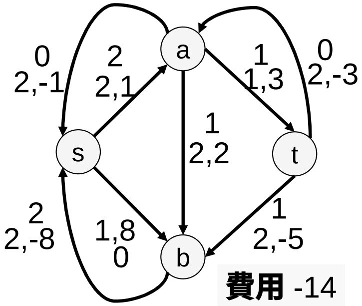
最小費用フロー x*



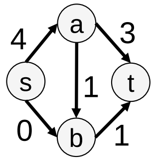
フロー x **に関する 残余ネットワーク上で** x* **と** x **の差を表現**

$x_{ij}^* - x_{ij} \geqq 0$ ⇒ **枝** (i, j) **に** $x_{ij}^* - x_{ij}$
$x_{ij}^* - x_{ij} < 0$ ⇒ **枝** (j, i) **に** $x_{ij} - x_{ij}^*$
その他の枝に0

性質の証明（その2）

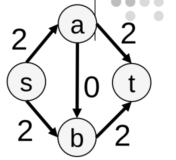


費用 -14



最小費用 フロー x*

費用 20



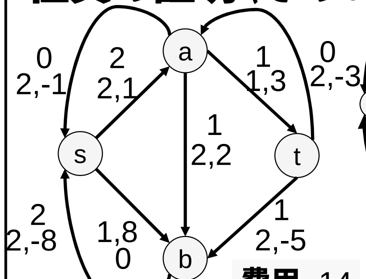
現在のフロー x

費用 34

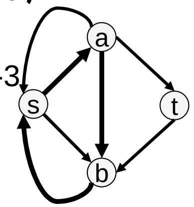
x* **と** x **の差：**
x **に関する残余ネットワーク上で 容量条件と流量保存則を満たす**
⇒ 残余ネットワーク上の「フロー」

差を表す「フロー」の費用
＝（x* の費用）
ー（x の費用）
＜ 0

性質の証明（その3）

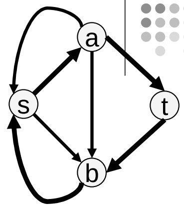


費用 -14



フロー量1

費用 -5



フロー量1

費用 -9

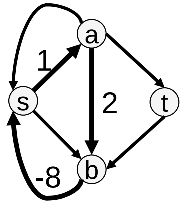
x* **と** x **の差を表す「フロー」：**
x **に関する残余ネットワーク上で 容量条件と流量保存則を満たす**

閉路上を流れる フローに分解可能

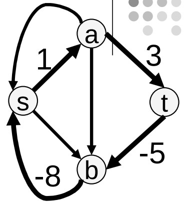
差を表す「フロー」の費用 ＝閉路上のフローの費用 の合計

性質の証明（その4）

閉路上のフローの
費用の合計　費用 -14
＝差を表す「フロー」の費用
＜ 0



フロー量1
費用 -5



フロー量1
費用 -9

閉路上を流れるフローの費用
＝（閉路の費用）
×（フロー量）

➡ いずれかの閉路の費用は負

➡ x の残余ネットワークに費用が負の閉路が存在
（証明終わり）

参考：目的関数が非線形関数の最小費用流問題

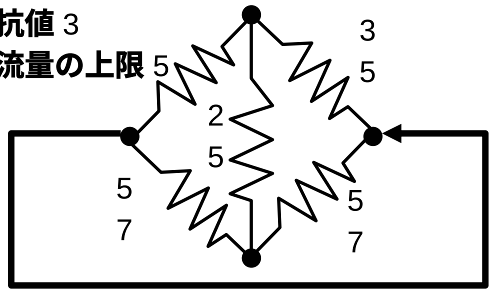
これまで扱ってきた最小費用流問題：
各枝の費用は線形関数 $f_{ij}(x_{ij}) = c_{ij} \times x_{ij}$

➡ 非線形関数 $f_{ij}(x_{ij})$ の場合に拡張

電気回路の例：
抵抗値 R の抵抗に電流が x アンペア流れると
消費されるエネルギー量は
$f(x) = (1/2)\ R\ x^2$　（2次の非線形関数）

参考：目的関数が非線形関数の最小費用流問題

抵抗値 3
電流量の上限 5



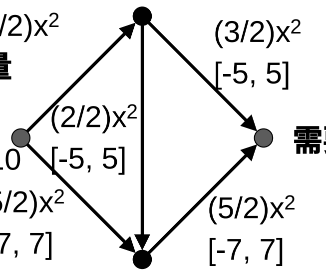
10アンペアの
電流を流す

各抵抗に流れる電流量はどのように決まるか？
⇒回路全体で消費される総エネルギー量を最小にする
⇒ 最小費用流問題へ

参考：目的関数が非線形関数の最小費用流問題

費用関数 $(3/2)x^2$
フローの容量
[-5, 5]

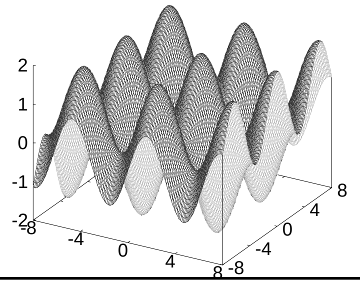
供給量＝10



需要量＝10

対応する非線形最小費用流問題（費用が2次関数）
ー負閉路消去法の拡張版を使って解くことが可能

第3章　非線形計画



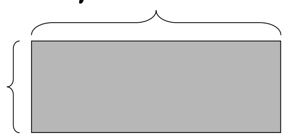
非線形計画問題とは？

目的関数や制約式が必ずしも線形でない数理計画問題

例：長方形の外周最小化問題

最小化	$2x + 2y$
条件	$x\,y \geqq 1$
	$x, y \geqq 0$

非線形の不等式

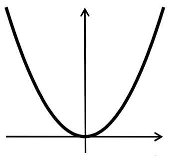


⇒ **非線形計画問題**

非線形関数の例（その1） [p.87]

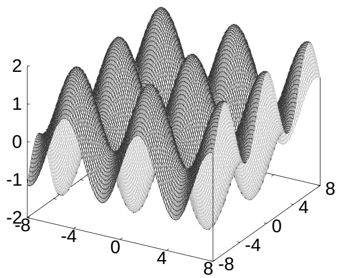
非線形関数　ー　線形でない関数

微分可能な非線形関数の例

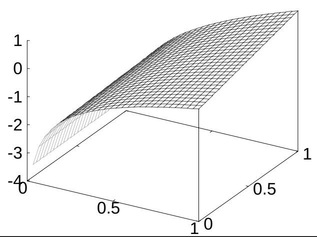


$$f_1(x) = x^2$$

$$f_2(x_1, x_2) = \sin x_1 + \cos x_2$$



$$f_3(x_1, x_2) = x_1 x_2 + \log x_1$$

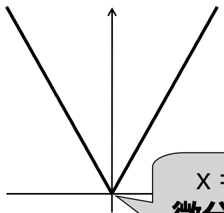


非線形関数の例（その2） [p.88]

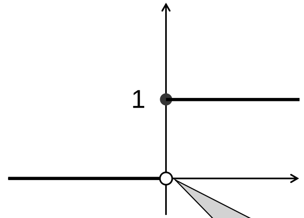
微分不可能な非線形関数の例

$$f_4(x) = |x|$$

$$f_5(x) = \begin{cases} 1 & (x \geq 0\text{のとき}) \\ 0 & (x < 0\text{のとき}) \end{cases}$$



x = 0 で
微分不可能



x = 0 で
微分不可能
不連続

この授業：
主に何回でも微分可能な関数を扱う

非線形計画問題の分類 [p.6,97,131]

制約なし最適化問題

入力：目的関数 f(x) **のみ**

最小化　f(x)　**条件　なし**　(x∈R^n)

制約つき最適化問題

入力：目的関数 f(x)
および制約を表す関数g_i(x) (i = 1, 2, …, m)

最小化　f(x)　**条件**　g_i(x) ≦ 0 (i = 1, 2, …, m)

この講義では、制約なし問題を主に扱う

非線形計画問題の分類 [p.6,97,131]

制約つき問題と制約なし問題の関係

- **制約つき問題は制約なし問題に変形できる**

最小化　f(x)　**条件**　g_i(x) ≦ 0 (i = 1, 2, …, m)

⇒

最小化　f(x) + h(x)　**条件　なし**

$$h(x) = \begin{cases} 0 & (g_i(\boldsymbol{x}) \le 0, i = 1,2,\ldots,m) \\ M & (\text{その他}) \end{cases}$$

M **は十分に大きい正数**

※この制約なし問題を直接解くことは難しい

- **制約なし問題を繰り返し解くことにより、制約つき問題を解くことが出来る**
  →p.146 **ペナルティ関数法、バリア関数法**

勾配ベクトル [p.89]

関数 f **の勾配ベクトル**

$$\boxed{\nabla f(\boldsymbol{x}) = \begin{pmatrix} \dfrac{\partial f}{\partial x_1} \\ \vdots \\ \vdots \\ \dfrac{\partial f}{\partial x_n} \end{pmatrix}}$$

一変数関数の場合は $\nabla f(x) = f'(x)$

例：

$$f_1(x) = x^2 \quad \Rightarrow \quad \nabla f_1(x) = 2x$$

$$f_2(x_1, x_2) = \sin x_1 + \cos x_2$$

$$\Rightarrow \quad \nabla f_2(\boldsymbol{x}) = \begin{pmatrix} \cos x_1 \\ -\sin x_2 \end{pmatrix}$$

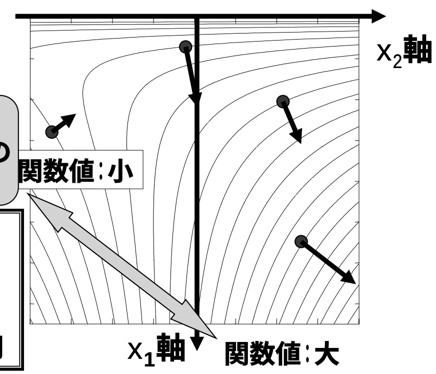
勾配ベクトル（続き） [p.89]

$$f_3(x_1, x_2) = x_1 x_2 + \log x_1$$

⇓

$$\nabla f_3(\boldsymbol{x}) = \begin{pmatrix} x_2 + 1/x_1 \\ x_1 \end{pmatrix}$$

関数f_3**の等高線と勾配ベクトルの方向**



等高線と勾配ベクトルの角度は９０度

勾配ベクトルのイメージ：
■関数という山を登るときに最も急な方向
■関数値が増加する方向

勾配ベクトルとテイラー展開[p.89]

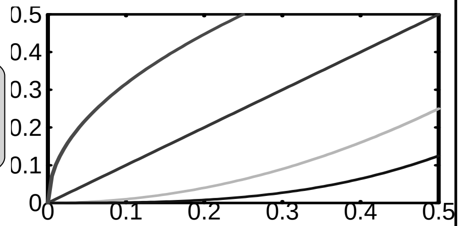
関数 f は勾配ベクトルを傾きとする線形関数により近似できる

（一次の）テイラー展開

$$f(\boldsymbol{x}+\boldsymbol{d}) = f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^T \boldsymbol{d} + o(\|\boldsymbol{d}\|)$$

記号 o(α)：α→0のときにg(α)/α→0となる関数g(α)を表す.

α², α³など，αより速く0に近づく関数を表す



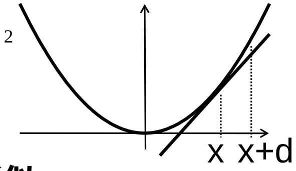
勾配ベクトルとテイラー展開[p.89]

$$f(\boldsymbol{x}+\boldsymbol{d}) = f(\boldsymbol{x}) + \nabla f(\boldsymbol{x})^T \boldsymbol{d} + o(\|\boldsymbol{d}\|)$$

例： $f_1(x) = x^2$ 　　 $\nabla f_1(x) = 2x$

$$\boxed{f_1(x+d)} = (x+d)^2 = x^2 + (2x)d + d^2$$

$$= \boxed{x^2 + (2x)d} + o(|d|)$$



関数 f を点 x において線形関数で近似

$(x+d)^2$ と $x^2 + 2xd$ の誤差 ー |d|が小さければ十分小さい

レポート問題（締め切り：12月15日）

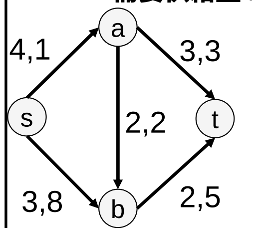
次の2つの最小費用フロー問題に対して、

（1）定式化せよ

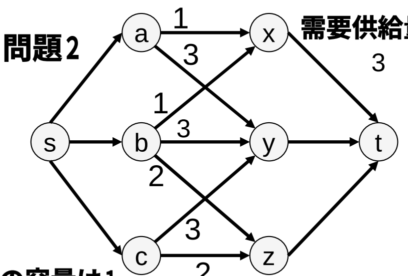
（2）人工問題を作って、需要供給量を満たすフローを求めよ

（3）負閉路消去法を用いて最小費用フローを求めよ

問題1 　需要供給量4



問題2 　需要供給量 3



各枝の容量は1

s から出る枝と t に入る枝の費用は0

それ以外は各枝の数値を参照