

データ構造と アルゴリズムII

第11回
最大フロー

560

Johnsonのアルゴリズム

- 疎なグラフ, 例えば $|E| \leq O(|V| \lg |V|)$ が仮定できる場合に向いている
- 隣接リスト表現を仮定する.
- 実行時間は $O(|V|^2 \lg |V| + |V| \cdot |E|)$.
- 上記の仮定の下で, Floyd-Warshallアルゴリズムよりも漸近的に高速

561

Johnsonのアルゴリズムのアイデア

- 辺重みが全部非負なら, Dijkstraのアルゴリズムをすべての点について実行すれば最短経路が得られる.
- 負辺があるが, 負閉路を持たない場合は, 再重み付けを行う.
- すなわち, 各頂点 v に関して $h(v)$ を定義し, $w'(u, v) = w(u, v) + h(u) - h(v)$ とする. これにより最短経路は変化しないことに注意.
- 新しい重みの元での最短経路長 $\delta'(u, v)$ から, 元の問題の最短経路長は以下で与えられる:
 $\delta(u, v) = \delta'(u, v) - h(u) + h(v)$.

562

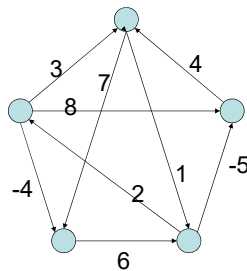
Johnsonのアルゴリズム

```

Johnson(G)
{
  compute G', where  $V[G'] = V[G] \cup \{s\}$  and
   $E[G'] = E[G] \cup \{(s, v) : v \in V[G]\}$  s.t.  $w(s, v) = 0$ .
  if Bellman-Ford( $G', w, s$ ) = False
    then print "∃ a neg-weight cycle"
  else for each vertex  $v \in V[G']$ 
    set  $h(v) = \delta(s, v)$ 
    computed by Bellman-Ford algo.
  for each edge  $(u, v) \in E[G']$ 
     $w'(u, v) = w(u, v) + h(u) - h(v)$ 
  for each vertex  $u \in V[G]$ 
    run Dijkstra( $G, w', u$ ) to compute  $\delta'(u, v)$ 
  for each  $v \in V[G]$ 
     $d_{uv} = \delta'(u, v) - h(u) + h(v)$ 
  return D
}
    
```

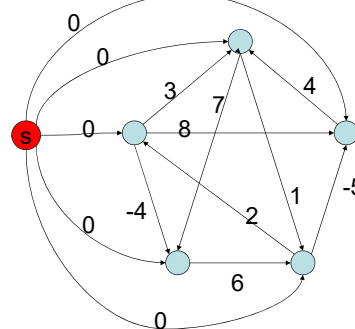
563

Johnsonのアルゴリズムの実行例



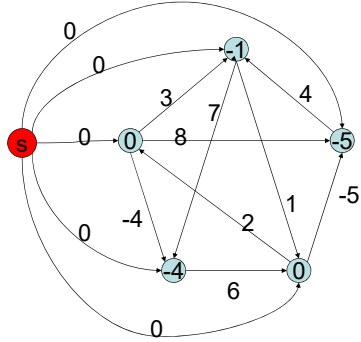
564

Johnsonのアルゴリズムの実行例



565

Johnsonのアルゴリズムの実行例

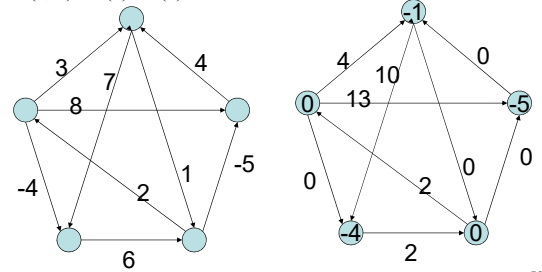


566

Johnsonのアルゴリズムの実行例

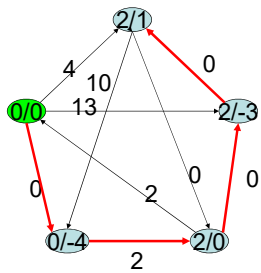
三角不等式より, u, v に関して,
 $h(u) + w(u, v) \geq h(v)$, よって,
 $w(u, v) + h(u) - h(v) \geq 0$.

再重み付け



567

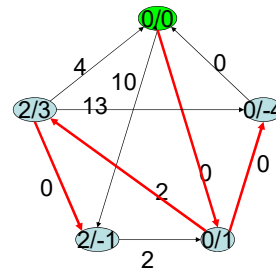
Johnsonのアルゴリズムの実行例



赤が最短経路.

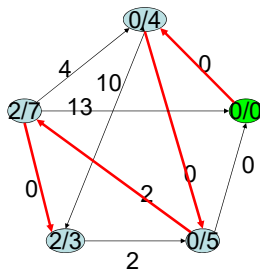
568

Johnsonのアルゴリズムの実行例



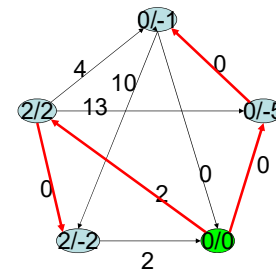
569

Johnsonのアルゴリズムの実行例



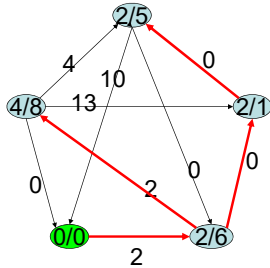
570

Johnsonのアルゴリズムの実行例



571

Johnsonのアルゴリズムの実行例



572

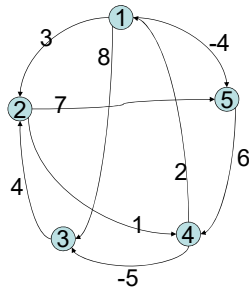
Johnsonのアルゴリズムの計算量

- Bellman-Fordの実行は $O(|V| \cdot |E|)$.
- Dijkstraを $|V|$ 回実行.
 - 凝ったヒープ (Fibonacci heap) を使えば
総計 $O(|V|^2 \log |V| + |V| \cdot |E|)$.
 - 普通のBinary heapだと、総計 $O(|V| \cdot |E| \log |V|)$.
- $|E|$ が少なければ ($\cong |V| \lg |V|$), Floyd-Warshall より良い.

573

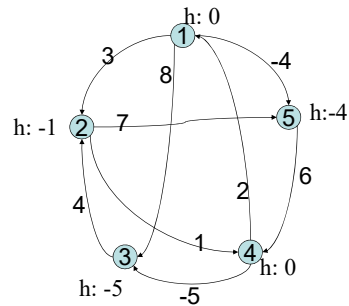
演習: Johnsonのアルゴリズム

- 右のグラフに関してJohnsonのアルゴリズムを適用せよ(最短経路は、1からの経路のみを求めるのみで良い)



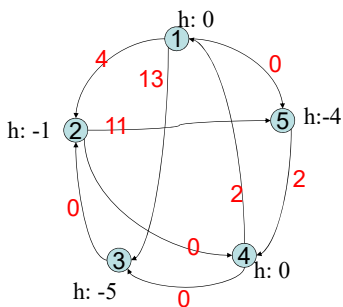
574

演習: Johnsonのアルゴリズム



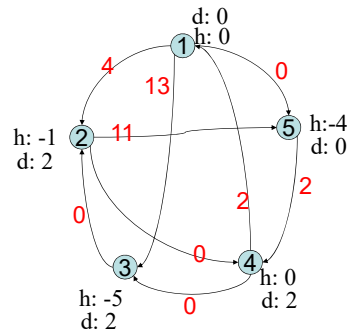
575

演習: Johnsonのアルゴリズム



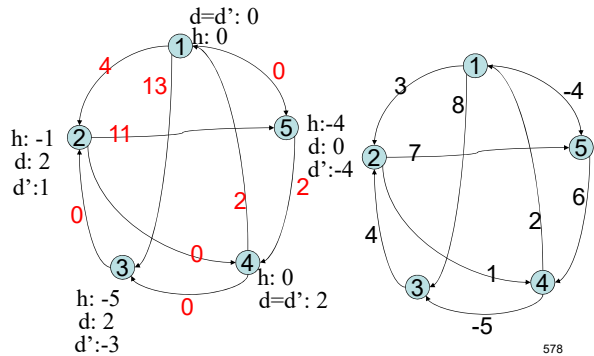
576

演習: Johnsonのアルゴリズム



577

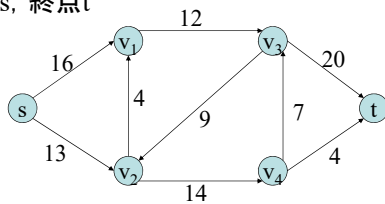
演習: Johnsonのアルゴリズム



26. 最大フロー

フローネットワーク

- フローネットワーク: $G=(V,E)$ は有向グラフで、各辺 $(u, v) \in E$ には容量 $c(u, v) > 0$ が定義される (u と v の間に辺がなければ $c(u, v)=0$ と仮定)
- 始点 s , 終点 t

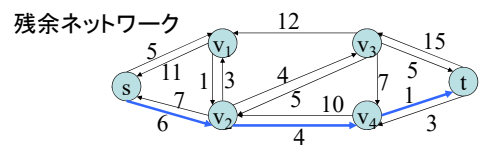
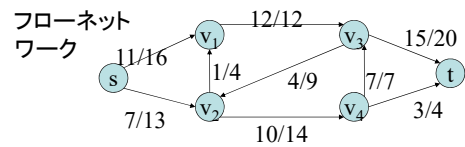
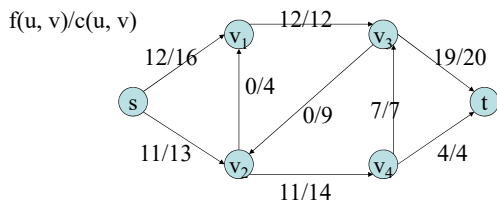


フロー

- フロー/流量は関数 $f: V \times V \rightarrow R$ で定義され、任意の u, v に関して以下の条件を満たす:
 - 容量制限: $f(u, v) \leq c(u, v)$
 - フロー保存則: すべての $u \in V - \{s, t\}$ に関して、 $\sum_{v \in V} f(v, u) = \sum_{v \in V} f(u, v)$ が成立.
- $|f| = \sum_{v \in V} f(s, v) = \sum_{v \in V} f(v, t)$ をフロー f の値と呼ぶ (絶対値とは無関係).

最大流量問題

- フローの値が最大となる f を求める.



→ 増加可能パス

Ford-Fulkerson法

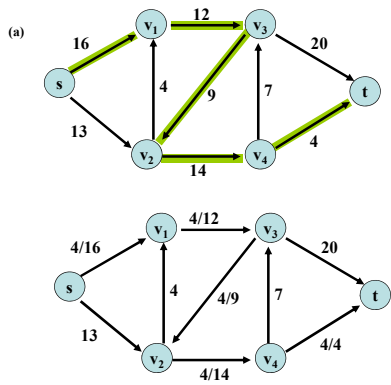
- 増加可能パスが存在する限り, 残余ネットワークにフローを加え続ける.
- 増加可能パスがなければ終了.

584

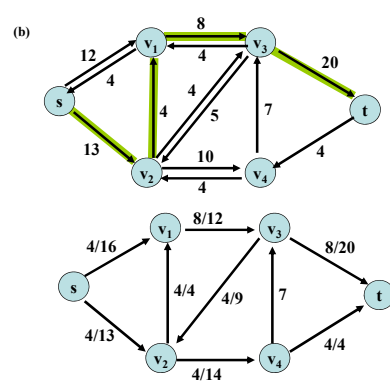
```

Ford-Fulkerson(G,s,t)
{ for each edge (u,v)∈E[G]
  do f[u,v]←0
    f[v,u]←0
  while ∃ path p from s to t on Gf
    do cf(p)←min{cf(u,v):(u,v) is in p}
      for each (u,v) in p
        if (u,v) ∈ E[G]
          f[u,v]←f[u,v]+ cf(p)
        else f[v,u]←f[v,u]-cf(p)
  return f
}
    
```

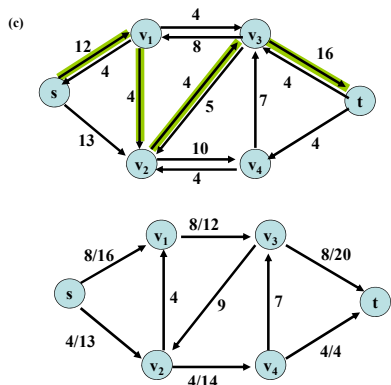
585



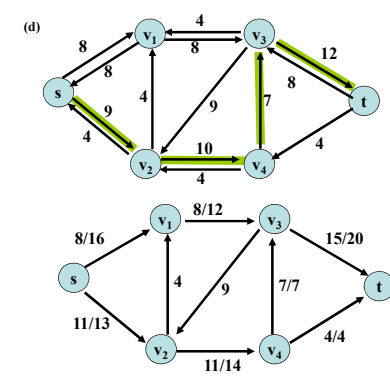
586



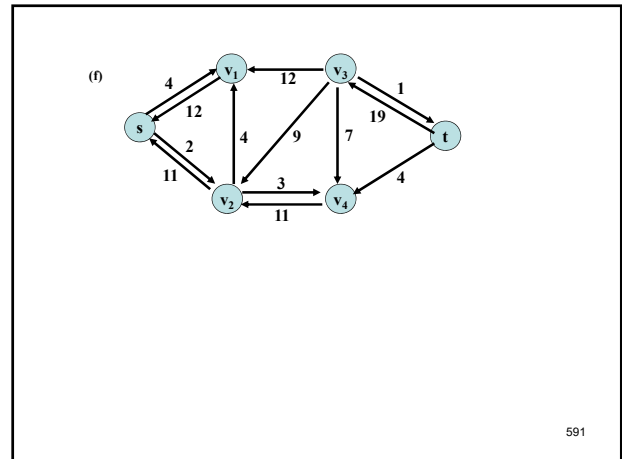
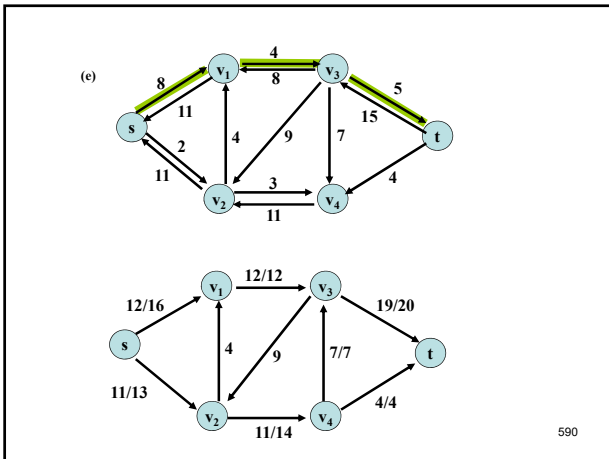
587



588



589



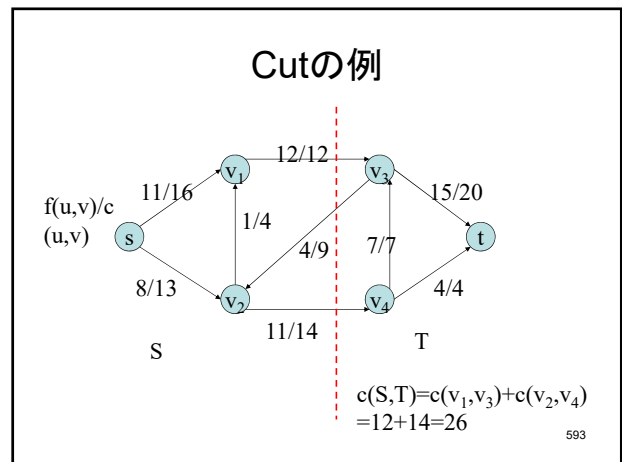
最大フローと最小カット

- フローネットワーク $G=(V, E)$ のカット (cut) (S, T) は、 V の S と T の分割で $s \in S$ 及び $t \in T$ を満たすもの。
- カット (S, T) とフロー f に関して、カットと交差する純フロー $f(S, T)$ は以下で与えられる:

$$\sum_{u \in S} \sum_{v \in T} f(u, v) - \sum_{u \in S} \sum_{v \in T} f(v, u)$$
- Cut (S, T) の容量 (Capacity), $c(S, T)$ は以下で与えられる:

$$\sum_{u \in S} \sum_{v \in T} c(u, v)$$
- ネットワークの最小カットは、このネットワークのすべてのカット中で容量が最小のもの。

592



補題26.4

- 任意のカット (S, T) とフロー f に関して、 (S, T) と交差する純フローは、フローの流量 $|f|$ に等しい。

594

補題26.4の証明

フロー保存則より、 s, t 以外の任意の頂点 u に関して、 $\sum_{v \in V} f(u, v) - \sum_{v \in V} f(v, u) = 0$ が成立。よって、 $\sum_{u \in S - \{s\}} (\sum_{v \in V} f(u, v) - \sum_{v \in V} f(v, u)) = 0$ が成立。

$$\begin{aligned}
 |f| &= \sum_{v \in V} f(s, v) - \sum_{v \in V} f(v, s) \\
 &= \sum_{v \in V} f(s, v) - \sum_{v \in V} f(v, s) + \sum_{u \in S - \{s\}} [\sum_{v \in V} f(u, v) - \sum_{v \in V} f(v, u)] \\
 &= \sum_{u \in S} [\sum_{v \in V} f(u, v) - \sum_{v \in V} f(v, u)] \\
 &= \sum_{u \in S} \sum_{v \in T} f(u, v) - \sum_{v \in T} f(v, u) + \sum_{u \in S} \sum_{v \in S} f(u, v) - \sum_{v \in S} f(v, u) \\
 &= \sum_{u \in S} \sum_{v \in T} f(u, v) - \sum_{v \in T} f(v, u) \\
 &= f(S, T)
 \end{aligned}$$

595

系26.6

- フローネットワークの流量は任意のカットの容量で上から抑えられる.
- 証明: 容量制限から明らか

596

最大フロー＝最小カット

- 定理26.6: 最大フロー最小カット定理
- 以下の条件は等価
 1. f は G における最大フローである.
 2. 残余ネットワーク G_f は増加可能経路を含まない.
 3. G のあるカット (S, T) に関して $|f| = c(S, T)$ である.

597

証明

- 1⇒2: 1だけ2でない, すなわち, 最大フローであるのに, 増加可能パスがあるとすると, 明らかに矛盾.
- 2⇒3: 残余ネットワークでは, s から t への経路が存在しない. s から到達可能な頂点集合を S , 残りを T とすると, これはカットになっている. $u \in S, v \in T$ に関して, $(u, v) \in E$ なら, この辺は容量一杯まで流れているはず(そうでないと v が s から到達可能). また, $(v, u) \in E$ なら, この辺のフローは0でないとは矛盾. 補題26.4より, (S, T) と交差する純フローは, フローの流量に等しい.
- 3⇒1: 系26.5より, すべてのフローの流量はカットの容量で上から抑えられる. よって, カットの容量に等しいことは, フローが最大であることの十分条件である.

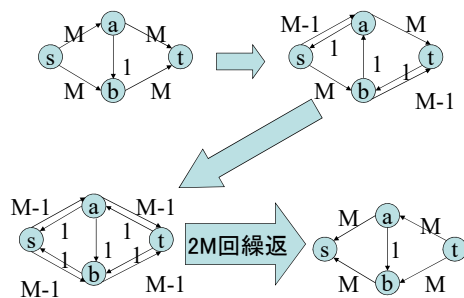
598

Ford-Fulkerson法の計算量

- 増加可能パスを求める方法に依存する.
- 良くない方法を使うと停止しないこともある(辺容量が無理数の場合).
- 容量が整数なら有限の繰り返しで停止する.
- 幅優先探索を用いて, 最短経路の増加可能パスを求めた方がよい(Edmonds-Karpアルゴリズムと呼ばれる).
- この場合のフロー増加操作の総数は高々 $O(V^2E)$ となる.

599

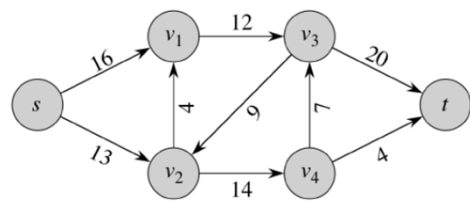
最短でない増加可能パス



600

演習: Ford-Fulkerson

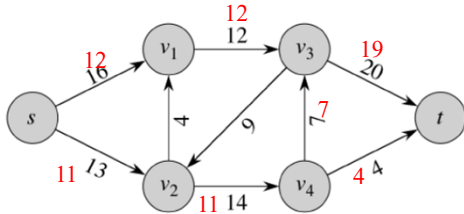
- 以下のグラフの最大フローを求めよ



601

演習: Ford-Fulkerson

- 以下のグラフの最大フローを求めよ



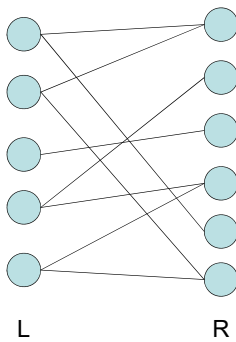
602

二分グラフの最大マッチング (26.3)

- 二分グラフは、頂点集合VがLとRの二つの集合に分割され、任意の辺 (u, v) は、LとRの頂点を結ぶ(L間, R間の辺は存在しない).

603

二分グラフの例



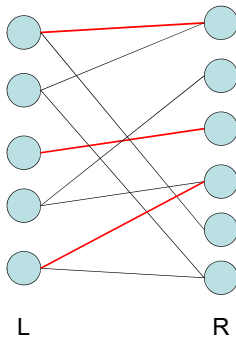
604

マッチング

- すべての頂点vに関して、vに接続する辺が高々一つしかない辺集合をマッチングと呼ぶ.
- 最大マッチングは、要素数が最大のマッチングである.

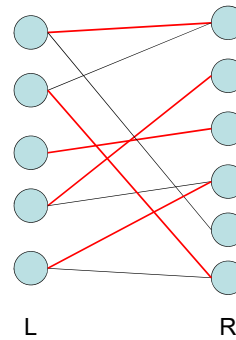
605

マッチングの例



606

最大マッチングの例



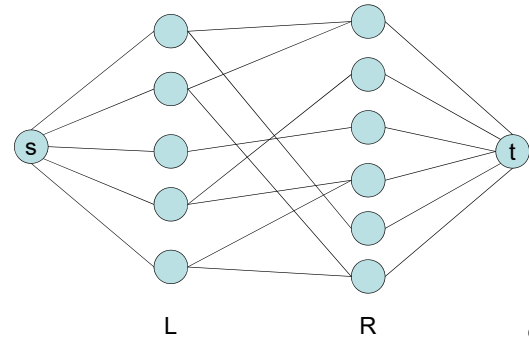
607

最大フローとしての表現

- 新しい頂点sとtを導入
- sから左側の頂点に容量1の辺を張る
- 右側の頂点からtに容量1の辺を張る

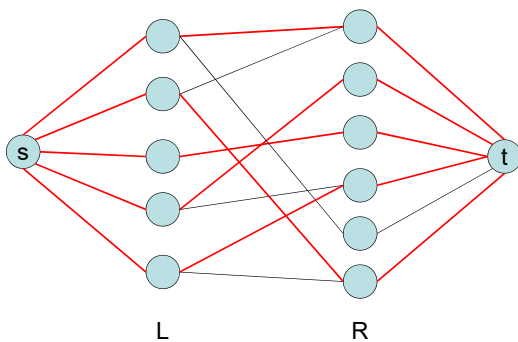
608

最大フローとしての表現



609

最大フロー=最大マッチング



610

予定 (II)

- 6/5: 第7回: 幅優先, 深さ優先探索, トポロジカルソート
- 6/12: 第8回: 強連結成分分解, 最小全域木
- 6/19: 第9回: 単一始点最短路 (I)
- 6/26: 第10回: 単一始点最短路 (II), 全点对最短経路
- 7/3: 第11回: 最大フロー
- 7/10: 小テスト(範囲は1-6回まで)
- 7/17: 休講
- 7/24: 定期試験(大講義室)

611

予定 (I)

- 4/10: 第1回: イントロダクション: マッチング
- 4/17: 休講(学生健康診断)
- 4/24: 第2回: 動的計画法1
- 5/1: 木曜日の講義日
- 5/8: 第3回: 動的計画法2
- 5/15: 第4回: 動的計画法3
- 5/22: 第5回: 貪欲法
- 5/29: 第6回: 制約充足問題

612

定期試験に関して

- 7月24日(火)
- 場所は, この部屋ではなく, 大講義室
- 内容は4題ぐらい
- 動的計画法, 貪欲法, 制約充足問題, 幅/深さ優先探索, 最短経路, 最大フロー等
- 2016年度より前とは傾向が変わるので注意

613