数理解析研究所講究録
202
第731巻 1990年 202-213

k－辺連結あるいはk－点連結全域部分グラフについて

豊橋技術科学大学　永持　仁 (Hiroshi Nagamochi)
京都大学　茨木 俊秀 (Toshihide Ibaraki)

1．はじめに

与えらえた無向グラフG=(V,E)および正整数k(≧2)に対し，辺数を最小にするようなGのk－辺連結全域部分グラフあるいはk－点連結全域部分グラフを求める問題は，kを固定してもNP－完全である[7]．最近，鈴木，高橋，西関らは辺数が高々O(k|V|)本であるようなk－辺連結（あるいはk－点連結）全域部分グラフを線形時間で見出す問題の重要性を指摘し，k≦3のときそのようなk－点連結全域部分グラフを見出す線形時間アルゴリズムを与えた[21]．一般のkに対しては，そのような全域部分グラフは多項式時間で見出せることが示されている．例えば，辺数がO(k|V|)本であるk－辺連結あるいはk－点連結全域部分グラフはそれぞれO(k|E|)，$O(|V|^{1/2}|E|^2)$時間で求まることが知られている[15]．

本報告で述べるアルゴリズム(FOREST)は，グラフ探索を1度行うだけで，辺数がk|V|本以下であるk－辺連結（あるいはk－点連結）全域部分グラフをO(|V|+|E|)時間で見出す手続きである．従って，このアルゴリズムを2点間の辺素あるいは内素なパスを最大本数見つける問題やグラフの連結度を判定する問題を解くアルゴリズムの前処理として用いれば，必要な計算時間を改善することができる．

次に，このアルゴリズムFORESTの持つ幾つかの有用な性質を利用して，多重グラフの辺連結度λ(G)，容量付き無向ネットワークの最小カット値，グラフの最小カットの総数などを求める問題を解く高速なアルゴリズムを開発する．

2．　定　義

G=(V,E)を節点集合V，辺集合Eからなるループを持たない無向グラフとする．グラフの2節点間に辺を多重に持つグラフは多重であると呼ばれ，2節点u,vを端点とする辺集合をE_{uv}で表わす．ただし，混同の恐れの無いときは$e\in E_{uv}$をe=(u,v)でも記す．多重辺を持たないグラフは単純と呼ばれる．G=(V,E)からE'(⊆E)の全ての辺を除去して得られるグラフをG-E'と書く．同様に，G=(V,E)からV'(⊂V)の全ての節点（およびVに接続している辺）を除去して得られるグラフをG-V'と書く．

グラフGのどのk－1本の辺を取り除いてもGが非連結にならないとき，Gはk－辺連結であるという．グラフGの辺連結度λ(G)とは，Gがk－辺連結であってk＋1－辺連結でないときλ(G)=kと定義される．グラフG=(V,E)の局所辺連結度λ(x,y;G)とは，G-E'においてxとyが非連結であるようなE'(⊆E)の位数の最小値である．明らかに，λ(G)=min{λ(x,y;G) | x,y∈V} が成り立つ．

グラフGがk＋1個以上の節点を持ちどのk－1個の節点を取り除いても非連結にな

らないとき，Gはk－点連結であるという（例えば，完全グラフK_{k+1}はk－点連結）．グラフGの点連結度$\kappa(G)$とは，Gがk－点連結であってk＋1－点連結でないとき$\kappa(G)=k$と定義される．局所点連結度$\kappa(x,y;G)$とは，隣接してないx,yに対し，G-V'においてxとyが非連結となるようなV'($\subseteq V$)の位数の最小値である．隣接するx,yに対しても$\kappa(x,y;G)\equiv|V|-1$と定めておけば，$\kappa(G)=\min\{\kappa(x,y;G)\mid x,y\in V\}$が成り立つ（実は，$\overline{\kappa}(x,y;G)$をx,y間の内素なパスの最大本数と定義しても，$\kappa(G)=\min\{\overline{\kappa}(x,y;G)\mid x,y\in V\}$　が成立する）．

グラフG=(V,E)に対して，V'=V,E'$\subseteq$Eを満たすグラフG'=(V',E')はGの全域部分グラフと呼ばれる．特に，G（連結でなくてもよい）の各連結成分上で，G'の作る部分グラフが全て連結であるとき，G'は極大であるという．閉路を持たないグラフTを森と呼び，Tが連結のとき木と呼ばれる．

グラフGの2節点x,yを1点に縮約して得られるグラフをG/{x,y}で表わす（ただし，発生したループは除去してあるとする）．部分節点集合X$\subseteq$Vに対して，$E(X)\equiv\{e\in E\mid e$の両端点の一方がXに，もう一方がV-Xに含まれる$\}$と定める．$E(\{x\})$はE(x)とも記す．|E(x)|は節点xの次数である．グラフGの節点次数の最小値を$\delta(G)$と書く．

連結無向グラフG=(V,E)において，G-Fが非連結となるような部分辺集合F$\subseteq$EをGのカットと呼ぼう．特に，|F|=kなるカットFの集合を$C^k(G)$と記す．同様に，2点s,tがG-Fにおいて非連結となるカットFをs-tカットと呼び，|F|=kなるs-tカットFの集合を$C^k(s,t;G)$で表わす．

図1に$\delta(G)=7$, $\lambda(G)=6$なる多重グラフGの例を示しておく．

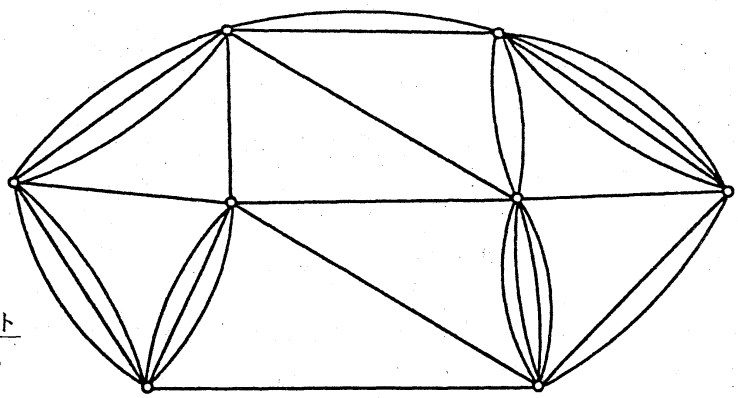


図1．　多重グラフの例．

3．　k－辺連結全域部分グラフを求めるアルゴリズム

まず，アルゴリズムの基礎となる定理を示す．

【定理3.1】[12] 連結グラフG=(V,E)に対し，$T_1=(V,E_1)$をGの任意の極大全域木とし，以下，$T_i=(V,E_i)$, i=1,2,...,kを$G-E_1\cup E_2\cup\ldots\cup E_{i-1}$の極大全域森とする．このとき，各$G_i=(V,E_1\cup E_2\cup\ldots\cup E_i)$, i=1,2,...,|E|は，任意のx,y$\in$Vに対し，$\lambda(x,y;G_i)\geqq\min\{i,\lambda(x,y;G)\}$　を満たす．□

この定理によれば，任意の$\lambda(G)\geqq k$に対し$G_k=(V,E_1\cup E_2\cup\ldots\cup E_k)$はk-辺連結であり，各極大全域森$T_i$は通常のグラフ探索を用いてO(|V|+|E|)時間で求まることから，高々k(|V|-1)本の辺を持つGのk-辺連結全域部分グラフG'はO(k|E|)時間で得られることが分かる．ところで，$T_1=(V,E_1)$を求める最初のグラフ探索において，E_1に含まれないと判

断された辺に対しても，その辺が含まれるべきE_iが$O(1)$時間で分かれば，1回のグラフ探索で全ての$E_1, E_2, \ldots, E_{|E|}$が求まる．以下に示すFORESTはそのようなグラフ探索アルゴリズムである．

```
Procedure FOREST; {入力: 多重グラフG=(V,E),
                   出力: E1,E2,...,E|E|}
begin
    E1:=E2:=...E|E|:=φ;
    各節点vのラベルr(v)を0にする.
    while 未走査節点が存在すれば do
      begin
        最大ラベルrの"未走査"節点xを選ぶ;
        for "未走査"辺によりxに隣接している各節点yに対し do
                                   {Exy内の辺は全て未走査}
          for 各辺e∈Exyに対し do
            begin
              Er(y)+1:=Er(y)+1 ∪ {e};
              r(y):=r(y)+1;
              辺eを"走査済"とマークする
            end;
          xを"走査済"とマークする
      end;
end.
```

一般に，ある森T_i上の2節点を結ぶ辺を新たにT_iに加えられるかどうか（閉路が発生するか）という判定を$O(1)$時間で行うことは難しい．アルゴリズムFORESTでは，次に走査すべき節点としては，なるべく大きな添え字iを持つE_iの辺に接続するものが選ばれる．この工夫により探索された辺の両端点が同じT_i上にあれば，常に閉路を作るという事実[12]が証明できるので，閉路判定の必要がなくなる（このために，各節点xに対し$E(x) \cup E_i \neq \phi$なる最大のiをラベルr(x)として保持させている）．

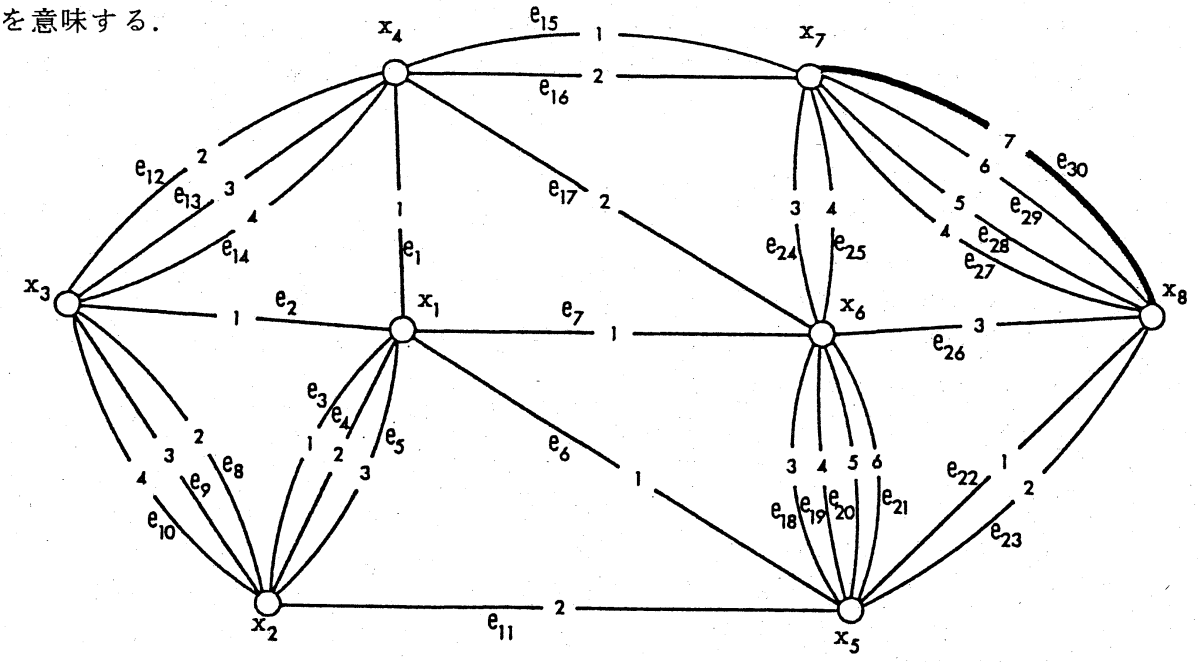
FORESTにおいて，同一ラベルrの節点を格納するためのバケットを$|E|$個用意し，非空バケットをリストでつないだデータ構造を用いれば，最大ラベルの未走査節点xは$O(1)$時間で見つかり，1回のラベルの更新も$O(1)$時間でできる（ラベルの変化量が常に定数1であることに注意）．従って，全体の計算時間は$O(|V|+|E|)$で済む．算法の正当性については比較的容易に証明でき，以下の結果を得る．

【補題3.1】[12] FORESTにk（≥ 1）-辺連結グラフG=(V,E)を入力して得られる(V, E_1)はGの全域木であり，各(V, E_i), $2 \leq i \leq |E|$は$G-E_1 \cup E_2 \cup \ldots \cup E_{i-1}$の極大全域森である．□

【定理3.2】[12] 連結グラフG=(V,E)に対し定理3.1を満たす辺集合E_i, $i=1,2,\ldots,|E|$を$O(|E|)$時間で見つけることができる．ただし，Gが単純グラフのとき$|E_i| \leq |V|-i$，Gが多重グラフのとき$|E_i| \leq |V|-1$を満足する．□

単純グラフGが完全グラフK_{k+1}である場合には，$|E_1|+|E_2|+\ldots+|E_k|=k(k+1)/2$，多重グラフGがk重の木であるとき，$|E_1|+|E_2|+\ldots+|E_k|=k(|V|-1)$が満たされる．

図1の多重グラフにFORESTを適用した例を図2に示す．ただし，図中の節点x_i, i=1, 2,...および辺e_k, k=1,2,...はそれぞれFORESTによりi番目，k番目に走査されたことを意味する．



G^1: $\delta(G^1)=7$

i : E_iの辺

7 : 次に縮約される辺

図2． 図1のグラフにFORESTを適用した例．

4．k－点連結全域部分グラフを求めるアルゴリズム

驚くことに，FORESTにそのままk(≧1)-点連結単純グラフG=(V,E)を入力すると，得られる全域部分グラフ$G'=(V,E'=E_1 \cup E_2 \cup \ldots \cup E_k)$はk-点連結になる．正確に言えば定理3.1,3.2に代わる次の命題が成り立つ．

【定理4.1】[12] 単純連結グラフ G=(V,E)に対し，各 $G_i=(V,E_1 \cup E_2 \cup \ldots \cup E_i)$, i=1,2,...,|E|が，任意のx,y∈Vに対し，$\kappa(x,y;G_i) \geq \min\{i, \kappa(x,y;G)\}$, $|E_i| \leq |V|-i$ を満たすようなEの分割$E_1, E_2, \ldots, E_{|E|}$をO(|E|)時間で見つけることができる．□

（実は，文献[12]における証明を拡張することにより，$\overline{\kappa}(x,y;G)$をx,y間の内素なパスの最大本数としたとき$\overline{\kappa}(x,y;G_i) \geq \min\{i, \overline{\kappa}(x,y;G)\}$ の関係も示せる．）

FORESTにより得られた$E_1, E_2, \ldots, E_{|E|}$に対し定理4.1が成立することの証明は，辺連結性の場合に比べればやや複雑である．結果を導くために用いられた次の補題を挙げておく．G=(V,E)の任意のE"(⊆E)およびG-E"における任意の分離節点集合W⊂Vを考える．

G-E"-Wの１つの連結成分を$G_X=(X,E_X)$,残りのG-E"-W-Xを$G_Y=(Y,E_Y)$で表わし，Wの節点をFORESTにより走査された順に，$s_1,s_2,\ldots,s_{|W|}$ と記すと次が示せる．

【補題4.1】[12] FOREST実行中t番目の$s_t\in W$が走査された直後，$\{e\in E"\mid e$は走査済辺$\}\subseteq E_d\cup E_{d+1}\cup\ldots\cup E_{|E|}$ なるd≧t+1が存在すれば，次の(i),(ii)が常に成り立つ．
(i) E_i(t+1≦i)の辺で連結しているx∈X,y∈Yはまだ存在していない．
(ii) E_tの辺で連結しているx∈X,y∈Yは存在し得るが，その任意のx-yパスは$s_t\in W$を必ず通る．□

この補題により定理4.1は容易に導ける．定理3.1の証明と同様に背理法を用いて$\kappa(x,y;G)\geq i+1$, $\kappa(x,y;G_i)=\kappa(x,y;G_{i+1})=i$なるx,y,iを仮定する．$E"=E_{i+2}\cup E_{i+3}\cup\ldots\cup E_{|E|}$ とすると，$G-E"-W=G_{i+1}-W$において$|W|=i$なるある点カットWにより，xはある連結成分Xに，yは残りのY=G-E"-W-Xに含まれる． $\kappa(x,y;G)\geq i+1$よりXとYの間にはE"の辺eが存在する．これは，G_iにおいて eの両端点x',y'は連結していることを意味し，補題3.1よりx'とy'は(V,E_{i+1})において連結している．ところが，補題4.1より$s_{|W|}$が走査された直後(その後永久に)，X,YはE_h(h≧i+1=|W|+1)では連結されない．これは矛盾である．

5．前処理アルゴリズムとしてのFOREST

FORESTは，グラフの連結度を保持したまま辺数を縮小でき，必要な計算時間も線形である．従って，グラフの連結度等を計算する際に，FORESTを前処理として実行しておけば必要な計算の手間を改良することができる．

例えば，与えられた正整数kに対し多重グラフGがk－辺連結度かどうかは$O(\min\{k,|E|^{1/2}\}|V||E|)$時間で判定できることが知られているが[4]，FORESTにより$G_k=(V,E_1\cup E_2\cup\ldots\cup E_k)$を求めた後，$\lambda(G_k)\geq k$の判定を行えばよいので，計算時間を$O(|E|+\min\{k,(k|V|)^{1/2}\}k|V|^2)$に短縮できる．また，この判定法に基づいて，$\lambda(G)$を効率良く決定するために，

$\lambda(G)\geq k$の判定をk=2,2^2,2^3,...の順に$\lambda(G)<k=2^i$となるまで行う． (5.1)

この場合，結局$O(|E|+\min\{\lambda(G),(\lambda(G)|V|)^{1/2}\}\lambda(G)|V|^2)$時間で済む[12]（これは[4]の算法では$\lambda(G)<k$と判定されたときには既に$\lambda(G)$が見出されていることを利用している）．この計算時間は，これまでに知られている多重グラフGの$\lambda(G)$を求める最良の$O(|E|^{3/2}|V|)$時間[4]に比べ短縮されている．

グラフGの点連結度$\kappa(G)$については，$\kappa(G)\geq k$を判定する$O(\max\{k,|V|^{1/2}\}k|V|^{1/2}|E|)$時間の算法が知られているが[6]，FORESTにより上記と同様に|E|をk|V|に置き換えることができ，$O(|E|+\max\{k,|V|^{1/2}\}k^2|V|^{3/2})$に減らせる[12]．$\kappa(G)$の決定に必要な手間$O(\max\{\kappa(G),|V|^{1/2}\}\kappa(G)|V|^{1/2}|E|)$[6]についても$O(|E|+\max\{\kappa(G),|V|^{1/2}\}\kappa(G)^2|V|^{3/2})$時間に減らせる[12]．

グラフG上の２点s,t間の辺素なパスの最大本数λを求める手間[4]$O(|V|^{2/3}|E|)$（Gが単純のとき）および$O(\min\{|V|,|E|^{1/2}\}|E|)$（Gが多重のとき），あるいは内素なパスの最大本数$\kappa$を求める手間[4]$O(|V|^{1/2}|E|)$についても同様にそれぞれFORESTにより，$O(|E|+\min\{\lambda,|V|^{2/3}\}|E|)$（Gが単純のとき），$O(|E|+\min\{\lambda,|V|,\lambda^{1/2}|V|^{1/2}\}\lambda|V|)$

(Gが多重のとき)，および$O(|E|+\min\{\kappa,|V|^{1/2}\}\kappa|V|)$時間に短縮できる[12]．また，これら2点間の辺素および内素なパスを見出す問題においてはFORESTにより平均的にどの程度効率が改善されるかという点についての計算機実験も行われている[10]．

2部グラフ$H=(V_1,V_2,E)$($|V_1|=|V_2|$)上で大きさ$k(<|E|/|V_1|)$のマッチングの実現するために，定理4.1を使って次のように予めグラフの辺数を$O(k|V_1|)$に縮小することができる(Hの最大マッチングの大きさαが$\alpha\geq|E|/|V_1|$を満たすことは大きさαの最小節点被覆が存在することから導けるので，大きさ$k<|E|/|V_1|$のマッチングの存在は自明である)．Hに新たに2点s,tおよびsとV_1間，tとV_2間を結ぶ辺を加えたグラフ$\bar{H}=(V_1\cup V_2\cup\{s,t\},\bar{E})$にFORESTを適用し$\bar{H}_k=(V_1\cup V_2\cup\{s,t\},\bar{E}_1\cup\bar{E}_2\cup\ldots\cup\bar{E}_k)$を見出す．定理4.1より2部グラフ$\bar{H}_k-\{s,t\}$には大きさkのマッチングが存在しており，辺数は$2k|V_1|$以下である(しかし，残念ながら最大マッチングなど，大きさ$k\geq|E|/|V_1|$のマッチング実現のためにはこのやり方では辺数のオーダーを減らせない)．

この他に，FORESTにより計算時間の短縮されるグラフアルゴリズムとして6－点連結グラフ$G=(V,E)$上で2組の2点対間の点素なパス(常に存在する)を見出す$O(|V||E|)$時間の算法[18]が挙げられる．これは，明らかに定理4.1により$O(|E|+|V|^2)$に短縮される．

6．多重グラフの辺連結度を求めるアルゴリズム

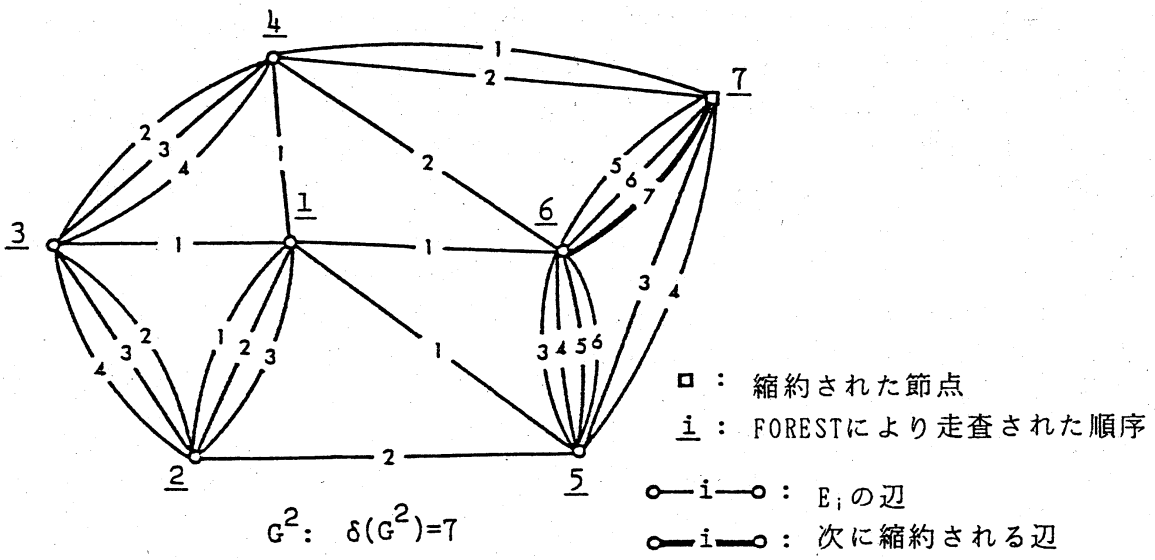
現在，単純グラフ$G=(V,E)$の辺連結度$\lambda(G)$は$O(\lambda(G)|V|^2)$時間で求まることが知られている[11]．グラフが多重の場合の$\lambda(G)$の計算時間は，前節で議論したようにFORESTを前処理として用いることにより$O(|E|+\min\{\lambda(G),(\lambda(G)|V|)^{1/2}\}\lambda(G)|V|^2)$で済むが，本節ではFORESTの持つ有用な性質を利用して，より高速な$O(|E|+\lambda(G)|V|^2)$時間アルゴリズムを開発する(この計算の手間は次節でさらに短縮される)．

定理3.1の条件を満足するEの分割$E_1,E_2,\ldots,E_{|E|}$において，ある辺eがE_hに含まれるとき，各$i\leq h$に対しeの両端u,vを結ぶE_iの辺から成るパスが全域森(V,E_i)上に存在することは容易に分かる(例えば図2の適用例を見よ)．これより次を得る．

【補題6.1】[12] 定理3.1の条件を満足するEの分割$E_1,E_2,\ldots,E_{|E|}$が見出されているとき，任意の辺$e=(x,y)\in E_i$に対し，$\lambda(x,y;G)\geq i$であり，x,y間の$\lambda(x,y;G)$本の辺素なパスは$O(|V|+|E|)$時間で見つかる．□

定理3.1の条件を満たすEの分割$E_1,E_2,\ldots,E_{|E|}$のうちをFORESTにより見出されたものは，特に以下の有用な性質を持つ．FORESTにより最後に走査された節点tについては，tに接続する辺E(t)は全てtに隣接する節点の走査中に走査され，その度にtのラベル$r(t)$が1ずつ増加していく．従って，tに接続する各辺は常に全て異なる分割E_iに属する，すなわち，$|E(t)\cap E_i|=1,i=1,2,\ldots,|E(t)|$が成り立つ(例えば，図2の$x_8$を見よ)．$E(t)\cap E_{|E(t)|}$の辺をe=(s,t)とすると補題3.1より$\lambda(s,t;G)\geq|E(t)|$．しかし，E(t)はs,tを分離するカットであるので$\lambda(s,t;G)=|E(t)|$を得る．これは次を意味する．

【補題6.2】 任意の無向多重グラフGにおいて $\lambda(x,y;G)=\min\{|E(x)|,|E(y)|\}$ $(\geq\delta(G))$なる辺$e=(x,y)$が必ず存在する．また，そのような辺はFORESTにより線形時間で見出せる．□



G^2: $\delta(G^2)=7$

図3. TESTECの適用例.

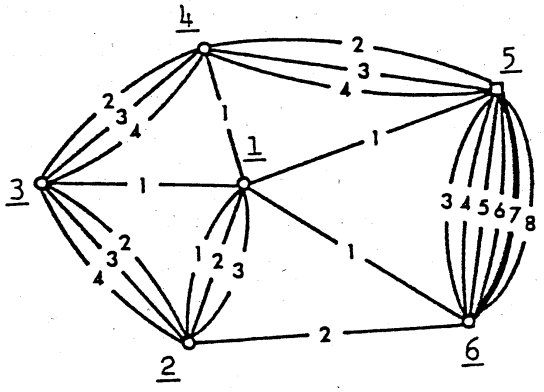


図4. G^3: $\delta(G^3)=7$

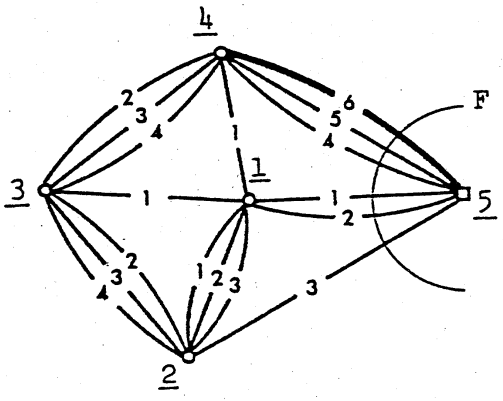


図5. G^4: $\delta(G^4)=6$

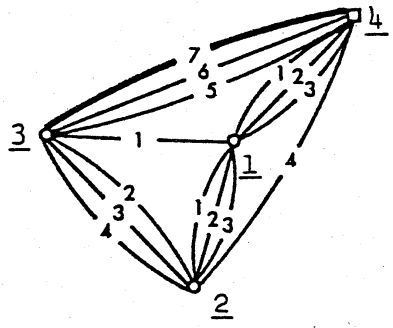


図6. G^5: $\delta(G^5)=7$

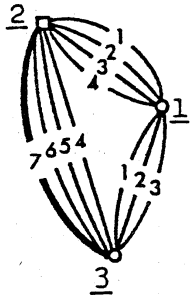


図7. G^6: $\delta(G^6)=7$

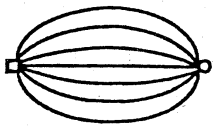


図8. G^7: $\delta(G^7)=7$

ところで，一般に多重グラフG=(V,E)(|V|>2)の任意の２点u,vを縮約しても辺連結度は減少しない．すなわち，$\lambda(G/\{u,v\})\geq\lambda(G)$．　逆に，$\lambda(u,v;G)\geq k$なるkに対しては，$\lambda(G/\{u,v\})\geq k$ ならば $\lambda(G)\geq k$ が成り立つ．次の補題はこれらの性質から容易に導くことができる．

【補題6.2】[13] $|V|\geq 3$のとき，$\lambda(u,v;G)\geq\delta(G)$なるu,vに対し，

$$\lambda(G)=\min\{\delta(G),\lambda(G/\{u,v\})\}\ .\ \square$$

補題6.2,6.3よりG=(V,E)の$\lambda(G)$を求めるには以下のようにすればよい：$G^1:=G$とし，FORESTにより$\lambda(u,v;G)\geq\delta(G)$なるu,vを見出し，$G^2:=G^1/\{u,v\}$とする．以下，節点数が２になるまで，$G^3,G^4,\ldots,G^{|V|-1}$を同様に構成した後，$\lambda(G)$は$\min\{\delta(G^1),\delta(G^2),\ldots,\delta(G^{|V|-1})\}$で与えられる．

図１のグラフにこのアルゴリズム（以下，TESTECと呼ぼう）を適用した例を図２～８に示す（この結果，$\lambda(G)=6$であることが判明する．また，図５のF⊆Eは$\lambda(G)$を与えるGの１つの最小カットである）．

アルゴリズムTESTECの計算時間はO(|V||E|)で済むことはFORESTの計算時間から容易に分かるが，(5.1)と同様の工夫を行うことにより，これをO(|E|+$\lambda(G)|V|^2$)時間に減らすことができる[13]（従って，Gが単純の場合にはこの手間は[11]の計算時間に一致する）．

７．実数容量付き無向ネットワークの最小容量カットを求めるアルゴリズム

本節では，無向グラフGの各辺eに正値の容量c(e)を持つネットワークN=(G,c)を考え，

最小容量カット値 $c(N)\equiv\min\{\Sigma_{e\in F}c(e)\mid F$はG-Fが非連結となるEの部分集合$\}$

を見出す効率の良いアルゴリズムを与える．

容易に分かることだが，c(N)の値は，Nの|V|(|V|-1)/2組の各２点対間の最大流の流量値のうちの最も小さい値と一致する．GomoryとHu[9]の結果によればNの全ての２点対間の最大流の値は|V|-1回最大流問題を解けば得られるので，c(N)は少なくともO(|V|T(|V|,|E|))時間で求まる．ただし，T(|V|,|E|)は最大流を計算するための手間で，現在，強多項式時間ではO(|V||E|log(|V|^2/|E|))が最良である[8]．以下では，c(N)を計算するO(|V||E|+|V|^{2}log|V|)時間のアルゴリズムを与える．

まず，ネットワークN=(G,c)の容量が全て整数値である場合を考えよう．Nの各辺eをc(e)本の多重辺に置き換えて得られるグラフ$\bar{G}=(V,\bar{E})$に対し，$\lambda(\bar{G})=c(N)$であるのは明らかである．従って，前節のアルゴリズムTESTECを用いれば，c(N)はO(|$\bar{E}$|+c(N)|V|^2)=O(Σc(e)+c(N)|V|^2)時間で求まる．しかし，この計算の手間はlog cの多項式ではない．この原因はFORESTの実行にO(|$\bar{E}$|)時間要するためである．

この点を改善するために，FORESTを直接ネットワークN=(G=(V,E),c)に適用しても正しく作動するよう修正する．$\bar{G}=(V,\bar{E})$を仮想しながらNにFORESTを適用してみよう．Nのある未走査辺e=(x,y)（$\bar{G}$では$|\bar{E}_{xy}|=c(e)$なる未走査辺集合）を走査する際，$\bar{G}$では$\bar{E}_{xy}$内の全ての辺は連続して走査されるが，Nに対するこの処理を単位時間で行うために，$\bar{E}_{xy}\cap E_i\neq\phi$なる最大のiだけをq(e)として保持しておく（$c(e)=|\bar{E}_{xy}|$を記憶しておけば，得られたq(e),e∈Eから$\bar{E}$の分割$\bar{E}_1,\bar{E}_2,\ldots$を求めることもできる）．このようにFOREST

を容量付きネットワークNが入力できるように修正したアルゴリズムCAPFORESTを以下に記述しておく.

```
Procedure CAPFOREST; {入力: N=(G=(V,E),c), 出力: q(e),e∈E}
begin
  全ての節点v∈Vに対して r(v):=0;
  全ての辺e∈Eに対して q(e):=0;
  while 未走査節点が存在すれば do
    begin
      最大ラベルrの"未走査"節点xを選ぶ;
      for "未走査"辺によりxに隣接している各節点yに対し do
                                        {E_xy内の辺は全て未走査}
        for 各辺e∈E_xyに対し do
          begin
            q(e):=r(y)+c(e);
            r(y):=r(y)+c(e);
            辺eを"走査済"とマークする
          end;
      xを"走査済"とマークする
    end;
end.
```

このCAPFORESTの計算時間は未走査節点の集合をフィボナッチヒープ[5]で管理しておくことにより$O(|E|+|V|\log|V|)$に抑えることができる[13].

辺連結度を求めるアルゴリズムTESTECに対してもNに直接適用できるよう同様の修正を施してやれば, CAPFORESTを$|V|-1$回用いて整数値ネットワークNの辺連結度に相当するc(N)を計算するアルゴリズムMINCUTが次のように得られる[13]. ただし, $\delta(N)\equiv \min\{\Sigma_{e\in E(v)}c(e)\mid v\in V\}$ とする.

```
Procedure MINCUT; {入力: N=(G=(V,E),c), 出力: c(N)および最小カットF}
begin
  N':=N; k:=+∞;
  while N'=(G'=(V',E'),c')において |V'|≧3 なら do
    begin
      N'にCAPFORESTを適用しq(e),e∈E'を見出す;
      Σ_{e∈E'(w)}c(e)=δ(N')なる節点w∈V'を選ぶ;
      if δ(N')<k then F:=E'(w);
      k:=min{k, δ(N')};
      q(e)≧δ(N')なる辺e=(u,v)∈E'を選び G':=G'/{u,v};
      N':=(G',c')
```

```
    end;
  {|V'|=2よりc(N')=δ(N')}
  if δ(N')<k then F:=E';
  k:=min{k, δ(N')};
  c(N)=kおよびFはNの最小カットと結論する
end.
```

明らかに，アルゴリズムMINCUTの計算時間は$O(|V||E|+|V|^2\log|V|)$で済む．MINCUTが有理数値容量を持つネットワークに対しても正常に作動することは容易に分かるであろう．さらに，容量が一般の実数であっても正しく答えを出力することも有理数近似の考え方を用いて証明されている[13]．

MINCUTの結果を用いると前節の$\lambda(G)$を求める手間を若干改善することができる．与えられたグラフ$G=(V,E)$から，各2点間の多重辺E_{uv}を$c(e)=|E_{uv}|$なる1本の辺$e=(u,v)$に置き換えて得られる容量付きネットワークN_Gのに対しMINCUTを適用してやれば，$O(|E|+p|V|+|V|^2\log|V|)$時間で$\lambda(G)=c(N_G)$が得られる（ただし，pは間に辺の存在する点対の総数である）．TESTECとMINCUTをうまく切り替えて$\lambda(G)$を見いだす$O(|E|+\min\{\lambda(G)|V|^2, p|V|+|V|^2\log|V|\})$時間のアルゴリズムを構成することができる[13]．

8．グラフの最小カットの総数を数えるアルゴリズム

グラフGの辺連結度$\lambda(G)$に対して明らかに，$C^k(G)=\phi$（$k<\lambda(G)$のとき），$C^k(G)\neq\phi$（$\lambda(G)\leq k\leq|E|$のとき）が成立するが，グラフの持つ最小カットの総数$|C^{\lambda(G)}|$を計算することは，辺の故障率を導入した確率グラフの連結確率の近似計算[16]やsuper-λと呼ばれるグラフの信頼性の検証[19,20]を行う際に必要となる．これまでに2点s,tを分離するs-t最小カットの総数$\mu=|C^k(s,t;G)|(k=\lambda(s,t;G))$を計算する$O(|E||V|+\mu|E|)$時間アルゴリズムやグラフの最小カットの総数$|C^k(G)|(k=\lambda(G))$を求める$O(|V|^2|E|)$時間のアルゴリズムが知られている[1]．

本節では，FORESTの性質に基づいてこれら2つのアルゴリズムの計算時間が短縮できることを示す．定理3.1の条件を満たすグラフGの辺集合Eの分割$E_1,E_2,\ldots,E_{|E|}$に対して次が成り立つことは定理3.1と類似の議論により証明できる．

【補題8.1】[14] $G_i=(V,E_1\cup E_2\cup\ldots\cup E_i)$としたとき，任意の2節点x,yおよび$\lambda(x,y;G)<i\leq|E|$なるiに対し，$C^k(x,y;G_i)=C^k(x,y;G)$ $(k=\lambda(x,y;G))$が成り立つ．□

BallとProvanによるs-t最小カット数μを数えるアルゴリズムは，s,t間の辺素なパスを最大本数求めた後，このパス集合に基づいてs-t最小カットを1つずつ線形時間で枚挙していくのもので，その手間$O(|E||V|+\mu|E|)$は，s,tの最大本数の辺素なパスを見つける手間をTとすると正確には$O(T+\mu|E|)$と表わせる[1]．従って，Tには5節の結果を用い，グラフを補題8.1より$G_{k+1}(k=\lambda(s,t;G))$に縮小しておけば，彼らのアルゴリズムの手間は，$O(|E|+\min\{\lambda(s,t;G),|V|^{2/3}\}m')$（グラフが単純のとき）および$O(|E|+\min\{|V|,\lambda(s,t;G),\sqrt{m'}\}m')$（グラフが多重のとき）に改善できる．ただし，$m'=\min\{|E|,k|V|\}$．グラフの最小カットの集合の総数$|C^k(G)|(k=\lambda(G))$を計算する

BallとProvanのアルゴリズムは次の容易に示せる事実に基づいている．すなわち，$|C^k(G)|(k=\lambda(G))$は，$|C^k(G/\{u,v\})|(\lambda(u,v;G)>k$のとき)，あるいは$|C^k(u,v;G)|+|C^k(G/\{u,v\})|$ $(\lambda(u,v;G)=k$のとき)に等しい．従って，$|C^k(G)|$を求めるには，まず$\lambda(G)$を決定した後（これは$O(|V|T)$時間で済む[4,17]），任意の２点u,v間の局所辺連結度$\lambda(u,v;G)$を調べ，$\lambda(u,v;G)>\lambda(G)$ならu,vを縮約し，そうでなければ$|C^k(u,v;G)|$を計算した後u,vを縮約するという操作を繰り返せば良い．このとき要する手間は，u-v最小カットが１つ当り線形時間で見出せることから$O(|V|T+|C^k(G)||E|)$で済む．ここで，$|C^k(G)|$が$|V|,|E|$の指数関数になる心配があるが,Bixby[2]により常に$O(|V|^2)$で抑えられることが知られている．この事実と文献[4]の$T\leqq O(|V||E|)$の結果からBallとProvanは当時$O(|V|^2|E|)$時間のアルゴリズムを与えたことになる．

この計算時間をFORESTにより改良するポイントは以下の２つである．

(1)補題8.1によりグラフを縮小しておく．

(2)補題6.2を用いる（すなわち，指定した２点u,v間の辺素なパスを求めるのではなく，FORESTにより線形時間で補題6.2を満たす２点見出す）．

(1),(2)の工夫により$|C^k(G)|(k=\lambda(G))$の計算は$O(|E|+\lambda(G)|V|^2+\lambda(G)|C^k(G)||V|)$時間に短縮できる[14]．

９．おわりに

グラフ探索アルゴリズムには，深さ優先探索，幅優先探索がよく知られており，これらは２点連結性や距離レベルなどのグラフの有用かつ基本的な性質を線形時間で見出す重要な役割を持っている．本稿で与えたアルゴリズムFORESTもグラフ探索の１つであり，主にグラフの連結度を調べるのに役立つ基本的な算法と考えられる．今後もこのアルゴリズムの利用可能な問題を調べていく予定である．

謝辞　日頃温かい御指導を賜る豊橋技術科学大学楠菊信教授に感謝致します．本研究は一部文部省科学研究費および中部電力助成金の援助のよる．

文　献

[1] M.O.Ball and J.S.Provan: "Calculating bounds on reachability and connectedness in stochastic networks", Networks, Vol.13 (1983) 253-278.

[2] R.E.Bixby: "The minimum number of edges and vertices in a graph with edge connectivity n and m n-bond", Networks, Vol.5 (1975) 253-298.

[3] F.T.Boesch: "Synthesis of reliable networks - A survey", IEEE Trans. Reliability, Vol.R-35 (1986) 240-246.

[4] S.Even and R.E.Tarjan: "Network flow and testing graph connectivity", SIAM J.Computing 4 (1975) 507-518.

[5] M.L.Fredman and R.E.Tarjan: "Fibonacci heaps and thier uses in improved optimization algorithms", 25th Symp. Found. Compt. Sci. (1984) 338-346.

[6] Z.Galil: "Finding the vertex connectivity of graphs", SIAM J.Computing 9

(1980) 197-199.

[7] M.R.Garey and D.S.Jhonson: Computers and Intractability, A Guide to the Theory of NP-completeness, Freeman, W.H. and Company, 1979.

[8] A.V.Goldberg and R.E.Tarjan: "A new approach to the maximum flow problem", Proc. 18th ACM Symp. on the Theory of Comput. (1986) 136-146.

[9] R.E.Gomory and T.C.Hu: "Multi-terminal network flows", J.SIAM Vol.9 (1961) 551-570.

[10] 近藤 健次: "k－連結全域部分グラフを求めるアルゴリズム(FOREST)の性能の検討", 豊橋技術科学大学情報工学科, 卒業論文(1990)

[11] D.W.Matula: "Determining edge connectivity in O(nm)", Annu. Symp. Found. Compt. Sci. 28th (1987) 249-251.

[12] H.Nagamochi and T.Ibaraki: "Linear time algorithms for finding k-edge-connected and k-node-connected spanning subgraphs", Technical Report #89006, Dept. of Applied Mathematics and Physics, Faculty of Engineering, Kyoto University (1989).

[13] H.Nagamochi and T.Ibaraki: "Computing edge-connectivity in multiple and capacitated graphs", Technical Report #89009, Dept. of Applied Mathematics and Physics, Faculty of Engineering, Kyoto University (1989).

[14] H.Nagamochi, Z.Sun and T.Ibaraki: "Counting the number of minimum cuts in multiple undirected graphs", Technical Report #89010, Dept. of Applied Mathematics and Physics, Faculty of Engineering, Kyoto University (1989).

[15] T.Nishizeki and S.Poljak: "Highly connected factors with a small number of egdes", Working paper (1989).

[16] J.S.Provan: "Bounds on the reliability of networks", IEEE Trans. Reliability, Vol.R-35 (1986) 260-268.

[17] C.P.Schnorr: "Bottlenecks and edge connectivity in unsymmetrical networks", SIAM J. Computing Vol.8 (1979) 265-274.

[18] Y.Shiloach: "A polynomial solution to the undirected two paths problem", J. of the Association for Computing Machinery, Vol.27 (1980) 445-456.

[19] 孫, 永持, 楠: "与えられた基数のカットセット数を最小にするグラフ", 信学論, Vol.J72-A (1989) 1601-1610.

[20] Z.Sun, H.Nagamochi and K.Kusunoki: "Multiple graphs minimizing the number of minimum cut-sets", 信学論 (submitted).

[21] 鈴木, 高橋, 西関: "3－連結全域部分グラフを求めるアルゴリズム", 情報処理学会研究報告 89-AL-7-2 (1989).