Graph など

Ryo Kawai

2021 年 10 月 4 日

前書き (memo)

TeX の環境をいじって色々試しているため, とても奇妙な PDF になってしまっている．環境としては, エディタとして VScode(LaTeX Workshop,LaTeX Utilities), TeX として TeXLive2021 を使用している．オンラインで動くので Overleaf とかも使うと良さそう．mac のみだが TeXPad が面倒な環境構築もいらずに, しかもとても便利そうだが, 日本語がもしかしたら面倒かもしれないのと, 有料 (PC:4040 円, ipad3180 円).

基本的に, 呼び名が複数あるものは最初に表記してある方で統一する.

雑多な日記

やっと TeX の移植が終わった．User Snippets は article は更新中．Beamer を余裕があれば作る．(09/20)

GitHub を活用していきたいと思いアカウントは作成したが, push に慣れていないので, GitHub で共有できるようになるのはずいぶん先になりそう．graphicx がなぜかエラーを吐くのはなぜなのか．参考文献を BibTeX をつかってやろうとしたら, エラーは出ないが参考文献が表示されないようになった．(09/23)

多分マクロをいじる必要がある (pLaTeX が処理されていない気がする)． 索引も同様 [11]． また, 洋書の出版社や著者名が怪しい．graphicx と mendex は対応するものがないとエラーを吐くっぽい．or,and などを描くのがめんどくさかったので論理記号で書いてしまった．(09/24)

出るようになった [5][6]．(09/25)

今度は figure がだめになった．jsarticle には chapter はないが part はある．本文をそのまま引用できる仕組みはないものか．(09/26)

崎が出る. 数式入りの footnote を数式内部に入れるとバグる．local history は無限にファイル生成する可能性が高いのでやめておくように．定義 1.2.1 など (09/27)

おかしなことに, index を begin の横に書いている. これはまずいが，とりあえず全部書いてから他のマクロでエラーが出ないか実験してから置換することにしよう． ちなみに index は). や)\ の間に入れると文字を詰めてしまうしたまにエラーを出すから注意する． なんで目次に索引と参考文献が出ないんだろうか．(09/28)

git を test している. 差分の参照などがよくわかっていない．(09/29)

git を private で作ってしまったので共有ができない． また, どうやら github 上だと hyperref が死んでしまうようだ． 今は A-Bpath のように書いているが. $A-B$path のようにした方がいいのか．強調 emph テスト．**通常**-*normal*数式 $-$ $mathematics$ 太字 $-$ $\boldsymbol{bm}$．英語と日本語が混じると統一感がでない問題がある. 今現在は textbf と bm を使っている (textbf だと数式が太字にならない). (09/30)

強調テスト．**通常-normal数式** $-$ $\boldsymbol{mathematics}$ **太字** $-$ $\boldsymbol{bm}$．ついにできた, 完璧だ. 数式モードでないときの英語も斜体にできる．**通常-*normal*数式** $-$ $\boldsymbol{mathematics}$ **太字** $-$ $\boldsymbol{bm}$．index では @,!,—,etc. などが特殊な役割を持つので, 数式モードで扱う時には””でエスケープするか他の textbar

などでだいようする. なぜか textbar が j になる, なぜ. どうやら vert だとうまくいく, package がわるいのか. (10/01)

同じ label をつけてもエラーは出ないが, ref では文章全体で一番最後の番号が参照される. Tikz で sin などを使う時には注意する, () の中に () は入れられない. 極座標表示にした方がいいかもしれない. (canvas polar cs:angle,radius) は (a:r)[7]. index では, 完全一致しない限りは別物だと判定される. (10/02)

目次

第 I 部
Graph

1　Graph のモチベーションと定義

1.1　モチベーション

いつか描きたい. いくつか例をあげる.

例 1.1.1 (The problem of the bridges of Königsberg(ケーニヒスベルグの橋の問題))**.** 有名な一筆書き問題

1.2　定義

数学的に扱いやすいように, グラフという言葉をきちんと定義していきたい. しかし単にグラフといっても実は様々な区別ができ, それに応じて多くの定義がある. ここではまず, そのようなさまざまなグラフの定義をしておく.

まず初めにこの PDF で通常使用するグラフの定義をしておく.

定義 1.2.1 (graph(グラフ))**. graph(グラフ)** , または **simple(単純)** グラフとは, **vertex(頂点)** と呼ばれる object の集合 V と, V の異なる 2 元からなる部分集合である **edge(辺)** の集合 E の組 $G = (V, E)$ のことである. すなわち $V, E : set$ が

$$E \subseteq \binom{V}{2}$$

[*1] を満たすとき, $G = (V, E)$ は graph であるという.

表記上の曖昧さを回避するために $V \cap E = \emptyset$ とする[*2].

定義 1.2.2. vertex の集合を **vertex set(頂点集合)** , edge の集合を **edge set(辺集合)** とよぶ. すなわち, 定義 1.2.1 の V, E はそれぞれ頂点集合, 辺集合である. 頂点集合 V を持つグラフを, **graph on V(V 上のグラフ)** という.

表記 1.2.3. グラフ G の頂点集合を $\boldsymbol{V(G)}$, 辺集合を $\boldsymbol{E(G)}$ で表す. 厳密に区別せず, 意味が明白な場合は $v \in V(G)$ を $v \in G$ と書いたり, $e \in E(G)$ を $e \in G$ と書いたりする.

定義 1.2.4. グラフ G の頂点の数 (頂点集合の濃度) を **order(位数)** といい, $\boldsymbol{|G|}$ で表す. 辺の数は $\boldsymbol{\|G\|}$ で表す.

[*1] $\binom{A}{k} = \{X \subset A : |X| = k\}$

[*2] 感覚的には頂点の集合と辺の集合は共通部分を持たないで欲しいので, 明記しておく. 例えば公理的集合論では $2 = \{0, 1\}$ であるから $V = \{0, 1, 2\}, E = \{\{0, 1\}\}$ とすると $V \cap E \neq \emptyset$ となってしまうような事態が発生する.

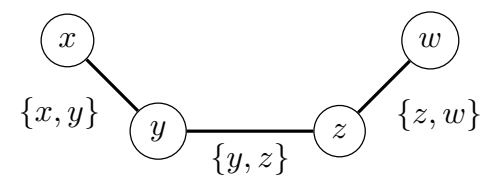
位数と辺の数が有限なグラフを **finite graph(有限グラフ)** , そうでないグラフを **infinite graph(無限グラフ)** という． グラフの定義から, 位数が有限なら辺の数も有限であるが, この後出てくる多重辺を持つグラフにも対応できるようにこのように定義する.

$(\emptyset, \emptyset)$ を **empty graph(空グラフ)** といい単に $\emptyset$ で表す． また, 位数が 0 または 1 であるグラフを **trivial(自明)** なグラフという (自明なグラフといったときには, 空グラフを無視することがある).

今後は基本的に有限グラフを扱うことにして, 無限グラフについては §9 に任せることにする.

通常は, グラフを絵で表すときには頂点を点で, 辺を頂点同士を結ぶ線で表す (絵で表されたグラフを考えるために数学の用語で整備した感じもするが)． この時に辺の形や点の位置は重要ではなく, どの頂点が結ばれているかが重要である.

例 1.2.5.



上のグラフは $V = \{x, y, z, w\}$ 上のグラフで, 辺集合は $E = \{\{x, y\}, \{y, w\}, \{w, z\}\}$ である.

通常のグラフ理論で考えられるグラフは前述のようなモデルだが, 例えば例 1.1.1 のように, ある 2 つの頂点を結ぶ辺が 2 本以上ある場合 (この辺のことを **multiple edges(多重辺)** という) や, 同じ頂点を結んでいる辺がある場合 (この辺のことを **loop(ループ)** という) は, 前述のグラフの定義では扱うことができない． このような多重辺やループを持つグラフを **multigraph(マルチグラフ)** という. グラフとマルチグラフの混同を防ぐために, グラフを単純グラフと言うことが多い.

マルチグラフの定義をしておこう.

定義 1.2.6 (multigraph(マルチグラフ))**. multigraph(マルチグラフ)** とは, 2 つの非交な集合 V, E(これらの要素をそれぞれ vertex,edge と呼ぶ) と, 辺に対して接続している頂点を対応させる写像 φ の組 $G = (V, E, \varphi)$ のことである． すなわち $V, E : set, \varphi : map$ が

$$V \cap E = \emptyset \ \wedge \ \varphi : E \to V \cup \binom{V}{2}$$

を満たすとき, $G = (V, E, \varphi)$ は multigraph であるという.

このように定義すると, ちゃんと多重辺やループを扱うことができる． 定義 1.2.1 では辺そのものが接続している頂点の情報を持っていたが, 定義 1.2.6 では φ が辺の接続している頂点の情報を持っているからである． ループは φ によって V に行くことに注意する.

マルチグラフについては §5 で詳しく触れる.

また, グラフの辺に向きをつけたモデルを考えたいときもある． そのような向きのついたグラフのことを **directed graph(有向グラフ)** といい, 省略して **digraph** とよくいう． この PDF でも digraph で統一する.

有向グラフの定義をしておこう.

定義 1.2.7 (digraph(有向グラフ))**. digraph(有向グラフ)** とは, **vertex(頂点)** と呼ばれる object の集合 V と, V の異なる 2 元の組である **arc(有向辺)** の集合 E の組 $D = (V, E)$ のことである. すなわち $V, E : set$ が

$$E \subseteq V^2 \backslash \{(x, x) : x \in V\}$$

を満たすとき, $D = (V, E)$ は digraph であるという.

表記上の曖昧さを回避するために $V \cap E = \emptyset$ とする.

定義 1.2.8. arc の集合を **arc set(有向辺集合)** という.

だいたいは $E \neq \emptyset$ の場合を考える. 図で表す際には, $(u, v) \in E$ を u から v への矢印で表すことが多い. 定義より, 辺 (u, v) と (v, u) は区別され, これらは向きを持っていると考えることが出来る. また, (u, v) と (v, u) のように向きが逆な辺であれば、2 頂点間に辺が 2 本接続することが可能である.

この定義は [8] にのっとったものだが, [10] などではグラフに向きをつけたものとして定義してある. すなわち $\{u, v\} \in E \ \Rightarrow \ \{v, u\} \notin E$ となっている. ここでは必要であればこの条件をつけることで対応する.

有向グラフについては §6 で詳しく触れる.

当然向きのついたマルチグラフも考えれるわけで, これを **quiver(箙)** という. 本によってさまざまであり, digraph を quiver と呼んでいたり, digraph を多重辺やループを含めて定義していたりもするが, この PDF ではこの呼び名で統一する[*3].

箙の定義をしておこう.

定義 1.2.9 (quiver(箙))**. quiver(箙)** とは, 2 つの非交な集合 V, E(これらの要素をそれぞれ vertex,arc と呼ぶ) と, 有向辺に対して接続している頂点を対応させる写像 φ_1, φ_2 の組 $G = (V, E, \varphi_1, \varphi_2)$ のことである. すなわち $V, E : set, \varphi_1, \varphi_2 : map$ が

$$V \cap E = \emptyset \ \wedge \ \varphi_1, \varphi_2 : E \to V$$

を満たすとき, $G = (V, E, \varphi_1, \varphi_2)$ は quiver であるという.

ただし, $v_1, v_2 \in V, \ \varphi_1(v_1) = \varphi_2(v_1), \ \varphi_1(v_2) = \varphi_2(v_2)$ のとき $\varphi_1(v_1) = \varphi_1(v_2) \ \Rightarrow \ v_1 = v_2$ と定める.

このように定義すると, ちゃんと多重辺やループを扱うことができる. 定義 1.2.7 では有向辺そのものが接続している頂点の情報を持っていたが, 定義 1.2.9 では φ_1, φ_2 が辺の接続している頂点の情報を持っているからである. 最後の条件は, 同じ頂点でのループが一つに定まることを意味している. まだあまり箙を勉強していないのでわからないが, これはいらないかもしれない.

箙については §8 で詳しく触れる.

[*3] 物理や環論ではどうやら有向なマルチグラフを quiver と呼ぶことが多いらしいので, こうした.

2 graph

2.1 basis

定義 2.1.1 (incident)**.** グラフにおいて, $v \in e$ であるとき, 頂点 v は辺 e に **incident(接続)** しているといい, e を v の **incident edge(接続辺)** という. 一つの辺に接続する 2 つの頂点をその辺の **end(端点)** といい, 辺はその端点を **join(結ぶ)** という.

辺 $\{x, y\}$ をよく省略して $xy(= yx)$ と表す. $x \in X \ \wedge \ y \in Y (X, Y \subseteq V)$ であるとき, 辺 xy を $\boldsymbol{X}$**-**$\boldsymbol{Y}$ **edge(**$\boldsymbol{X}$**-**$\boldsymbol{Y}$ **辺)** という. E に属する X-Y 辺全体の集合を $\boldsymbol{E(X, Y)}$ と表し, $E(\{x\}, Y)$ や $E(X, \{y\})$ のことを単に $E(x, Y)$ や $E(X, y)$ と表す. また, $v \in V$ の E 上の接続辺全体を $\boldsymbol{E(v)}$ と表す. すなわち $E(v) = E(V, v)$ である.

定義 2.1.2 (adjacent)**.** 2 つの頂点 x, y が $\{x, y\} \in G$ であるとき, x と y は **adjacent(隣接)** しているといい, 互いに他の **neighbour(隣接点)** という. また, 2 つの異なる辺 e, f が 1 つの端点を共有しているとき, すなわち $^{\exists} v \in G \ s.t. \ v \in e \ \wedge \ v \in f$ であるときも e と f は **adjacent(隣接)** しているという.

定義 2.1.3 (complete graph)**.** 全ての頂点が隣接しているグラフを **complete graph(完全グラフ)** といい, $|G| = n$ のものを $\boldsymbol{K^n}$ で表す[*4]. 特に K^3 は triangle(三角形) と呼ばれる.

定義 2.1.4 (independent)**.** グラフ G の頂点で, 他のどの頂点とも隣接していない頂点を **independent(独立)** した頂点という. 同じように, グラフ G の辺で, 他のどの辺とも隣接していない辺を **independent(独立)** した辺という. より一般に, $X \subseteq V(G) \veebar E(G)$ のすべての要素が独立しているときに X は **independent(独立)** しているという. $V(G)$ が独立しているとき, **stable set(安定集合)** ということもある.

表記 2.1.5. 2 つのグラフ $G = (V, E), G' = (V', E')$ に対して, グラフの間の写像 $\varphi : V \to V'$ を $\varphi : G \to G'$ と表す.

定義 2.1.6 (graph isomorphism)**.** 2 つのグラフ $G = (V, E), G' = (V', E')$ に対して, $\varphi : G \to G'$ が

$$\{x, y\} \in E \ \Rightarrow \ \{\varphi(x), \varphi(y) \in E'\}$$

を満たすとき, φ を **graph homomorphism(グラフ準同型写像)** であるという. 特にこのとき, $x' \in V'$ の φ による逆像 $\varphi^{-1}(x')$ は独立している.

φ が全単射であり φ^{-1} もグラフ準同型写像であるとき, φ を **graph isomorphism(グラフ同型写像)** という. またこのとき, G と G' は **graph isomorphic(グラフ同型)** であるといい, $\boldsymbol{G \simeq G'}$ と書き表す. 同型なグラフは区別せず, $G \simeq G'$ のことを, $G = G'$ と書くことが多い.

[*4] K_n と表記している本もある.

G から G へのグラフ同型写像を **automorphism(自己同型写像)** という.

定義 2.1.7. 同型写像の下で保存されるような性質を **graph property(グラフの性質)** といい, その中で引数を持つものを **graph invariant(グラフ不変量)** という.

グラフの頂点の数や辺の数などはグラフ不変量である.;

定義 2.1.8 (subgraph)**.** 2 つのグラフ $G = (V, E), G' = (V', E')$ が, $V' \subseteq V \ \wedge \ E' \subseteq E$ であるとき, G' を G の **subgraph(部分グラフ)** であるといい, $\boldsymbol{G' \subseteq G}$ で表す. よく G は G' を **contain(含む)** ともいう.

$G' \subseteq G \ \wedge \ G' \neq G$ であるとき, すなわち $G' \subsetneq G$ であるとき, G' を G の **proper subgraph(真な部分グラフ)** という,

表記 2.1.9. 2 つのグラフ $G = (V, E), G' = (V', E')$ に対して,

$$G \cup G' := (V \cup V', E \cup E')$$
$$G \cap G' := (V \cap V', E \cap E')$$

のように書き表す. これらは確かめるとグラフになっている.

$G \cap G' = \emptyset$ であるとき, G と G' は **disjoint(非交)** であるという.

2.2 path and cycle

定義 2.2.1 (path)**.** 空でないグラフ $P = (V, E)$ が

$$V = \{x_0, x_1, \cdots, x_k\}, E = \{x_0x_1, x_1x_2, \cdots, x_{k-1}x_k\} \ (x_0, x_1, \cdots, x_k\text{はすべて異なる})$$

とかけるとき, P を **path(道)** という.

またこの P について, x_0 と x_k は P で **link(結ばれている)** という. x_0 と x_k を P の **end(端点)** といい, $x_1, \cdots, x_{k-1}$ を P の **inner vertex(内点)** という.

複数の道が互いに内点を含まないとき, それらを **independent(独立)** な path といい, それぞれの path は **independent(独立)** であるという.

表記 2.2.2. 定義 2.2.1 の P を簡単に $P = x_0x_1 \cdots x_k$ と書いて, x_0 から $_k$ までの path という. また、$0 \leq i \leq j \leq k$ に対して,

$$Px_i := x_0 \cdots x_i$$
$$x_iP := x_i \cdots x_k$$
$$x_iPx_j := x_i \cdots x_j$$

のように書き表す. 他にも, 直観的にわかりやすいため, path $P(\ni x), Q(\ni x, y), R(\ni y)$ に対して $Px \cup xQy \cup yR$ を $PxQyR$ と書き表す.

定義 2.2.3 (A-B path)**.** 頂点集合 A, B に対して, path $P = x_0x_1 \cdots x_k$ が

$$V(P) \cap A = \{x_0\} \ \wedge \ V(P) \cap B = \{x_k\}$$

であるとき, P を ***A*-*B* path(*A*-*B* 道)** という.

表記 2.2.4. 上の $A = \{a\}$ のときは, $\{a\}$-B path の意味で単に a-B path と書く. また, A, B がグラフであるとき, $V(A)$-$V(B)$ path を単に A-B path と書く.

定義 2.2.5 (H-path)**.** グラフ H に対して, その端点のみで H と接しているような自明でない path のことを **H-path(*H*-path)** という. すなわち, path $P = x_0x_1 \cdots x_k$ が

$$P \cap H = (\{x_0, x_k\}, \emptyset) \ \wedge \ |P| > 1$$

であるとき, P は H-path であるという.

定義より, 長さが 1 の H-path x_0x_1 の辺は H の辺にはならない.

定義 2.2.6 (cycle)**.** 空でないグラフ $C = (V, E)$ が

$$V = \{x_0, x_1, \cdots, x_{k-1}\}, E = \{x_0x_1, x_1x_2, \cdots, x_{k-1}x_0\} \ (x_0, x_1, \cdots, x_k\text{はすべて異なる}, \ k \geq 3)$$

とかけるとき, C を **cycle(閉路)** という.

いいかえれば, path $P = x_0x_1 \cdots x_{k-1} (k \geq 3)$ に対して $C := P + x_0x_{k-1}$ を cycle という. 未定義語 (+)

2.3 degree

定義 2.3.1 (neighbours)**.** グラフ G の頂点集合 U に対して, その頂点の隣接点で U に属さないもの全体, すなわち,

$$N(U) = \{x \in V(G \backslash U) :^{\exists} y \in U \ s.t. \ xy \in E(G)\}$$

をグラフ G における U の **neighbours(近傍)** という.

特に $U = \{u\}$ のとき, $N(\{u\})$ を単に $N(u)$ と表す. すなわち $N(u)$ は u に隣接する頂点全体である.

定義 2.3.2 (degree)**.** グラフ G の頂点 v に対して, その頂点の接続辺の数, すなわち,

$$d(v) = |E(v)| = |\{vx \in E(G) :^{\exists} x \in V(G)\}|$$

をグラフ G における x の **degree(次数)** という.

(単純) グラフの場合, 頂点 v に対して, 接続する辺の数と隣接する頂点の数は等しいため, $d(v) = |N(v)|$ が成り立つ.

2.4 connectivity

定義 2.4.1 (connected)**.** グラフ G が **connected(連結)** であるとは, G の任意の 2 頂点 x, y に対してその 2 点を結ぶ G 上の path が存在することである. すなわち,

$$^{\forall}x, y \in G, ^{\exists} P \subset G : \text{path } s.t.\ P = x \cdots y$$

であるとき, G は connected であるという.

また, グラフ G が連結でないとき, グラフ G は **disconnected(非連結)** であるという.

定義 2.4.2. グラフ G の空でない極大な連結部分グラフを G の **component(連結成分)** という. すなわち, 各連結成分は共通部分を持たない. また, 空グラフは連結成分を持たないことに注意する.

定義 2.4.3 (k-connected)**.** $k \in \mathbb{N}, |G| > k$ で, $|X| < k$ [*5] である任意の頂点集合 (グラフ)X に対して $G - X$ が連結であるとき, グラフ G は ***k*-connected(*k*-連結)** であるという. 未定義語 (-)

X は任意だが, 仮に G の部分グラフではない A を取ったとしても $|G| > k, |A| < k$ より $G \cap A \subset B \subset G, |B| < k$ となる B が取れ, $G - A \supset G - B$ である. そのため, $X \subset G$ としても問題はない. 定義より, 全ての空でないグラフは 0-連結であり, 全ての連結なグラフは 1-連結である. また定義より $n, m \in \mathbb{N}, n < m$ のとき, グラフ G が m-連結ならば G は n-連結である.

定義 2.4.4. 定義 2.4.3 より, グラフ G は有限なので $\{x \in \mathbb{N} : G \text{ は } x\text{-連結 }\}$ は最大値をもつ. その値をグラフ G の **connectivity(連結度)** といい, $\boldsymbol{\kappa(G)}$ で表す. すなわち, グラフ G に対して k-連結であるが $k+1$-連結でない $k \in \mathbb{N}$ が存在し, $\kappa(G) = k$ を G の connectivity という.

2.5 contraction

定義 2.5.1. グラフ $G = (V, E)$ とその辺 $xy \in E$ に対して, $v_{xy} \notin V$ として

$$(V \backslash \{x, y\} \cup \{v_{xy}\}, \{vw \in E : \{x, y\} \cap \{v, w\} = \emptyset\} \cup \{v_{xy}w : w \in V \backslash \{x, y\}\ s.t.\ xw \in E \vee yw \in E\})$$

すなわち

$$G - \{x, y\} \cup \{v_{xy}\} + \{v_{xy}w : w \in V \backslash \{x, y\}\ s.t.\ xw \in E \vee yw \in E\}$$

で与えられるグラフを $\boldsymbol{G/xy}$ で表す.

ここからすぐに次のことがわかる. $N(\{x, y\}) = N(v_{xy})$. $G - \{x, y\} = G/e - v_{xy}$.

note

参考にしたのは [10][8].

[*5] $0 \in \mathbb{N}$

3　2-connected graph

最も単純な 2-連結グラフは cycle である. 他のすべての 2-連結グラフも, cycle に path を加えることで作ることが出来る.

定理 3.0.1. cycle に対して, 次のことが言える.

(i) cycle は 2-連結.
(ii) cycle 上の任意の 2 点 x, y に対して, x と y を結ぶ C 上の path が 2 本存在し, それらは独立である.

Proof. cycle を $C = P + x_0x_{k-1}, P = x_0x_1 \cdots x_{k-1}$: path(k ≥ 3) とする. $V(C) = \{x_0, \cdots x_{k-1}\}$ であるから, C 上の点は $x_i (0 \leq i \leq k-1)$ の形で書ける. 今, C 上の任意の点 x_i に対して $C - x_i = x_{i+1} \cdots x_{k-1}x_0 \cdots x_{i-1}$ は path となるため, C は 2-連結である. また, C 上の任意の 2 点 x_i, x_j に対して $x_ix_{i+1} \cdots x_{j-1}x_j$ と $x_jx_{j+1} \cdots x_{k-1}x_0 \cdots x_{i-1}x_i$ は x と y を結ぶ C 上の独立な path となる. □

定理 3.0.2. (有限) グラフ G に対して, 以下は同値である.

(i) G は 2-連結.
(ii) G は閉路から始めて, すでに構築したグラフ H に H-path を加えることで構築できる.
(iii) G は連結で, G の任意の 2 点に対してそれを含む G 上の cycle が存在する.

Proof. (3)⇒(1):G が 2-連結でないとすると, 仮定より 1-連結であるため, $z \in G$ *s.t.* $G - z$: 非連結 が存在する. $G - z$ が非連結より, $G - z$ 上で結ばれていないような 2 点 x, y が存在する. すなわち $^{\exists}x, y$ *s.t.* $^{\forall}P : x - y$path, P $\notin$ G. 今, $x, y \in G - z \subset G$ と仮定より, x, y を含むような G 上の cycle C が存在する. $z \notin C$ とすると, $C \subset G - z$ より x, y を結ぶ $G - z$ 上の path が存在し, 矛盾する. $z \in C$ とすると, cycle は 2-連結より $C - z$ は連結で $x, y \neq z$ であるため, x, y を結ぶ $G - z$ 上の path が存在し, 矛盾する. したがって G は 2-連結であり, (3)⇒(1) が示せた.

(1)⇒(2):G を 2-連結グラフとすると, G は cycle を含む[*6]. したがって, (2) のように構築される極大な部分グラフ H を含む. 仮に $xy \in E(G)\backslash E(H),\ x, y \in H$ が存在するとすると xy は H-path となり, H の極大性に反する. そのため $x, y \in H \Rightarrow xy \in H$ であるから H は G の誘導部分グラフである. 今 $G \neq H$ とすると, H が G の誘導部分グラフであることから $|G| > |H|$ がわかる. よって $G - H \neq \emptyset$ であり, G の連結性より vw *s.t.* $v \in G - H \wedge w \in H$ が存在する. また G は 2-連結であるため, v-H path $P \subset G - w$ が存在する. このとき, wvP は H-path であり, G に含まれている. したがって $H \cup wvP$ は H より大きい (H を含む)(2) のように構築されるグラフであり, これは H の極大性に反する. したがって $G = H$ であり, (1)⇒(2) が示せた.

[*6] (3)⇒(1) より

(2)⇒(3):G を cycle とすると, 明らかに (3) を満たす. 今, H を (3) を満たすグラフ, $P = x \cdots y$ を H-path , すなわち $H \cap P = (\{x, y\}, \emptyset)$ とする. H が (3) を満たすことから, $^{\forall}z \in H$ に対して x と y を結ぶ path $Q = x \cdots z \cdots y$ $s.t.$ $Q \subset H$ が存在する[*7]. よって $P \cap Q = (\{x, y\}, \emptyset)$ より $P \cup Q$ は cycle となり[*8]. この cycle は P 上の任意の点と H 上の任意の点を含む cycle となる. また, この cycle は P 上の任意の 2 点を含む cycle にもなっている. したがって, $H \cup P$ は (3) を満たすため, (2) で構築されるグラフは (3) を満たすことがわかり, (2)⇒(3) が示せた.

(1)⇒(2)⇒(3)⇒(1) より, (1),((2),(3) は同値である. □

note

(1)⇔(3) が [1] にある.

4　3-連結グラフについて

4.1　準備

補題 4.1.1. G: グラフ, $e = xy \in G$,
G: 連結 ⇔ G/e: 連結

Proof. G: 連結とすると, $^{\forall}a, b \in G, ^{\exists}P$:$a$ と b を結ぶ G 上の path . ここで $\{a, b\} \cap \{x, y\} = \emptyset$, $P' = G/e[P \cup \{v_{xy}\}]$ とすると,

(i) $P \cap \{x, y\} = \emptyset$ のとき, $P \subset G - \{x, y\} = G/e - v_e \subset G/e$.
(ii) $P \cap \{x, y\} = \{x\}(\text{or } \{\text{y}\})$ のとき, P: 連結と $P \ni x(\text{or y})$ より P' は連結であり, $a, b \in P' \subset G/e$.
(iii) $P \cap \{x, y\} = \{x, y\}$ のとき, P: 連結と $P \ni x, y$ より P' は連結であり, $a, b \in P' \subset G/e$.

であるから, a と b を結ぶ G/e 上の path が存在することがわかる. また, $\{a, b\} \cap \{x, y\} = \{x(\text{or y})\}$ の場合は (2) の最後を $v_{xy}, b(\text{or a}, \text{v}_{\text{xy}}) \in \text{P}' \subset \text{G/e}$ とすればよい. $\{a, b\} \cap \{x, y\} = \{x, y\}$ の場合は a, b は G/e 上で一点 $v_{x,y}$ になる. よって G/e: 連結である. 逆も同様に示せる. □

補題 4.1.2. G: グラフ, $e = xy \in G$, $x, y, v_{xy} \notin S$:vertices set,
$(G - S)/e = G/e - S$

[*7] $x, z \in H$ より, x, z を含むような cycle C_1 が存在し, ゆえに x と z を結ぶような path P_1 が存在する
[*8] $Q = x \cdots z \cdots y'y$ として, $yPxQy' + y'y$ は cycle であり, $yPxQy' + y'y = P \cup Q$ である.

Proof.

$$\begin{aligned}V((G-S)/e) &= (V(G)\backslash S)\backslash\{x,y\}\cup\{v_{xy}\}\\ &= (V(G)\backslash\{x,y\}\cup\{v_{xy}\})\backslash S(x,y,v_{xy}\notin S\ \text{より})\\ &= V(G/e-S)\end{aligned}$$

$$\begin{aligned}E((G-S)/e) &= \{vw\in E(G-S)|\{x,y\}\cap\{v,w\}=\emptyset\}\cup\\ &\quad \{v_{xy}w|w\in V(G-S)\backslash\{x,y\}\ s.t\ xw\in E(G-S)\vee yw\in E(G-S)\}\\ &= \{vw\in E(G-S)|\{x,y\}\cap\{v,w\}=\emptyset\}\cup\\ &\quad \{v_{xy}w|w\in V(G-S)\backslash\{x,y\}\ s.t\ xw\in E(G-S)\vee yw\in E(G-S)\}\\ &= \{vw\in E|(\{x,y\}\cup S)\cap\{v,w\}=\emptyset\}\cup\\ &\quad \{v_{xy}w|w\in V(G-S)\backslash\{x,y\}\ s.t\ xw\in E\vee yw\in E\}\\ &\quad \because V(G-S)=V-S,\\ &\qquad w\in V(G-S)\wedge x,y\notin S\Rightarrow(xw(yw)\in E\Leftrightarrow xw(yw)\in E(G-S))\\ &= \{vw\in E|(\{x,y\}\cup S)\cap\{v,w\}=\emptyset\}\cup\\ &\quad \{v_{xy}w|w\in V\backslash\{x,y\}\ s.t\ (xw\in E\vee yw\in E)\wedge w\notin S\}\\ &= \{vw\in E(G/e)|\{v,w\}\cap S=\emptyset\}\\ &= E(G/e-S)\end{aligned}$$

□

補題 4.1.3. G: グラフ, $e=xy\in G$, $x,y,v_{xy}\notin S$:vertices set,
$(G-S)$: 連結 $\Leftrightarrow$ $G/e-S$: 連結

Proof. 補題 4.1.1, 補題 4.1.2 よりわかる. □

定理 4.1.4. G:3-連結グラフ,
$|G|>4\Rightarrow^{\exists} e\in E(G)\ s.t.\ G/e:3-$連結

Proof. そのような辺が存在しない, つまり $^{\forall}e=xy\in G\ s.t.\ \kappa(G/e)\leq 2$ とする. すなわち $^{\exists}S\subset G/e$: vertices set s.t. $|\mathrm{S}|\leq 2\wedge \mathrm{G/e-S}$: 非連結. また, $e=xy$ によって G を縮約した際に得られる頂点を v_{xy}と書き表すこととする. 今, $v_{xy}\notin S$ とすると, $S\subset G/e-v_{xy}=G-\{x,y\}$ より $x,y\notin S$ である. よって補題 4.1.3 より $G-S$: 非連結となり, $\kappa(G)\leq 2$ となり G:3-連結グラフに矛盾する. よって $v_{xy}\in S$ である. また $|S|=1$ すなわち $S=\{v_{xy}\}$ とすると, これも $G/e-S=G-\{x,y\}$ より $\kappa(G)\leq 2$ となり矛盾する. したがって $|S|=2\wedge v_{xy}\in S$ がわかる. S の元で v_{xy} ではないほうの頂点を z とすると, $z\notin\{x,y\}$ より $G/e-S=G/e-\{v_{xy}\}-\{z\}=G-\{x,y,z\}$ である.

以上をまとめると $^{\forall}x,y\in G\ s.t.\ xy\in E(G),\ ^{\exists}z,\ G-\{x,y,z\}$: 非連結 が導ける. ここで, $G-\{x,y,z\}$ は非連結より 2 つ以上の連結成分を持ち, G が 3-連結であることから x,y,z はすべての連結成分と隣接していることに注意する. ここで $G-\{x,y,z\}$ の中で一番位数が小さい連結成分を C とし, $|C|$ が最小になるように x,y を取り直す. $v\in V(C)\ s.t.\ vz\in E(G)$ とすると v の存在性は明ら

か. また, 仮定より G/vz は 3-連結ではないため, $^{\exists}w \in G\ s.t.\ G-\{z,v,w\}$: 非連結. x と y が隣接していることから, $G-\{z,v,w\}$ は $D\cap\{x,y\}=\emptyset$ となる連結成分 D をもつ. $u \in V(D)\ s.t.\ vu \in E(G)$ とすると u の存在性は明らかであり, $v \in V(C)$ より $u \in V(C)\ i.e.\ C\cap D \neq \emptyset$ がわかる. $x,y,z \notin D$ より D は $G-\{x,y,z\}$ の連結成分の部分グラフであるため, $D \subseteq C$. また $v \notin D,\ v \in C$ であるため $D \subsetneq C$ である. ゆえに C の最小性に反する. □

定理 4.1.5. G が 3-連結グラフ (ただし K^3 は除く) であるための必要十分条件は, 以下を満たすようなグラフの列 $G_0,\cdots,G_n$ が存在することである.

(i) $G_0 = K^4 \wedge G_n = G.$
(ii) $^{\forall}i < n, ^{\exists}xy \in E(G_{i+1}), (d(x),d(y) \geq 3 \wedge G_i = G_{i+1}/xy).$

Proof. 必要性 ($\Rightarrow$):G:3-連結 $\Rightarrow$ グラフの列 $G_0,\cdots,G_n$ が存在.

位数が 4 である 3-連結なグラフは K^4 のみであるから, 定理 4.1.4 より, 各 G_i:3-連結となるようなグラフの列 $G=G_n,\cdots,G_0=K^4(^{\exists}xy \in E(G_{i+1}), G_i = G_{i+1}/xy)$ が構成することができる, また, 任意のグラフ H に対し, $\kappa(H) \leq \lambda(H) \leq \delta(H) = \min\{d(x)|x \in H\}$ であるから, この列は (2) を満たす.

十分性 ($\Leftarrow$):G:3-連結 $\Leftarrow$ グラフの列 $G_0,\cdots,G_n$ が存在.

(2) のときに, G_i:3-連結 $\Rightarrow$ G_{i+1}:3-連結を示す. これが示せると K^4 は 3-連結であることとグラフの列が有限であることから帰納的に $G_n = G$:3-連結が導ける.

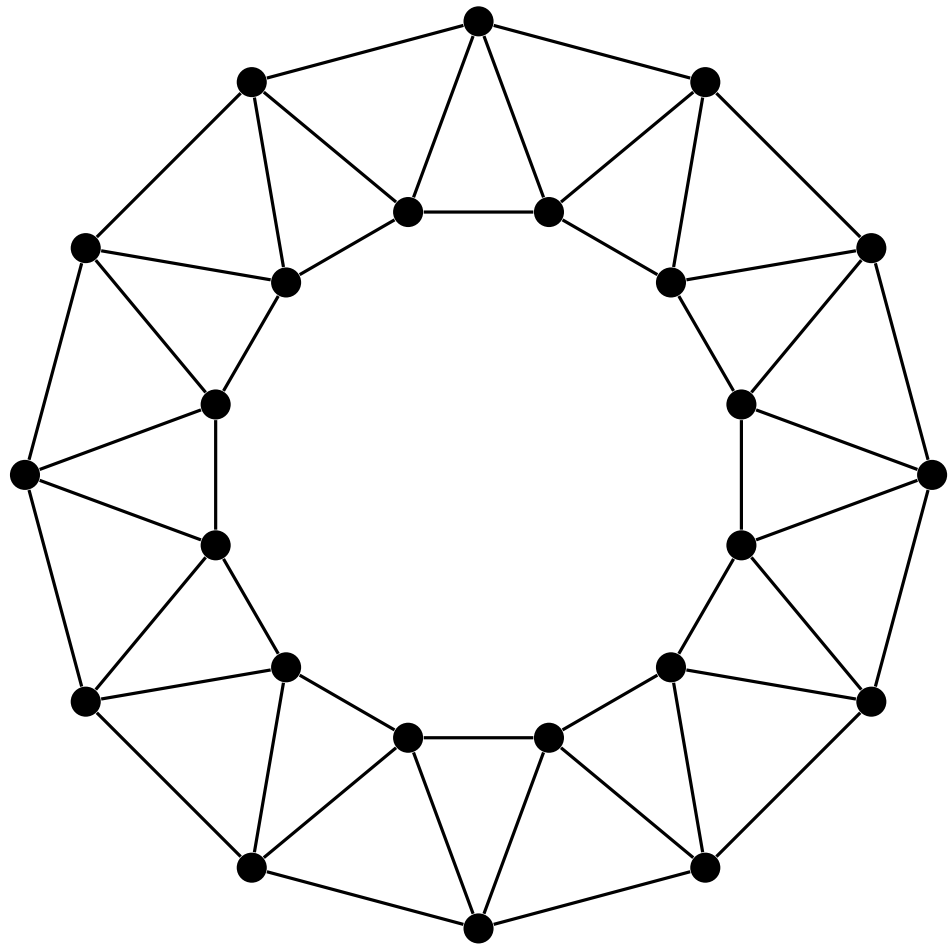
G_i:3-連結であり G_{i+1} が 3-連結でない, つまり $\kappa(G_{i+1}) \leq 2$ とする. すなわち $^{\exists}S \subset G_{i+1}$: vertices set s.t. $|\mathrm{S}| \leq 2 \wedge \mathrm{G_{i+1}} - \mathrm{S}$: 非連結. また, xy によって G_{i+1} を縮約した際に得られる頂点を v_{xy} と書き表すこととする. 今, $x,y \notin S$ とすると, $S \subset G-\{x,y\} = G/xy - v_{xy}$ より $v_{xy} \notin S$ である. よって補題 4.1.3 より $G_{i+1}/xy - S$: 非連結となり, $\kappa(G_i) \leq 2$ となり G_i:3-連結グラフに矛盾する. よって $\{x,y\}\cap S \neq \emptyset$ である. また $S \subseteq \{x,y\}$ とすると, これも $G_{i+1}/xy - \{v_{xy}\} = G_{i+1}-\{x,y\} \subset G_{i+1}-S$ より $\kappa(G_i) \leq 2$ となり矛盾する. したがって $|S|=2 \wedge x$ (resp y, 以降はxの場合で証明する) $\in S$ がわかる. S の元で x ではないほうの頂点を z とすると, $z \notin \{x,y\}$ より $G_{i+1}-S = G_{i+1}-\{x,z\}$ である.

ここで, $G_{i+1}-\{x,z\}$ は非連結より, 各連結成分 $C_k(k \in \mathbb{N})$ に分離することが出来, 特に $y \in C_1$ とすることが出来る. このとき, C_1 に y 以外の元 v が存在するとすると, $G_i-\{v_{xy},z\} = G_{i+1}-\{x,y,z\}$ であり G_i:3-連結であるため, $G_{i+1}-\{x,y,z\}$: 連結である. よって C_2-v path P が存在し, $P \subset G_{i+1}-\{x,y,z\} \subset G_{i+1}-\{x,z\}$ であるが, これは $v \in C_1$ であるため C_1 が連結成分であることに矛盾する. よって $C_1 = y$ である. y は $G_{i+1}-S$ における連結成分であるから, $N(y) \subset S \cup V(C_1 \backslash y) = S$ より $d(y) = |N(y)| \leq 2$ であるため, 仮定に反する.

□

これは 3-連結だけの場合で, 4-連結だと反例がある

例 4.1.6.



どの頂点を縮約しても 3-連結になる [2].

5 multigraph(マルチグラフ)

6 digraph(有向グラフ)

定義 6.0.1 (隣接 (adjacent))**.** グラフ D の有向辺 (u, v) に対して, u は v へ隣接している (u is adjacent to v) といい, 逆に v は u から隣接している (v is adjacent from u) という.

定義 6.0.2 (近傍 (out-neighborhood,in-neighborhood))**.** グラフ D の頂点 v に対して,

$$N^+(v) = \{x \in V | (v, x) \in E\}$$

$$N^-(v) = \{x \in V | (x, v) \in E\}$$

をそれぞれグラフ D における v の外近傍 (out-neighborhood), 内近傍 (in-neighborhood) という.

定義 6.0.3 (次数 (outdegree,indegree))**.** グラフ D の頂点 v に対して,

$$od(v) = |\{(v, x) \in E|^{\exists} x \in V\}|$$

$$id(v) = |\{(x, v) \in E|^{\exists} x \in V\}|$$

をそれぞれグラフ D における v の外次数 (outdegree), 内次数 (indegree) という. また, v の次数 $d(v)$ を $d(v) = od(v) + id(v)$ と定める.

つまり, 次数とは v に接続している辺の本数である. また, 明らかに $od(v) = |N^+|, id(v) = |N^-|$ である.

6.1 ネットワーク (Network)

定義 6.1.1 (ネットワーク (Network))**.** 有向グラフ $D = (V, E)$ が, source と sink という 2 つの異なる頂点 u, v をもち, また $c : E \to \mathbb{R}^+(\mathbb{R}^+ = \{|x||x \in \mathbb{R}\})$ が存在するとき, $N = (D, v, u, c)$ をネットワーク (Network) という.

D を N の underlying digraph , c を N の容量関数 (capacity function) , $e = (x, y) \in E$ に対する $c(e) = c(x, y)$ の値を e の capacity , v, u 以外の $N(D)$ の頂点を N の intermediate vertex という.

表記 6.1.2. 有向グラフ D, $g : E(D) \to \mathbb{R}$, $X, Y \subset V(D)$ に対して,

$$[X, Y] = \{(x, y)|x \in X, y \in Y\}$$

$$g(X, Y) = \sum_{(x,y)\in[X,Y]} g(x, y) \ ([X, Y] = \emptyset \Rightarrow g(X, Y) = 0)$$

また, $x \in V(D)$ のとき,

$$g^+(x) = \sum_{y\in N^+(x)} g(x, y) \ , \ g^-(x) = \sum_{y\in N^-(x)} g(y, x)$$

とし, より一般に $X \subseteq V(D)$ のとき,

$$g^+(X) = \sum_{x\in X} g^+(x) \ , \ g^-(X) = \sum_{x\in X} g^-(x)$$

6.2 フローネットワーク (Flow Network)

定義 6.2.1 (Flow)**.** ネットワーク $N = (D, u, v, c)$ に対して, $f : E(D) \to \mathbb{R}$ が以下を満たしているとき, f は N のフロー (flow) であるという.

(i) $^\forall a \in E(D), 0 \le f(a) \le c(a)$
(ii) $\forall x \in V(D)\backslash\{u, v\}, f^+(x) = f^-(x)$

$a = (x, y) \in E(D)$ のとき, $f(a) = f(x, y)$ を a に沿ったフローという. また, (2) をフローの保存則 (conservation equation) という.

他にも, $f : E(D) \to 0$ の場合, f はフローになる. これをゼロフローという. $X \subset V(D)$ に対して, $f^+(X) - f^-(X)$ を X から出ていくネットフロー, $f^-(X) - f^+(X)$ を X に入ってい

くネットフローという. とくに $x \in V(D)$ に対して, $f^+(x) - f^-(x)$ を x から出ていくネットフロー, $f^-(x) - f^+(x)$ を x に入っていくネットフローという. x が intermediate vertex であるとき, これらは (2) より 0 となる.

$a \in E(D)$ に対して $f(a) = c(a)$ であるとき, a は f について飽和している (saturated) といい, そうでないときには不飽和 (unsaturated) であるという.

定理 6.2.2. グラフ $N = (D, u, v, c)$, f を N 上のフローとするとき,

$$f^+(u) - f^-(u) = f^-(v) - f^+(v)$$

が成り立つ.

Proof.

$$\sum_{x \in V(D)} f^+(x) = \sum_{x \in V(D)} f^-(x) \; i.e. \; f^+(V(D)) = f^-(V(D))$$

であるから, 定義 6.2.1 の (2) より

$$f^+(u) - f^-(u) = f^-(v) - f^+(v)$$

が導ける. □

6.3 最大フロー (Maximum Flow)

定義 6.3.1 (value)**.** $N = (D, u, v, c)$ において, source u から出ていくネットフローをフロー f の value といい, val(f) で表す. すなわち val(f) $= f^+(u) - f^-(u)$ である.

定義 6.3.2 (最大フロー (Maximum Flow))**.** $N = (D, u, v, c)$ に対して, value が最大となるフロー f のことを N の最大フロー (Maximum flow) という. すなわち $^\forall f'$:N 上のフロー, val(f) $\geq$ val(f') である.

一意には定まらないが存在する. これは定義 6.2.1 の (1) より従う.

定義 6.3.3 (カット (Cut))**.** $N = (D, u, v, c)$. $X \subset V(D)$ に対して $\overline{X} = V(D) - X$ と定める. $u \in X \wedge v \in \overline{X}$ であるとき, $A = [X, \overline{X}] \subset E(D)$ を N のカット (cut) という.

u から v への path は必ず K を通らなければならない,

補題 6.3.4. $N = (D, u, v, c)$: ネットワーク, f: フロー, $X \subset V(D)$,

$$f^+(X) - f^-(X) = f(X, \overline{X}) - f(\overline{X}, X)$$

Proof.

$$\begin{aligned}
f^+(X) - f^-(X) &= \sum_{x\in X} f^+(x) - \sum_{x\in X} f^-(x) \\
&= \sum_{x\in X}\left(\sum_{y\in N^+(x)} f(x,y)\right) - \sum_{x\in X}\left(\sum_{y\in N^-(x)} f(y,x)\right) \\
&= \sum_{x\in X}\left(\sum_{y\in N^+(x)\cap X} f(x,y)\right) + \sum_{x\in X}\left(\sum_{y\in N^+(x)\cap \overline{X}} f(x,y)\right) \\
&\quad - \sum_{x\in X}\left(\sum_{y\in N^-(x)\cap X} f(y,x)\right) + \sum_{x\in X}\left(\sum_{y\in N^-(x)\cap \overline{X}} f(y,x)\right) \\
&= \sum_{a\in[X,X]} f(a) + \sum_{a\in[X,\overline{X}]} f(a) - \sum_{a\in[X,X]} f(a) - \sum_{a\in[\overline{X},X]} f(a) \\
&\quad ((\cdot\!:\!\cdot)[A,B] = \{(x,y)\in E(D) | x\in A, y\in B\} \\
&\qquad = \{(x,y)|x\in A, y\in B\cap N^+(x)\}(\{(x,y)|x\in A\cap N^-(x), y\in B\})\big) \\
&= \sum_{a\in[X,\overline{X}]} f(a) - \sum_{a\in[\overline{X},X]} f(a) \\
&= f(X,\overline{X}) - f(\overline{X},X)
\end{aligned}$$

□

定義 6.3.5 (容量 (Capacity))**.** $N=(D,u,v,c)$. $K=[X,\overline{X}]$ を N のカットとする. このとき, カットに含まれる arc の容量の合計値をカット K の容量 (capacity) といい,cap(K) と表す. すなわち

$$\mathrm{cap}(K) = c(X,\overline{X}) = \sum_{(x,y)\in[X,\overline{X}]} c(x,y)$$

である.

定理 6.3.6. $N=(D,u,v,c)$: ネットワーク, f:N のフロー, $K=[X,\overline{X}]$:N のカット, $\mathrm{val}(f) = f^+(X) - f^-(X) \le \mathrm{cap}(K)$

Proof. 仮定より $v\notin X$ であり, 定義 6.2.1 の (2) より, $^\forall x\in X-\{u\}, f^+(x)-f^-(x)=0$ である. よって

$$\begin{aligned}
f^+(X) - f^-(X) &= \sum_{x\in X} f^+(X) - \sum_{x\in X} f^-(X) \\
&= \sum_{x\in X}(f^+(x) - f^-(x)) \\
&= (f^+(u) - f^-(u)) \\
&= \mathrm{val}(f)
\end{aligned}$$

が成り立つ．また, ${}^{\forall}a \in E(D), 0 \leq f(a) \leq c(a)$ であるから, 補題 6.3.4 より,

$$\begin{aligned} f^+(X) - f^-(X) &= f(X, \overline{X}) - f(\overline{X}, X) \\ &\leq f(X, \overline{X}) \\ &\leq c(X, \overline{X}) \\ &= \mathrm{cap}(K) \end{aligned}$$

となり, 示せた. □

6.4 最小カット (Minimum Cut)

定義 6.4.1 (最小カット (Minimum Cut))**.** $N = (D, u, v, c)$ に対して，capacity が最小となるカット K のことを N の最小カット (Minimum cut) という．すなわち ${}^{\forall}K'$:N 上のカット, $\mathrm{cap}(K) \geq \mathrm{cap}(K')$ である.

一意には定まらないが, 存在する. これはネットワークの定義より従う.

系 6.4.2. $N = (D, u, v, c)$: ネットワーク, f:N のフロー, K:N のカット．このとき, $\mathrm{val}(f) = \mathrm{cap}(K)$ ならば, f は N の最大フローであり, K は N の最小カットである.

Proof. $\mathrm{val}(f) = \mathrm{cap}(K)$ とすると, 任意の N のフロー f', 任意の N のカット K' に対して, 定理 6.3.6 より, $\mathrm{val}(f') \leq \mathrm{cap}(K) = \mathrm{val}(f) \leq \mathrm{cap}(K')$ である. よって, f は N の最大フローであり, K は N の最小カットである. □

系 6.4.3. $N = (D, u, v, c)$: ネットワーク, f:N のフロー, $K = [X, \overline{X}]$:N のカット．このとき,

$$({}^{\forall}a \in [X, \overline{X}], f(a) = c(a)) \wedge ({}^{\forall}a \in [\overline{X}, X], f(a) = 0)$$

ならば, f は N の最大フローであり, K は N の最小カットである.

Proof. 定理 6.3.6 より,

$$\begin{aligned} \mathrm{val}(f) &= f^+(X) - f^-(X) \\ &= f(X, \overline{X}) - f(\overline{X}, X) \\ &= c(X, \overline{X}) - 0 \\ &= \mathrm{cap}(K) \end{aligned}$$

よって系 6.4.2 より f は N の最大フローであり, K は N の最小カットである. □

最大フローと最小カットであるための条件を与えている. 特に系 6.4.2 は定理 7.1.1 の十分条件を与えている．必要条件を与えるために, ここで一つ便利な概念を導入する.

定義 6.4.4 (semipath)**.** 有向グラフ D に対して semipath とは, 以下を満たすような空でない有向グラフ $P=(V,E)$ のことである.

$V=\{\omega_i|i=0,\cdots,k\}, E=\{a_i\in E(D)|a_i=(\omega_{i-1},\omega_i)\vee a_i=(\omega_i,\omega_{i_1}), i=1,\cdots k\}$(各$\omega_i$は異なる)

またこのとき, P を ω_0 から ω_k への semipath (ω_0-ω_k semipath) という. またこのとき E の元について, $a_i=(\omega_{i-1},\omega_i)$ を forward arc , $a_i=(\omega_i,\omega_{i_1})$ を backward arc という.

表記 6.4.5. 上の semipath を $P=(\omega_0,a_1,\omega_1,\cdots,\omega_{k-1},a_k,\omega_k)$ と書き表す.

6.5 f-Augmenting Semipaths

定義 6.5.1. $N=(D,u,v,c)$: ネットワーク, f:N のフロー, $P=(\omega_0,a_1,\omega_1,\cdots,\omega_{k-1},a_k,\omega_k)$:$D$ の semipath とする. P が以下の条件を満たしているとき, P は f-unsaturated な semipath であるという.

(i) $f(a_i)<c(a_i)$ (a_i:forward arc)
(ii) $f(a_i)>0$ (a_i:backward arc)

自明な semipath ($P=(\omega_0)$) は f-unsaturated とする. P が f-unsaturated な u-v semipath であるとき, f-augmenting な semipath であるという.

定理 6.5.2. $N=(D,u,v,c)$: ネットワークとする. このとき, f が最大フローであることと D 上に f-augmenting な semipath が存在しないことは同値である.

Proof. $\Rightarrow$)f を最大フローとし, D 上に f-augmenting な semipath P が存在するとする. $P=(\omega_0,a_1,\omega_1,\cdots,\omega_{k-1},a_k,\omega_k)$ とすると, $\omega_0=u,\omega_k=v$ である. P の forward arc a_{i_n} について, $c(a_{i_n})-f(a_{i_n})>0$ であり $n\le k<\infty$ であるため, $p_1=\min\{c(a_{i_n})-f(a_{i_n})|a_{i_n}\text{:forward arc}\}>0$ が存在する. 同様に P の backward arc a_{i_m} についても, $f(a_{i_n})>0$ であるため $p_2=\min\{f(a_{i_m})|a_{i_n}\text{:backward arc}\}>0$ が存在する. $p=\min\{p_1,p_2\}$ とすれば,

$$f'(a)=\begin{cases} f(a)+p & \text{if } a \text{ is a forward arc on } P\\ f(a)-p & \text{if } a \text{ is a backward arc on } P\\ f(a) & \text{if } a\notin E(P)\end{cases}$$

は N のフローとなり[*9],

[*9] ω_i $(0<i<k)$ に対して
(i) $f^+(\omega_i)+p=f'^+(\omega_i), f^-(\omega_i)+p=f'^-(\omega_i)$ (a_i,a_{i+1}:forward arc)
(ii) $f^-(\omega_i)+p-p=f'^-(\omega_i)$ (a_i:forward arc, a_{i+1}:backward arc)
(iii) $f^+(\omega_i)-p+p=f'^+(\omega_i)$ (a_i:backward arc, a_{i+1}:forward arc)
(iv) $f^+(\omega_i)-p=f'^+(\omega_i), f^-(\omega_i)-p=f'^-(\omega_i)$ (a_i,a_{i+1}:backward arc)
であるため $f^+(\omega_i)-f^-(\omega_i)=f'^+(\omega_i)-f'^-(\omega_i)$ であり, $x\in V(D)\backslash V(P)$ については明らかに $f^+(x)-f^-(x)=f'^+(x)-f'^-(x)$ である. また, f' は p のとり方から各 arc の容量を超えない. ゆえに f' は flow である.

(i) $f^+(u) + p = f'^+(u)$ (a_1:forward arc)
(ii) $f^-(u) - p = f'^-(u)$ (a_1:backward arc)

より $f^+(u) - f^-(u) < f'^+(u) - f'^-(u)$ *i.e.* $\mathrm{val}(f) < \mathrm{val}(f')$ であるから f が最大フローであることに矛盾する. よって f-augmenting な semipath P は存在しないことがわかる.

$\Leftarrow$)f が D 上 f-augmenting な semipath が存在しないような flow とする. このとき f が最大フローであることを示す. 今 $X = \{x \in V(D) | {}^\exists P{:}f\text{-unsaturated } u\text{-}x \text{ semipath}\}$ とすると, $u \in X, v \notin X$ であるため, $K = [X, \overline{X}]$ は cut となる. ${}^\forall a \in [X, \overline{X}], {}^\forall b \in [\overline{X}, X], f(a) = c(a), f(b) = 0$ であるから[*10], 系 6.4.3 より f は最大フローである. また, K は最小カットとなっている. □

7 最大フロー最小カット定理

7.1 maximum flow minimum cut theorem

定理 7.1.1 (maximum flow minimum cut theorem)**.** ネットワーク $N = (D, u, v, c)$ に対して, 最大フローと最小カットの値は一致する. すなわち, f:N のフロー, K:N のカットに対して,

$$f\text{: 最大フロー} \wedge K\text{: 最小カット} \Leftrightarrow \mathrm{val}(f) = \mathrm{cap}(K)$$

Proof. $\Leftarrow$) 系 6.4.2 より従う.

$\Rightarrow$)f: 最大フロー, K: 最小カットとする. 定理 6.5.2 より, D 上に f-augmenting な semipath は存在せず, $X = \{x \in V(D) | {}^\exists P{:}f\text{-unsaturated } u\text{-}x \text{ semipath}\}$ とすると $K' = [X, \overline{X}]$ は最小カットとなり,

$$f(a) = \begin{cases} c(a) & \text{if } a \in K' \\ 0 & \text{if } a \in [\overline{X}, X] \end{cases}$$

である. よって系 6.4.3 より $\mathrm{val}(f) = \mathrm{cap}(K') = \mathrm{cap}(K)$ となり示せた. □

7.2 The Ford-Fulkerson Algorithm

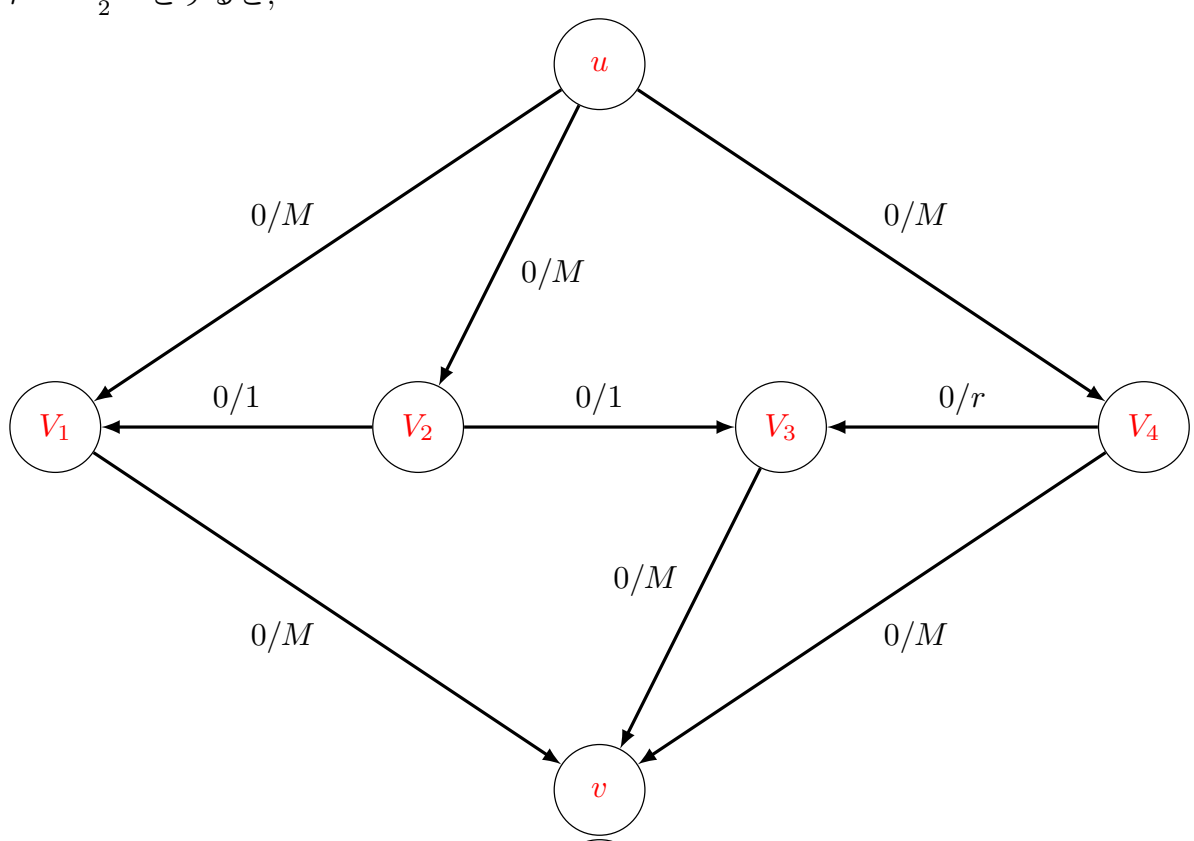
$N = (D, u, v, c)$: ネットワークとする.

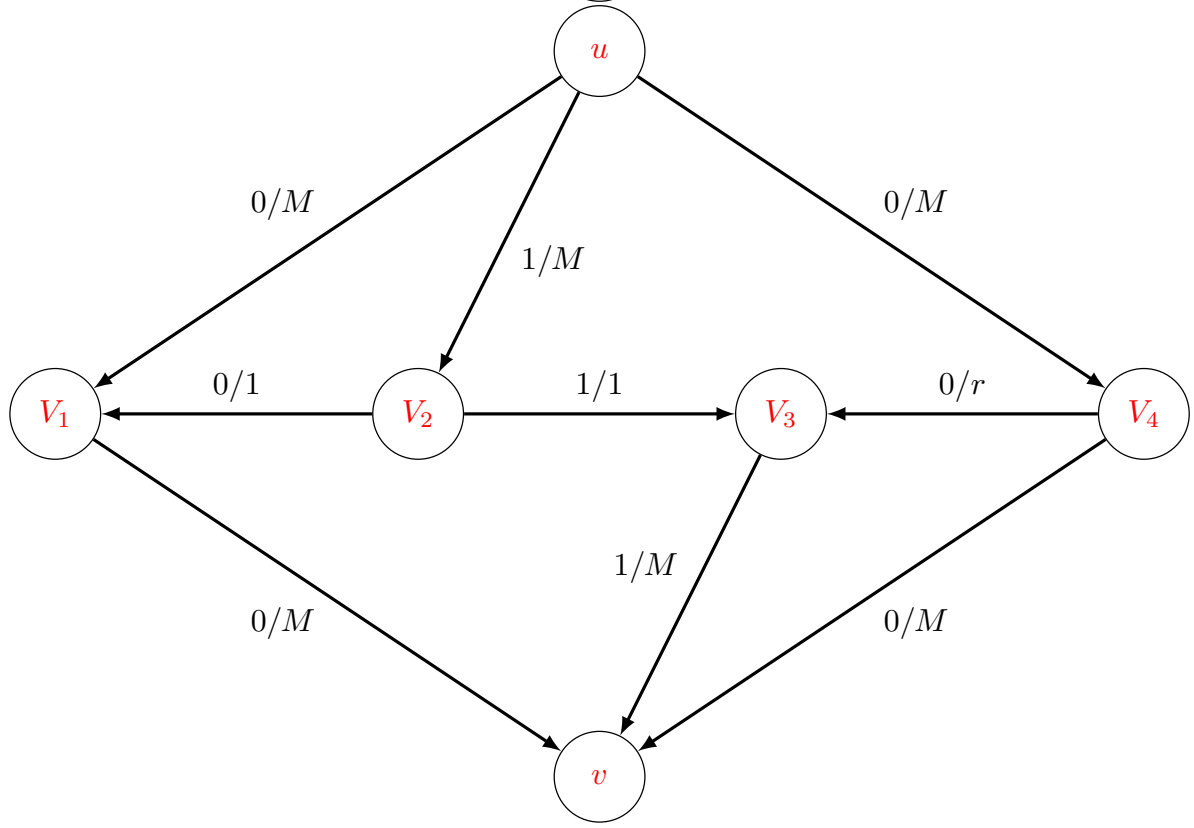
1. flow f を一つとる.
2. f-augmenting な semipath を見つける. 見つけれなかった場合は終了する.
3. 定理 6.5.2 の $\Rightarrow$ で作ったように f' を構成する.
4. $f = f'$ として Step 2 に戻る.

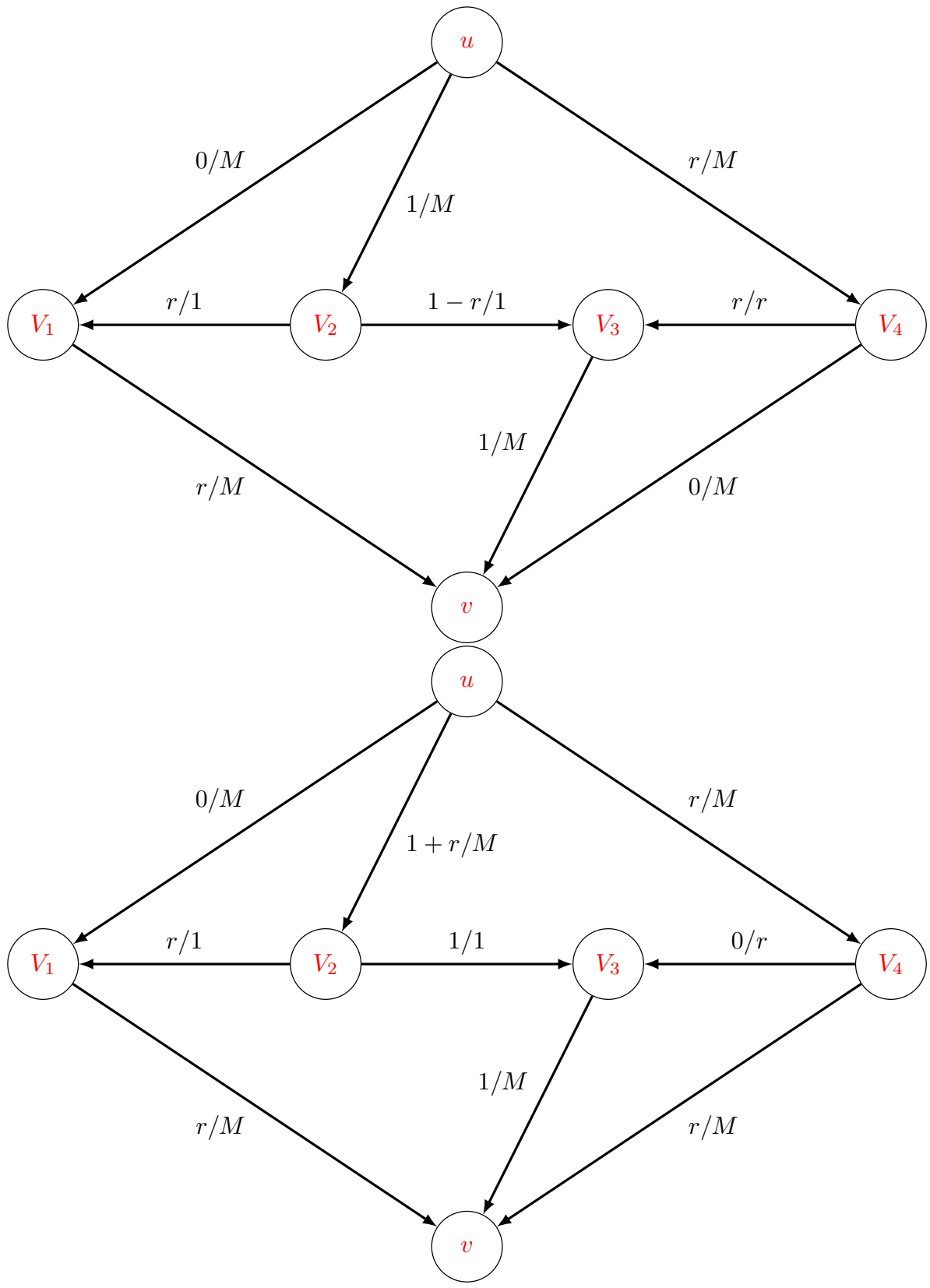
[*10] 任意の $(y, z) \in [X, \overline{X}]$ について, $y \in X$ より f-unsaturated u-y semipath が存在し, $z \in \overline{X}$ より f-unsaturated u-z semipath が存在しない. $f(y, z) < c(y, z)$ とすると, f-unsaturated u-y semipath に $f(y, z), z$ を加えたものは f-unsaturated u-z semipath になり矛盾する. よって $f(y, z) = c(y, z)$ である. 同様に, 任意の $(w, x) \in [\overline{X}, X]$ についても $f(w, x) = 0$ が言える.

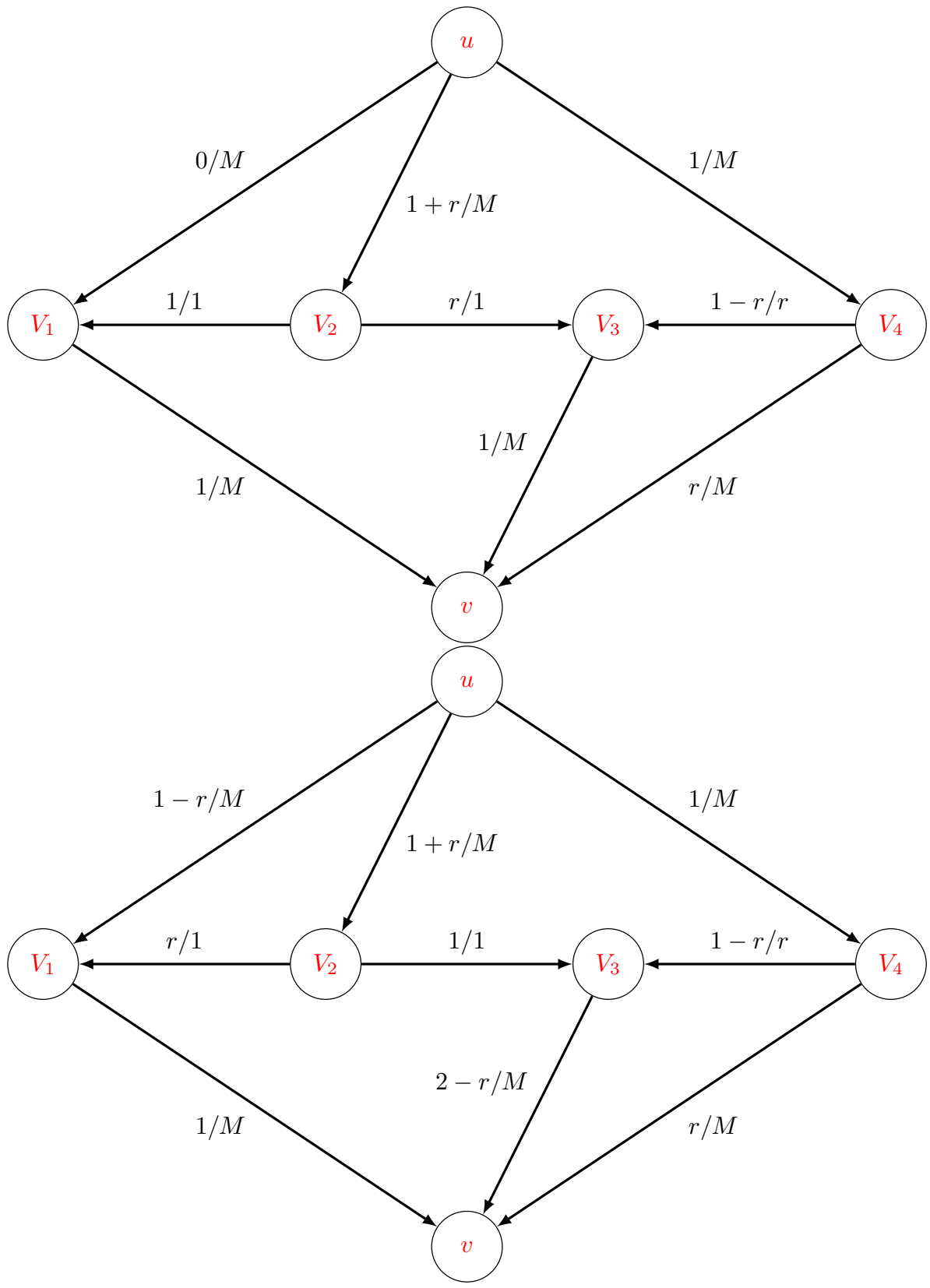
しかしこのアルゴリズムは欠点が多い. 一つはグラフと f-augment semipath の選び方によって, 計算量がとても大きくなるという点. もう一つは容量が無理数だとアルゴリズムが止まらなくなるという点だ [3].

$r = \frac{\sqrt{5}-1}{2}$ とすると,






u
0/M
1/M
r/M
V1
r/1
V2
1 − r/1
V3
r/r
V4
1/M
r/M
0/M
v
u
0/M
1 + r/M
r/M
V1
r/1
V2
1/1
V3
0/r
V4
1/M
r/M
r/M
v



7.3 The Edmonds-Karp Algorithm

$N = (D, u, v, c)$: ネットワークとする. 各点にラベルを付け, ラベル付けされているがスキャンされていない頂点のリスト L を用意する. ラベルは 2 つの値のペアである.

1. flow f を一つとる. N の intermediate vertex w において f-unsaturated な u-w semipath P が存在するときに, P の直前の頂点 x について $(x,w) \in E(P)$ であればラベルは $(x+, \epsilon(w))$, $(w,x) \in E(P)$ であればラベルは $(x-, \epsilon(w))$ とする.
2. u のラベルは $(-, \infty)$ とし, u をリスト L に加える.
3. $L = \emptyset$ ならばとめ. そうでなければ L の最初の元 x(ラベル $(z+, \epsilon(x))$ or $(z+, \epsilon(x))$ を持つ) について,
   3.1. flow f を一つとる.
   3.2. ラベルを付ける. u は $(-, \infty)$ とし, リスト L に加える.
4. v がラベルを持っている場合, Step 5 へ, そうでなければ 3 へ行く.
5. v がラベルを持っている場合, Step 5 へ, そうでなければ 3 へ行く.
6. ラベルを削除し, L から頂点を全て削除し, Step 2 に戻る.

8 quiver(箙)

9 infinite graph(無限グラフ)

[4]

第 II 部

origami(折り紙)

[9]

索引

参考文献

[1] ? 2-Connected Graphs. http://www.cs.rpi.edu/~goldberg/14-GT/08-block.pdf.

[2] ? Connectivity. http://www-sop.inria.fr/members/Frederic.Havet/Cours/connectivity.pdf.

[3] ? Ford Fulkerson algorithm. https://en.wikipedia.org/wiki/Ford\OT1\textendashFulkerson_algorithm#Non-terminating_example.

[4] ? Infinite graph. https://www.math.uni-hamburg.de/home/schacht/lehre/SS13/GT/Ch8prelims.pdf.

[5] ? latexmkrc の設定. http://www2.yukawa.kyoto-u.ac.jp/~koudai.sugimoto/dokuwiki/doku.php?id=latex:latexmk の設定.

[6] ? VScode による latex 環境の基本設定. http://www2.yukawa.kyoto-u.ac.jp/~koudai.sugimoto/dokuwiki/doku.php?id=latex:vscode による latex 環境の基本設定.

[7] Alg-d. TikZ の使い方 (圏論編). http://alg-d.com/math/kan_extension/TikZ_for_cat.pdf.

[8] G. Chartrand, L. Lesniak, P. Zhang. *GRAPH & DIGRAPH, Sixth Edition.* CRC Press?, 2016.

[9] ロベルト・ゲレトシュレーガー. 『折り紙の数学 ユークリッドの作図法を超えて 第 1 版』. 森北出版, 2008 年 10 月 20 日. 深川英俊 訳.

[10] R. Diestel. *Graph Theory, Fifth Edition.* Springer, 2017.

[11] TeX Wiki. 索引生成. https://texwiki.texjp.org/?%E7%B4%A2%E5%BC%95%E4%BD%9C%E6%88%90.