平成 30 年度
卒業論文

タイトル

名古屋工業大学　情報工学科

所属: 泉研究室

平成 26 年度入学　26115142 水谷　龍誠

目 次

第 1 章　はじめに **1**

1.1　研究背景 1

1.2　研究結果 1

第 2 章　準備 **2**

2.1　セパレータ 3

2.2　Subgraph Aggregation [1]li2018distributed 4

2.3　s-t 点素パス探索アルゴリズム 4

第 3 章　既存手法の概要 **6**

第 4 章　分散実装 **8**

4.1　カット探索 8

4.2　縮約アルゴリズム 9

第 5 章　まとめと今後の課題 **10**

第 6 章　謝辞 **11**

参考文献 **12**

第1章

はじめに

1.1 研究背景

あるグラフ $G = (V, E)$ が与えられたとき, グラフを非連結な二つの頂点集合に分割する小さな”セパレータ”の存在について考えられることがある. このセパレータの存在は, 高速なグラフアルゴリズム設計において非常に重要である. しかし, 一般のグラフに対して最小サイズのセパレータを求める問題は NP 困難であるため, 小さなセパレータを求める近似アルゴリズムがこれまで研究されている.

現在, 一般のグラフに対して小さなセパレータを近似する集中型のアルゴリズムはいくつか知られているが, 分散環境におけるアルゴリズムはまだあまり知られていない. この論文では, 既存の近似アルゴリズムをベースとする分散セパレータ近似アルゴリズムを提示する.

1.2 研究結果

第2章

準備

アルゴリズムは分散システムにおけるCONGESTモデルの下で動作する.n個の計算機ノード集合Vと通信リンクの集合Eである無向ネットワークグラフ$G=(V,E)$があるとする.CONGESTモデルにおいて計算機はラウンドに従って同期して動作を行う.各ラウンドにおいてノードは最大$O(\log n)$ビットのメッセージを各隣接ノードに送信,各隣接ノードからメッセージの受信,無制限の内部計算を行う事が出来る.
分散設定では,ノードは一意なID($O(\log n)$ビット)を持っており,隣接ノードのIDは知っているものとする.

グラフ$G=(V,E)$に対して,Dはその直径を表すとする.頂点$s,t\in V$に対して,sとtを結ぶ経路をs-tパスと呼ぶ.特に,いくつかのs-tパスに対して,それぞれのパスが同じ頂点を共有しないとき,これらをs-t点素パスと呼ぶ.また,sとtの間にパスが存在しなくなるように取り除かれた頂点集合をs-tカットと呼ぶ.

s-t点素パスとカットについて以下の定理が示されている.

定理 1 (メンガーの定理)**.** グラフGに対して,s,tのカットの最小値はs-t点素パスの最大本数と等しい

s-t点素パスの最大本数について考える.メンガーの定理によって各点素パスから頂点を一個づつ適切に取り出すと,その頂点集合はグラフGの最小s-tカットになる.もちろん,任意に各パスから頂点を取り出すとs-tカットにならない場合もある.この各パスから任意に取り出した頂点集合を特にスライスと呼び,以下のように定義する.

定義 1. Gをグラフとし,$s,t\in V$とする.$\{f_1,\ldots,f_k\}$をGにおける点素s-tパスの集合とする. この時,すべての$1\leq i\leq k$に対して,$w_i\in f_i$、$s\neq wi\neq t$であれば,$(f_1,\ldots,f_k)$に

関してタプル $(w_1, \ldots, w_i)$ をスライスと呼ぶ. X を V の任意の部分集合とし、$s, t \notin X$ とする。$G[V \backslash X]$ に s-t パスがない場合,X は s と t を分離すると言う。s と t を分離するスライスをカットと呼ぶ.

スライスの集合は,s に対する相対的な"近さ"によって部分的に並べることができる. ある s-t 点素パスの頂点について s 方向を前方,t 方向を後方として向きと順序を定義する.

定義 2. $\{f_1, \ldots, f_k\}$ を s-t 点素パスの集合とする. $U = (u_1, \ldots, u_k)$ および $W = (w_1, \ldots, w_k)$ を $(f_1, \ldots, f_k)$ に関するスライスとする. すべての $1 \leq i \leq k$ について、u_i は f_i における w_i の前方または $u_i = w_i$ とする. この時 U は W より s に近いと言い,$U \preceq W$ と書く.$1 \leq i \leq k$ に対して, さらに $u_i \neq w_i$ ならば,U は W より厳密に s に近いと言い,$U \prec W$ と書く. 同様に、W は U より厳密に t に近いと言う. 便宜上, タプル $(s, s, \ldots, s)$ と $(t, t, \ldots, t)$ についても同様に上記を定義. したがって, 例えば,$(s, s, ...s)$ はどのスライスよりも s に近いと言える." $\preceq$ " は一般的な順序の合計を定義するものではない.

カットを取り除くことでは G を少なくとも 2 つの連結成分に分離する. あるカットによって分離した各連結成分に対して以下の定義をする.

定義 3. U を任意のカットとする.s を含む $G[V \backslash U]$ の連結成分の頂点セットとして $V_s(U)$ を定義し,t を含む $G[V \backslash U]$ の連結成分の頂点セットとして $V_t(U)$ を定義し,V_r を $G[V \backslash U]$ の残りの連結成分の頂点集合の和集合として定義する. すなわち、$V_r(U)$ は s も t も含まない連結成分である.（したがって、$V_r(u)$ は空であり得る）

2.1 セパレータ

頂点集合 V の分割 $(A, S, B) : V = A \cup S \cup B$ について,$|A| \leq \frac{2}{3}|V|$ かつ $|B| \leq \frac{2}{3}|V|$ であり, 集合 A, B 間に辺が存在しないとき, 集合 S を G のセパレータと呼ぶ.$0 < \alpha < 1$ とし, 上記の分割 (A, S, B) に対して $|A|, |B| \leq \alpha|V|$ の時,S を α-セパレータと呼ぶ.

セパレータ, またはカットはある値 α に関して, セパレータやカットによって分割された全ての頂点集合のサイズが高々$\alpha|V|$ 以下である時, セパレータやカットについてバランスが取れていると言うことにする.

2.2 Subgraph Aggregation [1]li2018distributed

Ghaffari と Haeupler [2] のショートカットフレームワークは, 制限されたグラフ族の分散アルゴリズムを設計する上で有益であることが証明されている. さらに, このショートカットフレームワークの制約をさらに弱めて改良されたタスクが Jason [3] によって示されている. この論文では, このショートカットのフレームワークの内部動作には触れずに Subgraph Aggregation として定義されているタスクを利用する.

定義 4 (Subgraph Aggregation)**.** $G = (V, E)$ をネットワークグラフとし,$\mathcal{P} = (P_1, ...P_{|\mathcal{P}|})$ をパートの集合, 各 P_i について H_i を P_i のノード上の G の連結部分グラフとする. 必ずしもグラフ $G[P_i]$ から誘導されるとは限らない. 各部分グラフ H_i について、$V(H_i)$ 内のすべてのノードが部分グラフ H_i 内の隣接ノードを認識し、それ以外は何も知らないと仮定する. すべてのノード $v \in \bigcup_i P_i$ が O(log n) ビットの整数 x_v を持ち,$\oplus$ を長さ O（log n）の整数に作用する結合関数とする.P_i 内の各ノードは値 $\bigoplus_{v \in P_i} x_v$、すなわち P_i 内のすべての値 x_v の集合 $\oplus$ を知りたいとする. このようなタスクをオペレーター $\oplus$ における Subgraph Aggregation と呼ぶ.

SA ラウンドを,Subgraph Aggregation（SA）において各パートがその収集値 $\bigoplus$ を学習する一回の反復とする.SA は特別なグラフ構造に関して以下の定理が成り立つ.

定理 2. 結合演算子 $\oplus$ について、Q_G がグラフ G とその直径 D に依存するパラメータである場合,$\tilde{O}(Q_G)$ ラウンドで Subgraph Aggregation 問題を解くことができる.

全てのグラフ G : $Q_G = O(\sqrt{n} + D)$

種数 g のグラフ G : $Q_G = O(\sqrt{g+1}D)$

木幅 k のグラフ G : $Q_G = \tilde{O}(kD)$

H をマイナーとして含まないグラフ G : $Q_G = \tilde{O}(f(H) \cdot D^2)$,$f$ は H にのみ依存する関数

2.3 s-t 点素パス探索アルゴリズム

グラフプロパティの一種である木幅と呼ばれる値 k に対して以下の補題が示されている.

補題 1 ([3])**.** 木幅が高々kのグラフ$G = (V, E)$と二つの頂点$s, t \subseteq V$を与えると,k点素 s-t パスを見つけるか, サイズk以下の s-t ノードカットを$\tilde{O}(k^{O(1)}D)$ラウンドで出力する分散アルゴリズムが存在する. 前者の場合, 各ノードは自身がパス上にあるかどうかを知っており、そうであれば、そのパス上のその前方と後方を知る. 後者の場合、k点素パスが存在しないという事実と, カットに含まれるかどうかを各ノードが知っている.

アルゴリズムの詳細については触れないが, 点素パスの探索ステップ自体には木幅の値は関わっていない. 木幅が影響するのは SA ラウンドに関してのみである. 上記の補題と定理 2 より一般のグラフに置き換えた以下の系が成り立つ.

系 1. 補題 1 のグラフGを一般のグラフに置き換えると, 最大ℓ本の s-t 点素パスは$\tilde{O}(\ell^{O(1)}(\sqrt{n} + D))$ラウンドで見つけることができる.

第3章

既存手法の概要

既存の近似アルゴリズム [4] のプロセスを簡略化して説明する.

アルゴリズムのアプローチは最大 s-t フローに基づいている. 小さいセパレータを含むグラフ G が与えられ, そのセパレータの「異なる側に」頂点 s と t があるとする. この時, メンガーの定理により,s-t 点素パスの最大数も少なくなる.

アルゴリズムは s-t 点素パスの最大本数の集合を計算することから始める. 最大本数 k 本の各パスから 1 つの頂点を取ることによって,k 個の s − t カットを見つける. これらの頂点は s と t を分離するように選ぶ必要がある.s-t カットは, メンガーの定理によって存在することは明らかである.

次に, 二分探索法を使用して, 全ての s − t カットのうち 1 つは s に近く, もう 1 つは t に近い 2 つの「最良のバランス」を決定する. これら 2 つのカットのうちの 1 つが十分にバランスが取れていれば, 目的の小さいセパレータが発見できたことになる. バランスが取れていないとき, 連結成分が 2 つの s-t カットで分断されているされていると見なす. そして,s を含む連結成分を新たな頂点 s' に,t を含む成分を新たな頂点 t' に縮約する. 新しく得られたグラフのすべての s'-t' 頂点カットも G の s-t カットであり, さらに上記の 2 つの s-t カットよりもバランスが取れていることが示されている. そのため, 本数がある値 K に等しい s-t カットを得るまで点素パスを探索, 最良のバランスのとれた s-t カットをいくつか発見し, 頂点集合を縮約する上記のプロセスを繰り返す.

$\frac{2}{3} \leq \alpha < 1$ の時、s と t を分離する最大で K のサイズの α-セパレータの存在を考える. 上述した反復プロセス終了時, バランスが少なくとも α と同等であるカットを生じない場合,α-セパレータの少なくとも 1 つの頂点が実行された縮約のうちの 1 つに関与して

いるとみなす. したがって, プロセス全体を最大 K 回反復することによって, (プロセスで得られたすべての関連カットを集めることによって) セパレータを探索する.

ここまで, セパレータの「異なる側にある」頂点 s と t を見つけることができると仮定していたが, 実際には s と t を一様ランダムに選択している為, 上記の反復プロセスを適用してから得られた最大の連結成分に対して再度縮約アルゴリズムを反復する.

このようにして, 与えられたグラフが小さいセパレータを含んでいれば, 小さいセパレータを見つけるためのほぼ線形の実行時間を得る.

上記のプロセスをまとめると, ある値 $\frac{2}{3} \leq \alpha <$ と $0 < \varepsilon < 1 - \alpha$ について以下の結果が示される.

グラフ $G = (V, E), n \in V, m \in E$ について,G がサイズ K の α-セパレータを含んでいるとき, アルゴリズムはサイズ $O(\varepsilon^{-1}K^2 \log^{1+o(1)} n)$ の $(\alpha + \varepsilon)$-セパレータを高確率で $O(\varepsilon^{-1}K^3 m \log^{2+o(1)} n)$ 時間で見つけることができる.

第4章

分散実装

このセクションでは, セパレータを近似するための分散アルゴリズムを提示する. 具体的には, [4] の 5 つのアルゴリズムそれぞれを分散ネットワークで実行できるように適応させていく.

4.1 カット探索

あるスライスが与えられた時, そのスライスに近いカットについて定義する.

定義 5. $(f_1, \ldots, f_k)$ に関して U をスライスとする. $U \preceq X$ であり, かつ $U \preceq X' \prec X$ を満たすカット X' が存在しないようなカットを X とする. この時 $U^+ := X$ を定義する.
上記のような X が存在しない場合は、$U^+ := (t, t, \ldots, t)$ とする.
同様に,$Y \preceq U$ かつ $U \preceq Y' \prec Y$ を満たすカット Y' が存在しないようなカットを Y とする. この時,$U^- := Y$ を定義する.
上記のような Y が存在しない場合は $U^- := (s, s, \ldots, s)$ とする.

上記のカット U^+, U^- を計算するアルゴリズムを以下に示す.

1. 点素パス上の各頂点が隣接頂点にパス上のインデックスをブロードキャスト
2. 各点素パスの頂点を含まない G の各連結成分内 (仮 C) で, 受け取ったパスのインデックスの最大値最小値を収集 (2kSA ラウンド)
3. スライス U のインデックスを各点素パス上のノードが知る (kSA ラウンド)
4. パス上の各頂点がスライス U のインデックスを隣接頂点にブロードキャスト (k ラ

ウンド)

5. 各 C 内の頂点が点素パスの情報を 2 つ以上知っているとき
   - 各 C 内で受け取ったスライスのインデックスと大小比較
   - どれか一つでもスライスのインデックスより小さい時,C を X に追加
   - X 内でパスの最大値を収集 (kSA ラウンド)
6. loop 一番長い点素パスの長さ ℓ 分だけ繰り返す可能性
7. 収集したインデックスの最大値 (仮 w_i) の 1 つ前までのパスの頂点を X に追加
8. 最小値が x_i より小さい C を X に追加
9. 最大値収集,w_i を更新

全体で ℓkSA ラウンド

4.2 縮約アルゴリズム

縮約ステップでは,, まず初めに与えられた $s, t \in V$

第5章

まとめと今後の課題

第6章

謝辞

本研究の機会を与え，数々の御指導を賜りました泉泰介准教授に深く感謝致します．また，本研究を進めるにあたり多くの助言を頂き，様々な御協力を頂きました泉研究室の学生の皆様に深く感謝致します．

参考文献

[1] Bernhard Haeupler, D Ellis Hershkowitz, and David Wajc. Round-and message-optimal distributed graph algorithms. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing*, pages 119–128. ACM, 2018.

[2] Mohsen Ghaffari and Bernhard Haeupler. Distributed algorithms for planar networks ii: Low-congestion shortcuts, mst, and min-cut. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 202–219. SIAM, 2016.

[3] Jason Li. Distributed treewidth computation. *arXiv preprint arXiv:1805.10708*, 2018.

[4] Sebastian Brandt and Roger Wattenhofer. Approximating small balanced vertex separators in almost linear time. In *Workshop on Algorithms and Data Structures*, pages 229–240. Springer, 2017.