令和2年度　　修士論文概要		
k-極大独立集合検証問題の分散計算複雑性	片山・金研究室	佐藤 僚祐
	No. 31414050	Ryosuke Sato

1　はじめに

分散グラフアルゴリズムとは, 計算機を頂点, 辺を通信リンクとみなしてネットワークをモデル化したグラフ上において, そのネットワーク自身を入力として様々な問題を解く枠組みである. 分散アルゴリズムにおける代表的なモデルのひとつとして *CONGEST* モデルが存在する.*CONGEST* モデルにおいて, 各ノードは同期して同じアルゴリズムを実行して入力グラフ上の問題を解決する. 各ノードは各ラウンドで (i)b ビットのメッセージを隣接ノードに送信 (ii) 隣接ノードからメッセージを受信 (iii) 内部計算の 3 つの動作をする. 標準的には,$b = \log n$ を想定する. *CONGEST* モデルにおいて, ある 1 つのノードにグラフ全体のトポロジの情報を集め, そのノード上で逐次アルゴリズムを実行するという素朴なアプローチから自明に $O(n^2)$ ラウンドの上界を得ることができる.*CONGEST* モデルにおける下界の証明では, 下界をこの n^2 にどれだけ近づけることができるかに興味がもたれている.

ネットワーク上の最大独立集合を発見する最大独立集合問題に対する数多くの分散グラフアルゴリズムが研究されている. 大きなサイズの独立集合は経済学, 計算生物学, 符号理論, 実験計画法など様々な分野への応用に用いられる [1] が, そもそも最大独立集合問題は NP 完全であり, 頂点数 n に対して n の多項式時間で解くことは, その近似を含めて絶望的であるとされている. 内部計算に指数時間かかることを許した *CONGEST* モデルの下でラウンド数を n の多項式でおさえるような最大独立集合問題の複雑性としては, 頂点の最大次数を Δ としたとき, 最大重み付き独立集合の $(1+\varepsilon)\cdot\Delta$-近似を高確率で見つけるアルゴリズムに対する $(\frac{poly(\log\log n)}{\varepsilon})$ ラウンドの上界 [1] や, 最大独立集合の $(\frac{1}{2}+\varepsilon)$-近似を見つけるアルゴリズムに対する $\Omega(\frac{n}{(\log n)^3})$ ラウンドの下界, $(\frac{3}{4}+\varepsilon)$-近似を見つけるアルゴリズムに対する $\Omega(\frac{n^2}{(\log n)^3})$ ラウンドの下界 [2] が知られている.

しかし, 内部計算に指数時間かかることを許した下での分散グラフアルゴリズムの複雑性の議論の妥当性にはやや疑問が残る. そこで今回, 我々は最大独立集合の局所最適解である k-極大独立集合 (k-Maximal Independet Set, k-MIS) について考える. ネットワーク上の k-MIS を発見する k-MIS 問題は集中型アルゴリズムによって $O(n^{k+2})$ 時間で解くことができ,$k = O(1)$ のとき n の多項式時間になる.

今回は k-MIS の検証問題 (verification) に着目し, その複雑性について議論を行う. k-MIS 検証問題とはネットワーク上に独立集合が与えられ, それが k-MIS かどうかを判定する問題である. k-MIS 問題に対する素朴な局所探索アルゴリズムは, ある 1 つの独立集合からスタートし, (I) 現在の状態が k-MIS であるか判定する. (II)YES であればそれを出力,NO であれば解を更新, というフェーズを繰り返す. k-MIS 検証問題は上記の (I) に対応しており,k-MIS 問題と関連付いた問題ととらえることができる.

今回, 我々は極大独立集合検証問題に対する 3 つの複雑性の結果を証明する. 最初に,1-MIS 検証問題が $O(1)$ ラウンドで解けることを証明する. 次に,2-MIS 検証問題に対する $\tilde{\Omega}(\sqrt{n})$ ラウンドの下界を証明する. 最後に,3-MIS 検証問題に対する $\tilde{\Omega}(n)$ ラウンドの下界を証明する. この下界を一般の k に拡張できないかを現在検討中である. 後半の 2 つの証明のアイデアは 2 者間通信複雑性からの帰着に基づいている.

2　諸定義

CONGEST モデル

本稿で考える *CONGEST* モデルは, 単純無向グラフ連結グラフ $G = (V,E)$ により表現される. ここで V はノードの集合で $|V| = n$ とし,E は通信リンクの集合である.*CONGEST* モデルでは計算機はラウンドに従って同期して動作するものとする. 1 ラウンド内で, 隣接頂点へのメッセージ送信, 隣接頂点からのメッセージ受信, 内部計算を行う. 各辺は単位ラウンドあたり $O(\log n)$ ビットを双方向に伝送可能であり, 各ノードは同一ラウンドに異なる接続辺に異なるメッセージを送信可能である. また, 各ノードには $O(\log n)$ ビットの自然数値による ID が付与されており, 自身の隣接ノードすべての ID を既知であるとする. 各ノードはグラフのトポロジに関する事前知識を持たないものとする.

2 者間通信複雑性

2 者間通信複雑性の枠組みでは, アリスとボブの二人のプレイヤーがそれぞれ n ビットの 0/1 のデータ列で構成されるプライベートな入力 x および y を持っているとする. プレイヤーの目標は, 結合関数 $f(x,y)$ を計算することであり, 複雑性の尺度として $f(x,y)$ を計算するためにアリスとボブが通信によって交換する必要のあるビット数が用いられる.

この枠組みにおける重要な問題として, 交叉判定問題 (set-disjointness) がある. この問題では, アリスとボブはそれぞれ $x \in \{0,1\}^n$ と $y \in \{0,1\}^n$ を入力として持ち, 目的は $DISJ_n(x,y) := \bigvee_{i=1}^{n} x_i \wedge y_i$ を計算することである.n ビットの交叉判定問題を解くために, アリスとボブは通信によって $\Omega(n)$ ビット交換する必要があることが知られており, この事実を用いて最小カット発見 [3] や部分グラフ検出 [4] , 近似最大クリーク検出 [5] といったさまざまな問題に対する下界の証明がされている.

k-極大独立集合

定義 2.1 I が k-極大独立集合 $\Leftrightarrow$

$\neg(\exists I' \subseteq I |I'| = k, \exists S \subseteq V |S| \geq k+1, (I \setminus I') \cap S$ が独立集合$)$

3 結果

スペースの都合上,3 つの結果のうち,3-MIS 検証問題に対する $\Omega(n)$ ラウンドの下界の証明の概要を記載する.

3-MIS 検証問題を交叉判定問題から帰着する流れは次のとおりである. 最初にアリスとボブは特殊なグラフ $G = (V, E)$ の構築と G を G_A と G_B に分割するカット辺 C の決定を行う. 次に, アリスとボブは入力文字列に基づいてそれぞれ G_A と G_B に辺 E_A と E_B を追加する. ここで, カット辺 C は入力文字列に依存しない. グラフ G に辺を追加したグラフを $G' = (V', E')$ とすると $V' = V, E' = E \cap (E_A \cap E_B)$ 表すことができ,「G' 中に与えられている独立集合が, $DISJ_{N\times N}(x, y) = 1$ のときのみ 3-MIS でない」という特性を持つように構築を行う. グラフ G' を図 1 に示す.

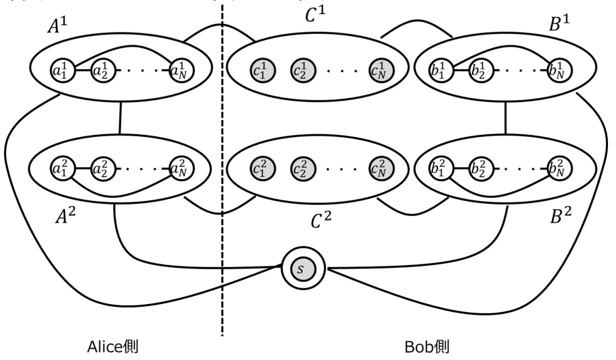

図 1: G'(V, E')

図中の頂点のうち, 灰色のものは独立集合に含まれる頂点とする. 図中で省略されている辺は次のとおりである.

- A^1, A^2, B^1, B^2 は N 頂点のクリーク K_N
- $\forall i((a_i^1, c_i^1), (a_i^2, c_i^2), (b_i^1, c_i^1), (b_i^2, c_i^2)) \in E$
- $\forall i((a_i^1, s), (a_i^2, s), (b_i^1, s), (b_i^2, s)) \in E$
- $(a_i^1, a_j^2) \in E_A \Leftrightarrow x_{i,j} = 0$, $(b_i^1, b_j^2) \in E_B \Leftrightarrow y_{i,j} = 0$

このグラフが上記の特性を満たしていることを示すために, 次の 2 点を確認する.

(i)$DISJ_{N\times N}(x, y) = 1$ のとき, グラフに与えられている独立集合が 3-MIS でない: $X_{i,j} = Y_{i,j} = 1$ とすると,s と c_i^1 と c_j^2 の 3 点を取り除いて $a_i^1, b_i^1, a_j^2, c_j^2$ の 4 点を追加できることから確認できる.

(ii)$DISJ_{N\times N}(x, y) = 0$ のとき, グラフに与えられている独立集合が 3-MIS: グラフに与えられている独立集合が 3-MIS でないと仮定する. このとき, ある 3 点を取り除くことで独立集合に追加できる 4 点が存在するが, A^1, A^2, B^1, B^2 がそれぞれクリークであるため,4 点を追加するためにはそれぞれから 1 点を選ぶ必要がある. c_i^1 を取り除いて a_i^1 と b_i^1 を追加し,,c_j^2 を取り除いて,a_j^2 と b_j^2 を独立集合に追加したとする. a_i^1 と a_i^2 が両方とも追加できるのは $x_{i,j} = 1$ のときのみであり,b_i^1 と b_i^2 が両方とも追加できるのは $y_{i,j} = 1$ のときのみであるが, これは $DISJ_{N\times N}(x, y) = 0$ に矛盾する. したがってグラフに与えられている独立集合は 3-MIS である.

今回,$N \times N$ ビットの交叉判定インスタンスをグラフに埋め込んでおり, カット辺のサイズ $|C| = 4N$ であることがわかる. アリスとボブは, このグラフ上に与えられた独立集合が 3-MIS であるかどうかを判定する分散アルゴリズムをシミュレートできる. 2 者間通信複雑性モデルでのシミュレートは, 次のように実行される.G_A 中の辺で送信されるメッセージ, あるいは G_B 中の辺で送信されるメッセージは, アリスとボブがそれぞれお互いと通信せずにシミュレートできる. カット辺 C を通じて送信されるメッセージに対しては, お互い情報を交換する必要がある.*CONGEST* モデルにおいてグラフ上に与えられた独立集合が 3-MIS であるかどうかを r ラウンドで判定するアルゴリズム $\mathscr{A}$ が存在したとすると, アリスとボブは $O(r \cdot |C| \cdot b)$ ビット通信したことになる. このグラフにおいてアルゴリズム $\mathscr{A}$ を実行すると同時に 2 者間交叉判定問題も解けていることになるので, 交叉判定問題の通信複雑性よりアリスとボブは少なくとも $\Omega(N \times N)$ ビットは通信しているはずである. したがって,$r = \Omega(N/4b) = \tilde{\Omega}(n)$ ラウンドの下界を得ることができる.

4 今後の課題

現在,3-MIS 検証問題に対する $\tilde{\Omega}(n)$ ラウンドの下界を一般の k に拡張できないか検討中である. それに加えて,3 つの結果に対する直感的な証明が思い浮かんでいる程度にとどまっているので, それを定式化してフォーマルな証明を書くことができるようにする必要がある.

参考文献

[1] Ken-ichi Kawarabayashi, Seri Khoury, Aaron Schild, and Gregory Schwartzman. Improved distributed approximation to maximum independent set. *arXiv preprint arXiv:1906.11524*, 2019.

[2] Yuval Efron, Ofer Grossman, and Seri Khoury. Beyond alice and bob: Improved inapproximability for maximum independent set in congest. *arXiv preprint arXiv:2003.07427*, 2020.

[3] Mohsen Ghaffari and Fabian Kuhn. Distributed minimum cut approximation. In *International Symposium on Distributed Computing*, pages 1–15. Springer, 2013.

[4] Orr Fischer, Tzlil Gonen, Fabian Kuhn, and Rotem Oshman. Possibilities and impossibilities for distributed subgraph detection. In *Proceedings of the 30th on Symposium on Parallelism in Algorithms and Architectures*, pages 153–162, 2018.

[5] Artur Czumaj and Christian Konrad. Detecting cliques in congest networks. *Distributed Computing*, 33(6):533–543, 2020.