MDD を用いた変更の少ない選挙区割の列挙

非会員　京都大学　*市野達也　ICHINO Tatsuya
05001214　京都大学　川原 純　KAWAHARA Jun
非会員　京都大学　湊 真一　MINATO Shin-ichi
01507360　文教大学　堀田敬介　HOTTA Keisuke

1.　はじめに

選挙区割問題は，指定された都道府県を指定された代表人数と同じ個数の選挙区に分割し，一票の較差がなるべく小さな区割を求める問題である [7]．一票の較差以外にも各選挙区の形状や地域間の親密度など，様々な評価尺度で「良い」区割が必要となる．現状の区割からなるべく少ない変更により，より良い選挙区割を求めたい．既存研究では，指定された較差以下となるすべての区割を求めて，零抑制型二分決定グラフ（ZDD）[6] として圧縮して表現する手法が提案されている [3]．本研究では，多分決定グラフ（MDD）[5] で表現される区割集合のそれぞれの区割について，与えられた自然数 m 個以下の市町村の選挙区の所属を変更することで得られる全区割を MDD で構築する手法を提案する．

2.　準備

1 つの都道府県は頂点重み付きグラフ $G = (V, E; w)$ で表され，V の要素は市町村に対応し，2 つの市町村が隣接する場合，またそのときに限り，対応する辺が E に含まれる．w は頂点から自然数への関数であり，対応する市町村の人口を表す．$V = \{v_1, \ldots, v_n\}$ とし，$v_1 < \cdots < v_n$ とする．本稿では選挙区の個数は k に固定する．V の分割 $(V_1, \ldots, V_k)$ を区割と呼ぶ．ここで，$\bigcup_{i=1,\ldots,k} V_i = V$ かつ，異なる i, j について $V_i \cap V_j = \emptyset$ を満たす．$\min_i\{\sum_{v \in V_i} w(v)\}$ と $\max_i\{\sum_{v \in V_i} w(v)\}$ の比を較差と定義する．

MDD は各変数が k 種類の値をとり，出力が 1 か 0 である多値論理関数を効率的に表現するデータ構造である．本稿では，MDD を用いて区割集合を表すため，MDD の説明を区割集合に沿って行う．MDD は非巡回有向グラフである．MDD は出次数が 0 の頂点を 2 つもち，それぞれ 0-終端と 1-終端と呼ぶ．終端以外の節点は非終端節点と呼び，k 本の出る枝をもち，それぞれ i-枝 $(i = 1, \ldots, k)$ と呼ぶ．非終端節点はラベルと呼ばれる V の要素をもつ．ラベルが $v_i \in V$ である節点は，ラベルが $v_{i+1} \in V$ である節点か，または終端節点を指す．$i = n$ のときは終端を指す．入次数が 0 である節点が 1 つ存在し，それを根と呼ぶ．

根から 1-終端までの有向パスを考える．この有向パス上ではラベルが v_i である節点を，各 i につきちょうど 1 回ずつ通る．この節点から出る有向パス上の枝が j-枝であるとき，v_i が選挙区 j に属すると解釈する．この解釈の下で，根から 1-終端までの有向パスは 1 つの区割に対応する．MDD が表す区割集合は，根から 1-終端までのすべての有向パスに対応する区割の集合であるとする．

3.　区割集合を表す MDD の構築

文献 [3] では，誘導部分グラフの辺集合によって区割を表現し，区割集合を，集合族を表現するデータ構造である ZDD を用いて表現していた．この方法では，頂点のある選挙区から別の選挙区への移動が表現しにくい．本稿では，区割集合を表現する MDD を構築する手法を新たに提案する．ここで，L, U を自然数とし，各区割は，各 $i \in \{1, \ldots, k\}$ について $L \leq \sum_{v \in V_i} w(v) \leq U$ を満たしているとする．L, U を適切に設定すると，各区割が指定した較差以下であるという条件を課すことが可能となるが，指定した較差以下の区割をすべて求めることはできず，今後の課題とする．

提案手法の概要を述べる．始めに，$G = (V, E)$ の連結成分（頂点集合）であり，重み和が L 以上 U 以下であるものの集合を ZDD（$k = 2$ の MDD とみなせる）として表現し，それを $\mathcal{Z}^1$ とする．区割集合を表現する MDD $\mathcal{M}^2$ をトップダウンに構築する．その際にサブセッティング法 [2] と呼ばれる手法を用いる．構築する MDD の各節点に，$\mathcal{Z}^1$ のいずれかの節点へのポインタを k 個保持させる．それらのポインタはそれぞれの選挙区を表している．最初に MDD $\mathcal{M}^2$ の根節点を作成する．根節

点には，$\mathcal{Z}^1$ の根節点へのポインタを k 個保持させる．以下，節点から i-枝 $(i=1,\ldots,k)$ の子節点の作成を，幅優先的に行う．親節点に，$\mathcal{Z}^1$ の節点へのポインタ $(n_1,\ldots,n_k)$ が保持されているとすると，i-枝の先の子節点には，$(n'_1,\ldots,n'_k)$ を保持させる．ここで，n'_i は n_i の 1-枝の先であり，n'_j $(j\neq i)$ は n_j の 0-枝の先である．$n'_1,\ldots,n'_k$ のいずれかが 0-終端になった場合，子節点も 0-終端となる．$n'_1,\ldots,n'_k$ のすべてが 1-終端になった場合，子節点も 1-終端となる．MDD $\mathcal{M}^2$ の根節点から 1-終端までの任意の有向パス P を考えると，i-枝を通った（ラベルの）頂点の集合は $\mathcal{Z}^1$ の要素であり，したがって重み和が L 以上 U 以下である G の選挙区である．これがすべての i について成り立つので，P は G の正しい区割である．

4. m 個の市町村を変更した区割の集合

区割集合を表現する MDD $\mathcal{M}$ が与えられた際，$\mathcal{M}$ に含まれる各区割について，頂点が属する選挙区を高々 m 個変更して得られる区割すべてを求め，それらを MDD として構築する手法を提案する．この MDD を $\mathsf{move}(\mathcal{M},m)$ と書く．提案手法は組合せ遷移の手法 [1] を基にしている．詳しい説明は省略し，MDD の再帰式のみを記述する．MDD の根節点の i-枝が指す節点から到達できるすべての節点と枝もまた MDD とみなすことができ，これを $\mathcal{M}_i$ と書く．$\mathcal{M}$ の根節点のラベルを v_1 とする．構築する MDD $\mathsf{move}(\mathcal{M},m)$ は，根節点のラベルは v_1 である．根節点の i-枝の先の MDD は以下の通りになる．根節点の i-枝の先は，v_1 の属する選挙区が i であることを意味する．元の区割で v_1 の属する選挙区が i である場合，残りの頂点のうち，高々 m 個変更する．これは再帰的に $\mathsf{move}(\mathcal{M}_i,m)$ と書ける．また，元の区割で v_1 の属する選挙区が j $(j\neq i)$ である場合，v_1 の選挙区を j から i に変更し，残りの頂点のうち，高々 $m-1$ 個を変更する．これは再帰的に $\mathsf{move}(\mathcal{M}_j,m-1)$ と書ける．これらの和集合が根節点の i-枝の先の MDD であり，$\mathsf{move}(\mathcal{M}_i,m)\cup\left(\bigcup_{j\neq i}\mathsf{move}(\mathcal{M}_j,m-1)\right)$ となる．和集合は MDD の演算により計算できる．再帰の末端については省略する．

区割が $(V_1,\ldots,V_k)$ で表現されている場合，V_i と V_j を交換しても同じ区割を表す．区割を一意に表現するために，正規形を定める．move 演算で得られる区割は正規形を満たすとは限らないが，区割集合のそれぞれの要素を正規形に変更する normalize 演算を設計できる．詳細は省略する．

move と normalize 演算を実行後の MDD と，前節で作成した MDD の共通部分演算を行うことで，人口の上下限を満たした区割を抽出できる．

表 1: move と normalize 演算の結果．時間は秒．

県	$m=3$ 時間	$m=3$ 区割個数	$m=5$ 時間	$m=5$ 区割個数
兵庫県	72	2.4×10^7	8,155	3.0×10^{11}
福岡県	127	5.9×10^7	16,230	1.4×10^{12}
愛知県	767	1.8×10^8	>20,000	時間超

5. 実験結果

計算機実験の結果の一部を示す．用いた計算機は CPU が Intel Xeon Gold 6134 (3.20GHz)、メモリが 3.0TB である．いくつかの都道府県に対して，2022 年衆議院議員選挙区の勧告案 1 つからなる MDD を作成し，move 演算と normalize 演算を施した結果を表 1 に示す．$m=3,5$ とした．講演ではより詳細な実験結果を示す.

謝辞

本研究の一部は，JSPS 科研費 JP20H00605, JP20H05794, JP20H05964 の助成を受けたものです.

参考文献

[1] Ito, T. et al., “ZDD-Based Algorithmic Framework for Solving Shortest Reconfiguration Problems,” CoRR, 2207.13959, 2022.

[2] Iwashita, H., Minato, S., “Efficient top-down ZDD construction techniques using recursive specifications,” TCS Technical Reports TCS-TR-A-13-69, 2013.

[3] Kawahara, J. et al., “Generating All Patterns of Graph Partitions within a Disparity Bound,” Proc. of WALCOM, pp. 119–131, 2017.

[4] Knuth, D. E., “The Art of Computer Programming, Volume 4A, Combinatorial Algorithms, Part 1,” Addison-Wesley Professional, 2011.

[5] Miller, D., “Multiple-valued logic design tools,” Proc. of ISMVL, pp. 2–11, 1993.

[6] Minato, S., “Zero-suppressed BDDs for set manipulation in combinatorial problems,” Proc. of DAC, pp. 272–277, 1993.

[7] 根本俊男, 堀田敬介, “選挙区最適区割問題のモデリングと厳密解導出,” 第 15 回 RAMP シンポジウム論文集, 104–117, 2003.