A-009

ZDD のトップダウン構築における変数順序付け法の実験と考察

Experiments and Considerations on Variable Ordering in Top-Down Construction of ZDDs

伊藤 華† 井上 祐馬† 湊 真一†
Hana Ito Yuma Inoue Shin-ichi Minato

1.はじめに

アイテムの組合せを要素とする「組合せ集合」を効率よく表現し，高速に演算処理する技法として，「ZDD」（ゼロサプレス型二分決定グラフ）と呼ばれるグラフ表現が知られている [1].

Knuth の示したパス列挙アルゴリズム「Simpath」[2]は ZDD をトップダウンに構築することで，与えられたグラフに対し高速に二点間の s-t パスの全列挙を行うことを可能とする．このアルゴリズムを拡張し，閉路や全域木等，与えられたグラフ構造に対して様々な制約条件を満たす部分グラフ構造を列挙することを可能にしたアルゴリズムを総称してフロンティア法と呼ぶ．この手続きは，電力網の最適化やパズル問題の生成と解答，地域に対する避難所の配置問題，電子回線の配置配線問題，そして選挙区割を行う問題など様々な現実的問題に応用可能である[3].

一般に ZDD のサイズは，グラフの変数順序によって大きく変化することが知られている．本研究では，フロンティア法で ZDD を構築する際の変数順序付けの方法について，実験による比較と考察を行う．

2.フロンティア法の変数順序付け

2.1 フロンティア法における ZDD と変数順序

ZDD(Zero-suppressed Binary Decision Diagram; ゼロサプレス型二分決定グラフ)とは2分決定木を簡約化して得られる非巡回有向グラフであり，組合せ集合の表現に適している．組合せ集合とは，「n個のアイテムから任意個を選ぶ組合せ」を要素とする集合である．例えば集合$\{a,b,c\}$があったとき，$\{a,ab\}$, $\{\Phi\}$はそれぞれ組合せ集合であると言える．

図1に示されているように，ZDD は1つの根節点と0または1の終端節点を持ち，それ以外の各節点は2値のラベル付けに対応した0-枝・1-枝と呼ばれる二つの出力枝を持つ．終端以外の節点は，組合せ集合のアイテムに対応するラベル（以後，変数と呼ぶ）を持つ．根節点から1の終端節点に至るパスが，組合せ集合に含まれる個々の組合せを表す．ZDD では変数順序は固定されており，根節点から終端に至る各パス上で，順序が逆転したり，同じ変数が2度出現したりすることはない．

フロンティア法は，頂点の集合 $V=\{v_1,...,v_m\}$と辺の集合 $E=\{e_1,...,e_n\}$で定められるグラフ $G=(V,E)$と制約条件が与えられた際，その条件を満たす部分グラフを列挙するアルゴリズムである．制約条件の例としては，与えられた始点 - 終点間の経路などがある．

フロンティア法においては処理過程で$\{e_1,...,e_n\}$を変数と

† 北海道大学大学院情報科学研究科

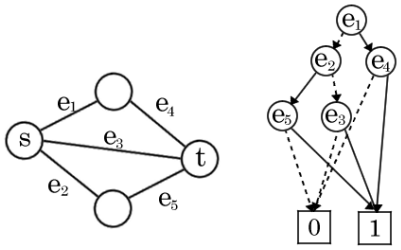

図1 入力グラフと s-t パスの ZDD

する ZDD が構築され，解の出力はその完成形として与えられる．フロンティア法では，各変数を使うか使わないかで2^n通りに場合分けしながら，求める部分グラフの集合を表す ZDD をトップダウンに（根節点から終端節点に向かって幅優先順に）構築していく．探索の途中状態は不完全な ZDD（多くの終端節点を持つ二分決定グラフ）として表現されるが，この各終端節点は mate と呼ばれるデータにひも付けされている．これはそれまでに選択された辺の状態を抽象化した形で保持している．探索の途中状態において，処理済みの辺 $e_1,...,e_{i-1}$と未処理の辺 $e_i,...,e_n$ の両方に接している頂点の部分集合をフロンティアと呼び，これに含まれる頂点は処理の進行と共に変動する．各節点の mate は，それまでに選択された辺の状態を抽象化した形で保持している．

フロンティア法においては，フロンティアのサイズ（頂点数）に対して，区別できる状態数が指数関数的に増大するため，ZDD の横幅がそれだけ増大する可能性がある．フロンティアサイズをなるべく小さく抑えることが ZDD サイズを抑える鍵となる．以下では、フロンティアサイズを抑制するような変数順序を割り当てる方法について考察する．

2.2 フロンティアサイズを抑制する変数順序

従来より公開されているフロンティア法の実装(Graphillion)[4]においては，元のグラフの各頂点を幅優先順に辿ったときに，最初に触れた辺の順序を用いて，各辺の変数順序付けを行っている（この従来手法を「幅優先順」と呼ぶ）.

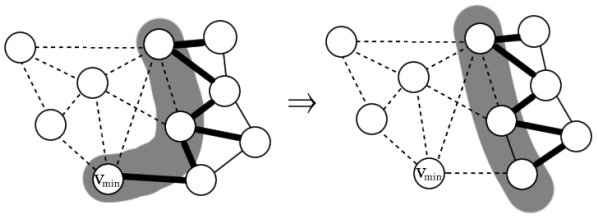

図2 v_{min} のフロンティアからの削除

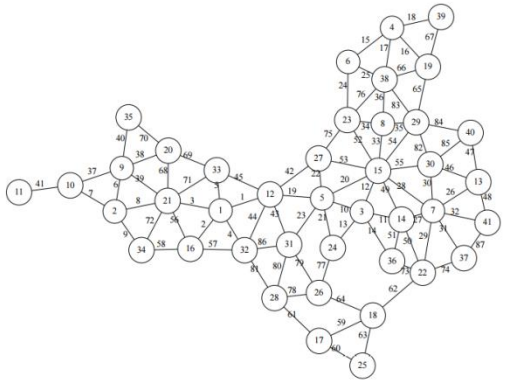

図3 茨城県内自治体の隣接関係を示したグラフ

	頂点数	辺数	幅優先探索			提案法		
			平均	最大	ZDD	平均	最大	ZDD
完全	8	28	4.96	7	578	4.96	7	724
グリッド	81	144	6.60	9	238657	7.38	9	167345
ランダム	15	53	8.22	11	59616	6.20	8	20831
茨城県	41	87	8.61	11	13166	3.83	5	716
大阪府	72	136	5.85	12	29675	4.13	6	3691
日本	47	104	4.39	8	1609	3.22	5	827
アメリカ	48	103	5.33	9	7029	4.29	6	2891

表1　平均・最大フロンティアサイズと ZDD サイズ

本稿では辺の順序を決める際，現在フロンティアに含まれている頂点の中で，未処理頂点と接続する辺の本数が最小の頂点に着目し，その頂点が早くフロンティアから抜け出すように辺を選んでいく，いわゆる貪欲法により順序を決定する方法を提案する．図2に，フロンティア内で未処理頂点と接続する辺の本数が最小である頂点 v_{min} がフロンティアから外される様子を示す．この図において，実線は未処理の，点線は処理済みの辺であることを示しており，両種類の辺と接続されている頂点がフロンティアである．なお、本手法では開始点の選択により異なる順序付けとなるが，現在の実装では、次数が最も少ない頂点を１つ選んで開始点としている．

3.評価実験

上記の貪欲法による変数順序付けのアルゴリズムを C++ で実装した．処理の各ステップにおけるフロンティアサイズの最大値と平均値，そして ZDD の節点数を計測した．また，同じグラフを幅優先順で処理した際のそれらと比較，評価する．

本稿においては，二つのアルゴリズムを用いて複数種類のグラフに処理を行った．基礎的な構造のグラフとしては K_8完全グラフ，9×9のグリッドグラフ，頂点数15のランダムグラフを処理した．現実的なものとしては，茨城県，大阪府，日本，米国の自治体・都道府県・州を頂点，その隣接関係を辺としたグラフをそれぞれ処理した．茨城県のグラフを図3へ示す．実験は OS が MacOSX 10.10.3 ，メモリ容量は8GB という環境下で行った．

4.実験結果と考察

結果は表1の通りである．提案法による順序付けをした結果，従来の幅優先探索による順序付けと比べてフロンティアサイズの平均値，最大値，ZDD のサイズは全体の傾向として削減されている．最も効果が大きかったのは茨城県であった．傾向としては，規則性のあるグラフと比べればランダムグラフや地図グラフのような構造にばらつきのあるグラフの方が大きく削減されているとも言える．実験結果より，フロンティアサイズを抑えることで ZDD のサイズを抑えるという当初の目的は多くの場合で達成されていると言える．ただし，グリッドグラフのように平均フロンティアサイズが増加しているのに ZDD のサイズが削減されている例外も存在する．また，完全グラフのように ZDD のサイズ自体が増加している例もあるため，あらゆるグラフに対して効果があるとは言えない．

5.まとめ

今回の実験を通じ，提案手法を用いてフロンティア法の変数順序付けを行いフロンティアサイズを削減することは，特に地図などの現実的なグラフを取り扱う際に大幅な ZDD のサイズダウンに結びつくことが分かった．

今後の課題としては，フロンティア法の変数順序付けに関してその他の手法についても検討していく必要がある．例えば，今回の貪欲法では先読みを一切行っていないが，数ステップ進んだ状態を考慮した順序づけ方法が考えられる．また，平均フロンティアサイズが増加しているのに ZDD が削減される例も見られることから，処理過程でのフロンティアサイズの推移を分析することも必要と思われる．

謝辞

実験にご協力頂いた本研究室・鈴木浩史さんと奈良先端大・川原純先生に感謝致します．なお本研究の一部は JST ERATO 湊離散構造処理系プロジェクトからの助成による．

参考文献

[1] 岩下 洋哲, 川原 純, 湊 真一, “再帰的仕様記述を用いた組み合わせ列挙 ZDD の効率的な構築方法” , 電子情報通信学会技術研究報告. VLD, VLSI 設計技術 112(320), 25-29, 2012-11-19.
[2] D. E. Knuth, “The Art of Computer Programming”, Volume 4A, Combinatorial Algorithms, Part 1, 1st edition, Addison - Wesley Professional, March 2011.
[3] 川原 純, 湊 真一, “組み合わせ問題の解を列挙索引化する ZDD アルゴリズムの汎用化”, 電子情報通信学会技術研究報告. COMP, コンピュテーション 112(93), 1-7, 2012-06-14.
[4] 井上 武, “Graphillion:巨大なグラフ集合を扱うソフトウェアライブラリ”, 電子情報通信学会技術研究報告. IN, 情報ネットワーク 113(140), 43-47, 2013-07-11.