分散検証環境 DiVinE を用いた分散 LTL モデル検査アルゴリズムの性能評価

渋谷 健介† 上田 和紀‡
†早稲田大学理工学研究科 ‡早稲田大学理工学術院

1 はじめに

LTL モデル検査で最適とされる逐次アルゴリズムは postorder の DFS に基づいているが，これは P-完全とされており，並列化は困難となっている．そのため，分散メモリ環境で動作する代わりのアルゴリズムがいくつか提案されている [1]．しかし，どのアルゴリズムが最も有用であるかはまだ判明しておらず，特定のアルゴリズムがどういう場合に有用であるかの情報も少ない．そこで本研究では実験により各分散 LTL モデル検査アルゴリズムの性能を評価し，それらのアルゴリズムの特性を明らかにすることを目的としている．

2 分散 LTL モデル検査

オートマトンベースの LTL モデル検査手法は，モデル検査問題をグラフの受理サイクル探索問題として解く．文献 [1] における並列アルゴリズムは基本的に到達可能性の検査を繰り返し行うことでサイクル探索を行っている．また，生成した状態はハッシュ関数により各ノードにランダムに分配される．ここでは各アルゴリズムについて簡潔に述べる．

Maximal Accepting Predecessors (MAP) 受理サイクルを作る受理頂点は自分自身を predecessor に持つという観点に基づく．各受理頂点について順序付けし，各頂点は最大の受理 predecessor (*MAP: Maximal Accepting Predecessor*) のみを保持するようにする．そしてこの *MAP* を successor に伝播させていき，最大のものに更新していく．すると，受理頂点が自分自身の *MAP* であるならば，それが受理サイクル上にあることが言える．

Back-Level Edges (BLE) 辺 (u, v) において v の属するレベルが u の属するレベル以下であるような辺のことを *Back-Level Edge (BLE)* と言う．本アルゴリズムは BFS を元にしており，全ての受理サイクルは少なくとも一つの *BLE* を含むという観点に基づく．現在のレベルから全ての *BLE* を探索し，それらの *BLE* に対して自己到達可能性のテストを行う，という二つのフェーズを交互に行うことでサイクル探索を行う．

One-Way-Catch-Them-Young (OWCTY) 受理サイクル上には存在し得ない頂点，つまり受理頂点から到達不可能な頂点と predecessor を持たない頂点をグラフから繰り返し取り除くことによりサイクル探索を行う．これ以上取り除く頂点が存在しなくなったとき，頂点集合が空であればサイクルが存在しないことになり，空でなければそれが反例となる．

Negative Cycle Detection (NC) 受理サイクル探索問題をより並列化しやすい単一経路最短問題における負閉路探索問題に帰着させる．受理頂点からの辺の長さを −1，それ以外の辺の長さを 0 に割り当てることにすると，負の長さのサイクルは受理サイクルと一致する．したがってこの問題は Bellamn-Ford アルゴリズムの分散バージョンと見ることができる．

3 評価実験

分散検証環境 DiVinE [2] を使用し，各アルゴリズムの性能評価を行った．DiVinE は分散環境における検証システムであり，前節のアルゴリズムを実装している．また，モデルとして BEEM ベンチマーク [3] から，状態数が 10^6 を超えるモデル 228 個を使用した．なお，実験に用いた環境は Dual Core AMD Opteron 2.0GHz を 2 個，主記憶容量 4GB を搭載したノード 9 台から構成されるクラスタ（最大 32 PEs）で，ネットワークは 1000 BASE-T Ethernet，実行環境は MPICH2 である．

3.1 実験結果

解けた問題数 制限時間を 20 分とした場合の解けた問題数は表 1 のようになり，MAP が一番多く解くことができ，NC, BLE と続くことがわかる．これらは on-the-fly に動作するため，特に受理サイクルを含む問題に対して良い性能を示した．一方，OWCTY アルゴリズムは最初にグラフ全体を構築する必要があるため，制限時間や特にメモリの制限で解けない問題が多数存在した．また，最も速く解けた問題数は表 2 のようになり，PE 数が増えるにつれて MAP アルゴリズムが最も速く動作する割合が高まっていることがわかる．

平均速度向上比 1 PE に対する PE 数毎の平均の速度向上比は図 1 のようになる．より精度の高い結果を得るために，ここでは対象を受理サイクルを含まない問題に限定している．MAP が一番性能向上しているが，PE 数を 16 台よりも増やすと性能を落としてしまっている．MAP 以外のアルゴリズムについても同様の傾向を示し，これは通信のオーバーヘッドが大きくなってしまうことが影響していると考えられる．また，32PE

Performance evaluation of distributed LTL model checking algorithms by DiVinE
†Kensuke SHIBUYA ‡Kazunori UEDA
†Graduate School of Science and Engineering, Waseda University
‡Faculty of Science and Engineering, Waseda University

表 1: 解けた問題数

	1x1	2x1	4x1	8x1	8x2	8x3	8x4
MAP	193	198	203	210	211	211	212
BLE	175	180	186	189	191	191	188
OWCTY	109	124	128	136	143	139	136
NC	165	186	192	201	201	207	203

表 2: 最も速く解けた問題数

	1x1	2x1	4x1	8x1	8x2	8x3	8x4
MAP	78	79	84	92	125	135	129
BLE	79	46	41	29	56	47	57
OWCTY	8	9	7	6	1	1	1
NC	40	72	76	85	35	35	30

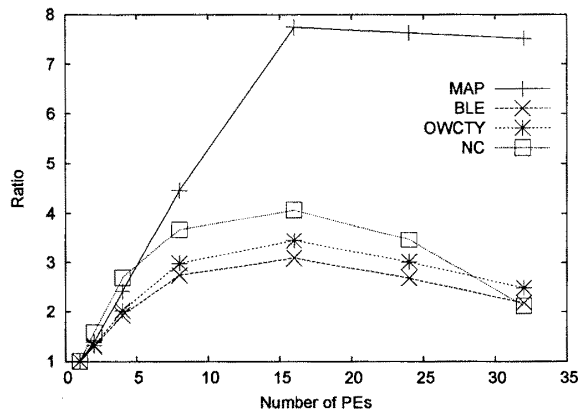

図 1: 平均速度向上比

を MAP だけで使用するよりも，MAP と BLE で 16PE ずつ同時に使用した方が速く解けるケースが 170 題あり，その場合の平均速度向上比は 7.91 となった．これは MAP だけで 16PE 以上使用したどの場合よりも高くなっており，より大規模な環境においては一つのアルゴリズムだけを使用するのではなく，複数のアルゴリズムを組み合わせることが有効であると考えられる．

通信量 一部のモデルに対して通信状況を詳しく調べてみると，通信関数の実行時間は全実行時間の 2〜3 割程度を占めていることがわかった．なお，これには idle 状態の時間が含まれていない．通信関数の実行時間と idle 状態の時間は PE 数を増やすにつれ共に長くなっており，これが 16PE 以上での性能低下に影響していると考えられる．通信時間や idle 状態の割合は状態空間の分割の仕方に強く影響されるため，いかに効率的に状態空間を分割するかが重要となる．

反例の長さ 反例の長さはどのアルゴリズムも最大数百程度と短く，多くの場合それよりずっと短い反例を発見していた．これは並列アルゴリズムが BFS に基づいているためだと考えられる．当然ながら反例は短い方が良いとされるため，これは並列アルゴリズムの利点の一つである．

3.2 各アルゴリズムのまとめ

各アルゴリズムの性能を比較してみると，MAP が安定して良い性能を示し，通信時間も比較的短かった．MAP は並列効果が一番出ており，他に比べ極端に遅いケースもほとんどないため非常にバランスの良いアルゴリズムであると言える．ただ，受理頂点を多く含む問題をやや苦手としているようである．BLE は他のアルゴリズムより特に良いケースは稀であり，並列効果もあまり出ていないなど際立った点は見受けられなかった．OWCTY は受理サイクルが存在しないモデルに対しては安定した性能を示したが，on-the-fly に動作しない欠点により受理サイクルを含むモデルでは他に劣る．NC は PE 数毎の速度の変動が大きく，分割の仕方に強く影響を受けるようである．全体的には MAP に次ぐ性能を示したが，極端に遅いケースも有り，安定感に欠ける結果となった．

4 まとめと今後の課題

分散検証環境 DiVinE を用いて各並列アルゴリズムの性能の評価実験を行い，MAP アルゴリズムが安定して良い性能であることがわかった．しかしどのアルゴリズムもある PE 数以上では性能を落としてしまい，通信量を抑えるためにより効率的な分割手法を考案する必要がある．

局所性を持つ分割手法としては性質オートマトンの性質に基づく分割手法 [4] が提案されており，これは分割数が限定されるという欠点はあるが，受理サイクルは全て 1 つのノード内に保持されるという特徴を持つ．また，モデル検査で生成される状態空間はいくつかの典型的性質を持っていると考えられるため，それらの情報は更なる探索の効率化や状態空間の分割手法の改良を行う際に有用であると考えられる．よって状態空間を完全にランダムに分割するのではなく，これらの性質を利用してヒューリスティックにより分割することは有効であると考えられ，その実装及び評価は今後の課題となる．

参考文献

[1] J. Barnat, L. Brim, and I. Cerna. Cluster-Based LTL Model Checking of Large Systems. In Formal Methods for Components and Objects, pp.259–279, 2005.

[2] J. Barnat, et. al. DiVinE — a tool for distributed verification. In Proc. CAV'06, LNCS 4144, pp. 278–281, 2006.

[3] R. Pelanek. Web portal for benchmarking explicit model checkers. Tech. Report FIMU-RS-2006-03, Masaryk University Brno, 2006.

[4] J. Barnat, et al. Property Driven Distribution of Nested DFS. in Proc. of 3rd Int'l Workshop on Verification and Computational Logic (VCL '02), 2002.