逆探索による単体的複体のシェラビリティー判定

RA 森山園子
情報理工学系研究科コンピュータ科学専攻

概要
これまでに，位相的性質と量子情報・量子計算との関連が指摘されてきた。そこで，トポロジーという概念の持つロバスト性から量子計算での超ロバスト性を確立することを目指して，組合せ的手法・トポロジカル法といった位相的手法の研究を遂行する．本研究では，トポロジーの基本構造である単体的複体のシェラビリティー判定を問題として取り上げ，組合せ的手法の１つである逆探索を利用した効率よい手法を開発した．

1 Introduction
トポロジーの基本構造である単体的複体のシェラビリティーは，トポロジーと組合せ論の両方の観点から研究されてきた重要な概念である．

しかし，与えられた単体的複体がシェラブルであるか否かを決定することは難しく，シェラビリティー判定を行うアルゴリズムやソフトウエアで，これまで実用的なものは存在しなかった．そこで[4]では，単体的複体の不変量である h-ベクトルを発展させた，森山が提案した新しい概念 h-assignment[2]に基づいて考案された，単体的複体のシェラビリティー判定を行う効率的アルゴリズム the hash-DFS method を実現するソフトウエア SHIDE (SHellability decIDEr)[3] を構築した．SHIDE は，シェラブルな単体的複体については，既存の判定方法では非常に困難な 40 程度のファセット数を持つものを次元に拠ることなく 3 時間以内に汎用の計算機上で判定結果を返し，シェラブルでないものについても，既存の判定方法では困難な例について判定結果を与える画期的なソフトウエアである．
The hash-DFS method では，探索空間を深さ優先的に探索して，シェラブルな h-assignment を見つけることにより単体的複体のシェラビリティーを判定する．再探索の回避のため，探索に行き詰まった場合は，探索木をバックトラックして新たな探索を試みると同時に，展開に行き詰まったノードの h-assignment を inappropriate h-assignment としてハッシュに登録した．
このハッシュの実装には工夫を施した．ハッシュへのメモリ割り当てが大きいほど機能性が向上するのは勿論だが，汎用の計算機というメモリ制約下での動作を目的としたため，ハッシュへの登録の際にハッシュ用メモリサイズを越える可能性が高い．その際に，できるだけ有用な情報をハッシュに残すことを目指した工夫をハッシュに施した．

しかし，探索空間が非常に大きいために，汎用の計算機のメモリ制約の元では，探索が進行するにつれて，既にハッシュに登録された不適切な h-assignment を棄却せざるを得ない．そこで，再探索にかかる手間に応じてノードに優先度を付け，ヒューリスティックスを使うことにより，再探索を回避すべきノードを優先的に残すようにしているが，the hash-DFS method で完全に再探索を回避することは不可能である．

そこで，本研究では，shellable h-assignment の探索に逆探索[1]を使った手法を提案する．我々の手法 the RS method では，探索空間 S に全域木 T(S) を与え，この T(S) を深さ優先的に探索して shellable h-assignment を見つける手法である．本手法では，理論的意味から再探索を完全に回避することができる．これは the hash-DFS method でなし得なかったことである．

更に，the hash-DFS method ではハッシュテーブルに多くのメモリを使用していたにも関らず，逆探索では現在の探索過程のノードのみを保存すれば良いので，非常に小さなメモリを使用するだけで良いという利点もある．

2 The RS method

The hash-DFS method・the RS method の探索空間 S は，単体的複体に定義される全ての *h*-assignment 集合である．*h*-assignment とは，単体的複体の不変量である *h*-ベクトルから得られる，単体的複体の各ファセットに与えられたラベル集合である．このラベル付けから，条件を満たす *h*-assignment A,B に順序 A➔B を定義できる．A➔B において，A を parent assignment，B を child assignment と呼ぶ．そこで，各ノードを *h*-assignment とし，A➔B を満たす *h*-assignment A,B に有向枝を与えて構成した acyclic 有向グラフを G(S) とする．

The hash-DFS method ではこの G(S) を深さ優先的に探索して shellable *h*-assignment の存在を調べていた．探索がある *h*-assignment A に至った時，その A に child assignment が存在しない場合は，1 ステップだけバックトラックして再度探索を開始する．このように，バックトラックの対象となった *h*-asignment、つまり shellable *h*-assignment に拡張できないものを inappropriate *h*-assignment と呼ぶ．

G(S) は tree ではないので，深さ優先探索において再度 inappropriate *h*-assignment を探索する可能性がある．従って，the hash-DFS method では，再探索の回避のため，inappropriate *h*-assignment をハッシュに登録して，探索の各ステップでハッシュを参照し，ハッシュに登録されていない child assignment を選択していた．しかし，メモリ制約の問題から，ハッシュに登録された inappropriate *h*-assignment を棄却せざるを得ず，結果的に再探索を回避できない状況になっていた．

そこで，本研究では，逆探索を使うことで，理論的に再探索を回避することに成功した．逆探索は Avis and Fukuda[1] により提案された手法で，グラフ G(S) の（ルートを除いた）各ノードの親ノード集合から親をユニークに定義できる場合は，グラフ G(S) の全域木 T(S) が得られることから，この T(S) を探索することで，再探索せずに探索空間 S の探索を可能にする手法である．再探索を完全に回避できることから，逆探索は様々な組合せ列挙問題において使われてきた．

従って，逆探索を用いるため，G(S) の各ノードの親ノードをユニークに定める．単体的複体 C の全てのファセットにインデックス C(I) を与え，G(S) の各ノード A の parent assignment 集合の中で，C(I) から導かれる辞書式順序で 1 番目に位置する parent assignment をユニークな親として定義する．この定義から，A が B のユニークな親であるときのみ A➔B と枝を与えることで，G(S) から全域木 T(S) を導くことができ，この T(S) を深さ優先探索するだけで，再探索を完全に回避して探索空間 S を探索できる．

更に，注目すべきことは，T(S) を深さ優先探索する過程で保存すべきデータは，現在のノードの *h*-assignment のみであることである．The hash-DFS method でハッシュテーブルに大量にメモリを使用していたことから，メモリ効率の点でも the RS method の優位性を確認できる．

また，the hash-DFS method では，探索過程でハッシュから情報が棄却されるため，探索空間の全てを探索できない可能性が否定できないが，the RS method では再探索無く探索空間の全てのノードを探索することが出来るので，shellable *h*-assignment の列挙を行うこともできる．

References

[1]D. Avis, K. Fukuda, Reverse search for enumeration, Discrete Appl. Math. 65 pp.21-46 (1996).

[2]S. Moriyama, *h*-assignments of shellable simplicial complexes, preprint,available at http://www-imai.is.s.u-tokyo.ac.jp/~moriso/hassign031114.pdf.

[3] Sonoko Moriyama and Masahiro Hachimori, *h*-assignments of simplicial complexes and a reverse search, submitted.

[4]S. Moriyama, A. Nagai, and H. Imai, Fast and space-efficient algorithms for deciding shellability of simplicial complexes of large size using *h*-assignments, in Proc. of International Congress of Mathematical Software (ICMS'02) pp.82-92 (2002).