Vol.2015-MPS-103 No.62
Vol.2015-BIO-42 No.62
2015/6/25

フラグメント伸長型タンパク質-化合物ドッキングのビームサーチによる高速化

小峰 駿汰[1,2,a)]　石田 貴士[1,2]　秋山 泰[1,2,b)]

概要：タンパク質-化合物ドッキングシミュレーションは，標的となるタンパク質と薬剤候補化合物の立体構造データを用いてそれらの結合状態を予測する手法である．ドッキングシミュレーションは分子の内部自由度や，回転，タンパク質との相対位置による自由度があり，原子数が数千あるタンパク質との相互作用を計算しなければならないので計算量が膨大になってしまうという問題点がある．探索空間の大きいこの探索問題への対処方法として，初期探索においてリガンドをフラグメントに分割する手法がいくつか提案されている．その中でベースとなるフラグメントを段階的に伸ばしていく方法があるが，この手法では一度計算したエネルギー値の再利用が不十分となる．また、フラグメント間の距離に基づいて辺を張ったグラフを利用してクリーク探索をする方法では，最大クリークを全て列挙した後にエネルギー値の悪いものを枝刈りしているため，効率が悪いという問題点がある．そこで，本研究では全てのフラグメントについてエネルギー値を再利用するためにエネルギーグリッドを作成し，探索部分ではビームサーチを利用する手法を提案する．その結果，ビームサーチを用いない全解探索と比較して約 9.3 倍，フラグメント分割を用いない手法と比較して 2000 倍以上の高速化を達成した．

キーワード：タンパク質-化合物ドッキング，化合物フラグメント，バーチャルスクリーニング，ビームサーチ

Fast protein-ligand docking based on fragment extension using beam search

Komine Shunta[1,2,a)]　Ishida Takashi[1,2]　Akiyama Yutaka[1,2,b)]

Abstract: Protein-ligand docking simulation is a method of predicting bound forms of a protein and a drug like compound by using their 3-D structure information. There is a problem such that this simulation has enormous computational complexity because compounds have degrees of freedom from dihedral angle, rotation, position and proteins have a lot of atoms. To deal with this large search space, some methods to devide compounds into fragments are suggested. One of these methods extends base fragments. But this method cannot reuse energy values sufficiently. Another method of fragment based methods using clique search of a graph based on distances of fragments cannot calculate efficiently because it enumerates maximum clique and then prunes candidates with bad energy values. Here, we propose a new method which makes energy grids to reuse energy values of all fragments and uses beam search in search part. Proposed method is 9.3-fold faster than the same exhaustive fragment building approach when without using beam search, and over 2000-fold faster than a naive method which does not divide compound into fragments.

Keywords: protein-ligand docking, fragment of compound, virtual screening, beam search

[1] 東京工業大学 大学院情報理工学研究科 計算工学専攻
Department of Computer Science, Graduate School of Information Science and Engineering, Tokyo Institute of Technology
[2] 東京工業大学 情報生命博士教育院
Education Academy of Computational Life Sciences, Tokyo Institute of Technology
[a)] komine@bi.cs.titech.ac.jp
[b)] akiyama@cs.titech.ac.jp

Vol.2015-MPS-103 No.62
Vol.2015-BIO-42 No.62
2015/6/25

1. 序論

ドッキングシミュレーションとは，レセプターとなるタンパク質の立体構造情報と，リガンドとなる化合物の立体構造情報を基にして，それらの分子がうまく相互作用するようなリガンドの位置及び姿勢 (配座) をコンピュータを用いて予測する手法である．リガンド内部の自由度を考慮しないリジッドドッキング [1] も知られているが，現在ではより高精度なリガンドの結合の回転などの自由度も考慮するフレキシブルドッキングの方が主流になっている [2]．ドッキングシミュレーションの問題はエネルギー関数の構築と最小のエネルギー関数値の探索の 2 つの問題に分割して考えることができる．

リガンドの探索における並進移動の間隔は 0.5 Å, 二面角の間隔は 5° 以内が望ましいが，少なめに見積もって一辺 10 Å の立方体内部で探索するとしても，3 次元方向の並進移動で $(10/0.5)^3 = 20^3$，化合物の 3 方向の回転で $(360/5)^3 = 72^3$，回転可能な単結合を平均 6 個とすると二面角は $(360/5)^3 = 72^6$ であり，$20^3 \times 72^3 \times 72^6 \approx 2 \times 10^{20}$ 通りのポーズを探索する必要があるため，スーパーコンピューターを利用しても数十億年かかってしまうと言われている [3]．そのため，探索空間の大きいこの探索問題への対処方法として様々な手法が提案されてきた．例えば，現在最も予測精度の良いドッキング手法の一つである Glide[4] で実装されているように，粗い探索から始めて段階的に細かい探索にしていく手法や，GOLD[5] や AutoDock[6] で実装されているように，遺伝的アルゴリズムを用いた手法がある．

一方，化合物は通常単結合しか回転しないため，回転不可能な部位が存在する場合が多い．そのような部位をフラグメントと呼ぶ．フラグメントを用いない手法では，全てのポーズにおいて同じフラグメント内での原子の位置関係に変化がないにもかかわらず，同じエネルギー値を何度も 1 原子ずつ計算している．そこで，無駄な計算を省くためにフラグメントに分割してドッキングシミュレーションを行う手法がいくつか提案されている．ドッキングシミュレーションの中でフラグメントへの分割を取り入れている手法としては，FlexX[7] で実装されているように，ベースとなるフラグメントから段階的に伸ばしていく方法や，eHiTS[3] で実装されているように，フラグメント間の距離に基づいて辺を張ったグラフを利用してクリーク探索をする方法がある．

これらのフラグメントに分割して探索する手法は，フラグメントに対する冗長なエネルギー計算を避けられ，最新のドッキングアルゴリズムと同等以上の精度，計算速度を実現している [8]．しかし，一方で既に提案されたフラグメントベースのドッキング手法にはそれぞれ問題が存在している．例えば，FlexX のようなやり方では最初に置くベースとなるフラグメント以外は毎回違う位置・回転角度で出現するため，エネルギー値の計算を再利用することがほとんどできていない．また，eHiTS のようなやり方では，最大クリークを全て列挙した後にエネルギー値の悪いものを枝刈りしているため効率が悪い．フラグメントベースのドッキング手法は他の既存手法と比較して，十分に高速ではあるが，このような問題が解決できればより高速にできるはずである．

そこで本研究では，ドッキングシミュレーションのための新たな探索手法を提案する．実験した結果，ドッキングポーズの探索が高速化されていることが確認できた．

2. 提案手法

2.1 レセプター側での事前計算

提案手法ではまず，ある原子がある位置に置かれた時のエネルギー値を事前に計算することで原子ごとのエネルギーグリッドを作成する．この計算は，リガンドとは独立にレセプターのみで計算できる．バーチャルスクリーニングは通常，一つのレセプターに対し数百万もの化合物の中から薬剤候補を選ぶものであるので，レセプターのみで計算できるこの値を使い回すことで効率化ができる．事前計算に関するパラメータは以下に示す．

- 使用した原子種 (6 種)
  C, C(aromatic), S, H, N, O
- 原子が置かれる空間
  リガンドに含まれる原子を点とみなしたときの重心を中心とする，一辺 $(L_{ligand} + 10)$Å の立方体（ただし，L_{ligand} はリガンド内の最遠原子対間の距離）

2.2 全体の流れ

アルゴリズム全体について大まかな流れを説明する．まず最初にリガンド (化合物) とレセプター (タンパク質) の立体構造情報，事前計算したエネルギーグリッドをファイルから読み込み，リガンドをフラグメントに分割する．そして，それぞれのフラグメントをノードとするグラフを作成し，フラグメントのエネルギーグリッドを作成する．その後フラグメントの種類，移動，回転の組み合わせをノードとしたグラフを作成し，それらの情報を基に探索を行う．そして，ドッキング計算後に最もエネルギー値の良かったリガンドの候補ポーズの立体構造情報を出力する．

2.3 リガンドのフラグメント分割

提案手法では以下の基準に基づいてフラグメント分割を行った．

- 単結合以外の結合は回転できないものとし，水素原子は他原子と比べて非常に小さいため，フラグメント分割のための基準としては無視する．
- 回転不可能な単結合以外の結合の両端の 2 原子は，全

て同一フラグメントに含める．
- 環構造の場合も同一のフラグメントに含める．
- その他の単結合の場合でも，その結合を切断したときにどちらか片方に 1 原子のみが残される場合は，回転させてもその原子のみがその場で回転するだけなので位置関係に変化はないので同一フラグメントに含める．
- 切断した時に孤立する原子が更に他の 2 原子以上と結合している場合，つまりその原子が他の 2 つ以上のフラグメントと結合している場合には結合角が崩れてしまう可能性があるので，その場合に限り同一フラグメントには含めないが，それ以外で片方の原子が孤立する場合は同一フラグメントに含める．
- 最後に，全ての水素原子をその原子が結合している原子の属するフラグメントに含める．

図 1 にフラグメント分割の例を示す．

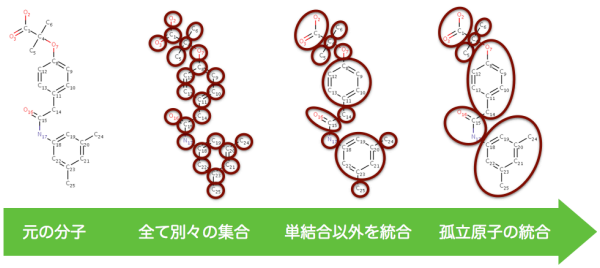

図 1 フラグメント分割の例

2.4 フラグメントグラフ作成

分割されたフラグメントについて，元々のリガンドではフラグメント同士がどのように結合していたかの情報を有するグラフを作成する．具体的には，グラフにおけるノードはフラグメントを表し，辺は 2 つのフラグメント間にまたがる結合の両端の原子の位置の情報を持つ．

2.5 フラグメントの回転

2.5.1 回転方向の列挙

提案手法では，回転において均等なサンプリングを行うために正十二面体に基づく 60 通りの回転を用いている．$\phi = (1+\sqrt{5})/2$，$\mathbf{v}_a = (0, \phi^3, \phi^2), \mathbf{v}_b = (0, 1, \phi^2)$ としたとき，$\mathbf{v}_a$ がある面の中心を貫き，$\mathbf{v}_b$ がその面のある頂点を貫くような原点を中心とした正十二面体が作れる．この正十二面体を P とする．$\mathbf{v}_a$ を $\mathbf{v}_b$ を軸として $2k\pi/3$(k は整数) 回転させたとしても，$\mathbf{v}_a$ は P 上のある面の中心を貫いている．同様に，$\mathbf{v}_b$ を $\mathbf{v}_a$ を軸として $k\pi/5$(k は整数) 回転させたとしても，$\mathbf{v}_b$ は P 上のある頂点を貫いている．回転列挙のアルゴリズム内では，このような回転を適切な順番で適用している．

2.5.2 フラグメントのエネルギーグリッド

原子のエネルギーグリッドに基づいて，フラグメントのエネルギーグリッドを作成する．フラグメントのエネルギーグリッドは，あるフラグメントをある回転方向で空間上に配置した時，そのフラグメント内の全ての原子に関するエネルギー値の合計を計算したものである．

2.6 トリリニア補間

エネルギーグリッドは，格子点上の座標におけるエネルギー値しか保持していない．一方，原子の座標は任意の実数値を取ることができるので，原子の座標 (x, y, z) が与えられた時にその点におけるエネルギー値を推定しなければならない．そのために，線形補間を 3 次元に拡張したものであるトリリニア補間 [9] を用いてエネルギーグリッドの補間を行った．

2.7 フラグメント伸長

2.7.1 探索

探索について大まかな流れを説明する．

(1) フラグメントに順序をつける．
(2) 最初のフラグメントから順にそのフラグメントが置ける場所に置く．置けるかどうかの判定は，そのフラグメントと結合しているフラグメントとの位置関係と，他の原子が重なっていないかをもとに判定する．
(3) 最後のフラグメントを置く時は，それまで置いてきたフラグメントのエネルギーの合計と置いた場所を記録する．
(4) 最終的に全てのフラグメントのエネルギーの合計が良い (低い) ポーズを出力する数だけ選び，改めて通常の方法によりタンパク質原子との 2 体間のエネルギーを計算する．

2.7.2 結合グラフ作成

探索時に直前に選んだフラグメントの位置・回転に結合できる次のフラグメントの位置・回転を毎回列挙していては時間がかかってしまう．これに対し，事前計算として各フラグメントのそれぞれの回転に対して結合可能な次のフラグメントの位置・回転への辺を張ったグラフを作る．この結合グラフを利用して，探索中にかかる計算コストを削減している．

2.7.3 両立可能性判定

結合グラフの辺を張る際に 2 つのフラグメントが満たしているべき条件について説明する．フラグメント 1 とフラグメント 2 は原子 a_1, a_2 間の結合でつながれているとする．フラグメント 1 とフラグメント 2 は独立に移動・回転をするので，1) a_1 から見た a_2 への結合の端点 a_2' と，フラグメント 2 上にある a_2 の実際の位置や，2) a_2 から見た a_1 への結合の端点 a_1' と，フラグメント 1 上にある a_1 の実際の位置，は必ずしも一致するわけではない．そこで，少

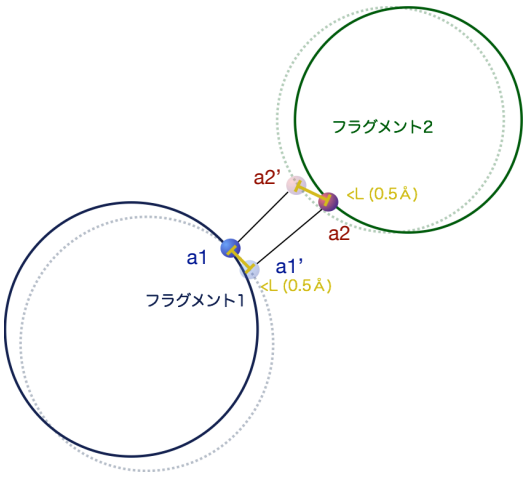

図 2 フラグメント間の結合が回転不可能

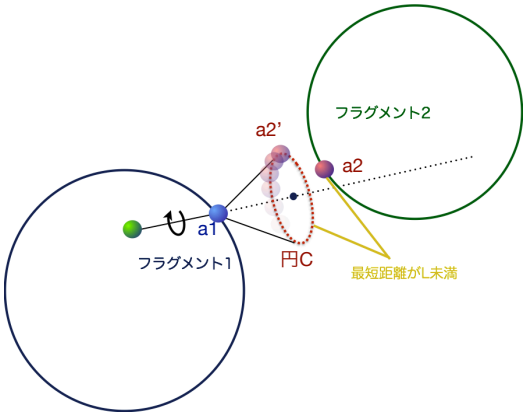

図 3 フラグメント間の結合が回転可能

しのずれを許して，$\max(|a_1 - a_1'|, |a_2 - a_2'|) < L$ のときにこれらのフラグメントを結合可能とみなす．今回は L を並進移動のサンプリング間隔である 0.5 Å にした．a_1 や a_2 の結合の中で同一フラグメントに含まれる原子同士を結ぶものがある場合，フラグメント間を結ぶ結合は回転することができるので，場合分けをして対処する．

- フラグメント間の結合が回転不可能な場合
  結合が回転不可能な場合は，a_1, a_2 が固定された時に a_2' や a_1' が一通りしか取りようがないので，単純に a_2', a_1' の位置を算出して距離を計算すれば良い．(**図 2**)
- フラグメント間の結合が回転可能な場合
  原子 a_1 が a_2 以外の原子 a_11 と結合していて，その結合が単結合の場合，a_1,a_2 間の結合は回転可能となる．この場合は，a_2' は a_1 と a_11 を貫く軸に直交する円 C 上に存在できる．よって，a_2 と a_2' の距離はこれらの取り得る距離の中で最小のものとする．(**図 3**)

2.7.4 ビームサーチ

可能な探索を全て行ってしまうと，計算すべきノードの数はフラグメントの個数に対し指数的に増加してしまう．フラグメントの数は 1〜15 個程度であり，1 つのノードからの分岐は数十本程度なので，例えば 30 本とすると $30^{15} \fallingdotseq 10^{22}$ となり，全て計算するのは現実的に不可能である．そこで，計算量を減らすために本研究ではビームサーチを用いた．ビームサーチは，探索木においてそれぞれの深さ d $(d = 1, 2, 3...)$ のノードの探索回数をビーム幅 w_{beam} に絞る探索手法である．ビームサーチは一般的に 2 種類の方法が知られている．

- 幅優先探索ベース
  各深さごとに評価値が上位 w_{beam} 個選び探索する．その深さについて全て探索し終わったら，次の深さについて探索する．w_{beam} が 1 のときに山登り法，w_{beam} が無限大のときに幅優先探索と同等の探索となる．
- 最良優先探索ベース
  今まで探索したノードから遷移可能な全てのノードを保持しておき，最も評価値の良い (エネルギー値の低い) ものから優先的に選んでいく．同じ深さのものを既に w_{beam} だけ探索してしまっている場合は，無視する．w_{beam} が 1 のときに山登り法，w_{beam} が無限大のときに最良優先探索と同等の探索となる．

最良優先探索は負辺がない場合，最初に見つかる解が最適解であることが保証されている [10]．探索空間が木で表現できる (始点からあるノードへのパスが一通りしか存在しない) とき，同様に最初に見つかる N 個の解は上位 N 個の解であると保証される．ただしこの問題の場合，最良優先探索ベースのビームサーチをそのまま用いてしまうと，負辺があるので最初にみつかる上位 N 個の解がその探索で見つかる上位 N 個の解と一致するとは限らない．そこで本研究では最良優先探索を改良して Dijkstra 法ベースの手法を実装した．Dijkstra 法は負辺が存在しないときの最良優先探索であるので，そのまま適応することができない．そこで，負辺を減らすために各フラグメントごとにそのフラグメントを追加した際のエネルギー値の最小値をあらかじめ計算しておき，探索においてはその最小値をエネルギーから引いた値を比較基準にした．Dijkstra 法ベースのビームサーチによって得られる解は，幅優先探索によって得られる解と等しい．しかし，Dijkstra 法ベースの場合は全ての深さにおける探索がビーム幅に到達する必要はないので，探索するノード数は幅優先探索よりも少ない．

フラグメントの個数を F，ビーム幅を $w_{beam}(\gg F)$，探索ノード 1 つごとにかかるコストを C_{node} としたとき，幅優先探索ベースで上位 w_{beam} 個のノードを選ぶ際の実装で優先度付きキューやソートを用いると，計算時間は $\Theta(F \cdot w_{beam} \cdot (C_{node} + \log(w_{beam})))$ である．Dijkstra 法ベースでは，探索するノード数を $N(\leq F \cdot w_{beam})$ とすると，計算時間は $\Theta(N \cdot (C_{node} + \log(N))) \subset \mathcal{O}(F \cdot w_{beam} \cdot (C_{node} + \log(w_{beam}))$ である．よって，Dijkstra 法ベースの計算量は漸近的に幅優先探索以下となる．

一方，幅優先探索では $\Theta(N)$ で要素 N 個のリストから

表 1 実行環境

計算機	TSUBAME 2.5 Thin ノード
CPU	Intel Xeon X5670 2.93[GHz] (6 core) ×2
メモリ	54GB

値の上位 k 個を選択するアルゴリズムが適用でき [11]，そのときのビームサーチの計算時間は $\Theta(F \cdot w_{beam} \cdot C_{node})$ となる．このとき，Dijkstra 法ベースとの計算量の大小関係は N や C_{node}，w_{beam} の値に影響されるので一概にどちらの計算時間が少ないかは決定できない．本研究では，実際に計算機実験を行うことで決定した (4.4 節)．

2.7.5 繰り返し探索

提案手法では，ビームサーチを一度だけ行うのではなく，最初に選ぶフラグメントを変えながらフラグメントの個数と同じ回数だけビームサーチを繰り返す．繰り返し探索を行う理由として，最初に探索するフラグメントに不適切なものを選んでしまうと，ビームサーチのビーム幅がきわめて大きいものでない限り最適解が見つけられなくなってしまうことがある．このような場合への対処として，最初に選ぶフラグメントを変更して探索を繰り返すことが有効である．

2.8 エネルギー関数

本研究でのエネルギー関数に，CHARMM36[12] のファンデルワールス相互作用と，静電相互作用を用いた．

3. 評価実験

3.1 比較対象

本研究では探索アルゴリズムを提案したが，既にあるドッキングソフトウェアとの比較はエネルギー関数の違いがあるため，探索問題の定義や難易度そのものが異なってしまい，比較が困難である．そこで，同じエネルギー関数を用いて自分で実装した他の探索手法と以下に示す 3 つの実験によって比較を行った．実行環境を**表 1** に示す．

3.1.1 全解探索版との比較 (実験 1)

ビームサーチは，必ず最適な解を求められるようなアルゴリズムではない．そのため，ビームサーチによる計算の高速化によって精度にどの程度影響が出ているかを調べる必要がある．そこで，ビームサーチにおけるビーム幅を無限大にすることで，全ての解を探索するようにした手法と比較した．ただし計算量の関係から，フラグメントに分割した時にフラグメント数が 5 個以下となるようなデータのみで測定した．

3.1.2 フラグメント分割しない手法との比較 (実験 2)

提案手法ではフラグメントに分割しているが，そのような手法でどれだけ計算速度が向上したか，精度にどのような影響が出るのかを調べる必要がある．そこで，フラグメントに分割しないナイーブな方法と比較した．

分割しない手法として，以下のような手法を用いた．

- 回転可能な結合を $2\pi/5$ 刻みで回転させて配座を生成する．(多面体に基づいた回転における軸方向の回転角度と一致させるため，$2\pi/5$ という角度を用いた．)
- 自己衝突するような配座を除く．
- 自己衝突しない配座と，提案手法のフラグメントと同じ回転，同じ並進移動を用いてリガンド全体を移動させ，各ポーズについてエネルギーグリッドに基づくエネルギーを計算する．
- タンパク質の原子と衝突していないもののうち上位 N_{out} 個のポーズについて更に 2 体間のエネルギーを計算し，エネルギーの一番良いものを出力する．

計算量の関係から，本研究では，回転可能な結合が 5 本以下のデータのみで測定した．

3.1.3 幅優先探索ベースと Dijkstra 法ベースとの比較 (実験 3)

提案手法 (幅優先探索をベースとしたビームサーチ) と Dijkstra 法をベースとしたビームサーチでは漸近的評価ではどちらの方が速いか明らかではなかった．そこで，幅優先探索をベースとしたビームサーチと Dijkstra 法をベースとしたビームサーチでの計算時間を比較した．

3.2 データセット

データセットとして Astex Diverse Set[13] を用いた．このデータセットはタンパク質-化合物ドッキングの性能評価のために The Cambridge Crystallographic Data Centre (CCDC)[14] によって The Protein Data Bank (PDB)[15] からタンパク質-化合物複合体の構造を選別されたデータセットである．85 種類のタンパク質リガンド複合体で構成されている．

3.3 初期配座変更

評価実験に用いるリガンドは，共結晶構造のリガンドを引き離しただけのものであり，正解の構造そのものである．これをそのまま使ってしまうと，回転方向の数を少なくしたり並進移動の粒度を粗くしたほうが，正解の構造である最初の構造を見つけやすくなってしまう．そこで，あらかじめ分割されたフラグメントをそれぞれランダムに回転・並進移動することで，正しく評価を行えるようにした．

4. 結果と考察

4.1 提案手法の結果

提案手法で解を見つけられなかった例題は 6 個 (PDB ID: 1SQN, 1OF6, 1N46, 1S19, 1Z95, 1R55) あった．これらのフラグメント数は順に 2, 4, 6, 8, 12, 15 であった．このうち 1SQN のみフラグメント分割しない手法で解が見つかり，その他の 5 個は他の手法でも解を見つけることができなかった．解を見つけられた 79 個でのエネルギー平均

Vol.2015-MPS-103 No.62
Vol.2015-BIO-42 No.62
2015/6/25

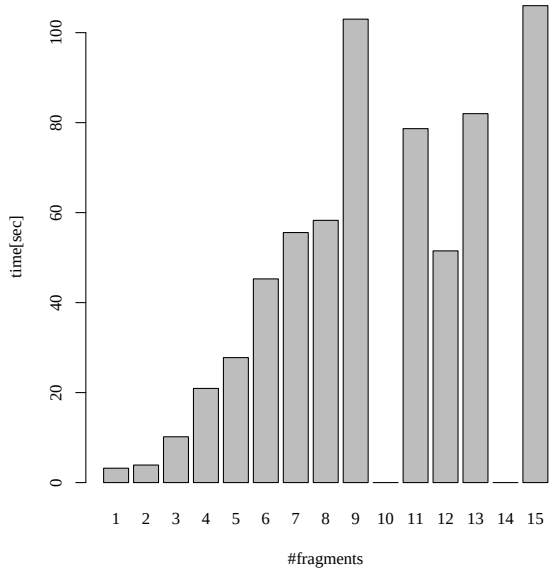

図 4 提案手法の実行時間

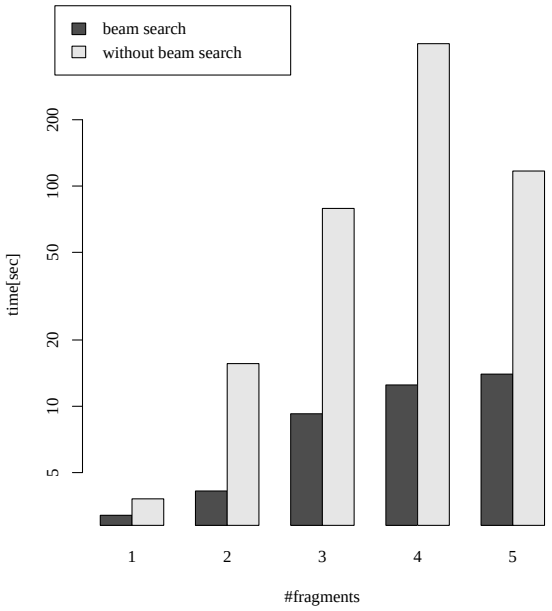

図 5 ビームサーチの有無によるの実行時間の比較

は −126.2 kcal/mol であった．

85 個の平均実行時間は 34.6 sec，解が見つかった例題についての平均実行時間は 35.1 sec であった．フラグメントごとの平均実行時間は**図 4** に示す (フラグメント数が 10, 14 の例題は存在しなかった)．

4.2 全解探索版との比較 (実験 1)

今回のデータでフラグメントの数が 5 個以下の例題は 49 個あった．これらのうち，21 個はメモリ不足のため全解探索で計算することができなかった．残りの 28 個のうち，どちらの手法でも解が見つからなかった例題は 2 つ (PDB ID: 1SQN, 1OF6) あった．実行時間の平均は，全解探索でメモリ不足の例題を除いた 28 個で，提案手法では 7.9 sec であり全解探索では 73.1 sec で，提案手法は約 9.3 倍高速であった．計算量は提案手法ではフラグメント数に対して線形だが，全解探索ではフラグメント数に対して指数関数になっているため，フラグメント数が多いリガンドについては，より差が出るはずである．

エネルギーの平均は，解が見つかった 26 個で提案手法が −102.1 kcal/mol, 全解探索で −99.4 kcal/mol であった．**図 5** にフラグメントごとの実行時間の比較，**図 6** にエネルギーの比較を示す．全解探索の方が，提案手法よりエネルギー平均が高くなっているが，これは最終的に全てのフラグメントのエネルギーの合計が低いものを N_{out} 個だけ選び，それぞれのポーズについて全てのタンパク質原子との 2 体間のエネルギーを計算しているため，エネルギーグリッドに基づくエネルギー値と最終的なエネルギー値で順位が入れ替わってしまうことがある．そのため，提案手法の方がエネルギー値が低い例が存在する．この結果から，

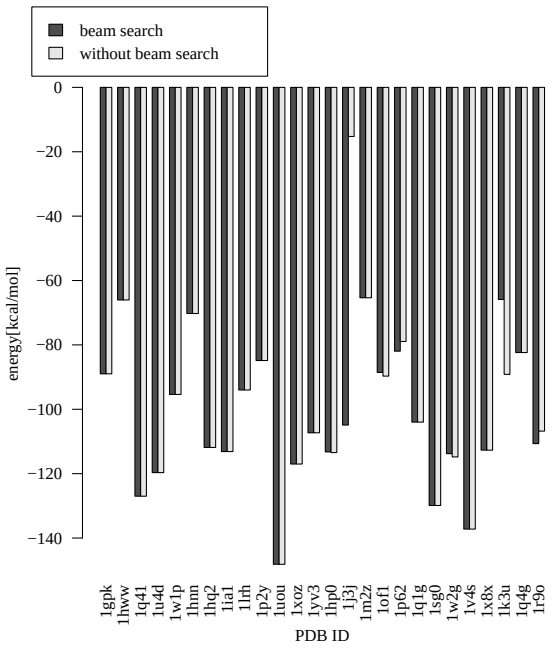

図 6 ビームサーチの有無によるのエネルギーの比較

提案手法が全解探索と同等以上の精度を出していると考えられる．

4.3 フラグメント分割しない手法との比較 (実験 2)

回転可能な結合の数が 5 本以下の例題は 29 個あった．これらのうち両方の手法で解が発見できなかった例題は 1 個 (PDB ID: 1OF6) あり，1 個 (PDB ID: 1SQN) は提案手法で解が見つけられず，2 個 (PDB ID: 1UOU, 1YV3) はフラグメント分割しない手法で解が見つけられなかった．実行時間の平均は，27 個で提案手法が 9.52 sec，非分

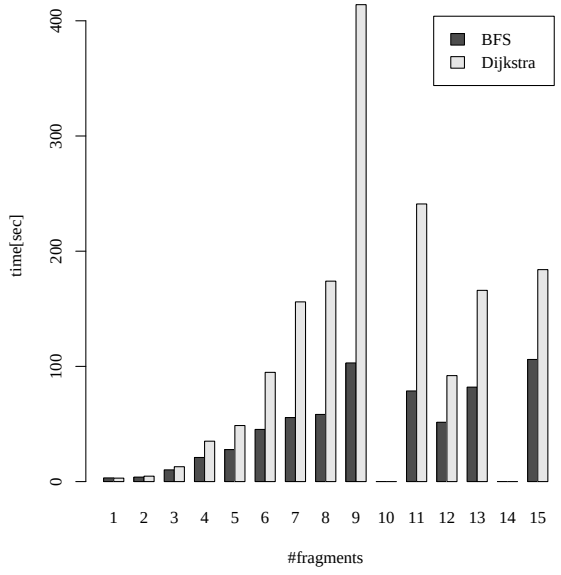

図 7 幅優先探索ベースと Dijkstra 法ベースの実行時間

割手法が 21141.4 sec で，提案手法は約 2220 倍高速であった．計算量は提案手法では回転可能な結合の数に対して線形だが，フラグメント分割しない手法では回転可能な結合の数に対して指数関数になっているため，回転可能な結合数が多い例題については，より差が出るはずである．エネルギーの平均は，どちらも解が発見されたのは 27 個で提案手法が −126.2 kcal/mol, 非分割手法が −120.8 kcal/mol であった．

4.4 幅優先探索ベースと Dijkstra 法ベースとの比較 (実験 3)

理論上，解は全て同じものが出るため実験結果のエネルギー値は全て同一のものだった．全体の平均値は幅優先探索が 34.6 sec，Dijkstra 法ベースが 87.0 sec であった．フラグメントごとの実行時間の比較を**図 7** に示す．フラグメント数が 1 のときのみ Dijkstra 法の方が速いが，平均 3.0 sec と 3.2 sec のためほとんど差がないと言ってよいだろう．その他の場合は全て幅優先探索ベースの手法の方が速かった．

幅優先探索ベースの計算時間が $\Theta(F \cdot w_{beam} \cdot C_{node})$ であり，Dijkstra 法ベースの計算時間は $\Theta(N \cdot (C_{node} + \log(N)))$ であることから，Dijkstra 法によって減らせるノード数の割合 $N/(F \cdot w_{beam})$ の影響よりも，上位 w_{beam} 個を選ぶときのコストの削減率 $C_{node}/(C_{node} + \log(N))$ による影響が大きかったと考えられる．よって提案手法を幅優先探索ベースのものとする．

5. 結論

本研究ではドッキングシミュレーションの高速化に向けて，フラグメント伸長の探索にビームサーチを用い，更にビームサーチを繰り返し適用することで精度を保ちつつ高速な探索を実装した．ビームサーチの実装方法として，幅優先探索ベースの手法と，Dijkstra 法ベースの手法の 2 種類が考えられる．前者の計算時間は $\Theta(F \cdot w_{beam} \cdot C_{node})$ であり，Dijkstra 法ベースの計算時間は $\Theta(N \cdot (C_{node} + \log(N)))$ であることから，どちらが速いかは一概に言うことができない．そこで，2 つの実装の違いの影響についても比較したところ，幅優先探索ベースの手法が高速であることが分かった．エネルギー関数の問題から，既に提案されている手法との比較は困難であったので，自分で実装した別の 2 つの手法との比較を行った．一つはビームサーチを用いた部分を全解探索に改変した手法との比較で，結果として精度をほとんど変えずに約 9 倍高速に計算できた．もう一つは，フラグメントを分割せず回転可能な結合の回転を網羅的に探索するナイーブな手法との比較で，2000 倍以上高速に計算できた．

6. 今後の課題

2 番目以降に探索するフラグメントの順序を本研究では原子の数を基準に決めていたが，フラグメントのエネルギーグリッド値を参考にして順序を決めたほうが良い解が見つけやすいかもしれない．また，今回の実験ではビーム幅を 5000 に固定していたが，どの程度まで精度を保ったままビーム幅を小さくできるか調べる必要がある．並進移動や回転のサンプリング間隔を変化させることによる結果への影響も調べるべきである．

更に，既存のソフトウェアと比較して実験的に性能差を示す必要がある．

謝辞 本研究の一部は，日本学術振興会 科研費 基盤研究 (A) (24240044) の支援によって行われた．

参考文献

[1] Kuntz I.D., Blaney J.M. Oatley S.J., Langridge R., Ferrin T.E.: “A geometric approach to macromolecule-ligand interactions”, *Journal of Molecular Biology* 161(2) pp.269–288, 1982.

[2] Sousa S.F., Fernandes P.A., Ramos M.J.: “Proteinligand docking: current status and future challenges”, *Proteins Struct Funct Genet* 65(1), pp.15–26, 2006.

[3] Zsoldos Z., Reid D., Simon A., Sadjad S.B., Johnson A.P.: “eHiTS: a new fast, exhaustive flexible ligand docking system”, *Journal of Molecular Graphics & Modelling* 26(1), pp.198–212, 2007.

[4] Friesner R.A., Banks J.L., Murphy R.B., Halgren T.A., Klicic J.J., Mainz D.T., Repasky M.P., Knoll E.H., Shelley M., Perry J.K., Shaw D.E., Francis P., Shenkin P.S.: “Glide: a new approach for rapid, accurate docking and scoring. 1. Method and assessment of docking accuracy”, *Journal of Medicinal Chemistry* 47(7), pp.1739–1749, 2004.

[5] Jones G., Willett P., Glen R.C. Leach A.R., Taylor R.: “Development and Validation of a Genetic Algorithm for

Flexible Docking", *Journal of Molecular Biology* 267(3), pp.727–748, 1997.

[6] Morris G.M., Huey R., Lindstrom W., Sanner M.F., Belew R.K., Goodsell D.S., Olson A.J.: "AutoDock4 and AutoDockTools4: Automated docking with selective receptor flexibility", *Journal of Computational Chemistry* 30(16), pp.2785–2791, 2009.

[7] Rarey M., Kramer B., Lengauer T. Klebe G.: "A fast flexible docking method using an incremental construction algorithm", *Journal of Molecular Biology* 261(3), pp.470–489, 1996.

[8] Plewczynski D., Lazniewski M., Augustyniak R., Ginalski K.: "Can we trust docking results? Evaluation of seven commonly used programs on PDBbind database", *Journal of Computational Chemistry* 32(4), pp.742–745, 2011.

[9] Steve H.: "Tri-linear Interpolation", *Graphics Gems IV*, pp.521–525, 1994.

[10] Dijkstra E.W.: "A note on two problems in connexion with graphs", *In Numerische Mathematik 1*, pp.269–271, 1959.

[11] Musser D.R.: "Introspective Sorting and Selection Algorithms", *Software Practice and Experience* 27(8), pp.983–993, 1997.

[12] Huang J., MacKerell A.D. Jr.: "CHARMM36 all-atom additive protein force field: validation based on comparison to NMR data", *Journal of Computational Chemistry* 34(25), pp.2135–2145, 2013.

[13] Hartshorn M.J., Verdonk M.L., Chessari G., Brewerton S.C., Mooij W.T., Mortenson P.N., Murray C.W.: "Diverse, high-quality test set for the validation of protein-ligand docking performance", *Journal of Medicinal Chemistry* 50(4), pp.726–741, 2007.

[14] Allen F.H., Bellard S., Brice M.D., Cartwright B.A., Doubleday A., Higgs H., Hummelink T., Hummelink-Peters B.G., Kennard O., Motherwell W.D.S., Rodgers J.R., Watson D.G.: "The Cambridge Crystallographic Data Centre: computer-based search, retrieval, analysis and display of information", *Acta Crystallographica Section B Structural Crystallography and Crystal Chemistry* 35(10), pp2331–2339, 1979.

[15] Berman H.M., Westbrook J., feng Z., Gilliland G., Bhat T.N., Weissig H., Shindyalov I.N., Boume P.E.: "The Protein Data Bank", *Nucleic Acids Research* 28(1), pp.235–242, 2000.