

共通な部分構造の 再利用アルゴリズムを用いた タンパク質リガンドドッキング手法の開発

2018/02/13

本研究の概要

2

- ▶ 化合物の**部分構造**に着目し，部分構造の結合スコアの計算結果を再利用することによって，1タンパク質－複数化合物の**高速なドッキングツールを開発**した

- ▶ 部分構造の計算結果の再利用効率を改善するため

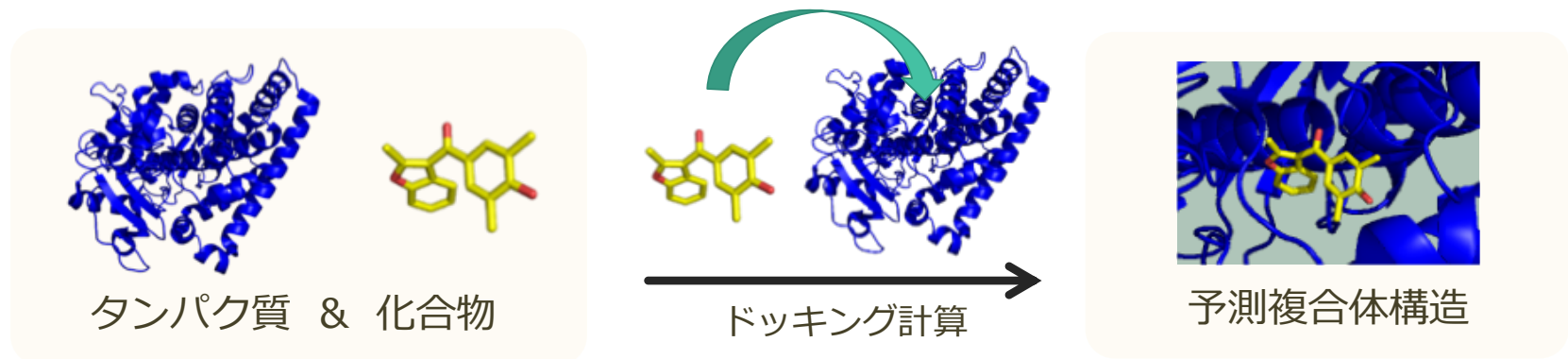
- ▶ 改善① – 化合物の評価順序の変更
- ▶ 改善② – メモリ戦略の最適化

という2つの手法を実装し，改善前より **1.77** 倍の高速化をした

- ▶ 比較対象としてフラグメントを用いず**原子グリッド**のみを用いるツールを開発した結果，原子グリッドのみのツールからの **2.34** 倍の高速化を確認できた

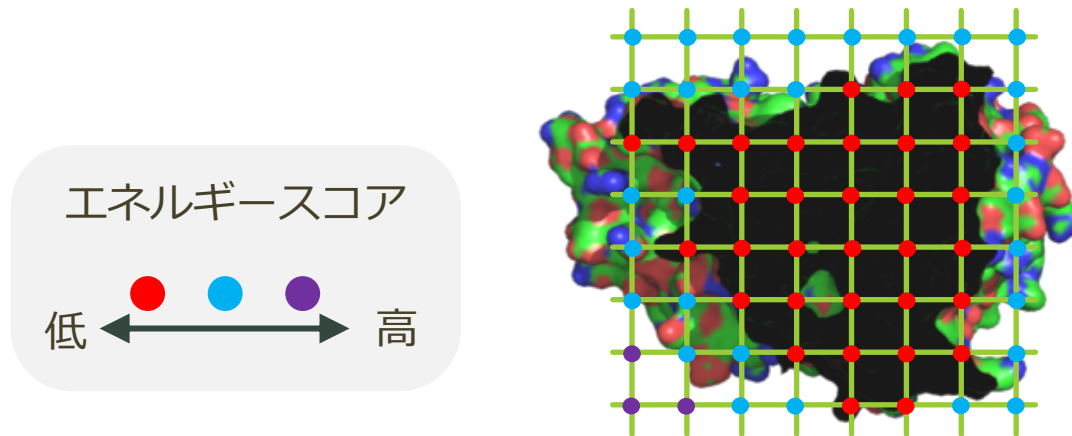
背景 – タンパク質リガンドドッキング

- ▶ 薬剤になり得る化合物は膨大な数存在し，全てに化学実験を行うのは **コスト**（時間・費用）**がかかる**
- ▶ 計算機で薬剤候補化合物を絞り込むために，**ドッキング**という手法がよく用いられる
 - ▶ ドッキング タンパク質と化合物の立体構造情報を用いて結合スコアを計算し，結合の起こりやすさ・複合体の構造を予測する手法
- ▶ 多数の化合物の評価のため，**ドッキング計算の高速化が求められている**



背景 – 原子グリッドを用いたドッキング

- ▶ ドッキングにおいて**原子**の計算結果を再利用する手法
 - ▶ 原子一つを格子点上に配置したときの結合スコアを事前計算し**原子グリッド**として保存し，化合物の評価速度を**高速化する**
 - ▶ Glide[1], AutoDock Vina[2] などのドッキングツールにおいて幅広く用いられている



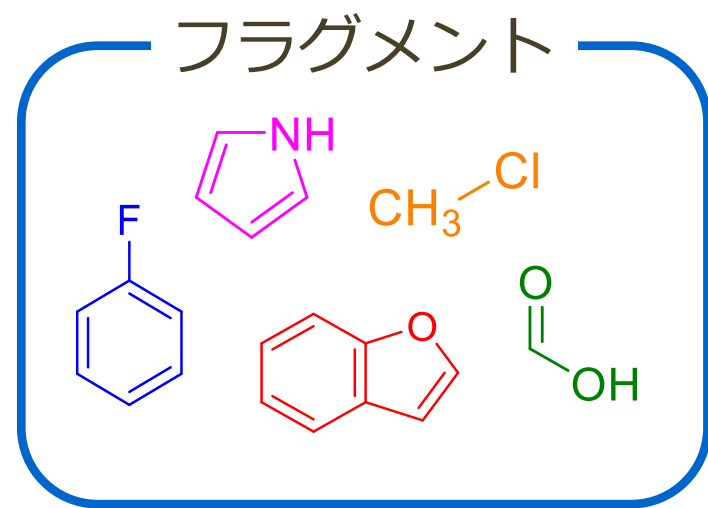
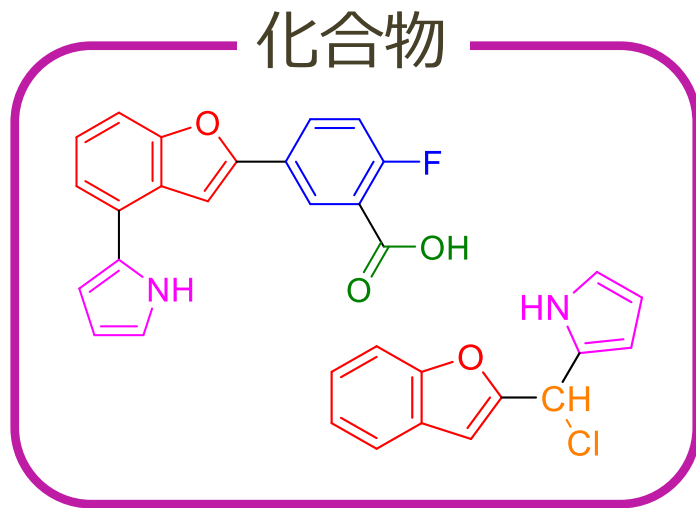
- ▶ しかし，ドッキング計算の速度はまだ十分に速いとは言えない

[1] Richard A. Friesner, *et al.*, *J. Med. Chem.*, **2004**, 47 (7), pp. 1739–1749

[2] Oleg Trott, *et al.*, *J. Comput. Chem.*, **2010**, 31 (2), pp. 455–461

背景 – フラグメントに基づいたドッキング

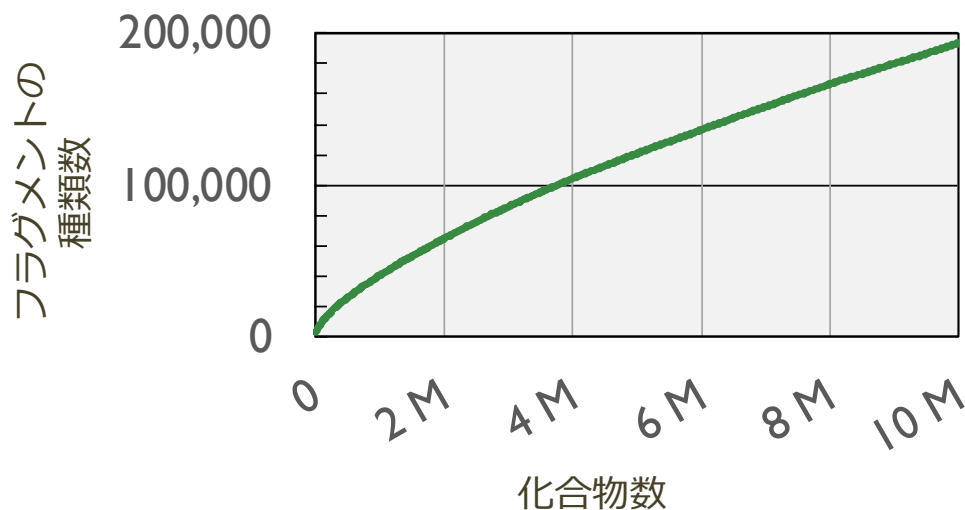
- ▶ ドッキング手法の中でも特に化合物の持つ部分構造（**フラグメント**）に注目する手法をフラグメントに基づいたドッキングと呼ぶ
- ▶ 本研究では特に、内部に回転可能な結合がない部分構造を**フラグメント**と定義することで、**原子グリッド**と同様に複数の化合物間での計算結果の再利用を行う



背景 – フラグメント

6

- ▶ 薬剤候補化合物は、**共通なフラグメント**を持つことが**多い**ことが知られている^[3]
 - ▶ 複数の化合物を評価する際に、**フラグメント**の計算結果を再利用することが**高速化に有用である**と考えられる

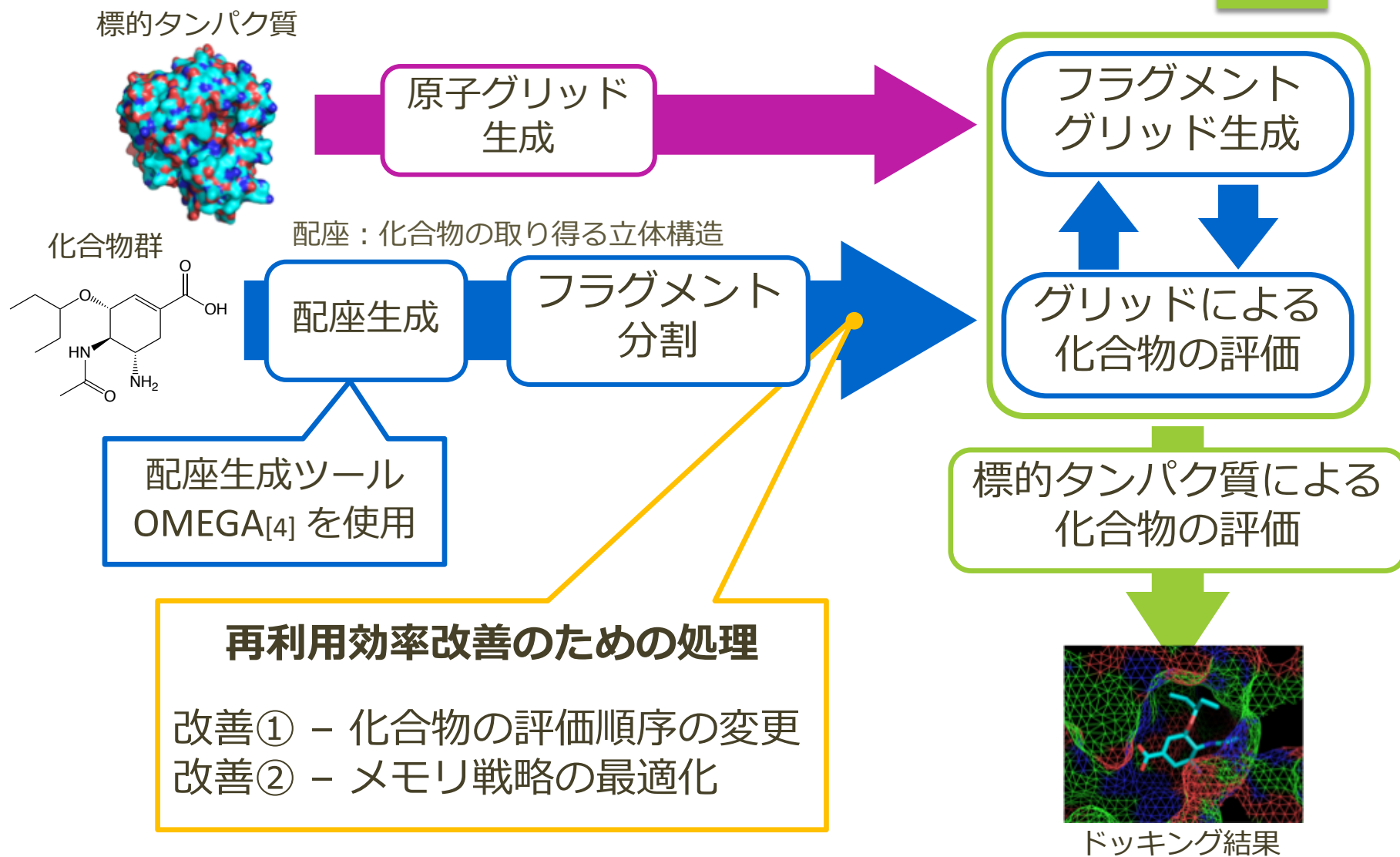


1,000 万化合物内に
フラグメントは
20 万種類しかない

ZINC Drugs Now Subset (10,639,555 entries)
に対してフラグメント分割を行った結果

本研究のドッキングツール全体の流れ

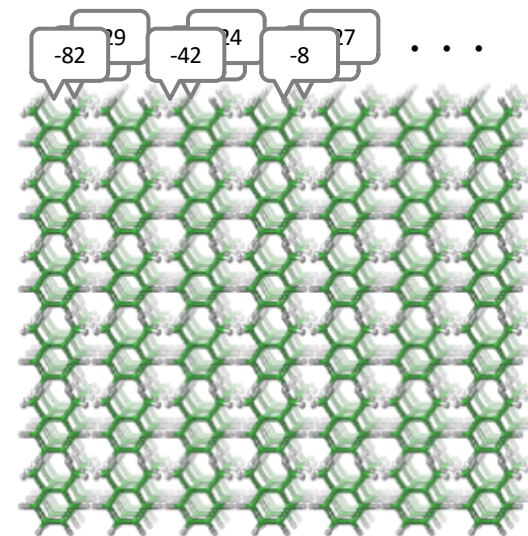
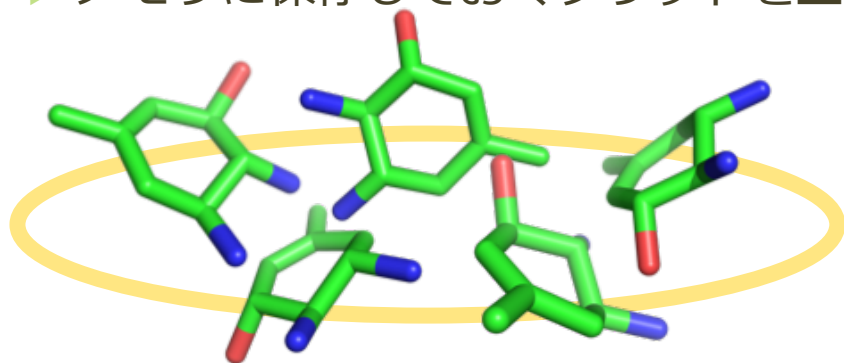
7



手法 – フラグメントグリッド

8

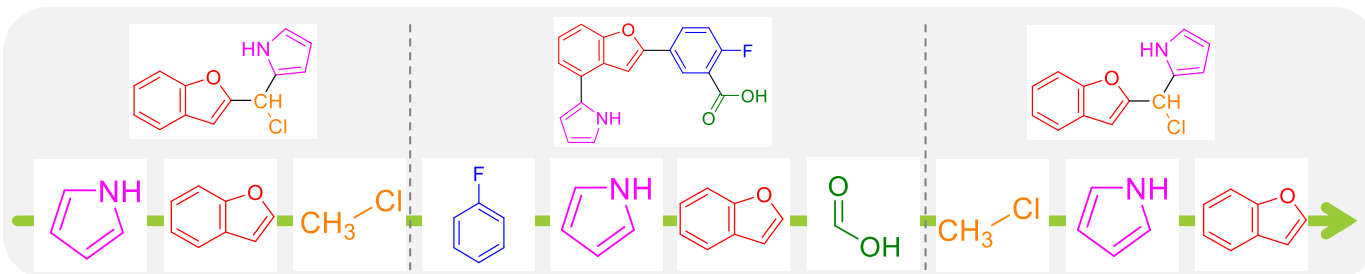
- ▶ **フラグメント**の計算結果を保存・再利用する
 - ▶ フラグメントを格子点上に配置したときの結合スコアを**フラグメントグリッド**として保存し、化合物の**評価速度を高める**
 - ▶ **回転**も踏まえて保持するので、**メモリに乗り切らない**
 - ▶ メモリに保存しておくグリッドを**上手く取捨選択する必要がある**



- ▶ **再利用効率改善**のため以下の2つの処理を行った
 - ▶ 改善① – 化合物の評価順序の変更
 - ▶ 改善② – フラグメントグリッドのメモリ戦略の最適化

改善① – 化合物の評価順序の変更

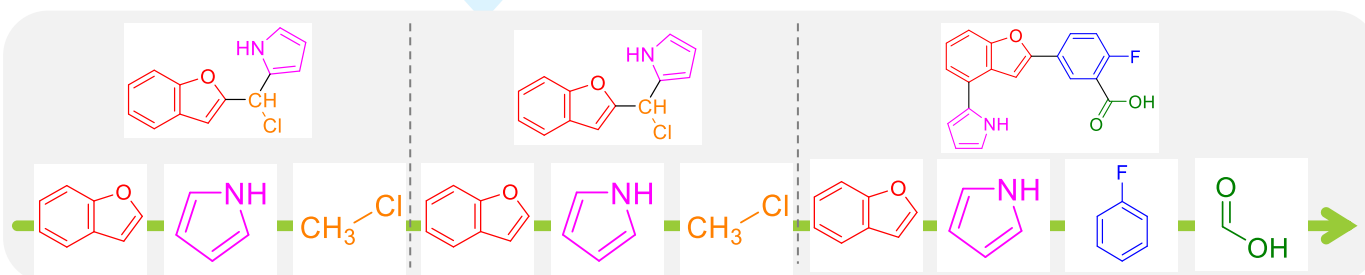
9



原子数 × (出現数 - 1)
で**再利用の重要度**を
定義した

再利用が起こりやすくなるように
化合物，フラグメントの並び替えを行う

フラグメント	原子数	出現数	重要度	rank
	9	3	18	1
	5	3	10	2
	2	2	2	3
	7	1	0	4
	3	1	0	5

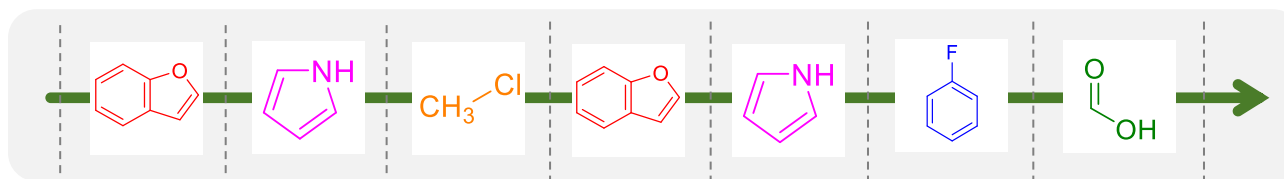


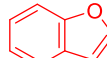
rank を用いて
**フラグメント，
化合物をソートする**


改善② – メモリ戦略の最適化

10

フラグメントグリッドのメモリ戦略 = **オフラインキャッシュ問題**



 のグリッドを
保持するとコスト削減

 のグリッドを
保持するとコスト削減

最小費用流問題 に帰着できる[5]

(\longrightarrow は cost : 0, capacity : ∞)

cost : -9, capacity : 1

cost : -5, capacity : 1

cost : その辺を経由することで削減できるコスト
capacity : その辺を経由できる上限回数

速度評価実験 – 比較対象・データセット

11

▶ 比較対象


- ▶ **比較手法① – 再利用効率改善を行わないフラグメントグリッド再利用**
 - ▶ 化合物の評価順序の変更なし
 - ▶ フラグメントグリッドのメモリ戦略に
オンラインキャッシュアルゴリズム (Least Recently Used (LRU)) を使用
- ▶ **比較手法② – フラグメントに着目せず原子グリッドのみ使用**

▶ データセット

- ▶ 化合物 : ZINC Drug Database (ZDD)[6] (2,924 entries)
 - ▶ 配座生成に失敗したものがあり, そのうち 2,886 件を用いた
 - ▶ その内 10, 100, 1000 件をランダムに選んでサブセットを作成
- ▶ タンパク質 : Influenza neuraminidase N1 (PDBID : 2HU4)

速度評価実験 – 実験結果

12

	フラグメント 利用	化合物の 並び替え オフライン キャッシュ	化合物件数			
			10	100	1,000	2,886
比較手法①	○	×	2.93 分	21.05 分	212.85 分	613.55 分
比較手法②	×		1.73 分	17.71 分	196.69 分	810.83 分
提案手法	○	○	2.93 分	19.49 分	142.14 分	346.60 分

- ▶ 化合物 2,886 件のケースにおいて,
比較手法①の **1.77** 倍, 比較手法②の **2.34** 倍の高速化が見られた
- ▶ 化合物数が多くなると共通なフラグメントが増加していくので,
これらの高速化率はさらに化合物数の大きなケースにおいて
さらに向上すると考えられる

- ▶ 化合物の共通部分構造の計算結果を再利用する高速なドッキングツールを開発した
化合物の評価順の変更やメモリ戦略の最適化により再利用効率を改善し、計算結果を変えずに改善前より **1.77** 倍高速化した
- ▶ 今後の課題
 - ▶ **より多数の化合物に対する実験**
 - ▶ より多数の化合物に対して本当に速度がさらに向上するか検証する必要がある
 - ▶ **精度の評価及び改善**
 - ▶ 本研究では精度の評価を行っていないがそこまで精度が良くないことが想定されるので、評価及び改善をしたい

補足 – エネルギースコア関数

14

- ▶ 今回, 結合スコアの計算には AutoDock Vina[2] のものを用いた
- ▶ 化合物とタンパク質の結合スコアは, 化合物の原子とタンパク質の原子のすべてのペアに対する以下の関数の値の総和で表される

$$\begin{aligned} \text{Energy}(a_1, a_2, d) = & (-0.035579) e^{-(2d)^2} \\ & + (-0.005156) e^{-\left(\frac{d-3}{2}\right)^2} \\ & + (0.840245) f_1(d) \\ & + (-0.035069) f_2(a_1, a_2, d) \\ & + (-0.587439) f_3(a_1, a_2, d) \end{aligned}$$

$$\begin{aligned} f_1(d) &= \begin{cases} d^2 & (d < 0) \\ 0 & (\text{otherwise}) \end{cases} \\ f_2(a_1, a_2, d) &= \begin{cases} 1 & (a_1 \text{ と } a_2 \text{ が共に疎水性} \wedge d \leq 0.5) \\ 1.5 - d & (a_1 \text{ と } a_2 \text{ が共に疎水性} \wedge 0.5 < d < 1.5) \\ 0 & (\text{otherwise}) \end{cases} \\ f_3(a_1, a_2, d) &= \begin{cases} 1 & (a_1 \text{ と } a_2 \text{ がそれぞれ水素結合の受容体と供与体} \wedge d \leq -0.7) \\ -1.428571 d & (a_1 \text{ と } a_2 \text{ がそれぞれ水素結合の受容体と供与体} \wedge -0.7 < d < 0) \\ 0 & (\text{otherwise}) \end{cases} \end{aligned}$$

補足 – フラグメント分割

15

- ▶ フラグメント分割は、最初すべての原子を独立な集合とした状態から以下のように集合の統合を行う
 - ▶ これは既存研究^[3,7]において提案された手法である
- ▶ Open Babel^[8] に実装されたメソッドを用いて結合が回転可能か判断する
- ▶ 回転不可能だった場合、結合の両端の原子は同一集合に含める
- ▶ 回転可能な結合においても、どちらか片方が 1 原子のみとなる場合は同一集合に含める
- ▶ 環構造の場合は、その環を構成するすべての原子を同一集合に含める

[7] 小峰 駿汰, *et al.*, 情報処理学会研究報告 バイオ情報学 (BIO), 2015-BIO-42(62), pp. 1-8

[8] Noel M O'Boyle, *et al.*, *J. Cheminform.*, **2011**, 3 (1):33

補足－データセットの詳細

16

化合物 件数	配座総数	フラグメント 種類数	フラグメント 総数	総原子数
10	579	35	3,559	19,821
	861	34	6,929	33,074
	924	42	7,817	40,548
100	7,521	186	74,004	339,887
	7,522	192	57,620	292,343
	9,326	196	80,221	401,948
1,000	79,846	1,011	653,834	3,304,839
	81,218	968	658,274	3,307,017
	82,583	1,007	674,511	3,395,115
2,886	241,034	2,028	1,973,311	9,915,444

補足 – 実験結果の詳細（比較手法①）

17

化合物 件数	メモリ戦略：オフライン		メモリ戦略：オンライン	
	並び替え：有	並び替え：無	並び替え：有	並び替え：無
10	2.93 分	2.94 分	2.93 分	2.93 分
100	19.49 分	19.22 分	20.60 分	21.05 分
1,000	142.14 分	166.33 分	168.72 分	212.85 分
2,886	346.60 分	469.96 分	404.56 分	613.55 分

補足 – 実験結果の詳細（比較手法①）

18

