目 次

図 目 次

表 目 次

第 1 章

序論

1.1　背景

computer-aided drug discovery について、G. Sliwoski, et al., 2014**ref** の論文の章立てに触れながら説明。SBDD, LBDD, ADMET の予測、という三種類の立場で計算手法が役立っているということを話せれば OK。ここに VS についての説明も入るだろう。

創薬にかかる時間やお金の具体数は記述しない予定

1.2　研究目的

SBDD では計算時間がかかるということに言及。docking 計算手法の高速化が一般的に行われている一方、フィルタリングを行うということもされている。

フィルタリングに関しては基本的に構造を見ていない、ligand-based な手法**ref**。glide HTVS**ref:HTVSって論文ある？**くらいしか構造を見ていない。glide HTVS に関しても、速度を高めるために強い仮定を置いているため、精度が十分とはいえない。したがって、より高速に、より精度よくフィルタリングを行うことが非常に大切となる。

1.3　本論文の構成

2 章では SBDD による創薬について詳しく説明、3 章で提案するフィルタリング手法の説明を行う。4 章で実験について述べ、5 章ではこの実験の結果についての考察を加える。最後に 6 章で結論および今後の展望について述べる。

第2章

SBVS(Structure-based virtual screening)による創薬

2.1　SBVSについて

- SBVS とは
  - 大量の化合物から薬物のタネとなる化合物（リード化合物）をコンピュータを用いて選別することを指す。
  - ligand-based の手法に比べて、多様な候補化合物が得られる（要出典）
- SBVS におけるフィルタリングの必要性、および問題点の説明
  - SBVS は比較的に時間を要する
  - したがって、フィルタリングをする必要がある。
  - しかし、多くのフィルタリング手法は化合物の構造だけを見てタンパク質の構造を見ていない。この手法は既に阻害剤として知られている化合物がそれなりに知られていないとどうすることもできない
  - 一方、粗く、高速なドッキングをすることでフィルタリングを行おうとする手法がある。glide の高速ドッキングモード (HTVS, high throughput virtual screening) がそれにあたる。

2.2　タンパク質-化合物ドッキングシミュレーション

glideref, GOLDref, Autodock Vinaref の 3 種類は最低でも説明する。特に glide に関しては自分の手法でも用いるので詳しく説明する。

2.3 化合物のフィルタリング

- Ligand-based なフィルタリング
  リピンスキー **ref** など、タンパク質の構造によらない手法について

- Structure-based なフィルタリング
  glide HTVS のようなドッキングに基づいた手法を説明する。

第3章

化合物の部分構造を利用したフィルタリング手法の開発

3.1　提案手法へのアイデア

3.1.1　ドッキング計算における探索自由度の削減

- 化合物の内部自由度を図で示す。イメージ図：図 3.1

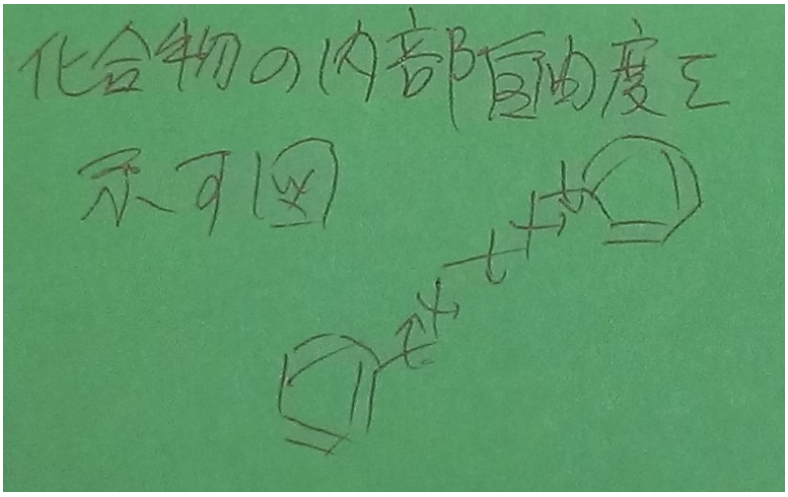

図 3.1　化合物の内部自由度

- 探索自由度の削減手法を示す。 図 3.2 的な。

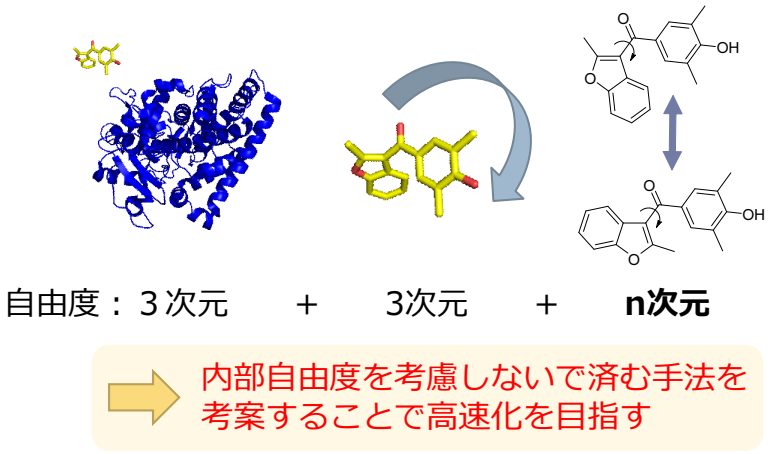

図 3.2　探索自由度について

3.1.2　化合物のスコア算出

スコア統合のイメージを示す (図 3.3 的な？これは雑すぎだが)。ここでは、計算量を O(n) に抑えることで高速化を目指す、ということを話せば OK。探索などはしない。

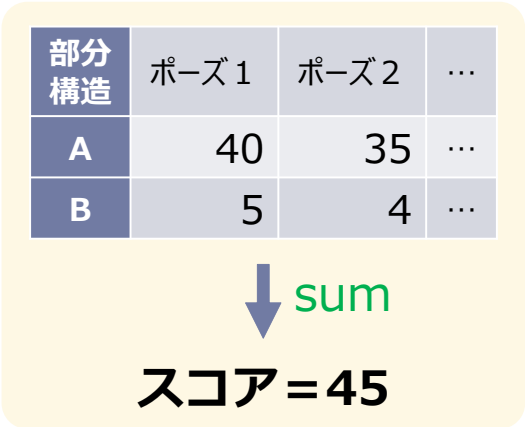

図 3.3　スコア統合イメージ

3.2　部分構造への分割

- 分割手法に関しては、小峰の SIGBIO の論文を参照する。 @cite
- ここの時点で小さい部分構造は除去する、ということを述べる？**@todo 要検討、score_max の評価をどうするかに依存。**

たぶんすることになると思う。その場合は分割の実例に除去の例を載せる。

- 分割について、実例を図で示す。

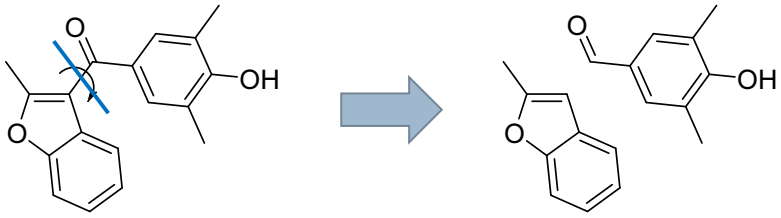

図 3.4　フラグメント分割例

- 分割することで発生する、共通部分の存在について述べる。**@todo これも図で示す。**
- 大量の化合物を分割した場合に図 3.5 のようになる、ということを示す。データセットがここで突然出てきてしまうが … 。

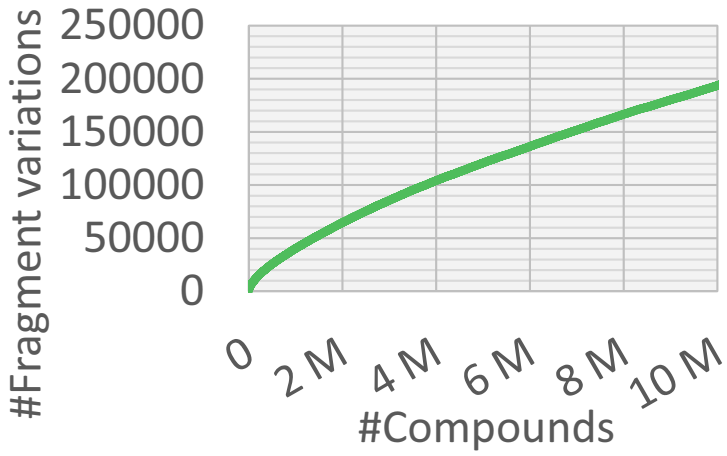

図 3.5　大量の化合物を分割した場合の例

3.3　部分構造単位のドッキング

glide 通常モード (SP)、および glide 高速モード (HTVS) を用いることを記述。

@todo 部分構造スコアはベストをとることを図示する (図 3.6 のような)。

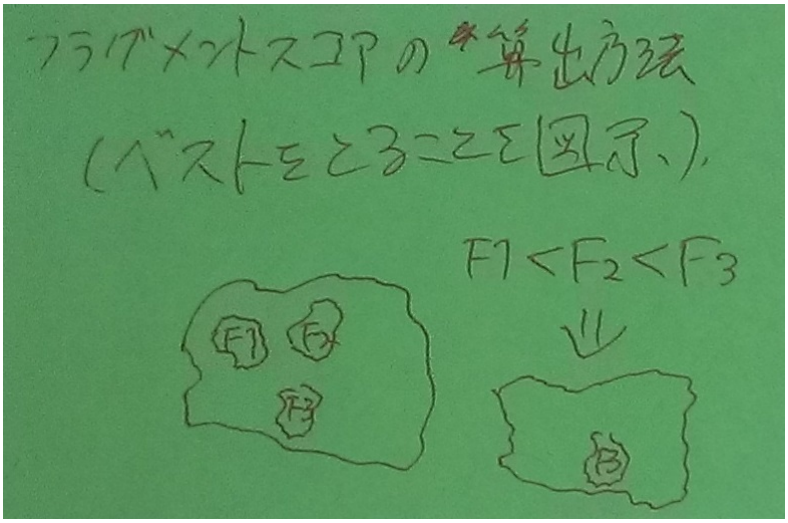

図 3.6　部分構造スコアの取得

3.4　部分構造スコアの統合

最初に、フラグメントごとのドッキング結果は構造を保持しないことを図で示し、したがってポーズを出力しないことを明記する。**@todo 図 3.7 のようなものを示す。**（なぜか図 3.7 が表示されないが、図 3.6 と同様に手書きなので無視。）

図 3.7　統合結果の構造

3.4.1　部分構造スコアの総和

score_sum について記述。numHvyAtom <= 2 のフラグメントを除去している。

3.4.2 部分構造スコアの最良値

score_max について記述。numHvyAtom <= 2 のフラグメントを除去していない。**@todo 除去しても結果はほとんど変わらないはず？**

3.4.3 総和法と最良値法の線形和

maxsumBS について記述。numHvyAtom <= 2 のフラグメントを除去している。

第4章

実験

ここで話すことは大きく二つ。
どこかで評価軸の説明をする

- それぞれの手法の単独性能について
- それぞれの手法を filtering 手法として用いた場合について

4.1　データセット

DUD-Eref を利用した、だけではなく、input と output を明確にする。

表 4.1　DUD-E diverse subset の詳細

ターゲット名	タンパク質名	正例		負例	
		化合物数	平均分割数	化合物数	平均分割数
akt1					
ampc					
cp3a4					
cxcr4					
gcr					
hivpr					
hivrt					
kif11					

4.2 計算環境

TSUBAME Thin ノード

4.3 実験結果

4.3.1 ドッキングにかかる計算時間

4.3.2 予測精度

ここでは単独手法としての精度を示す。ROC 曲線は Appendix として載せていることを記述

表 4.2 フィルタリング手法の予測精度

手法	フラグメントドッキング	ROC-AUC	Enrichment Factor			
			EF(1%)	EF(2%)	EF(5%)	EF(10%)
総和 (score_sum)	glide 通常モード					
	glide 高速モード					
最良値 (score_max)	glide 通常モード					
	glide 高速モード					
線形和 (maxsumBS)	glide 通常モード					
	glide 高速モード					
従来手法 (glide 高速モード)						

4.3.3 filtering と通常のドッキングとを合わせた場合の計算時間および精度

filtering 時に何%の化合物を残すか、などの議論と合わせて、精度と速度のバランスを示す。

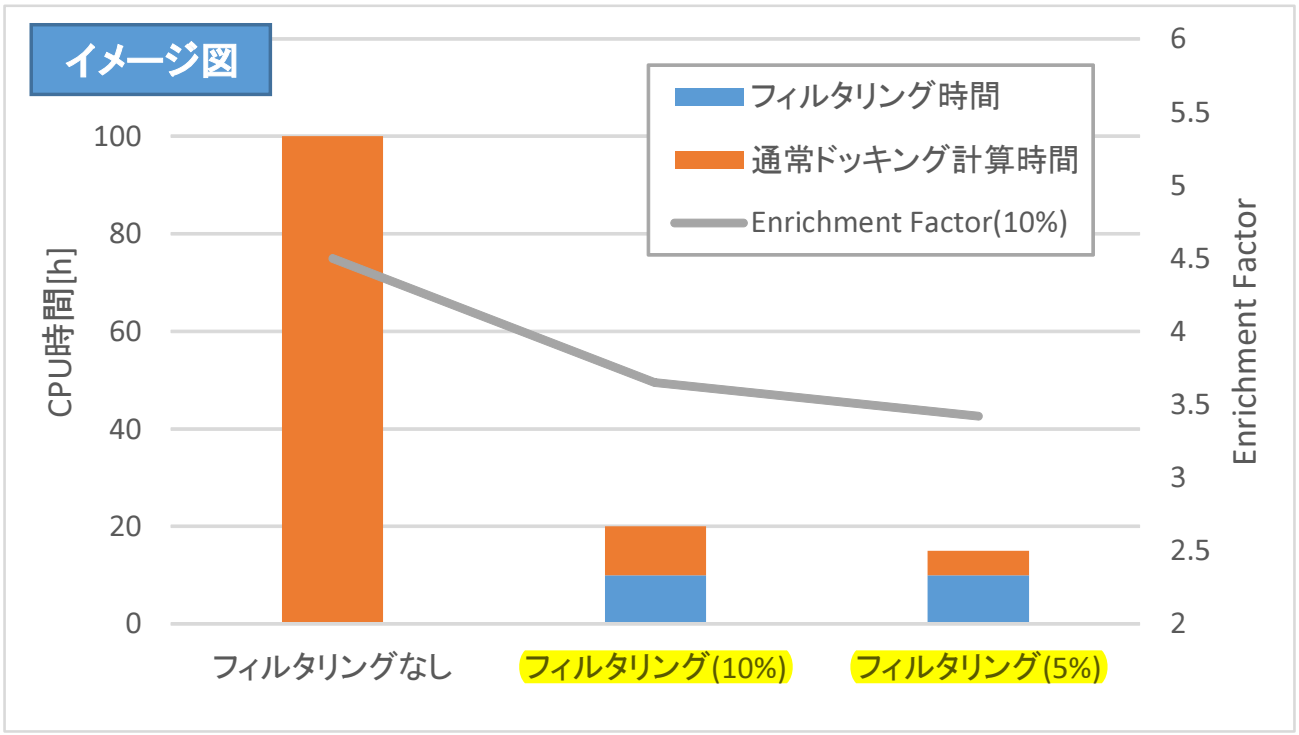

図 4.1 計算時間と精度のトレードオフ

第5章

考察

5.1　フラグメントスコア計算式の改善

5.1.1　各種エネルギースコアの寄与率の評価

hbond, evdw, ecoul など、それぞれ別々に用いた場合の評価がどうなっているのか。
ターゲットの正解化合物の傾向との関係が見つけられればうれしいが。

5.1.2　Glide GScore のフラグメント向け修正

フラグメントドッキングにおいては gscore をそのまま使うのではなく、少しパラメータを調整した方が良い。
公正な評価のために、通常のドッキングに関しても同様のパラメータチューニングをかける。

5.2　提案手法の得手・不得手

ターゲットの active ligand について、何らかの考察を加えることで提案手法が向くタイプのターゲット、向かないタイプのターゲットを評価する。

- 大きな DB に適用した場合の話をする（？　その場合は実験の段において「一致率」による評価を行う）

第6章

結論

6.1　本研究の結論

6.2　今後の課題

謝辞

ほげほげ。

参考文献

[1] ほげほげ

付録 A

ROC曲線

求められたROC曲線をひたすらかく。