隠れマルコフモデルと適応的閾値を用いた KEGG オーソログ予測法の開発とウェブツールの構築

統合ゲノミクス分野　　荒巻　拓哉

序論

2019 年の現在、次世代シーケンサーによるゲノムの配列決定は一部の研究者のものではなく、多くの生物学・医学研究者が利用する技術となっている。解読されるゲノム配列が増加するに従い、配列データの解釈に必要なバイオインフォマティクス技術もますます重要性を増している。遺伝子機能予測（機能アノテーション）も重要な技術の一つであり、現在までに様々な方法が開発されている。

遺伝子機能予測では、個々の遺伝子機能を表す ID が利用される。ここで用いられる機能 ID のセットは様々なものが知られており、Gene Ontology (GO) [1]、Clusters of Orthologous Groups (COG) [2]、eggNOG (evolutionary genealogy of genes: Non-supervised Orthologous Groups) [3]、KEGG Orthology (KO) [4] が代表的である。この中で KEGG Orthology は、生命科学に関するリソースを集めた Kyoto Encyclopedia of Genes and Genomes (KEGG) を構成するデータベースの 1 つであり、遺伝子間のパスウェイのデータベース KEGG PATHWAY と連携して、KO でのアノテーション情報を元にパスウェイの再構築が容易にできることが特徴である。実際に、配列決定されたゲノムの遺伝子を収録する KEGG GENES データベースの遺伝子の機能アノテーションも、KO の ID である K 番号を各遺伝子に割り当てることで実現されている。

遺伝子機能予測は、アミノ酸配列が類似していれば機能も似ているとの仮定のもと、データベースから相同配列を検索するという方法が一般的である。しかし、どの程度類似していれば機能を割り当てられるのかについての基準は必ずしも明確ではない。現在 KO による機能予測を自動で行うツールとして KAAS [5]、BlastKOALA、GhostKOALA [6]の 3 つが知られている。KAAS は 2007 年から提供されているツールで、KEGG GENES の各生物種の遺伝子とユーザーのクエリ遺伝子による双方向の BLAST 検索に基づいてアノテーションを行う。BlastKOALA、GhostKOALA は 2016 年から提供されているツールで、KEGG GENES データベースに対し、それぞれ BLASTP [7]、GHOSTX [8]による検索を行い、その結果をもとにアノテーションを行う。一般に GhostKOALA は BlastKOALA よりも高速であるため、大量の配列を扱うメタゲノム解析に向いているとされる。

以上 3 つのツールは、いずれも KEGG GENES データベースに対するペアワイズの相同性検索に基づいている。しかし KEGG GENES は 2019 年 1 月現在 2600 万以上のアミノ酸配列を収録する巨大なデータベースであり、全ての配列に対する検索は非常に時間がかかる。そこで、KAAS は検索対象をいくつかの代表生物種に絞ることで、BlastKOALA と GhostKOALA は科や属ごとの代表配列を選び出すことで参照データベースの縮小を試みている。しかしながら私は異なったアプローチによってより高速で精度の高いアノテーションを実現できないかと考え、隠れマルコフモデル (hidden Markov model, HMM) [9]に注目した。HMM は未知データのラベリングなどに使われる

確率モデルの一種で、しばしば配列検索にも応用される。HMM による配列検索では、データベース中の相同な配列をアラインメントし、各位置におけるアミノ酸残基や挿入・欠損の出現頻度がモデル化される。検索はこのモデルに対して行われるため、実際の配列は参照データベースに含まれない。このため、HMM による配列検索は BLAST や GHOSTX に比べてデータベースのサイズは小さくなり、一般に高速でかつリモートホモログも発見できるとされている。HMMER パッケージ [10]はアラインメントのモデル化やそれを用いた検索などのアルゴリズムを実装したプログラムのパッケージで、バイオインフォマティクスの分野では広く利用されている。

そこで本研究では、KO データベースの配列をプロファイル HMM 化した KOfam データベースの構築と、それを用いた高速かつ高精度の遺伝子機能予測法の開発を目指した。特に、ある配列がある KO に含まれるかどうかを判定するための類似度スコアの基準を KO ごとに設定することにより、予測精度の向上を図った。本研究におけるプロファイル HMM の作成や HMM による配列検索などは、全て HMMER パッケージ (version 3.1b2)を用いた。また用いた KEGG Orthology および GENES データベースは 2018 年 11 月 22 日時点の Release 88.2 である。

方法

KO 配列を元にしたプロファイル HMM の作成

プロファイル HMM の作成は、まず各 KO のアミノ酸配列が必要となるが、KEGG Orthology にはこうしたファイルは用意されていなかったため、各 KO の遺伝子名のリストと KEGG GENES のアミノ酸配列から FASTA 形式のファイルとして得た。この各 KO の配列群に対し閾値 1.0 で CD-HIT (version 4.7) [11]を実行し、冗長性を除去した配列セット（以下「非冗長配列セット」という）を得た。この操作の目的は、配列数を減らしデータを扱いやすくすることと、後述する閾値決定において、完全一致する配列が複数のグループに含まれることを避けることである。次に、非冗長配列セットを多重配列アラインメントした。アラインメントには MAFFT (version 7.310) [12] を用い、--amino --anysymbol オプションを利用した。ここまでの操作で得られた各 KO 配列の多重配列アラインメントから、HMMER パッケージの hmmbuild を用いプロファイル HMM を作成した。

KO ごとの適応的閾値の決定

HMMER の配列検索プログラム hmmsearch によって得られる配列とプロファイル HMM の類似度の指標には、配列全体とそこに含まれるドメインのそれぞれにおけるビットスコアと E-value がある。本研究では、ある配列に KO を割り当てるかどうかの判定基準としてビットスコアを用いている。E-value は入力の配列の本数によって変化するものであり、一定の数値を基準とすることが難しいためである。以下では、配列全体のビットスコアを全体スコア、ドメインのビットスコアをドメインスコアと言う。

ビットスコアによる判定においては、全ての KO で単一の閾値を用いることはできない。なぜなら、KEGG Orthology では各 KO の粒度が異なっており、非常に類似度の高い配列同士のみを含むものから、比較的幅広い配列を含むものまであるからだ。こ

のため本研究においては、KO ごとに個別の適応的閾値を採用した。この閾値は、各 KO の配列に交差検証に似た操作を適用することによって得られる。その詳細を以下に示す。

各 KO について、非冗長配列セットを 3 グループに分割した。なお、ここで非冗長配列セットの配列が 3 本未満の場合は三分割できないため、閾値の計算を行わず KOfam データベースにも含まないこととした。3 グループのうち 1 つを正解データとし、残りの配列から上記の方法でプロファイル HMM を作成した。このプロファイルと、正解データおよびその KO の配列を除く非冗長配列セット（以下「不正解データ」と言う）の配列を用いて hmmsearch を実行する。ここで得られた各配列の全体スコアについて、より多くの正解データを正解、より多くの不正解データを不正解と判定する閾値を設定するため、正解データの各配列のビットスコアを閾値とした場合の F 値を計算した。F 値は、再現率 (recall) と適合率 (precision) の調和平均として定義される（式(1)）。

$$F = \frac{2}{\frac{1}{recall} + \frac{1}{precision}} \tag{1}$$

ここで再現率と適合率は、その閾値での真陽性（閾値以上の正解配列）、偽陽性（閾値以上の不正解配列）、偽陰性（閾値未満の正解配列）の数をそれぞれ TP、FP、FN としたとき、それぞれ式(2)、(3)で表される。

$$recall = \frac{TP}{TP + FN} \tag{2}$$

$$precision = \frac{TP}{TP + FP} \tag{3}$$

以上のようにして F 値が最も高くなる閾値を計算した。残る 2 グループの非冗長配列セットについてもそれぞれを正解データとして F 値と閾値を計算し、3 回の計算で得られた閾値の平均をその KO の適応的閾値とした。

精度向上のための検討

利用するスコアの種類による予測精度の変化

上述の閾値決定法において、用いるビットスコアをドメインスコアに変更して再び閾値計算を行った。各 KO について 2 種類の閾値計算それぞれにおける平均の F 値を求め、散布図にプロットした。

配列フィルタリングによる予測精度の変化

多重配列アラインメントの質を上げるため、三分割後のアラインメントに trimAl (version 1.4.rev15) [13] を適用し、他の配列とのオーバーラップが少ない配列を除去した。具体的には、-resoverlap 75 -seqoverlap 0.75 オプションを用いた。このフィルタリングで得られた配列アラインメントからプロファイル HMM を作成し、同様に閾値を計算した。各 KO についてフィルタリングの有無それぞれでの平均の F 値を求め、散布図にプロットした。ただし、フィルタリングの結果 3 グループいずれでも配列が除去されなかった KO についてはここでは解析しなかった。

遺伝子機能予測法の性能評価

本予測法の未知の生物種に対する予測性能の評価のため、既にアノテーションされている KEGG GENES の遺伝子の機能を本予測法で予測した。クエリ遺伝子には、KEGG GENES データベースからランダムに選択した原核生物、真核生物各 20、計 40 の生物（表 1）の全遺伝子を用いた。この 40 の生物が属する属の配列を除いた残りの配列で KOfam データベースを構築し、これを用いてクエリ遺伝子の機能を予測し、計算に要した CPU 時間を測定した。KAAS、BlastKOALA、GhostKOALA でも同じクエリを用い、それぞれのツールのウェブサイト上でデフォルトとなっているデータベースから、同様に配列を除いて機能予測を行い CPU 時間と予測精度を比較した。また、各ツールの原核生物向けのデータベースを使い同様の方法で表 1 の原核生物の配列の機能を予測した。以下、前者を全体データベース、後者を原核生物データベースと言う。KOfam に関しては、真核生物の配列しか含まない KO を除外することで原核生物データベースとした。

予測結果の評価にあたっては、再現率と適合率を式(4)、(5)の通り定義した上で F 値を式(1)で計算して用いた。ただし、KEGG GENES でアノテーションされている KO（以下「正解の KO」と言う）と予測した KO が同じ KO だった数を match、正解と予測の KO が異なる数を unmatch、正解の KO が存在するが予測の KO がない数を missing、正解の KO がなく予測の KO が存在する数を excess とした。

$$recall = \frac{match}{match + unmatch + missing} \tag{4}$$

$$precision = \frac{match}{match + unmatch + excess} \tag{5}$$

表 1　テストセットに利用した生物種

真核生物	*Ailuropoda melanoleuca*, *Apis mellifera*, *Aspergillus fumigatus*, *Aspergillus nidulans*, *Aspergillus oryzae*, *Bos taurus*, *Brugia malayi*, *Candida glabrata*, *Drosophila melanogaster*, *Leishmania infantum*, *Magnaporthe oryzae*, *Malassezia globosa*, *Mus musculus*, *Nematostella vectensis*, *Ornithorhynchus anatinus*, *Plasmodium knowlesi*, *Strongylocentrotus purpuratus*, *Thalassiosira pseudonana*, *Toxoplasma gondii*, *Trichomonas vaginalis*
原核生物	*Alicycliphilus denitrificans* BC, *Bacillus licheniformis* ATCC 14580, *Bordetella avium*, *Borrelia afzelii*, *Borrelia duttonii*, *Burkholderia* sp. CCGE1002, *Comamonas thiooxydans*, *Deinococcus radiodurans*, *Geobacter daltonii* FRC-32, *Halorhodospira halophila*, *Leuconostoc mesenteroides* subsp. *mesenteroides* ATCC 8293, *Methanoregula boonei*, *Pseudonocardia dioxanivorans*, *Pyrococcus horikoshii*, *Rhodobacter sphaeroides* KD131, *Saccharophagus degradans,* *Synechococcus* sp. CC9902, *Thermotoga* sp. RQ2, *Teredinibacter turnerae*, *Tropheryma whipplei* Twist

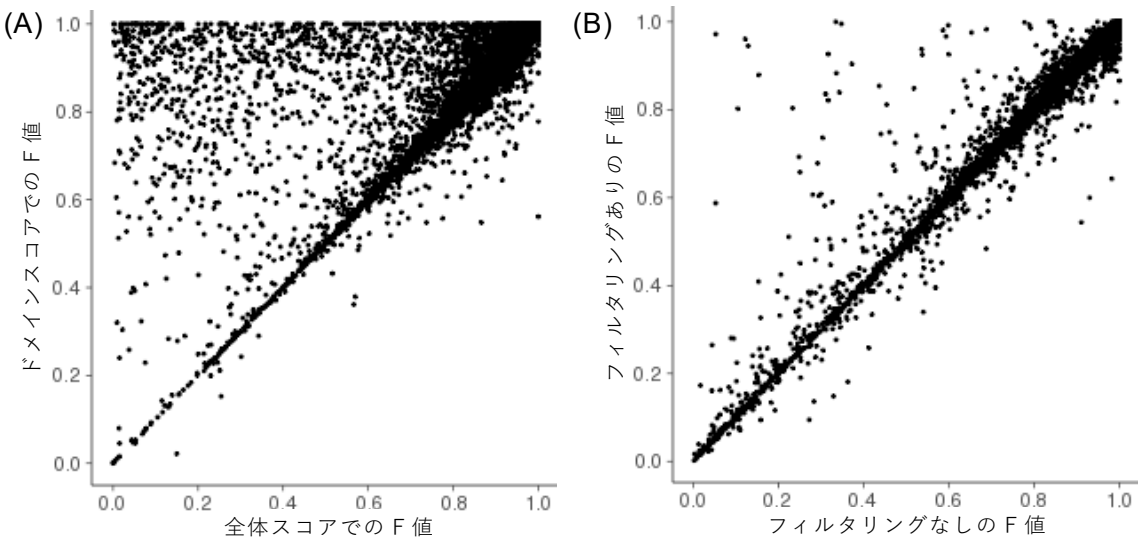


図 1　(A) 全体スコアとドメインスコアによる閾値計算時の F 値の変化。(B) フィルタリングの有無による F 値の変化。いずれの図も各点は 1 つの KO を表す。

結果・考察

KOfam データベースの構築

22403 個の KO のうち、アミノ酸配列を持つ（RNA 遺伝子等でない）KO は 22070 個で、閾値計算に必要な 3 本以上の非冗長配列を持ちプロファイル HMM を作成できた KO は 20654 個 (93.6%) だった。またそのうち原核生物の配列を含むものは 9414 個だった。

精度向上のための検討

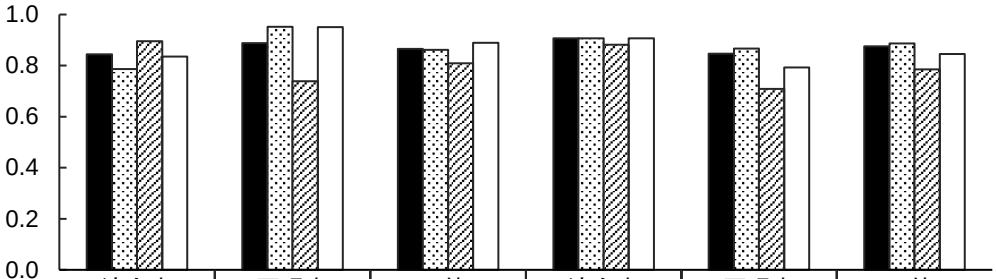
全体スコアとドメインスコアの利用、および配列フィルタリングの有無による F 値の変化は図 1 のとおりである。ドメインスコアの利用によって F 値が上昇したものは 6522 個、下落したものは 5245 個あり、F 値が 0.1 以上変化したものに限るとそれぞれ 1273 個と 65 個だった。以上より、ドメインスコアの利用が予測精度の向上につながる場合とそうでない場合があるものと考え、いずれのビットスコアを利用するかについては両方のビットスコアによる閾値を計算し、得られた F 値によって KO ごとに個別に決定するものとした。

次に、配列フィルタリングによって F 値が上昇したものは 4073 個、下落したものは 3631 個あり、そのうち F 値が 0.1 以上変化したものはそれぞれ 116 個と 39 個だった。以上より、配列フィルタリングに関しても予測精度の向上につながる場合とつながらない場合があると考え、フィルタリングを行う場合と行わない場合の両方で閾値計算を行い、F 値の高い方を採用することとした。

性能評価

KOfam の予測結果、計算時間の他ツールとの比較結果を図 2、表 2 に示した。

F 値は KAAS が少し低いものの、全体的に大きな差は見られなかった。KAAS の再現率が低いのは、BlastKOALA や GhostKOALA のデータベースは KO を網羅するよう代表配列を選んでいるのに対し、KAAS は代表生物種が KO を網羅していないためと



	全体データベース			原核生物データベース		
	適合率	再現率	F値	適合率	再現率	F値
■KOfam	0.8442	0.8881	0.8656	0.9064	0.8462	0.8752
▨GhostKOALA	0.7871	0.9522	0.8618	0.9068	0.8668	0.8864
▨KAAS	0.8954	0.7388	0.8096	0.8813	0.7086	0.7856
□BlastKOALA	0.8348	0.9502	0.8888	0.9064	0.7925	0.8456

図 2　各アノテーションツールによるアノテーションの適合率、再現度、F 値。

表 2　一生物種の全遺伝子アノテーションにかかった平均 CPU 時間

	全体データベース				原核生物データベース	
	全クエリ	速度比	原核生物クエリ	速度比	原核生物クエリ	速度比
KOfam	2:26:18	68.99	0:24:42	87.17	0:11:59	82.80
GhostKOALA	2:34:50	65.19	0:31:56	67.43	0:21:02	47.15
KAAS	5:03:32	33.25	1:24:13	25.57	0:23:11	42.80
BlastKOALA	168:12:41	1.00	35:53:27	1.00	16:31:53	1.00

考えられる。実際、正解となる KO 16184 個のうち 1885 個が KAAS のデータベースに含まれていなかった（KOfam は 213 個、BlastKOALA・GhostKOALA は 117 個）。また、理由は明らかでないものの、GhostKOALA の適合率が特にクエリが真核生物の場合に低かった。

計算時間は KOfam と GhostKOALA が同じ程度で速く、KAAS はその約 2 倍の時間がかかり、BlastKOALA が圧倒的に遅かった。原核生物データベースの場合、KOfam はデータベースサイズに比例して計算時間も約半分になったのに対し、GhostKOALA の計算時間は 3 分の 2 程度の短縮にとどまった。

他ツールと比較した利点

本研究で開発した遺伝子機能予測法は、生物学者に広く利用されている KAAS の性能を予測精度、計算時間の両面で上回った。また、予測精度において BlastKOALA と GhostKOALA と同等の結果を得た。KEGG GENES のアノテーションは BlastKOALA や GhostKOALA と類似の KOALA アルゴリズムに基づいてマニュアルで決定されていることを考慮すると、隠れマルコフモデルと閾値の学習に基づき根本的に原理が異なる本予測法が KOALA グループと同等の予測精度を示したことは特筆に値する。一方、計算時間の面においては BlastKOALA、GhostKOALA と比べてそれぞれ 68 倍以上、1.05～1.7 倍の高速化に成功した。さらに、それ以外の面において KOfam による機能予測法が優れている点がいくつかある。一つは、各 KO の閾値計算時の F 値から、予測された機能の信頼性がわかる点である。従来の手法でも予測精度の高い KO とそうでない KO があるが、それらを利用者が区別することは難しかった。KOfam による予

```
# gene name  KO      thrshld  score   E-value KO definition
#----------- ------ ------- ------ --------- --------------------
* seq1       K02313  118.63  477.0  5.8e-144 chromosomal replication initiator protein
  seq1       K10763  163.53  111.5   4.6e-33 DnaA-homolog protein
  seq1       K02315  147.30   88.4   7.1e-26 DNA replication protein DnaC
  seq1       K11144  159.07   57.9   1.3e-16 primosomal protein DnaI
  seq1       K14575  824.10   40.6   1.3e-11 AAA family ATPase
  seq1       K13338  524.23   38.5   5.6e-11 peroxin-1
  seq1       K13525  792.27   36.1   2.5e-10 transitional endoplasmic reticulum ATPase
* seq2       K02338   49.43  291.0   1.7e-87 DNA polymerase III subunit beta [EC:2.7.7.7]
  seq2       K11621  136.47   13.5    0.0028 lia operon protein LiaG
* seq3       K14761   58.63   62.4   2.8e-18 ribosome-associated protein
* seq4       K03629   91.07  308.5     6e-93 DNA replication and repair protein RecF
  seq4       K07459  119.60   46.2   4.4e-13 putative ATP-dependent endonuclease of the OLD
```

図 3　アノテーションパイプラインの出力例

測では、予測の信頼性も考慮した結果の解釈が可能である。また、一部の KO のみについて解析を行いたい場合に、KOfam であればその KO だけに絞って予測を行うこともできるため、計算時間を短縮できる利点もある。KAAS ではアルゴリズムの性質上、全ゲノム配列を使用することが前提となる。

遺伝子機能予測パイプライン

KOfam データベースを用いてアミノ酸配列の機能を予測するパイプラインを作成した。このパイプラインは大きく分けて 2 つのステップから成る。前半のステップでは HMMER パッケージの hmmsearch コマンドで、入力として受け取った配列と KOfam データベースによる配列検索を行い、両者の類似度を示すビットスコアを得る。このステップでは GNU parallel [14]を用い複数のプロファイル HMM に対して並列処理を行うことで、計算にかかる実時間の短縮を図っている。後半のステップでは、得られたビットスコアを適応的閾値と比較し、ビットスコアが閾値以上であればその配列はその KO であると予測する。

最終的な結果の出力形式は、タブ区切り形式と詳細形式の 2 つから選択可能である。前者は配列名と予測した KO の K 番号をタブ区切りで表示するもので、そのまま KEGG Mapper [15]に入力として渡し、パスウェイの再構築などが可能である。後者は配列名と K 番号、ビットスコア、E-value、KO の説明といった、より詳細な情報を表示するもので、閾値を下回った KO も確認することができる（図 3)。

このパイプラインは Linux 上で実行可能なプログラムとして実装されているほか、ウェブツールとしてウェブブラウザからも利用可能である（図 4)。ウェブツールではクエリ配列をアップロードするとサーバー上で機能予測が行

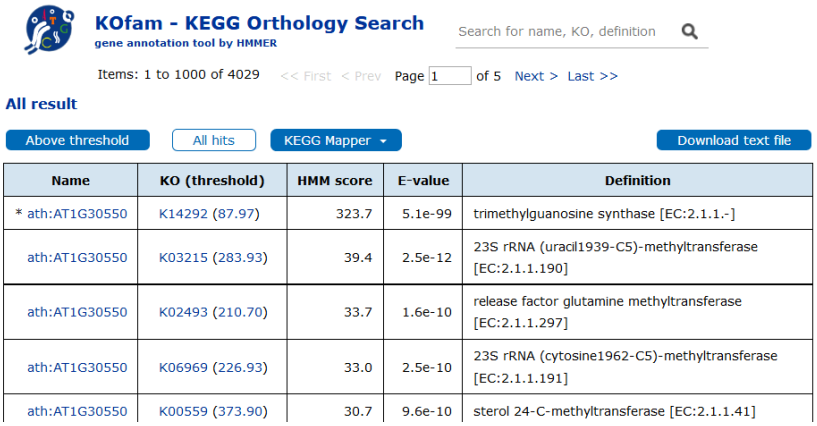


図 4　ウェブツールの結果表示画面

われ、図 3 と同様の情報を得られる他、ハイパーリンクをたどることで KO の詳細情報や配列とプロファイル HMM のアラインメントの情報を簡単に確認することができる。また、KEGG Mapper によるパスウェイ再構築も 1 クリックで行える。

結論と今後の課題

予測精度、計算時間のいずれの面においても既存のツールと対等以上の性能を持つ遺伝子機能予測法を開発することができた。また、GenomeNet (https://www.genome.jp/) でウェブツールを公開予定である。今後は本予測法が正しく予測できなかった、あるいは F 値の低かった KO について、その配列上の特徴（ドメイン構成など）や進化距離の情報に関する傾向の有無を解析し、プロファイル HMM の作成方法やスコアの計算方法を改善できないかを検討する余地があると考えている。また、メタゲノム等から得られた断片配列の予測精度に関しても評価することにより、利用価値を高めることも検討できるであろう。

謝辞

本研究の遂行にあたり、多くの方にご指導、ご支援を頂き感謝申し上げます。緒方博之教授、Romain Blanc-Mathieu 助教、遠藤寿助教には大変多くのご指導を頂きました。京都大学化学研究所 金久實特任教授、情報・システム研究機構 データサイエンス共同利用基盤施設 ライフサイエンス統合データベースセンター 五斗進教授にも重要な助言を頂きました。BlastKOALA のプログラムは金久ラボラトリーズ 森嶋佳苗氏に、KAAS と GhostKOALA のプログラムは日本ヒューレット・パッカード株式会社 上原英也氏にご提供いただきました。同社の大久保宏一氏にはウェブツールの実装にあたり多大なご協力を頂きました。

本研究成果は京都大学化学研究所スーパーコンピュータシステムを利用して得られたものです。

参考文献

[1] Ashburner, M., Ball, C. A., Blake, J. A., *et al.*, *Nature Genetics*, **25**, 25-29 (2000).
[2] Tatusov, R. L., Fedorova, N. D., Jackson, J. D., *et al.*, *BMC Bioinformatics*, **4**(41), (2003).
[3] Huerta-Cepas, J., Szklarczyk, D., Forslund, K., *et al.*, *Nucleic Acids Research*, **44**, D286-D293, (2016).
[4] Kanehisa, M., Sato, Y., Kawashima, M., Furumichi, M., Tanabe, M., *Nucleic Acids Research*, **44**, D457-D462, (2016).
[5] Moriya, Y., Itoh, M., Okuda, S., *et al.*, *Nucleic Acids Research*, **35**, W182-W185, (2007).
[6] Kanehisa, M., Sato, Y., Morishima, K., *Journal of Molecular Biology*, **428**(4), 726-731, (2016).
[7] Altschul, S. F., Madden, T. L., Schäffer, A. A., *et al.*, *Nucleic Acids Research*, **25**(17), 3389-3402, (1997).
[8] Suzuki, S., Kakuta, M., Ishida, T., Akiyama, Y. *PLOS ONE*, **9**(8), e103833, (2014).
[9] Eddy, S. R., *Nature Biotechnology*, **22**(10), 1315-1316, (2004).
[10] Eddy, S. R., *PLoS Computational Biology*, **7**(10), e1002195, (2011).
[11] Fu, L., Niu, B., Zhu, Z., *et al.*, *Bioinformatics*, **28**(23), 3150-3152, (2012).
[12] Katoh, K., Standley, D. M., *Molecular Biology and Evolution*, **30**(4), 772-780, (2013).
[13] Capella-Gutierrez, S., Silla-Martinez, J. M., Gabaldon, T., *Bioinformatics*, **25**(15), 1972-1973, (2009).
[14] Tange, O. (2018).
[15] Kanehisa, M., Goto, S., Sato, Y., *et al.*, *Nucleic Acids Research*, **40**, D109-D114, (2012).