BioRubyの系統樹データ構造の実装

大阪大学微生物病研究所附属
遺伝情報実験センター
ゲノム情報解析分野

後藤　直久

2007年3月2日

序論（2005年）

データベース: 719件以上

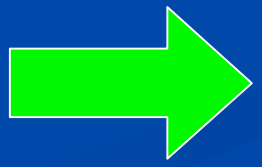
例: GenBank, EMBL, DDBJ, PDB, KEGG, …

Galperin, M.Y. (2005) The Molecular Biology Database Collection: 2005 update. *Nucleic Acids Research*, 33: D5-D24.

解析ソフトウェア: 129～448種類以上

例: BLAST, FASTA, CLUSTAL W, …

http://bioinformatics.org/software/
http://sourceforge.net/ のBioinformaticsカテゴリ

組み合わせ  新たな生物学的知見

序論

データベース: 968件以上

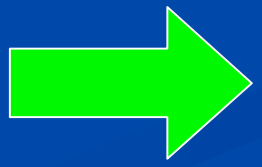
例: GenBank, EMBL, DDBJ, PDB, KEGG, …

Galperin, M.Y. (2007) The Molecular Biology Database Collection: 2007 update. *Nucleic Acids Research*, 35: D3-D4.

解析ソフトウェア: 133～1063種類以上

例: BLAST, FASTA, CLUSTAL W, …

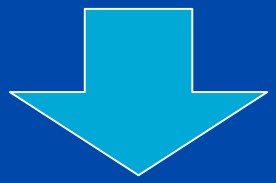
http://bioinformatics.org/software/
http://sourceforge.net/ のBioinformaticsカテゴリ

組み合わせ  新たな生物学的知見

データベース: 968件以上

解析ソフトウェア: 133～1063種類以上

- データ形式（フォーマット）はそれぞれ別々で、よく使われるフォーマットはいくつか存在するが、基本的には統一されていない
- データを読み込み解釈する機能（パーサ）は、あるフォーマットについて一回プログラミングしたら、流用可能
- データの読み書き以外にも、バイオインフォマティクスに必要な定型処理はたくさんある



統合的に扱えるライブラリ（ソフトウェア部品集）や
ソフトウェア環境の整備が必要

プログラム言語のライブラリとして実装するメリット

- 大量データ処理
  - ゲノム全体など数千~数万個の遺伝子に対する処理
- 複数処理の組み合わせ
  - 例: BLASTを実行しヒットした遺伝子をCLUSTAL W でマルチプルアライメントする
- 条件分岐
  - 例: BLASTのe-valueが0.1以下のときは処理Aを、それより大きいときは処理Bを実行

BioRuby

バイオインフォマティクスにおいて

頻繁に使用する機能・あったら便利な機能

- 塩基・アミノ酸配列の処理・解析
- データベースのデータ処理
- 解析ソフトウェアの結果処理
- ファイル入出力・ネットワークとの通信
- …

統一されたインターフェース・使用法

個別に深く理解する必要なく使える

Rubyで実装したライブラリ
（ソフトウェア部品集）

Rubyとは？

- オブジェクト指向スクリプト言語
- http://www.ruby-lang.org/
- 日本で開発され、海外にも普及したプログラム言語
  - 作者: まつもとゆきひろ氏
- Perlとの類似点
  - テキスト（文字列）処理が得意
  - スクリプト言語（コンパイル不要）
- Javaとの類似点
  - オブジェクト指向

他言語による先行プロジェクト

- Perl　　BioPerl
- Java　　BioJava
- Python　　Biopython

言語により得意分野が異なるので共存

- Open Bioinformatics Foundation (OBF) を結成
  - 情報交換や開発協力など
- データ入出力形式の標準化 (OBDA)

Rubyを選択した理由

- オブジェクト指向
  - データ構造を容易に記述できる
  - データとデータに対する処理を一括管理可能
    →構造化されたデータが数多く存在する生物学分野では特に有用
- 簡潔な文法
  - 書きやすく読みやすい
  - 開発効率が高い
- 日本で誕生し海外にも普及した言語

BioRuby

- フリーソフトウェア
  - 誰でも自由にコピーや配布ができる
  - ソースを公開しており改造も自由
    - 改造した物の再配布も可能
- オープンな開発体制
  - インターネットを活用
  - 誰でも開発に参加可能
- http://bioruby.org/

BioRuby

- 2000/11/21　BioRubyプロジェクト開始
- 2001/06/21　バージョン0.1をリリース
  - …（この間, リリース20回以上, 学会発表10回以上）
- 2005/06-2006/02　IPA未踏ソフトウェアプロジェクト
- 2006/02/24　バージョン1.0をリリース
- 2006/12/25　1.1.0-pre1

- 現在（1.1.0-pre1）
  - ファイル数:　173　（ドキュメント・テストを除く）
  - 行数:　約38,000行　（空行・コメントのみの行を除く）
  - 30種類以上のデータ形式の読み込みサポート
  - 20種類以上のアプリケーションに対する何らかのサポート
  - 開発者:　累計 10人以上（うち海外3-4人以上）

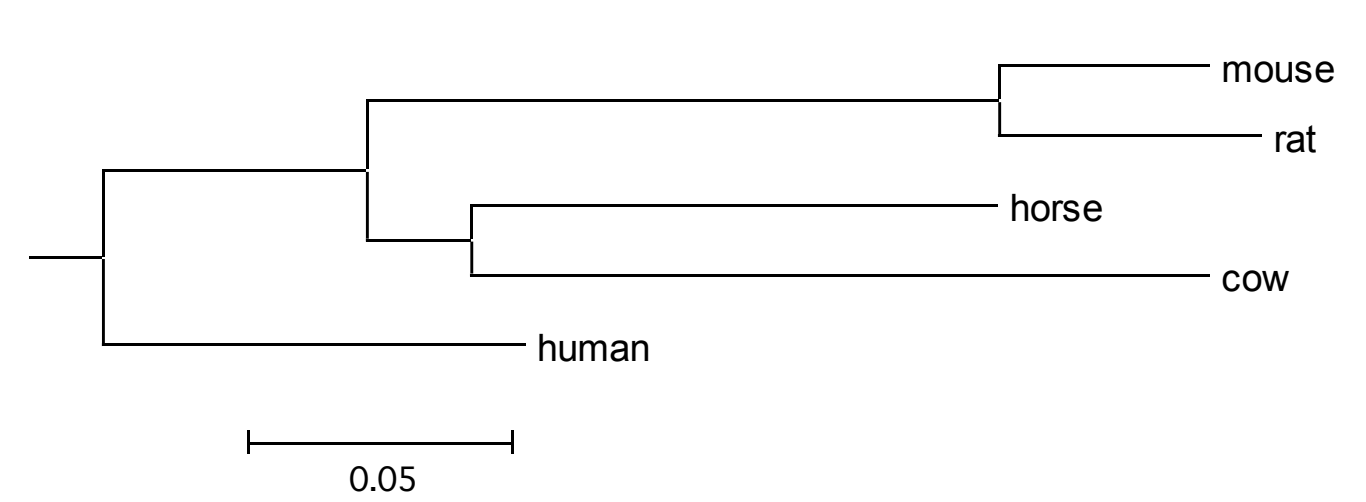
http://bioruby.org/

系統樹の文字列表記

- Newick形式（別名New Hampshire形式）
  - 系統樹の文字列表記のデファクトスタンダード

例:

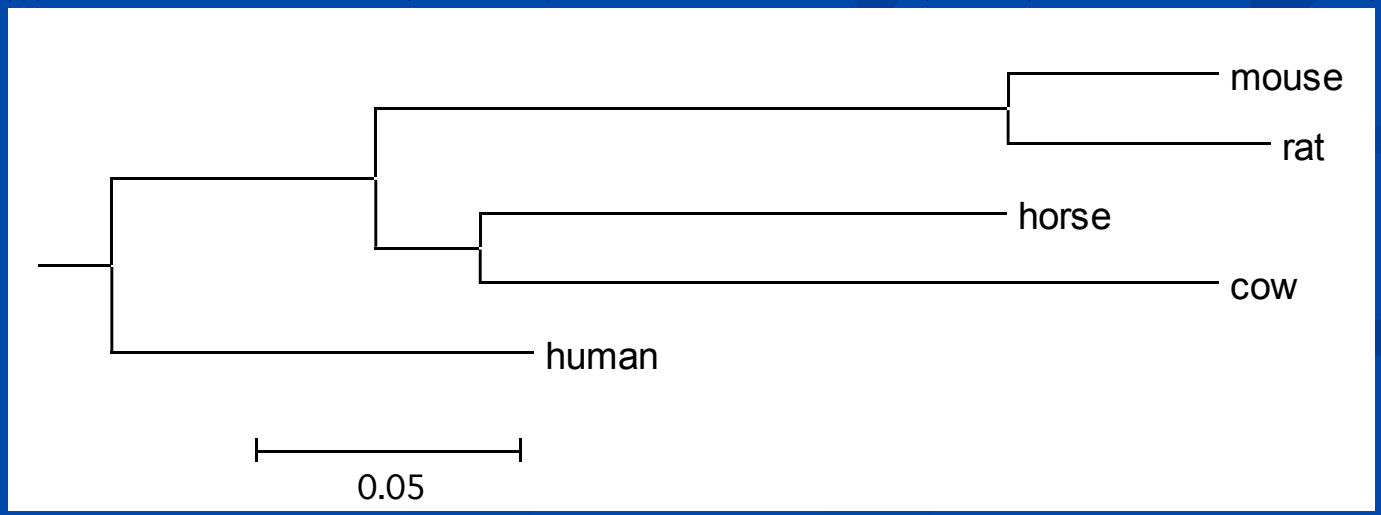
```
(((mouse:0.04, rat:0.05):0.12,
  (horse:0.10, cow:0.14):0.02):0.05,
 human:0.08);
```



系統樹のデータ構造

- 階層的クラスタリングとして扱う
  - Newick形式を素直に解釈するとそうなる
  - ノードや辺の追加・削除、rootの変更が面倒
  - 親子関係を得るのは容易
- グラフ(Graph)として扱う
  - ノードや辺の追加・削除、rootの変更、複数系統樹の結合などが容易

(((mouse:0.04, rat:0.05):0.12, (horse:0.10, cow:0.14):0.02):0.05, human:0.08);



Bio::Tree

- 系統樹を格納するデータクラス
  - 主に分子系統樹を想定
- 内部データ構造としてグラフ構造を使用
  - 無向グラフとして系統樹を保持
    - 隣接リスト+辺リストとしてグラフを保持
  - 内部でBio::Pathwayクラスを利用
    - BioRubyの提供するグラフ構造クラス
    - ただしユーザーはBio::Pathwayのメソッドは直接利用不可
      - 将来Ruby標準で高性能グラフ構造クラスが提供されたらそちらを使うかも
  - インターフェースはBoost Graph Libraryを参考に

Bio::Treeクラスの成立経緯

- 2005年6月頃、Newick形式のパーサーを試作
  - 内部データ構造はグラフではなく階層的クラスタリング
    - ノードや辺の追加・削除、rootの位置変更は未実装
  - そのままお蔵入り
- 2006年7月、Daniel AmelangさんがBioRubyメーリングリストにNewick形式のパーサークラスを作成したと投稿
  - やはり内部データ構造は階層的クラスタリング
    - ノードや辺の追加・削除、rootの位置変更は未実装
- そこで奮起して開発開始
- 2006年10月、最初のバージョンが完成

Bio::Treeの簡単な例

```
#!/usr/bin/env ruby
require 'bio'
str = '(((mouse:0.04, rat:0.05):0.12,
         (horse:0.10, cow:0.14):0.02):0.05,
         human:0.08);'
# Newick形式のデータクラス
newick = Bio::Newick.new(str)
# Bio::Treeオブジェクトを得る
tree = newick.tree
# ノードの一覧をArrayとして返す
p tree.nodes
# 終端ノード(leaf)一覧をArrayとして返す
p tree.leaves
# 辺(edge)とその両端のノードの一覧をArrayとして返す
p tree.edges
```

Bio::Treeクラスの概要

- Bio::Tree
  - 系統樹（グラフ）のトポロジーを表現
    - 内部にBio::Tree::NodeとBio::Tree::Edgeのオブジェクトを格納
- Bio::Tree::Node
  - ノードを表現
    - 名前やブートストラップ値などの情報を持つ
  - 内部ノードと終端ノード(leaf)を区別しない
    - rootも特別扱いしない
- Bio::Tree::Edge
  - 辺(edge)を表現
    - 距離などの情報を持つ
  - 両端のノードの情報を持たない

根(root)を変更する

```
#!/usr/bin/env ruby
require 'bio'
str = '(((mouse:0.04, rat:0.05):0.12,
         (horse:0.10, cow:0.14):0.02):0.05,
         human:0.08);'
tree = Bio::Newick.new(str).tree
# 現在のrootを表示
p tree.root
# "mouse"という名前のノードを得る
mouse = tree.get_node_by_name('mouse')
# mouseのすべての祖先ノードを表示
p tree.ancestors(mouse)
# "horse"および"cow"という名前のノードを得る
horse = tree.get_node_by_name('horse')
cow = tree.get_node_by_name('cow')
# horseとcowのもっとも最近の共通祖先をrootに設定
tree.root = tree.lowest_common_ancestor(horse, cow)
# mouseのすべての祖先ノードを表示
p tree.ancestors(mouse)
# Newick形式のデータを表示
print tree.output_newick
```

Bio::Treeのその他の主なメソッド

each_node	ノードを順にたどるイテレータ
each_edge	辺を順にたどるイテレータ
adjacent_nodes(node)	nodeに隣接するすべてのノードを得る
out_edges(node)	nodeに接続するすべての辺を得る
add_node(node)	新しいノードnodeを追加
remove_node(node)	nodeを削除
add_edge(node1, node2, edge)	node1とnode2の間に辺edgeを追加
remove_edge(node1, node2)	node1とnode2の間の辺を削除
distance(node1, node2)	node1とnode2の間の距離を得る
path(node1, node2)	node1とnode2の間の経路のノードを得る
parent(node)	nodeの直近の親ノードを得る
children(node)	nodeの直近の子ノードを得る
descendents(node)	nodeの下のすべての子ノードを得る
leaves(node)	nodeの下のすべての終端ノードを得る
distance_matrix	距離行列を得る
adjacency_matrix	隣接行列を得る
subtree([node1, node2, ...])	subtreeを得る
remove_nonsense_nodes	枝分かれしていない内部ノードを刈り取る

注: いくつかのメソッドはtreeではない(networkの)データを与えるとうまく動かない

課題

- アルゴリズムや解析手法の追加
  - グラフのアルゴリズム
  - 進化系統学の解析メソッド
    - 系統樹のトポロジー比較 --- 文字列表記に変換してから比較？
- メソッドの充実
  - ノードや辺の指定方法
    - 名前のないノードの指定はどうすれば簡単か？
  - Deep copy
- 速度
  - 例：大きな系統樹の距離行列を得ようとすると遅い
    - 将来Rubyに高性能なグラフ構造ライブラリが標準添付されることを期待
- 入出力
  - PhyloXMLなどのデータ形式
  - 描画・作図
- アプリケーションとの連携
    - Phylip, Molphy, PAUP, Hyphyなど
- ドキュメント・サンプル

リンク

- BioRuby
  - http://bioruby.org/
- ChemRuby
  - http://chemruby.org/
- オープンバイオ研究会
    - オープンソースソフトウェアを活用・推進
  - http://open-bio.jp/
- bioinformatics-jp メーリングリスト
  - http://groups.yahoo.co.jp/group/bioinformatics-jp/
- 生物学者による情報処理技術研究会
    - 実験系生物学者のための研究会
  - http://www.kazusa.or.jp/infobiologist/

Phyloinformatics Hackathon

- 期間: 2006/12/10-15
- 場所: NESCent
  - National Evolutionary Synthesis Centerの略
  - アメリカ合衆国ノースカロライナ州Durham
  - Duke大学構内に存在
- 目的:進化系統情報学のソフトウェアの整備
- 参加者数:約25名
- https://www.nescent.org/wg/phyloinformatics/

Phyloinformatics Hackathon

（諸事情により写真は省略しました）