Rubyによる地球流体データの扱い

北海道大学大学院
理学研究科数学専攻
後藤謙太郎
gotoken@math.sci.hokudai.ac.jp

謝辞

□ **地球流体電脳倶楽部**

○ http://www.gdf-dennou.org

□ **科学技術振興事業団**

○ **計算科学技術活用型特定研究開発推進事業**

○ http://www.jst.go.jp

□ **日本UNIXユーザ会**

○ http://www.jus.or.jp

おことわり

このプロジェクトは始まったばかりで、一部の仮実装はしましたが、まだきちんとしたプロダクトはできてません。2000年3月をメドに鋭意作成中です。

あらまし

- 地球流体

- したいこと
  - 膨大なデータを共有したい
  - 絵を書きたい

- 背景となる技術
  - 電脳ライブラリ(DCL, gtools)
  - netCDF

- 進行中プロジェクト
  - netCDFの読み書きとラッパ
  - 新たなデータ書式をRubyで
  - DCLラッパとクラスライブラリ化

地球流体

□おもに、温度、風速

□観測方法もさまざま

○固定、船上、ゾンデ(風船)

□モデルによる計算データも扱いたい

膨大なデータを共有したい

- 100Mバイトは当り前、1Gのファイルがあったりする世界
- FORTRAN でも扱えないと困る
- データの説明をデータファイル自体に書いてないと面倒
  - 自然言語だと面倒
  - FORTRANベッタリだと構造を定義しにくい

絵を書きたい

- □とにかくプロットしたい
- □地球に依存した事情がある
  - ○地図投影
  - ○軸の間隔やラベル
- □後者はDCL(後述)で解決されている

netCDF

netCDF (network Common Data Format)

- **配列指向のデータの共有を目指した書式仕様とライブラリ群.**
- **汎用なので、制限を付けなければ共有できない.**
  - **各種**convention**が存在する**
- University Corporation for Atmospheric Research **発**
  - UNIDATA **の一つ** <URL:http://www.unidata.ucar.edu/>
  - The NASA CDF **に由来**
- FORTRAN, C, C++, Perl, Java **のインターフェイスがある.** (Ruby **がないのは悔しい** ^^;)

DCL

特徴

- FORTRAN ライブラリである．
  - C 言語全盛の時代にあって，古式ゆかしい FORTRAN を 守っている
  - 大型機ではまだ FORTRAN には勝てない
  - ただし現在Cに書き換え中
- 移植性が高い
  - 機種依存する部分はいくつかのサブパッケージにまとめてあるので，これらが移植できれば OK
- 地球科学特有の事情が考慮されている
  - 欠損値処理や地図投影などの機能がもっとも基本的な部分で サポートされている

どこにRubyを活かすか

- プロトタイピング
  - やっぱりインタプリタだと楽チン
  - 数学的なモデルも容易に実装できる
  - 拡張ライブラリ版と使い分けも可能
- データフォーマット
  - パーザーを書かなくていいので楽そう
  - 構造の定義は自由
  - 名前空間を使って仕様を拡張しやすい
- ラッピングのテスト
  - 電脳ライブラリはあまり包まれた経験がない
  - FORTRANのライブラリのインタフェイスの指針(?)

進行中プロジェクト

- netCDFの読み書きとラッパ
- 新たなデータ書式をRubyで
- DCLラッパとクラスライブラリ化

netCDFの読み書きとラッパ

netCDFデータにはテキスト形式とパック形式がある．テキスト形式はヘッダを見るのには有効だが流通には使われない．

パック形式は読みとりやすい形式なので，早速書いた．

- このために pack と unpack を拡張させてもらった (e,E,g,G)．

問題点

- 遅い．10M程度のファイルに含まれる数値を全てオブジェクトにするのはキツい．
  - 多次元配列クラスとnetCDFのラッパで対処予定

新たなデータ書式をRubyで

- 観測データとその機能を分けて記述できる
  - 観測データは固有の座標軸を持たない
    - 位置や時刻も本質的には測定量
    - 格子データと言っても欠損したデータもありうる
    - ゾンデデータの場合格子ですらない
    - 地球は球ではない
  - 一方，公開する場合はデフォルトのプロット対象は指定すべき
    - 温度と風速のどちらをプロットするのかとか
- 自由なのはいいけどFORTRANのことも考えないといけない
  - FORTRANでは基本的には配列によるハッシュになる
  - メソッドはサブルーチン渡しで実現(?)
  - 動的な機能(procとか)はどこまで許すかとか

DCLラッパとクラスライブラリ化

- 各種プラットフォームに対応したため名前の制約などがある
  - 現在FORTRAN90とCでラップ中
- 関数へのデータの受渡しが不便なのでこれも改良の余地あり
  - どの引数に影響が及ぶかなど
- 現在，一部関数のみ実装
- 近いうちにRubyでいくつかのクラスに整理
  - 現在は単にFORTRANのサブルーチンを呼んでるだけ

まとめ

感触

- Rubyだとプロトタイピングとアイデアの実装はやっぱり早い
- packを本家Rubyでも拡張してもらえた小気味よさが嬉しい
- 実行は遅いが，モデル計算なら拡張ライブラリで何とかなりそう
- FORTRANからくらべると一気に抽象度があげれるので作戦会議重要

問題点

- スピードをある程度確保しなければならない
  - 文字列で別言語を書いて呼び出すとかしてオブジェクトを生成せずに数値を触る方法は必要