大規模計算機システム 利用講習会
スーパーコンピュータ利用入門

大阪大学 サイバーメディアセンター
大規模計算研究部門
吉野　元
yoshino@cmc.osaka-u.ac.jp

参考資料

「大規模計算機システム 利用講習会
スーパーコンピュータ利用入門（2013年9月10日）」
大阪大学 サイバーメディアセンター
大阪大学 情報推進部 情報基盤課 研究系システム班

目　次

- UNIX環境を利用するための基礎知識

OSAKA UNIVERSITY

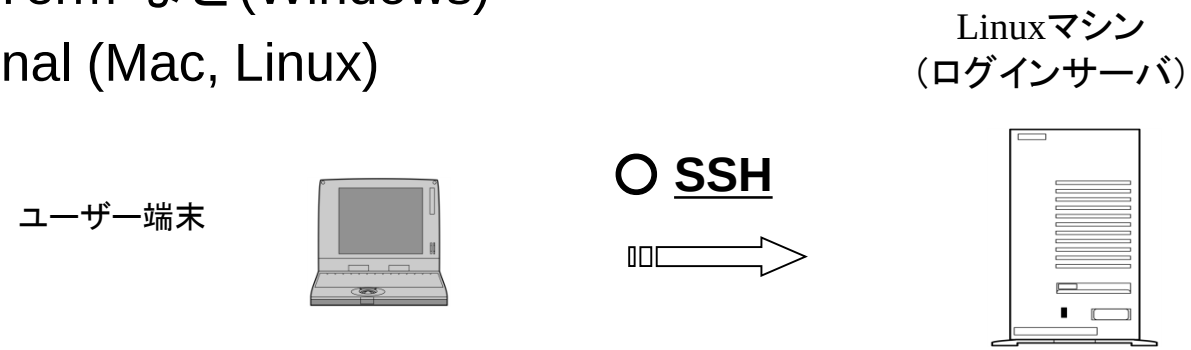
Cybermedia Center
Osaka University

UNIX 環境を利用するための基礎知識

(1) ssh loginとlogoutをやってみる
(2) ファイルシステムの階層構造を探検
(3) 簡単なファイル操作を試す
(4) Emacs はじめの一歩
(5) 標準入出力、シェルスクリプト

UNIX (Linux)の利用

- ログインして利用する
  - リモートマシンを利用する場合
    - sshプロトコルが使えるアプリケーションを利用
      - TeraTerm など(Windows)
      - terminal (Mac, Linux)

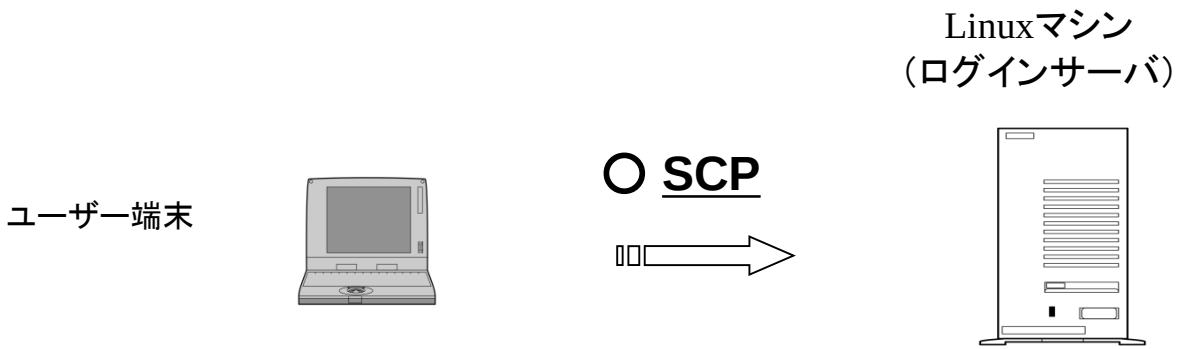


次の例は、ユーザ名v6a022で接続する例です。
% ssh login.hpc.cmc.osaka-u.ac.jp –l v6a022

- 終了時はログアウトする　%logout

UNIX (Linux)の利用

- ファイルの転送
  - プロトコルが使えるアプリケーションを利用
    - TeraTerm など(Windows)
    - terminal (Mac, Linux)



次の例は、ローカルマシンにあるファイル”foo”を
ユーザ名v6a022のホームディレクトリに転送する例です
% scp foo v6a022@login.hpc.cmc.osaka-u.ac.jp :~/

逆の操作は
% scp v6a022@login.hpc.cmc.osaka-u.ac.jp :~/foo .

UNIX (Linux)とシェル

- UNIX (Linux)のバージョン
  - 開発過程などの違いからさまざまなバージョンがある
    - Redhat, CentOS, SUSE, Debian, Ubuntsu, Fedora, Vine Linux, ….
- シェル (Shell)
  - 利用者はシェル上でコマンドを実行する
  - シェルのバージョン
    - csh, tcsh, bash, ….

コマンドラインでwhoを実行してみる。
コマンドラインでps auxを実行してみる。

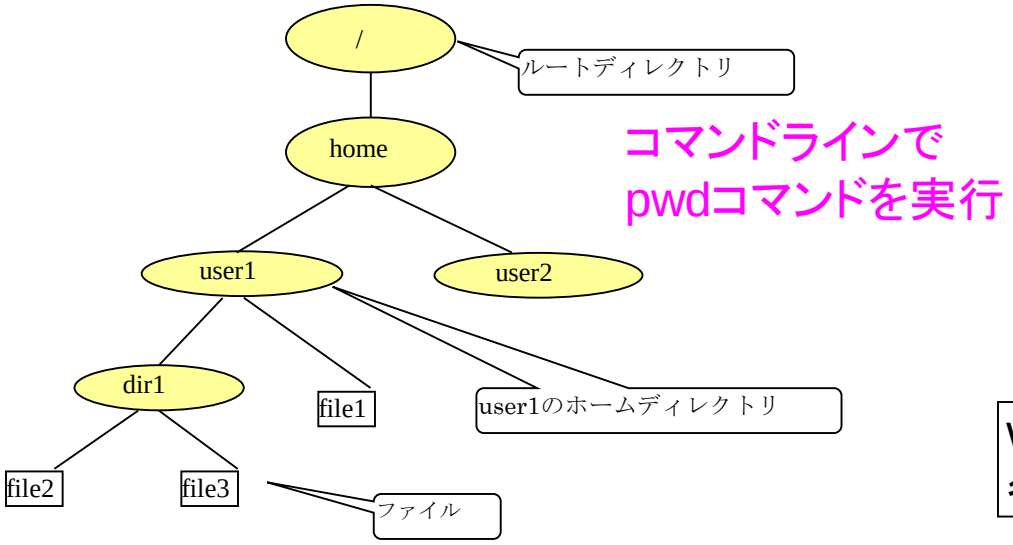
UNIX

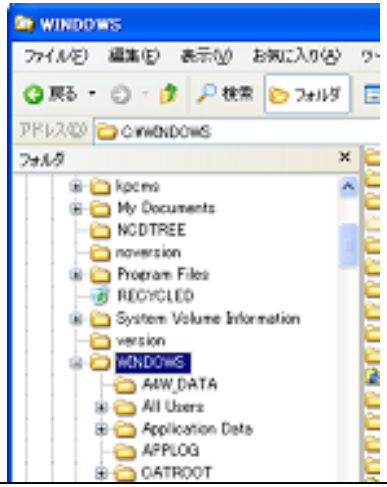
- 1968～69年頃にアメリカAT&T社のベル研究所で開発されたオペレーティングシステム（OS）
  - C言語で記述される

- マルチタスク
  - 複数のジョブをほぼ同時に実行可能
- マルチユーザ
  - 複数のユーザで同時に利用可能
- ネットワーク
  - ネットワーク機能が充実

ディレクトリ・ファイルの構造

- ディレクトリ構造　：　Windowsと同様の階層構造
- UNIX 環境での違い
  - 全てがルートディレクトリの下に展開
  - １つのディスクドライブがルートに割り当て
  - 他のディスクドライブはいずれかのディレクトリに割り当て
  - 統一的に管理･運用が可能





Windowsでは、ツリー構造が
各ディスクドライブの下に広がる

特別なディレクトリ

- ホームディレクトリ
  - UNIX：そこから下位層は個人用となるディレクトリ
  - Windows：特に存在しない
    - 敢えてあげるならデスクトップ、マイドキュメント
    - 環境変数HOMEで指定すれば利用可能

- ルートディレクトリ
  - 単に「/」（スラッシュ）だけで表現
  - 最上位の場合のみ該当
    - それ以外では「/」は別の意味を持つ

Unixコマンドを使ってファイルシステムの階層構造を探検！

特別なディレクトリ

- その他のディレクトリ
  - カレントディレクトリ(current directory)：現在いるディレクトリ
    - 作業ディレクトリ(working directory)とも呼ぶ
  - 親ディレクトリ：
    カレントディレクトリの１つ上のディレクトリ
- ディレクトリの表記方法
  - / ： ルートディレクトリ
  - ~/ ： ユーザのホームディレクトリ
  - ./ ： カレントディレクトリ
  - ../ ： 親ディレクトリ

パス

- パスとは
  - あるディレクトリやファイルが
    ディレクトリツリーのどこにあるかという情報

- UNIXでの記述方法
  - / → home → user1　⇒　/home/user1
  - 先頭以外の「/」は階層の区切りを示す

- （参考） Windowsでの記述方法
  - C: → Windows → Task　⇒　C:¥Windows¥Task
  - 欧米環境だと「￥」は「 ＼ 」

パスの種類

- 絶対パス
  - ルート（Windowsだとドライブ名）からのパス表現
    - UNIX　　：「/」から始まる
    - Windows　　：「C:¥」　等から始まる
  - カレントディレクトリに左右されず一意に決定
  - 下の方の階層になるとパスが長くなる

- 相対パス
  - カレントディレクトリを基準としたパス表現
    - ディレクトリ名、「./」や「../」から始まる
  - 下の方の階層でも短いパスで記述が可能
  - カレントディレクトリが変わると基本的に使えない

パスの使い分け

- 一般的な基準
  - 絶対パス：必ず特定のディレクトリやファイルを指定したい場合に利用
  - 相対パス：ユーザの個人作業で利用
    ⇔　ホームディレクトリ以下での作業　等

- 表現例（アカウント　user1 の場合）
  - 絶対パス：　/home/user1/program/samples/source.f90
  - /home/user1/program にいる時の相対パス
    - ./samples/source.f90
    - samples/source.f90
    - ~/program/samples/Lesson001.txt

UNIX コマンド

- コマンドとは
  - ユーザがキーボードなどで特定の文字列を入力して
    コンピュータに与える命令のこと
  - 記述は１行（入力後は必ずEnterキーを押す）
  - 実行結果は文字列で返ってくる
  - 引数やオプションを付ける場合もある

  - % command [オプション] [引数]

- 引数（argument）とは
  - 命令に対する目的語

- オプションとは
  - コマンドの働きをいろいろと修飾する
  - 一般にコマンドの後に「-[文字]」の形で記述する

主なUNIXコマンド

- ファイル操作コマンド
  - ls： ファイル・ディレクトリの一覧表示
  - cd： 作業ディレクトリを引数で指定したディレクトリに移動
  - mkdir： ディレクトリの作成
  - rm： ファイルの削除
  - cp： ファイルのコピー
  - mv： ファイルの名称変更・移動

- ファイル閲覧コマンド
  - cat : ファイル内容を表示
  - more : ファイルの内容をページごとに表示
  - less : moreの高機能版

ls

- 概要
  - ファイル・ディレクトリの一覧表示
  - list の略
  - ディレクトリ移動後は基本的に最初に実行

- 引数
  - 一覧を表示したいディレクトリへのパス
  - 省略時はカレントディレクトリの一覧を表示
    - 通常はこちらの形式で利用

ls

- 主要なオプション
  - -l ： 詳細情報(long format)　ファイルの所有者、権限、大きさ など
  - -a ： 隠しファイル（ . で始まるファイル名）も表示(all)
  - -t ： タイムスタンプでソート(time)
  - -F ： ファイルとディレクトリを区別して表示(File)
  - -r ： 逆順にソートして表示(reverse)
  - -R ： 下方のディレクトリ内も再帰的に表示(Recursive)
  - -h : 　-lオプションと共に用いるとファイルの大きさの表記がわかりやすくなる
- 利用方法
  - 複数のオプションはまとめて記述
  - よく使われる形式
    - ls -l,　ls -a,　ls -ltr,　ls -lR

cd

- 概要
  - 作業ディレクトリを引数で指定したディレクトリに移動
  - change directory の略
- 引数
  - 絶対パス・相対パスのどちらも利用可能
  - 省略するとhomeに戻る
- パス情報に関するコマンド
  - pwd：カレントディレクトリの絶対パスを表示

mkdir

- 概要
  - ディレクトリの作成
  - make directory の略
- 引数
  - 作成したいディレクトリの名前
- 類似コマンド
  - rmdir：ディレクトリの削除（後述）
  - touch：空のファイルを作成

rm

- 概要
  - ファイルの削除
  - remove の略
  - 削除されたファイルを元に戻すことは不可能

- 引数
  - 削除するファイル名(複数指定可能)
  - ワイルドカード「*」の利用が可能
    - 「*」以外が一致するファイルは全て処理対象
      - 例1：a*.txt ⇒ a1.txt, a123.txt, abc.txt, …
      - 例2：* ⇒ そのディレクトリにある全てのファイル

rm

- 主要なオプション
  - -i：ファイルの削除前に問い合わせる
    - 「yes」か「y」を入力しなければ削除しない
  - -f：警告せずに削除
  - -r：ディレクトリごとファイルを削除
    ⇔　ディレクトリの削除コマンド：rmdir
    ・ディレクトリの中が空の場合のみ利用可能
  - -v：処理内容を表示
- 利用例
  - rm hoge　　：ファイルhoge を削除
  - rm -vi hoge　：ファイルを削除してよいかの確認があり、結果も表示
  - rm hoge*　　：hoge で始まるファイルをすべて削除
  - rm -rf hoge　：ディレクトリhoge 以下のすべてのファイルと
    ディレクトリを削除

cp

- 概要
  - ファイルのコピー
  - copy の略
  - ディレクトリにも利用可能

- 引数
  - 引数は２つ指定
  - 第一引数：コピー元のファイル名
  - 第二引数：コピー先のファイル名

cp

- 応用
  - ディレクトリのコピー
    - 実行時に「-r」オプションを付ける
    - 中のファイルごとコピーされる
    - 第二引数はディレクトリ名
  - 第一引数のファイル名にワイルドカードを利用
    - 複数のファイルを一度にコピーできる
    - 第二引数はディレクトリ名
      ⇒　コピー先のファイルは元ファイルと同じ名前

mv

- 概要
  - ファイルの名称変更・移動
  - move の略

- 引数
  - 引数は2つ指定
  - 第一引数：処理対象のファイル名
  - 第二引数：記述する形式によって動作が変化

mv

- 第二引数
  - ファイル名　⇒　その名前に変更
    - mv hoge hogehoge
      ⇒　hoge というファイル名を hogehoge に変更
  - ディレクトリ名　⇒　そのディレクトリに移動
    - mv hoge dir/
      ⇒　ファイル hoge をディレクトリ dir の下に移動
      - ディレクトリ名であることを明示するために
        後ろに「/」を付ける
  - パス付きでファイル名を指定すれば両方を同時実行

cat

- 概要
  - ファイルの内容を出力（表示）
  - catenate（連結する）から
    - 本来はファイルを連結する操作
- 例
  - % cat hoge1
  - % cat hoge1 hoge2 >hoge3

more

- 概要
  - ファイル内容をページ（画面）単位で表示
- 操作
  - SPACE: 1ページ（画面）進む
  - Enter: 1行だけ進む
  - q: 終了
  - / : 下方向への検索
    - n: 同じ検索を繰り返す

less

- 概要
  - moreの高機能版
  - ページを戻ったり進んだりできる
- 操作
  - moreの操作コマンドに加えて
  - f: 1ページ（画面）進む
  - b: 1ページ（画面）戻る
  - g: ファイル先頭へ移動
  - G: ファイル末尾へ移動
  - 数字n+操作コマンド: n回の操作コマンドを実施する

tar

- 概要
  - 多くのファイル(ディレクトリを含んでも良い)を一つのアーカイブにまとめる
  - アーカイブを展開
- 操作

  まとめ方

  ```
  tar -cf archive.tar foo1 foo2 foo3
  ```

  展開の仕方

  ```
  tar -tvf archive.tar
  ```

du

- 概要
  - ディレクトリ内のファイル容量を表示
- 操作

  -s 引数で指定したファイルやディレクトリ(サブディレクトリを含め)総計を表示

  -h 見やすい表示

  du -sh . カレントディレクトリ以下の総計

作業環境

- 効率的に作業を行うために
  - 各作業ごとにディレクトリを分ける
    - ディレクトリが違えば同じ名前のファイルを作成可能
  - 文字列補完
    - [Tab]キーを使って入力している文字列を補完できる
  - ヒストリ機能
    - [↑]や[↓]で以前に入力したコマンドを呼び出せる（ctl-pやctl-nも可）
  - コマンド「cp」の利用
    - 編集前に対象ファイルのコピーを作成し、
      バックアップを取る習慣を付けておく方がよい

標準入出力、リダイレクション、シェルスクリプト

1. 標準入力=キーボード、標準出力=ディスプレイ、標準エラー=ディスプレイ
   (例) %pwd コマンドを実行すると結果は標準出力(画面)に表示される。

2. リダイレクション(redirection)によって標準入出力を変更できる。またパイプ(pipe)は標準出力を標準入力につなぐ。
   (例) %pwd > foo
   %date >> foo
   % who | sort
   % who | sort –k 3 > goo

3. コマンドを並べてゆくとスクリプト(script)ができる。
   (例) %cat foo.scrpt
   #!/bin/sh (シェルスクリプトの場合)
   cal
   echo “today =” `date | awk ‘{print $1,$2,$3} ` (注：`はbackquote)

   % chmod u+x foo.scrpt (実行権限を付与)

   %./foo.scrpt で実行してみると。。

OSAKA UNIVERSITY

Cybermedia Center
Osaka University

エディタ　Emacsの基本的な使い方

OSAKA UNIVERSITY

Cybermedia Center
Osaka University

エディタ Emacs の基本的な使い方 (1)

1. emacs起動　ターミナルでemacsコマンド
2. ファイルを開く　cntl-x cntl-f ファイル名
3. 文字入力　Hello World とタイプ
4. ファイル保存　cntl-x cntl-s
5. emacs終了　cntl-x cntl-c

ファイルが出来ていることを確認

ターミナルでlsコマンド ファイル名のリストを確認
ターミナルでcat ファイル名 で(あるいはmore, less コマンドなどを使って)中身を確認

エディタ Emacs の基本的な使い方 (2)

ショートカットキー	動作
C-f	カーソルを1文字分右へ
C-b	カーソルを1文字分左へ
C-p	カーソルを1文字分上へ
C-n	カーソルを1文字分下へ
C-a	カーソルを行頭へ
C-e	カーソルを行末へ
C-d	カーソル位置にある文字を削除
C-m	カーソル位置に改行、カーソルも次の行頭へ
C-o	カーソル位置に改行、カーソル位置は移動しない
C-v	カーソル位置を1画面分下へ
M-v	カーソル位置を1画面分上へ
M->	カーソルをファイルの先頭へ
C-f	カーソルをファイル末尾へ
C-l	カーソルがある行がウィンドウの中央になるようスクロール
C-_	編集をUndo
C-g	コマンド入力/実行をキャンセル

参考「Emacs 超入門」 長島浩道

http://sourceforge.jp/magazine/09/04/06/1138226