TSUBAME利用法 入門編
(Linux基礎)

目次

- TSUBAME3概要
- Linuxの概要
- ファイル操作
- 各種コマンド
- プログラムの実行
- 利用環境

TSUBAME3概要

- TSUBAME3とは
- 機器詳細
- ISVアプリケーション
- サービス内容
- 計算サーバの利用
- センターホームページの活用

TSUBAMEの名称は

Tokyo-tech **S**upercomputer and **UB**iquitously **A**ccessible **M**ass-storage **E**nvironment

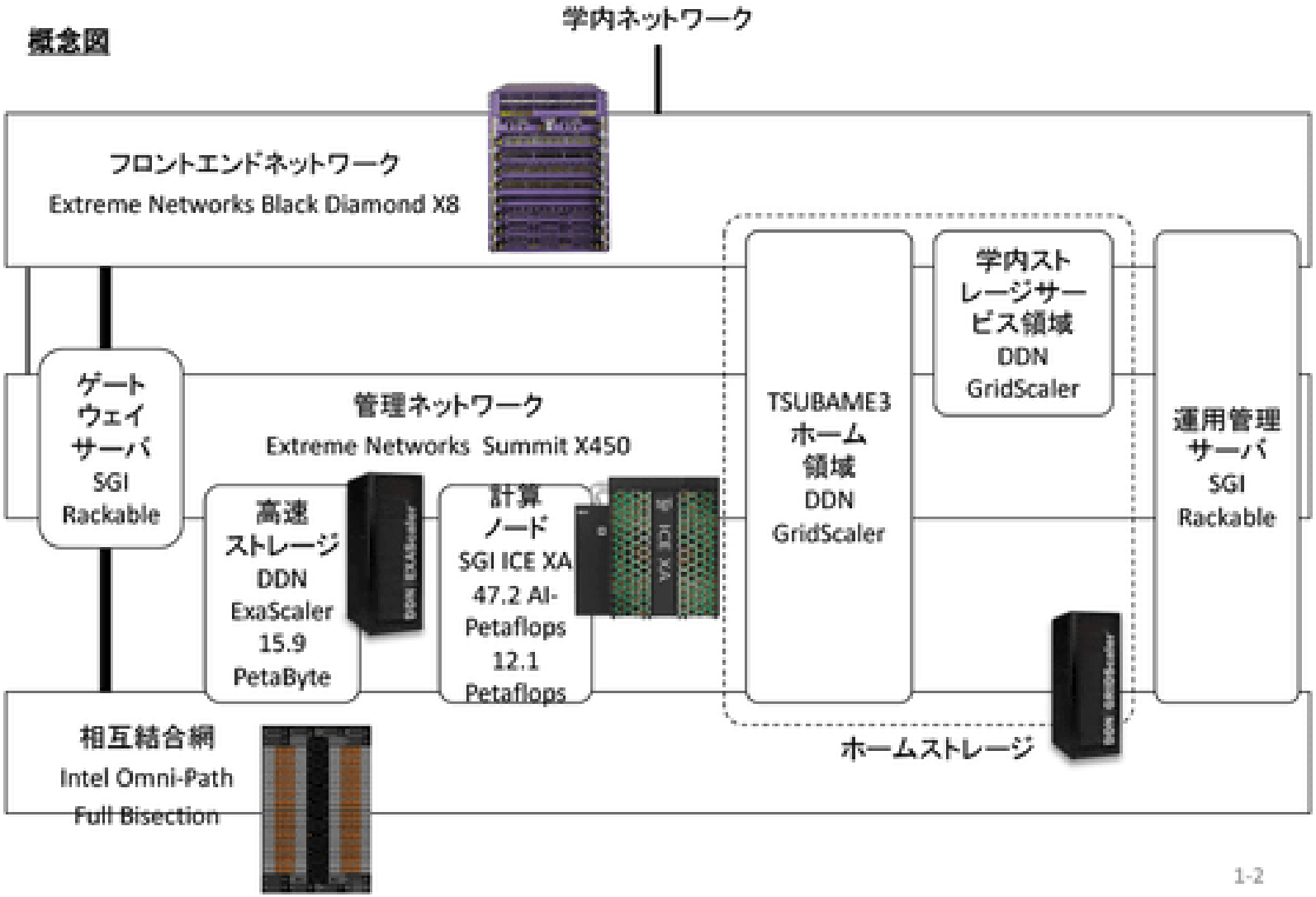
の略で公募により命名されています

TSUBAME3とは

- 2017年8月1日に東京工業大学学術国際情報センター(GSIC)に導入されたGreen500 List 1 位(2017年6月版)のグリーンスーパーコンピュータ
- 倍精度総理論演算性能は12.15PFLOPS
- **半精度**総理論演算性能は**47.2**PFLOPS

- 総主記憶容量は135TiB
- 磁気ディスク容量は15.9PB
- Omni-Pathによる100Gbps高速ネットワーク

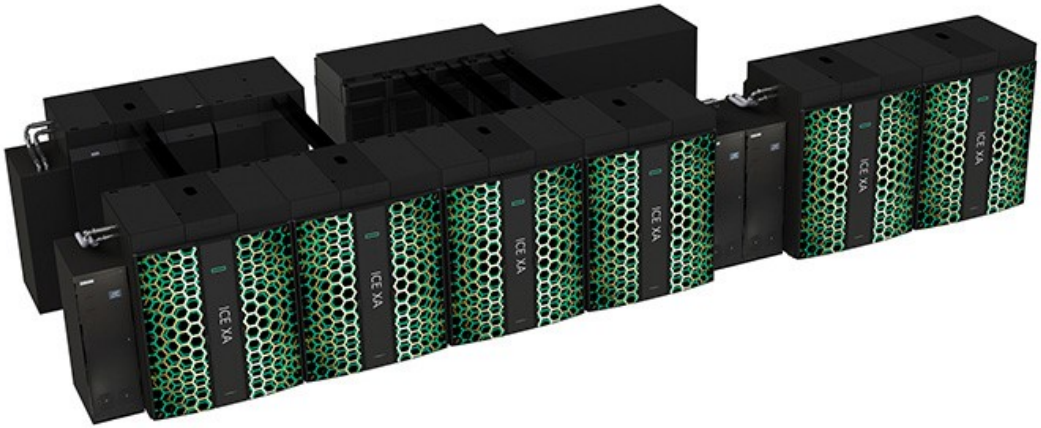
TSUBAME3システム概念図



機器詳細（1）

- 計算ノードSGI ICE XA 540ノード

演算部名		計算ノード　540台
ノード構成（１台あたり）	CPU	Intel Xeon E5-2680 v4　2.4GHz× 2CPU
	コア数/スレッド	14コア / 28スレッド×2CPU
	メモリ	256GiB
	GPU	NVIDIA TESLA P100 for NVlink-Optimized Servers ×4
	SSD	2TB
	インターコネクト	Intel Omni-Path HFI 100Gbps　×4



機器詳細 (2)

- ストレージ

用途	マウント	容量	ファイルシステム
Homeディレクトリ	/home	40TB	GPFS+cNFS
共有アプリケーション配備	/apps		
高速ストレージ領域1	/gs/hs0	4.8PB	Lustre
高速ストレージ領域2	/gs/hs1	4.8PB	Lustre
ローカルスクラッチ領域	/scr	各ノード1.9TB	ext4（SSD)

- ソフトウェア
  - OS:SUSE LINUX Enterprise Server12 SP5
  - ISVアプリケーション別ページ参照

ISVアプリケーション一覧

- ANSYS/Fluent
- ABAQUS/ABAQUS CAE
- MSC One
- Gaussian/GaussView
- AMBER
- Materials Studio
- Discovery Studio
- Mathematica
- Maple
- AVS/Express
- AVS/Express PCE
- LS-DYNA
- LS-PrePost
- COMSOL
- Schrodinger
- MATLAB
- ARM Forge
- Intel Compiler
- PGI Compiler

提供アプリケーションはmoduleコマンドで環境のロード/パージが可能
Gaussian,ARM Froge,およびコンパイラ**以外**は原則学内ユーザのみ利用可能

TSUBAME3.0のアプリケーション

- 全体的にライセンス数は少ないので占有するような利用は避ける
- 今後アプリケーションの利用に対しては課金を予定
- 公平利用を促す課金収入によりライセンスを追加購入できる仕組みを検討中

TSUBAME利用の流れ[事務手続]

[受講者]は2まで終了済みという前提
円滑な利用のためにはポイントが必要となります。

1. アカウント取得
2. SSH鍵ペアの作成と公開鍵の登録(本講習はパスワードでOK)
3. グループの作成[グループ管理者のみ]
4. グループメンバの登録[グループ管理者とグループメンバ]
5. ポイント購入[グループ管理者のみ]
6. グループディスクの設定[グループ管理者のみ]

参考:TSUBAME3の利用までの流れについて

サービス内容

- 計算ノード利用
  - 540ノードの計算ノードSGI ICE XAを利用可能
    計算ノード利用イメージは，別ページ参照

- ストレージ利用
  - homeディレクトリ(制限 **25**GB)
    - 各自のホームディレクトリは，/home/［0-9］/アカウント名
    - システムの全マシンから，アクセス可能
  - 高速ストレージ領域
    - Lustreファイルシステムで構成されるグループディスク領域
    - グループディスクの購入方法は「TSUBAME3.0ポータル利用説明書」参照
  - スクラッチ領域
    - ローカルスクラッチ領域
      - SSDで構成されるローカルスクラッチ領域
    - 共有スクラッチ領域
      - ローカルスクラッチ領域を束ねた共有ファイルシステム
      - 資源タイプFのf_nodeのみ

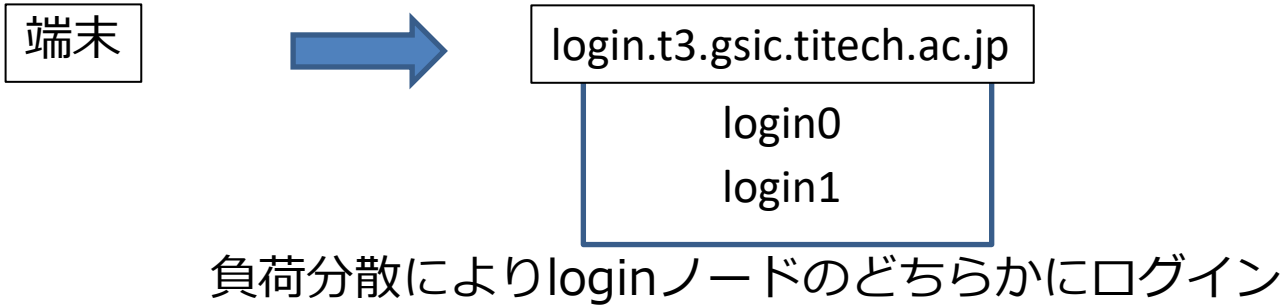
計算ノード利用イメージ (1)

- ログイン(SSH鍵認証のみ)
  - 研究室内PCなどより，sshコマンドを実行．負荷分散を考慮し，自動的にログインノードのいずれか(login0/login1)にログインされる．

※ログインノードでは計算実行は禁止されており、全ての実行ログは保存されております。制限にかかった場合は自動的に実行しているプログラムが停止されます。

※ログインに使うSSH公開鍵をアップロードする必要がある

「TSUBAME3.0ポータル利用説明書」参照



計算ノード利用イメージ (2)

ジョブは540ノードある計算ノードのいずれかで実行される

ユーザによるノード指定は基本的に出来ない

- インタラクティブジョブ(小、中規模ジョブ)
  - ログインノードから，ジョブ投入用コマンドqrshを利用して実行

    $ qrsh -l資源タイプ-l 確保時間 -g TSUBAME3グループ
  - インタラクティブノード上で実行

    $ ./a.out
- バッチジョブ(中、大規模ジョブ)
  - ログインノードから，ジョブ投入用コマンドqsubを利用して実行

    $ qsub -l 資源タイプ -l確保時間 -g TSUBAME3グループ job.sh

ノードは，いくつかの資源タイプに分かれている

→資源タイプは，「利用の手引き」参照

センターホームページの活用

http://www.t3.gsic.titech.ac.jp/

TSUBAME計算サービス

TSUBAME3.0

Tools

コンテンツを追加

メニュー

- トップ
- TSUBAMEポータル
- お知らせ
- 規則・利用細則
- 年間スケジュール
- アカウント取得方法
- 利用の手引き

現在のTSUBAMEの運用状況 / Current TSUBAME Operational Status

2018.9.20 10:00：利用できます / Available

重要なお知らせ

2018-08-23
メンテナンス後の変更点について(2018年8月)
2018-08-17
【復旧】TSUBAMEの再開の大幅な遅れについて(21日14:30追記)
2018-07-05
2018年度 夏季一斉休業・停電に伴うTSUBAME3.0計算サービスの停止について(8/17)

詳しくは

- 重要なお知らせ、お知らせ
  - 障害情報が掲載されているので，定期的に参照してください.
- TSUBAME計算サービス

  https://www.t3.gsic.titech.ac.jp/
- 利用前に必ず利用の手引きを読んでください.
  - 利用の手引き

    https://helpdesk.t3.gsic.titech.ac.jp/manuals/handbook.ja/pdf/handbook.ja.pdf
  - 簡単な問答集+手引きの補助(FAQ)

    https://www.t3.gsic.titech.ac.jp/faq
  - 問い合わせ先

    https://www.t3.gsic.titech.ac.jp/contact-t3

Linuxの概要

- Linuxの概略
- ログイン
- リモートログイン、ログアウトの仕組み
- ターミナル
- TSUBAMEへのログイン
- ログアウト

Linuxの概略

- LinuxはLinuxカーネルを利用したOS

- Linuxはマルチタスク/マルチユーザーのOS
  - 複数のユーザーが同時に使える
  - 利用者が誰か、正当な利用者かどうか

- 多様性の高いLinuxディストリビューション
  - debian
  - slackware linux
  - SUSE Linux Enterprise Server (SLES)
  - Red Hat Enterprise Linux (RHEL)
  - CentOS

  …

ログイン

- 利用者が誰か、利用者が正当な利用者であるかどうかの認識をする。
- Linuxにログインするときはログイン名とパスワードが必要となる。

_____-000 へようこそ
ログイン名 : _____
パスワード : _____

必要事項を入力後に認証されて使用可能となる

リモートログイン

- 遠隔地にあるホストを、現在のホストから使用する
- コマンドとしては telnet, rlogin, ssh など
- TSUBAMEではセキュリティの高い鍵認証sshを導入

認証 →

作業 →

Terminal

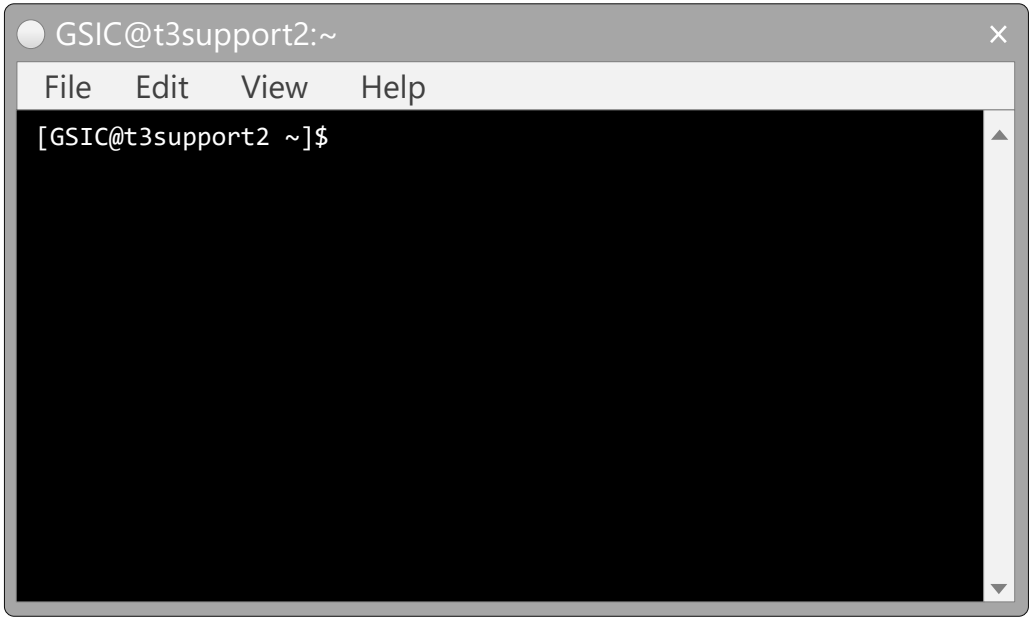
```
[GSIC@t3support2 ~]$ ssh login.t3.gsic.titech.ac.jp -l GSIC -i ~/.ssh/id_ecdsa
Last login: Tue Oct  3 09:26:54 2017 from 131.112.3.100
GSIC@login0:~>
GSIC@login0:~> top
GSIC@login0:~> exit
logout
Connection to login0.t3.gsic.titech.ac.jp closed.
[GSIC@t3support2 ~]$
```

← 接続終了

以降でてくるユーザ名のGSICやGSICUSERは例です
実際は自分のTSUBAME3のアカウント名を利用して下さい
端末ではシェルによって異なりますが以下のような表記となります
ユーザ名@ホスト名 : ディレクトリ

端末(ターミナル)

- Linux利用時は標準的な端末(ターミナル)を使用する



Windows用ターミナルからの利用

- Cygwin/PuTTY/RLogin/MobaXtermなど様々なソフトウェアがあります。

- おすすめはRlogin/MobaXtermですがいろいろ試して気に入ったものをお使い下さい。

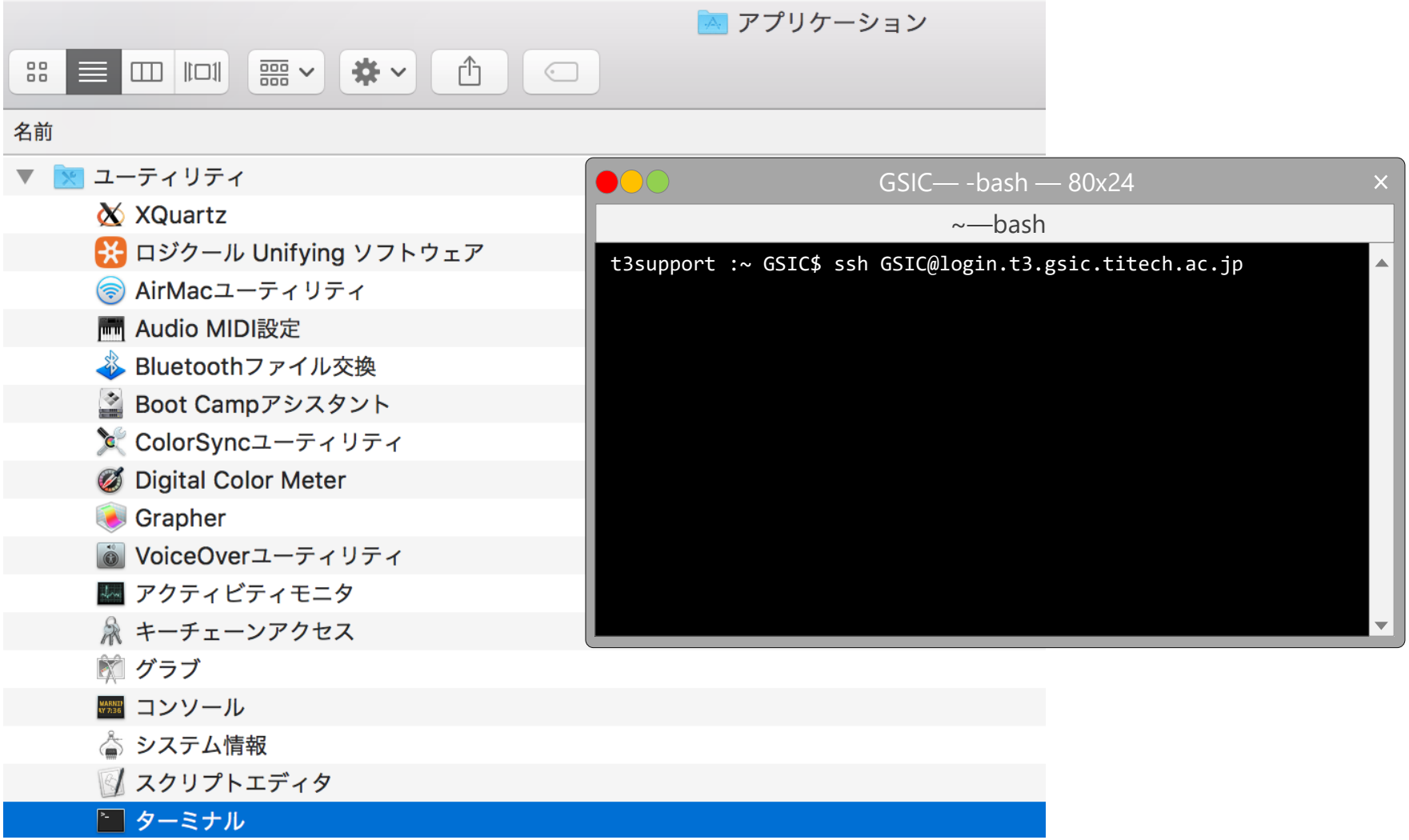
各ターミナルアプリケーションの利用方法についてはお問い合わせいただいてもお問い合わせ窓口ではサポートできません。

個人環境となるため、出来ても助言程度となります。

参考:Windowsで利用できるSSHクライアントについて

Macからの利用

- アプリケーション>ユーティリティ>ターミナル.app

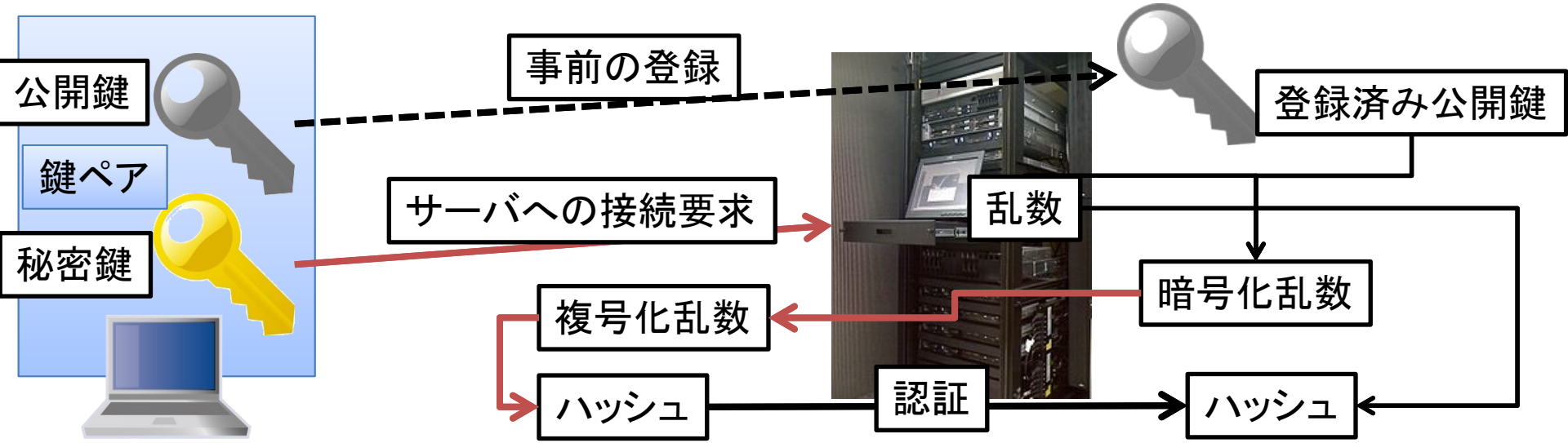


SSH鍵認証

公開鍵(Public)/秘密鍵(Private)のペアを利用した認証システム

- メリット
  - 鍵を所有しているマシンからのみアクセスが可能=高セキュリティ
- デメリット
  - 作成/管理のコスト

*秘密鍵は絶対に流出させない(秘密鍵から公開鍵を作成できる)



TSUBAME3へのログイン

- ログイン操作 → % や $ という記号を表示(ターミナル)
- これをプロンプトと呼び、コマンドを入力できる状態
- この状態からTSUBAMEを使うためにリモートログイン

  →ssh を使用

実習室ではポータルに登録した「パスワード」を利用して下さい。

GSIC@t3support2:~

File Edit View Help

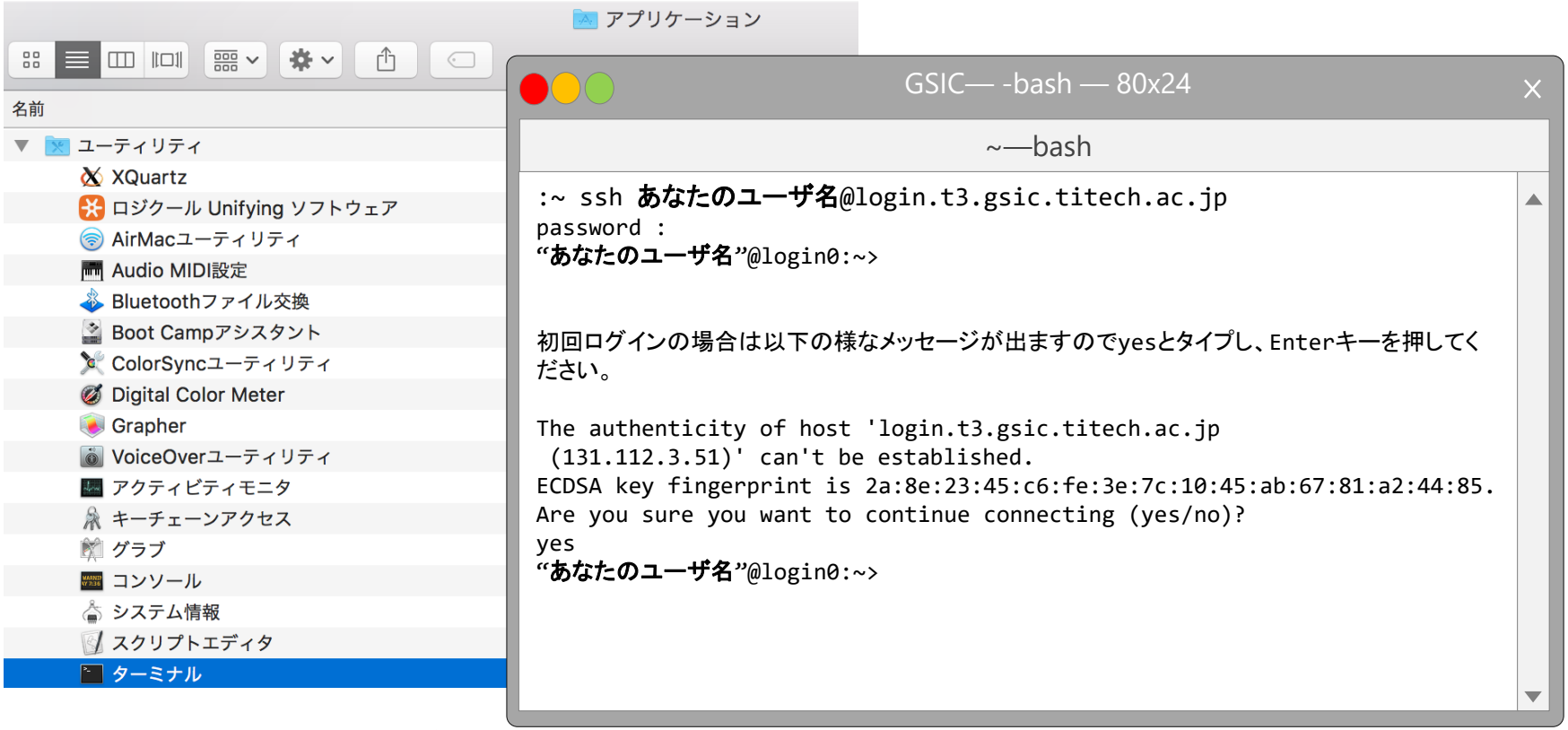
```
[GSIC@t3support2 ~]$ ssh あなたのユーザ名@login.t3.gsic.titech.ac.jp -i 鍵ファイル名
Last login: Tue Oct  3 09:26:54 2017 from 131.112.3.100
GSIC@login0:~>

以下のようにしても同じ
$ sshあなたのユーザ名@login.t3.gsic.titech.ac.jp
```

実際は自分のTSUBAME3のユーザ名を利用して下さい
15B0000 ,18M0000,18D0000,18R0000など

実際にログイン操作を行います

- アプリケーション>ユーティリティ>ターミナル.app



実習室ではパスワードログインとなりますので、ポータルに登録した「**パスワード**」を利用して下さい

ユーザ名は学生証、職員証、アクセスカードに記載されているidや学籍番号などです(お立場によって記載が異なります)

TSUBAMEログインノードのlogin0もしくはlogin1にログインできていればOKです

ログアウト

- 操作を終了したい場合はログアウト操作
- 現在作業しているマシンの確認
  - データの整理
  - 実行状態の確認

- ログアウト操作
  - Ctrlキーと小文字のdを同時に入力(Ctrl-dと表現)
  - logoutコマンドを使用
  - exitコマンドを使用

Tips

- 研究室や自宅から自分が利用するクライアントから接続する際は、KeepAliveなどの設定を適切に行なって下さい。

- 行わない場合はTSUBAMEとの通信が切断されます。

- クライアントによって作業が異なります。
  - mac,cygwin,linuxでは以下のような設定を追記する

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ cat ~/.ssh/config
        ServerAliveInterval 120
        ServerAliveCountMax 30
```

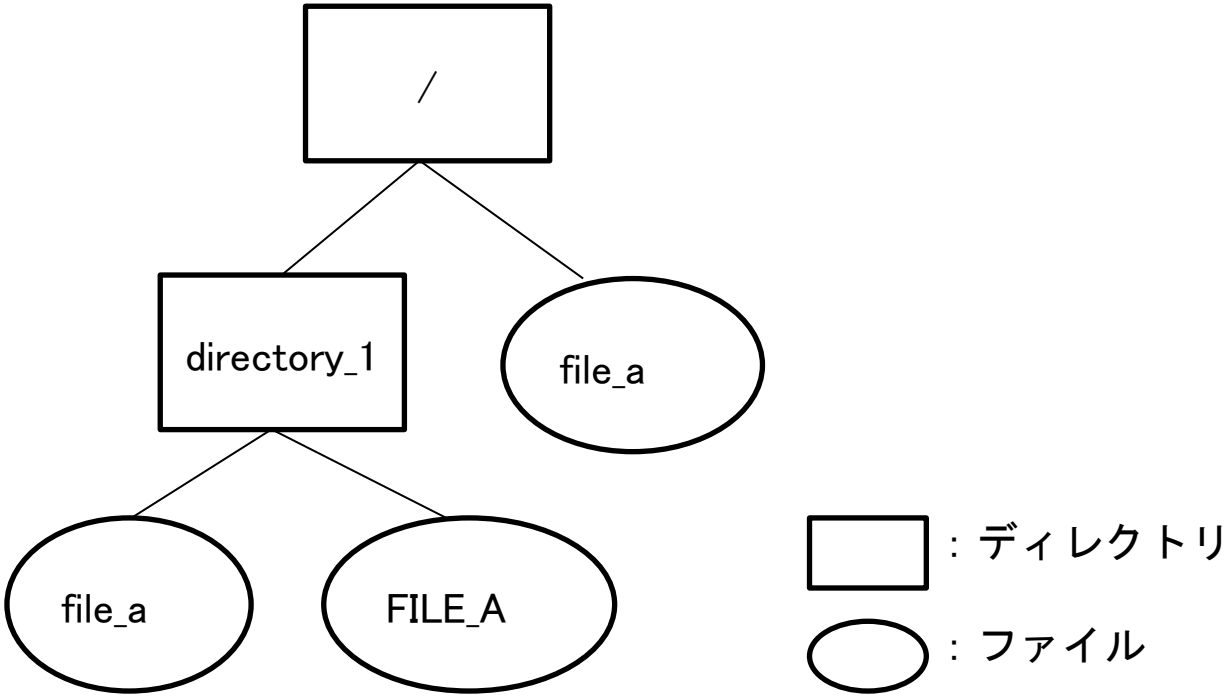
  - sshコマンドに-o ServerAliveInterval=120をつけても良い

ファイル操作

- ファイルシステム
- ファイルの指定
- ファイル指定に使用する記号
- 情報表示コマンド
- ファイルを指定する特殊文字
- ファイルのパーミッション
- ファイル管理コマンド
- マシン(OS)によるファイルの差異

ファイルシステム

- 階層構造で表現
- ディレクトリによるファイル管理
- 一般ファイルと特殊ファイル



ファイルの指定

- 階層構造のトップを / と表現し、”ルート”と呼ぶ
- ファイルへのアクセス方法
  - 絶対指定：ルート(/)からフル指定
  - 相対指定：自分の位置(ディレクトリ)からファイルが存在する位置を指定
    - ../　一つ上の階層を表現
    - ./　現在の階層を表現

GSIC@t3support2:~

File　Edit　View　Help

```
[GSIC@t3support2 ~]$ ls /home/GSIC
Desktop
[GSIC@t3support2 ~]$ ls ../GSIC
Desktop
```

ファイル指定に使用する記号

- ホームディレクトリ　　~
- 一つ上のディレクトリ　../
- 現在のディレクトリ　　./

GSIC@t3support2:~

File　Edit　View　Help

```
[GSIC@t3support2 ~]$ cd ..
[GSIC@t3support2 home]$ cd ~
```

※カレントディレクトリ(現在位置)を表現する場合．(ドット)を使用

情報表示コマンド

- pwd (自分が今何処にいるのかを表示するコマンド)

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ pwd
/home/GSIC
```

- ls　(欲しい情報に合わせてオプションを指定)

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ ls
Desktop
```

ファイルを指定する特殊文字

- ファイル名を指定するために特殊文字表現が利用可能。

メタキャラクタ	機能	使用例
*	全ての文字列に対応	ls *
?	1文字に対応	ls ?
【文字列】	文字列の中の1文字に対応	ls bc
【文字1-文字2】	文字1と2の間にある1文字に対応	ls b[a-c]d

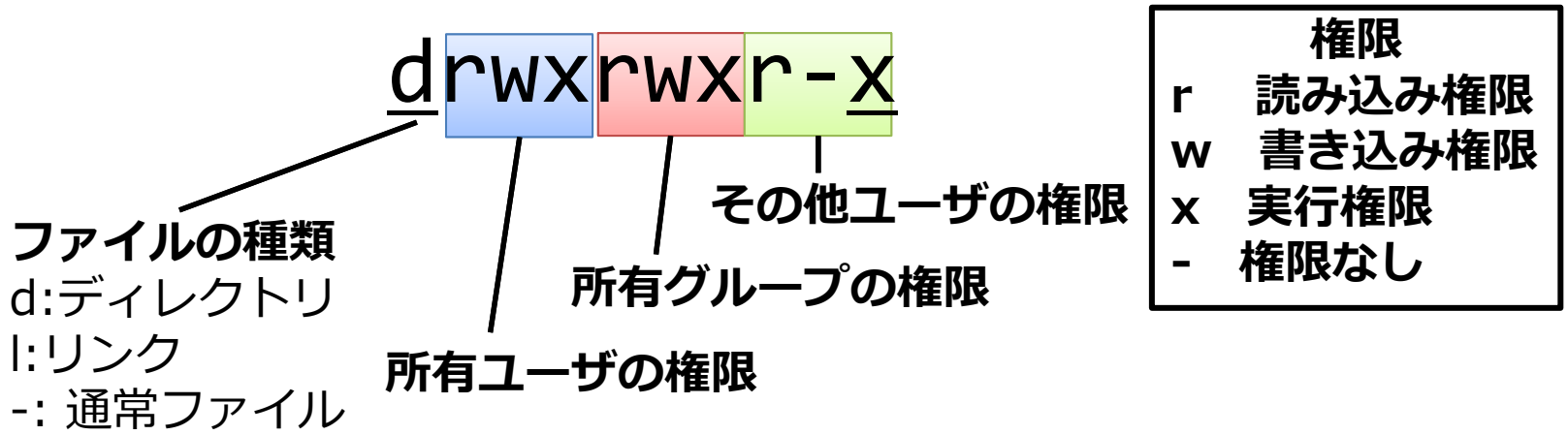
※メタキャラクタ
それ自体は意味を持たないが、他の文字と組み合わせることで全体で
意味を持たせる働きを持つ記号のこと。メタ文字とも呼ぶ。

ファイルのパーミッション

- ls –l コマンド(-lオプション付き)で確認

```
GSIC@g3support2:~
File  Edit  View  Help
[GSIC@t3support2 hogehoge]$ ls -l
合計 0
drwxrwxr-x. 2 GSIC users 6 10月  3 15:35 hoge
-rw-rw-r--. 1 GSIC users 0 10月  3 15:35 hogedoc
モード  サイズ 所有者 グループ 日付      名称
```

- モードについて



ファイル管理コマンド

- ディレクトリの作成　　% mkdir aaa
- ディレクトリの削除　　% rmdir aaa
- ファイル属性の変更　　% chmod 755 aaa

※ 755はオクテットまたはビット表現と呼び、所有者、グループ、その他のユーザーに対するアクセス権を表現する

0　– (権限なし)
1　x (実行可)
2　w (書き込み可)
3　1+2
4　r (読み込み可)
5　1+4
6　2+4
7　1+2+4

(以下はマシン管理者の操作)

- 所有者の変更　　% chown necapps aaa
- グループの変更　　% chgrp procon aaa

マシン(OS)によるファイルの差異

- テキストファイルの改行コード
  - Unix/linux　　¥n LF(line feed)
  - Macintosh　　¥r CR(carrage return)
  - Win/Dos　　¥r¥n CRLF

- 日本語の表示コード
  - Unix/Linux　　UTF-8/EUC
  - Macintosh/windows　　S-JIS

- nkf コマンドにより変換(UTF-8に変換)

  nkf –w abc.txt > abc_utf8.txt

- 改行コードがWin/DosのままだとUnix/Linuxでは動きません

  nkf コマンドによる変換(改行コード変換)

  nkf -Lu abc_*crlf.sh* > abc_*fl.sh*

圧縮・解凍

- 圧縮

  gzip atom45.tar → atom45.tar.gz
  zip atom45.zip atom45 → atom45.zip
  tar czvf atom45.tgz atom45 → atom45.tgz
  tar cjf smpl.tar.bz2 smpldir → smpl.tar.bz2
  bzip2 sample.txt → sample.txt.bz2

- 解凍

  zcat atom45.tar.Z | tar -xvf -
  tar jxf sample.tar.bz2
  gzip -d atom45.tar.gz → atom45.tar
  unzip book2nd.zip
  tar xzvf atom45.tgz
  bzip2 -d sample.txt.bz2 → sample.txt

各種コマンド

- よく使用するコマンド
- コマンド操作
- ファイル操作コマンド
- alias(別名)の機能
- エディタ
- コマンドの結合
- コマンドとジョブ

よく使用するコマンド

- ssh
- exit
- mkdir
- rmdir
- chmod
- chown
- chgrp
- nkf
- cd
- cp
- mv
- rm
- pwd
- ls
- vi もしくはemacs
- view
- tail
- cat ,more ,less
- find
- file
- grep
- diff ,sdiff
- man

コマンド操作

- コマンドのみで実行
- コマンドに引数(オプション)をつける
- コマンドに引数(ファイル)を指定する
- 複数のコマンドを組み合わせる

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ ls
Desktop
[GSIC@t3support2 ~]$
[GSIC@t3support2 ~]$ ls -l
drwxr-xr-x  2 GSIC users    512 Sep 13 10:15 Desktop
[GSIC@t3support2 ~]$
[GSIC@t3support2 ~]$ cal 10 2017
    October 2017
Su Mo Tu We Th Fr Sa
 1  2  3  4  5  6  7
 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31
```

ファイル操作コマンド

- cd
  change directoryの略
  ディレクトリの移動
- cp
  copyの略
  ファイル、ディレクトリの複製
- mv
  moveの略
  ファイル、ディレクトリの移動
- rm
  removeの略
  ファイル、ディレクトリの削除

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 hogehoge]$ ls
hoge  hogedoc
[GSIC@t3support2 hogehoge]$ cd hoge
[GSIC@t3support2 hoge]$ ls
cast dust host
[GSIC@t3support2 hoge]$ cp cast fast
[GSIC@t3support2 hoge]$ ls
cast dust fast host
[GSIC@t3support2 hoge]$ mv host test
[GSIC@t3support2 hoge]$ ls
cast dust fast test
[GSIC@t3support2 hoge]$ rm dust
[GSIC@t3support2 hoge]$ ls
cast fast test
```

※cp,mv,rmは –i オプションで確認処理をする
※一度消去したファイルの復活コマンドはありません。

注意事項

危険なコマンド例

- **$ rm -rf ***

大事なものが入っているディレクトリで叩いては駄目です
* ←　全てを表す記号

スクリプト化した際には以下の記載にもご注意下さい
ホームディレクトリの中身が消え、ログインできなくなります

- **$ rm -rf ~**

alias(別名)の機能

- よく使うコマンドやコマンド列に別名をつける。
- 別名もまた、コマンドとして利用できる。
- 別名を設定するコマンドを alias と呼ぶ。
- 別名定義を解除するコマンドを unalias という。

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ cat .bashrc
# rm Opthion
alias rm='rm -i'
```

※このように.bashrcに書いておくとミスによるファイルの喪失を防止できる。
例えば、rm a* とすると、aで始まるファイルだけを削除する。
間違って、rm a * のように間にスペースが入ると、aというファイルと全てのファイルを削除してしまう。 aliasを設定することで、- i オプションが利き、削除するか確認が来る。

viエディタ

Linuxマシンにはほぼインストールされているエディタ

```
GSIC@t3support2:~
File  Edit  View  Help
[GSIC@t3support2 ~]$ vi hogedoc
~
~
~
~
~
"hogedoc" 0L,                                    0C 0,0-1        全て
```

i または a で入力モード

Escで待機モード

xで1文字削除、ddで1行削除

:wqでエディタを保存終了

:q! で廃棄終了

uで一つ前に戻れる

待機モード時はhjklキーで←↓↑→移動

viewコマンド

- viエディタと同じエディタ内コマンドが利用可能。
- 上書き禁止モードでエディタを開く。
- ファイルの中身を確認する場合に使用。

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ view hogedoc
~
~
~
~
~
"hogedoc" [readonly] 0L,
```

tailコマンド

- ある出力の最後の一部分を標準出力に表示

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ tail -5 /usr/share/doc/python-2.7.5/LICENSE
FITNESS, IN NO EVENT SHALL STICHTING MATHEMATISCH CENTRUM BE LIABLE
FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT
OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
```

※計算結果を定期的にファイルに出力して、tailコマンドで進行状況をみる。
ただし、プログラムでのテキストの連続出力はしない
tailコマンドの連続発行はしない (マシン負荷が増大するため)

cat ,more ,lessコマンド

- ファイルの内容を表示したい場合にcatを利用
- サイズの大きいファイルをページ出力したい場合はmore,less

more,lessはページ単位で停止するので、スペースキーでページ送りします。

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ more /usr/share/doc/python-2.7.5/LICENSE
A. HISTORY OF THE SOFTWARE
==========================

Python was created in the early 1990s by Guido van Rossum at Stichting
Mathematisch Centrum (CWI, see http://www.cwi.nl) in the Netherlands
as a successor of a language called ABC.  Guido remains Python's
principal author, although it includes many contributions from others.

In 1995, Guido continued his work on Python at the Corporation for
National Research Initiatives (CNRI, see http://www.cnri.reston.va.us)
in Reston, Virginia where he released several versions of the
--続ける--(29%)
```

findコマンド

- ファイルの存在する位置を知る
  → ファイル名やファイル名の一部がわかっている場合
  → 実行権のないディレクトリに関する検索は不可能
- 類似コマンドとしてwhereisやwhichコマンド

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ find . -name hoge -print
./hoge

[GSIC@t3support2 ~]$ which python
/usr/bin/python

[GSIC@t3support2 ~]$ whereis python
python: /usr/bin/python /usr/bin/python3.4 /usr/bin/python3.4m
/usr/bin/python2.7 /usr/bin/python2.7-config /usr/bin/python3.4-config
/usr/bin/python3.4m-config /usr/lib/python3.4 /usr/lib/python2.7
/usr/lib64/python3.4 /usr/lib64/python /usr/lib64/python2.7
/usr/include/python3.4m /usr/include/python2.7 /usr/include/python
/usr/share/man/man1/python.1.gz
```

fileコマンド

- ファイルの型を調べる
- 指定されたファイルの文字列を調べ内容を判断

※判断ミスもあるので、全面的に信用しないこと

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ file hoge
hoge: directory

[GSIC@t3support2 ~]$cd hoge
[GSIC@t3support2 hoge]$ file hogedoc
hogedoc: empty

[GSIC@t3support2 hoge]$ file hogedoc2
hogedoc: ASCII text
```

grepコマンド

- ファイル内の文字列検索
- 標準出力(パイプ)からの文字列検索を行なう

viエディタやmoreコマンドなどの文字列検索を行なう前に実行すると便利

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 examples]$ grep mpi *.c
connectivity_c.c:#include <mpi.h>
hello_c.c:#include "mpi.h“

[GSIC@t3support2 examples]$ grep mpi *.c
connectivity_c.c:#include <mpi.h>
connectivity_c.c:    MPI_Status  status;
(略)
hello_c.c:#include "mpi.h“
hello_c.c:    char version[MPI_MAX_LIBRARY_VERSION_STRING];
(略)
[GSIC@t3support2 hoge]$ grep -i mpi *.c | more
(略)
```

diffコマンド

2つのファイルの違いを表示する

diff

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ diff hello_c.c hello_cxx.cc
18,20c18,20
<     MPI_Init(&argc, &argv);
<     MPI_Comm_rank(MPI_COMM_WORLD, &rank);
<     MPI_Comm_size(MPI_COMM_WORLD, &size);
---
>     MPI::Init();
>     rank = MPI::COMM_WORLD.Get_rank();
>     size = MPI::COMM_WORLD.Get_size();
```

sdiff

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ sdiff hello_c.c hello_cxx.cc
    MPI_Init(&argc, &argv);                          |     MPI::Init();
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);            |     rank = MPI::COMM_WORLD.Get_rank();
    MPI_Comm_size(MPI_COMM_WORLD, &size);            |     size = MPI::COMM_WORLD.Get_size();
    MPI_Get_library_version(version, &len);                MPI_Get_library_version(version, &len);
```

オンラインマニュアル

- man コマンド名/ファイル名
- man -k キーワード

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 examples]$ man ls
Man: find all matching manual pages (set MAN_POSIXLY_CORRECT to avoid this)
 * ls (1)
   ls (1p)
Man: What manual page do you want?
Man:
NAME
       ls - list directory contents
SYNOPSIS
       ls [OPTION]... [FILE]...
DESCRIPTION
       List information about the FILEs (the current directory by default).  Sort
entries alphabetically if none of -cftuvSUX nor --sort is specified.
       Mandatory arguments to long options are mandatory for short options too.
```

POSIX
【Portable Operating System Interface for UNIX】
IEEEによって定められた、UNIXベースのOSが備えるべき最低限の仕様のセット。
(e-wordより抜粋)

コマンドの結合

コマンドを結合させて、プログラムのように一括処理ができる

- コマンドをパイプ(|)でつなぐ
- コマンドの実行結果をファイルに書き出す
- ファイルをコマンドに入力する

セミコロン

- 複数のコマンドをセミコロンでつなぐ(グルーピング)
- コマンド1の実行後に、コマンド2,コマンド3が順次実行される

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ uname;arch;hostname
Linux
x86_64
t3support2
```

パイプ

- 例)catの標準出力をgrepに渡す

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ cat /usr/mpi/gcc/openmpi-1.10.4/include/mpi.h | grep INDEX
#define MPI_T_ERR_INVALID_INDEX       57
  MPI_COMBINER_INDEXED,
  MPI_COMBINER_HINDEXED_INTEGER,
  MPI_COMBINER_HINDEXED,
  MPI_COMBINER_INDEXED_BLOCK,
  MPI_COMBINER_HINDEXED_BLOCK
```

リダイレクション

- > の例　　lsの結果を`ls.txt`として作成
- >>の例　　lsの結果を`ls.txt`に追記

GSIC@t3support2:~

File　Edit　View　Help

```
[GSIC@t3support2 ~]$ ls
Desktop hoge
[GSIC@t3support2 ~]$ ls > ls.txt
[GSIC@t3support2 ~]$ cat ls.txt
Desktop
hoge
[GSIC@t3support2 ~]$ ls >> ls.txt
[GSIC@t3support2 ~]$ cat ls.txt
Desktop
hoge
Desktop
hoge
ls.txt
```

ヒアドキュメント

- 標準入力からデータをコマンド(プログラム)へ渡す
  - ファイルからの入力

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ a.out < input.dat
```

  - 標準入力からの入力

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ a.out << EndOfFile
100
EndOfFile
$
```

バックグラウンド実行

- 実行時間の大きなコマンド(プログラム)
- コマンド実行中に他の事をしたい

  →バックグラウンドジョブとして実行

  ※フォアグラウンドでコマンドの実行は問題ない
- コマンドの後ろに&をつけて実行

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ find . -name xinit -print &
[1] 53254
```

プロセス

- プロセスはOS上の処理の実行単位のこと
  - 下の例では3プロセス

GSIC@t3support2:~

File Edit View Help

```
GSICUSER00@r6i3n2:~> top
 PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
354797 hpe_use+  20   0 49.897g 456836 408116 R 14.286 0.173   0:02.09 pmemd.cuda.MPI
354798 hpe_use+  20   0 49.897g 453680 407172 R 14.286 0.172   0:02.08 pmemd.cuda.MPI
  3207 hpe_use+  20   0  425428  60592   1356 S  2.640 0.092 30:26.49 a.out
```

ジョブ

- ジョブはコマンド/プログラムをまとめたシェルの実行単位のこと
  - コマンドをパイプなどでつなげた場合もジョブ

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ sleep 30 &
[1] 3423
 [GSIC@t3support2 ~]$ jobs
[1]+  Running                 sleep 30 &
[GSIC@t3support2 ~]$
```

練習

- ここまでに出てきたコマンドを使ってみます

```
cd
mkdir lesson
cd lesson
cp /gs/hs0/soudan/UNIX/* .
cp sample.sh sample.txt
file sample.txt
vi sample.txt
```

- iもしくはaキーを押し入力モードにし、どこかを適当に編集し,ESCキーを押し(待機状態)、:wq (保存終了)とタイプしてください。

```
diff sample.sh sample.txt
sdiff sample.sh sample.txt
```

解説

```
cd                                  #ホームに移動
mkdir lesson                        #lessonディレクトリを作成
cd lesson                           #lessonディレクトリに移動
cp /gs/hs0/soudan/UNIX/* .          #UNIX以下のファイルをコピー
cp sample.sh sample.txt             #sample.shをsample.txtとしてcp
file sample.txt                     #sample.txtの型を確認
vi sample.txt                       #sample.txtの修正
diff sample.sh sample.txt           #比較
sdiff sample.sh sample.txt
```

プログラムの実行

- moduleコマンドについて
- コンパイラ
- バッチジョブ

moduleコマンドについて

- 利用するソフトウェアに関係する環境設定を、前もってmoduleコマンドで行う
  - 例： $ module load intel　　Intelコンパイラ
    - module load intel/16.0.4.258 のようにバージョン指定も可能
  - 例： $ module load gnuplot
- 用意されているモジュール一覧： $ module avail
- モジュールによって、さらに依存モジュールのロードが必要
  - たとえば、gromacsモジュールはintel-mpiモジュールに依存
- moduleコマンド自体が動かないときは

  $. /etc/profile.d/modules.sh 先頭は「ピリオド・スペース」

コンパイラ

- 利用できるコンパイラ
  - Intel Compiler(Intel Parallel Studio XE Cluster Edition)
  - Intel oneAPI
  - PGI Compiler
  - NVIDIA HPC SDK
  - gcc

利用可能な最新バージョンは、module avコマンドで確認してください

コンパイル(基本型)

- 例1. Cプログラム「pi.c」の場合

  $ gcc pi.cまたは $ icc pi.c $pgcc pi.c

- 例2. 実行オブジェクトファイルを明示的に指定する場合

  $ gcc -o addition addition.c

- 例3. ライブラリを指定する場合(openMP並列)

  $ g++ thred.c –pthread

- 例4. MPI とOpenMPを使用する「sample.f」の場合

  $ mpifort –openmp -o sample sample.f

ライブラリのリンク

- リンク時にオプションでライブラリ名を指定
  → アーカイブファイルを検索してリンク
- lib***.a というライブラリをリンクする
  → -l*** とオプションの直後に記述

- 例1. Fortran,Cプログラムに「libm.a」をリンクする場合
  $ ifort sample.f –lm
  $ icc sample.c –lm
- 例2. 特定の場所にあるlibblas.aをリンクする場合
  $ ifort sample.f –L~/BLAS/lib –lblas
- 例3. 数値演算ライブラリをリンクする場合
  $ module load intel
  $ ifort -o sample -fast sample.f wclock.o -lmkl_blacs_lp64
  -L$MKLROOT/lib/intel64/lib/intel64

コンパイラのマニュアル

- ユーザーズマニュアルの利用方法
  - INTEL https://www.xlsoft.com/jp/products/intel/tech/documents.html?tab=1#doc-psxe
  - PGI http://www.softek.co.jp/SPG/Pgi/document.html

プログラムの実行環境

- JOBの投入にあたっての注意点
  - 資源タイプf_node以外は複数のユーザで共用
  (CPU、メモリ等は資源タイプ毎で指定されるが、スクラッチ領域は共有)

  - グループディスクの活用
  /gs/hs0/グループ名　もしくは　/gs/hs1/グループ名

  /homeディレクトリは25GBのみ

プログラムの実行例

- 例1. Fortranプログラム「sample」を実行する場合

```
$ ./sample
```

- 例2. MPIの実行ファイル「sample」を2並列で実行する場合

```
$ module load openmpi
$ mpirun -np 2 ./sample
```

- 例3. OpenMPの実行ファイル「sample」を2並列で実行する場合

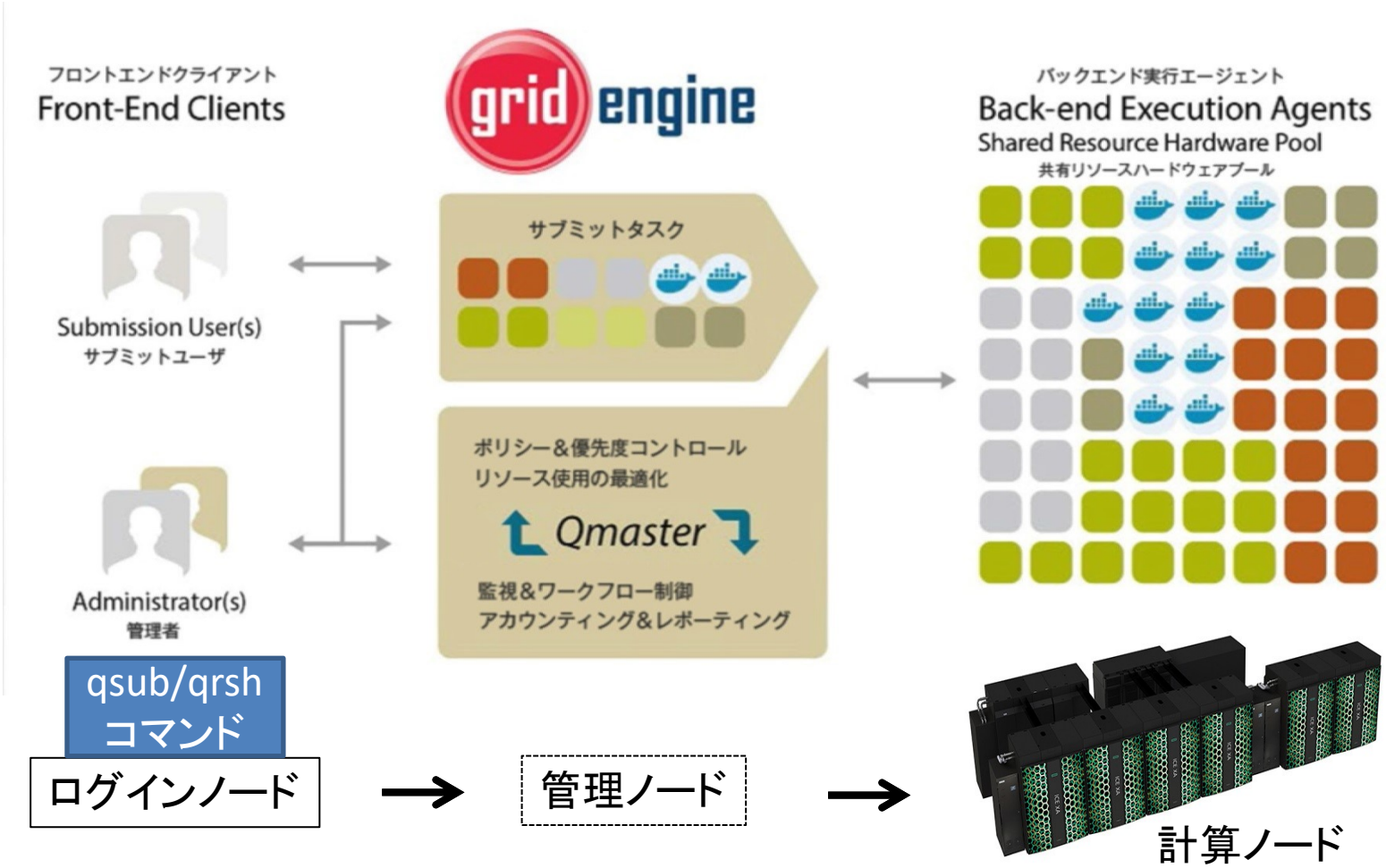
```
$ export OMP_NUM_THREADS=2
$ ./sample
```

- 例4. pthreadの実行ファイル「sample」を2並列で実行する場合

```
$ export NCPUS=2
$ ./sample
```

バッチジョブ

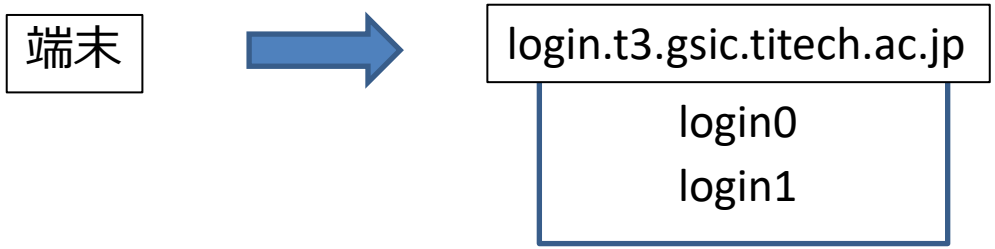
- ジョブスケジューラにUGEが導入されています



計算ノード利用イメージ (1)

- ログイン(SSH鍵認証のみ)
  - 研究室内PCなどより，sshコマンドを実行．負荷分散を考慮し，自動的にログインノードのいずれかにログインされる．

「TSUBAME3.0ポータル利用説明書」参照



負荷分散によりloginノードのどちらかにログイン

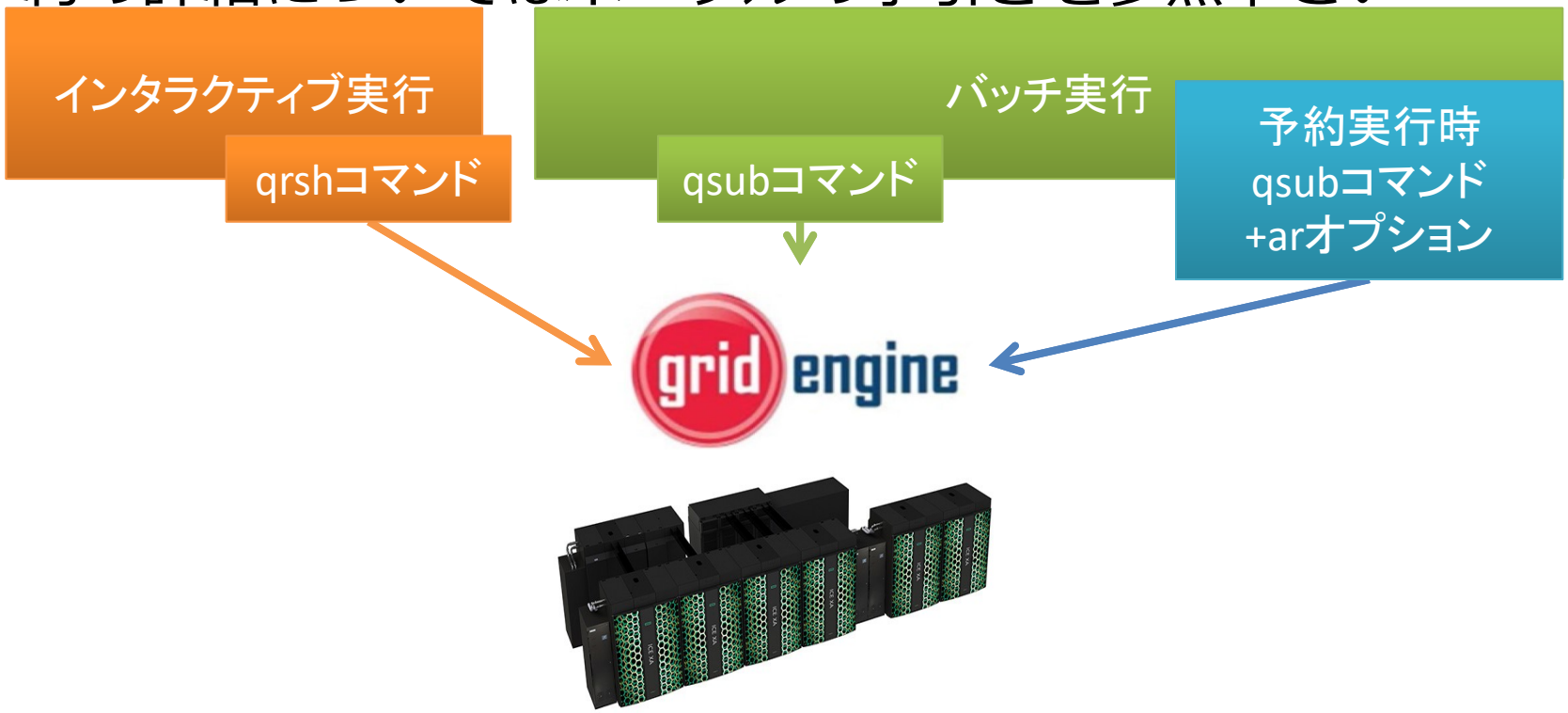
計算ノード利用イメージ (2)

ジョブは540ノードある計算ノードのいずれかで実行される

ユーザによるノード指定は基本的に出来ない

インタラクティブ実行かバッチ実行(通常/予約)

予約の詳細についてはポータルの手引きを参照下さい



資源タイプの構成

- 6つの資源タイプが存在し、SSH接続が可能なのはf_nodeのみ
- 同一ジョブで資源タイプを組み合わせて利用することは不可
- 同じ資源タイプは最大72個、24時間確保可能(例外:予約)

資源タイプ	資源タイプ名	使用物理CPUコア数	メモリ (GB)	GPU 数
F	f_node	28	240	4
H	h_node	14	120	2
Q	q_node	7	60	1
C1	s_core	1	7.5	0
C4	q_core	4	30	0
G1	s_gpu	2	15	1

- コマンドオプションとしては-lで資源タイプ名を指定する

計算ノード利用の制限事項

- 制限事項
  - 1ジョブの利用可能時間
    - 最大24時間(予約は条件により1週間程度利用可能)
  - 利用可能ノード数
    - 一人あたり最大72ノード(2016スロット=物理CPUコア)
  - 実行可能なジョブ数
    - 一人あたり最大150ジョブ

たとえば、#$ -l f_node=72の設定のジョブが実行されている場合や#$ -l s_core=1の設定のジョブが、150個実行されている場合は、自分の他のジョブは「待ち」となり実行されません。

詳細は各種制限値一覧をご確認ください

プログラムの実行（シェルの作成）

- バッチで実行するためのジョブスクリプトの作成

詳細はTSUBAME3.0利用の手引きを参照

```
#!/bin/bash                 <-シェバン、スクリプトのインタプリタ指定
#$ -cwd                     <-カレントドライブで実行するためのバッチジョブスケジューラー用オプション
#$ -N unix_test_job         <-ジョブ名を指定
#$ -l s_core=1              <-資源タイプを指定、s_core(1cpu)を1つ
#$ -l h_rt=0:10:0           <-計算ノードを確保する時間を指定、10分

. /etc/profile.d/modules.sh

echo "this host is" `hostname` "."
```

プログラムの実行（バッチ実行）

- qsub -l 資源名 -l 確保時間 -g グループ ジョブスクリプト

詳細はTSUBAME3.0利用の手引きを参照

GSIC@t3support2:~

File Edit View Help

```
[GSICUSER@t3support2 ~]$ qsub -g グループ sample.sh

#お試し実行の場合は10分、2ノード以内であることを確認し、-gオプション無しで実行する
[GSICUSER@t3support2 ~]$ qsub sample.sh

#資源タイプf_nodeでインタラクティブ実行した場合
[GSICUSER@t3support2 ~]$ qrsh -g GSIC -l f_node=1 -l h_rt=8:0:0
```

バッチキューの確認

- ジョブが投入されていることを確認

GSIC@t3support2:~

File Edit View Help

```
GSICUSER00@login0:~> qstat
job-ID     prior   name       user         state submit/start at     queue    jclass slots ja-task-ID
-----------------------------------------------------------------------------------------------------
93501      0.55500 PDF        GSICUSER00   r     10/04/2017 07:39:58 all.q@r2i4n6    56
93578      0.55500 QRLOGIN    GSICUSER00   r     10/04/2017 11:39:58 all.q@r6i3n2    28
```

- ジョブを消去

GSIC@t3support2:~

File Edit View Help

```
GSICUSER00@login0:~> qdel 93578
GSICUSER00@login0:~> qstat

GSICUSER00@login0:~>
```

プログラムの実行サイズ

- 実行時のメモリサイズの調べ方
- ※topの場合は確認次第qキーを押してtopを停止する。

GSIC@t3support2:~

File Edit View Help

```
GSICUSER00@r6i3n2:~> ps aux
USER              PID %CPU %MEM    VSZ   RSS    TTY   STAT START   TIME COMMAND
GSICUSER00     354797 13.2  0.1 52321684 490304 pts/1 Rl   13:19   0:02 pmemd.cuda.MPI
GSICUSER00     354798 13.2  0.1 52321348 487148 pts/1 Rl   13:19   0:02 pmemd.cuda.MPI

GSICUSER00@r6i3n2:~> top
 PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
354797 hpe_use+  20   0 49.897g 456836 408116 R 14.286 0.173   0:02.09 pmemd.cuda.MPI
354798 hpe_use+  20   0 49.897g 453680 407172 R 14.286 0.172   0:02.08 pmemd.cuda.MPI
```

プログラムの強制終了

- 実行中のジョブのプロセス番号の確認

  例. psコマンドによる確認/ Topコマンドによる確認

GSIC@t3support2:~

File Edit View Help

```
GSICUSER00@r6i3n2:~> ps aux | grep GSICUSER00
354797 13.2  0.1 52321684 490304 pts/1 Rl   13:19   0:02 pmemd.cuda.MPI -O -i input -p top -o test
354798 13.2  0.1 52321348 487148 pts/1 Rl   13:19   0:02 pmemd.cuda.MPI -O -i input -p top -o test

GSICUSER00@r6i3n2:~> top
354797 hpe_use+  20   0 49.897g 456836 408116 R 14.286 0.173   0:02.09 pmemd.cuda.MPI
354798 hpe_use+  20   0 49.897g 453680 407172 R 14.286 0.172   0:02.08 pmemd.cuda.MPI
```

- Kill コマンドの実行

GSIC@t3support2:~

File Edit View Help

```
GSICUSER00@r6i3n2:~> kill 354797 354798

# 上記コマンドで消えない場合
GSICUSER00@r6i3n2:~> kill -9 354797 354798
```

処理情報収集コマンド(2)

- 実行jobを表示(topコマンド)

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ top
Tasks: 784 total,   1 running, 781 sleeping,   2 stopped,   0 zombie
%Cpu(s):  0.1 us,  0.1 sy,  0.0 ni, 99.9 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem:  26377451+total, 14203016 used, 24957150+free,     3352 buffers
KiB Swap:        0 total,        0 used,        0 free. 10473748 cached Mem
   PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
225488 root      20   0  371260  20796   4816 S  2.990 0.008 118:58.30 nv-hostengine
  1111 root      20   0  110076  66204  61604 S  0.332 0.025   0:19.26 systemd-journal
```

- 実行jobを表示(psコマンド)

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ ps aux

[GSIC@t3support2 ~]$ ps aux | grep
ps aux | grep 1111
root        1111  0.0  0.0 110076 66224 ?        Ss   Sep28   0:19 /usr/lib/systemd/systemd-journald
GSIC      352412  0.0  0.0  10540  1640 pts/0    S+   12:29   0:00 grep --color=auto 1111
```

利用環境

- シェル

シェル

- コマンドを実行する事が出来る環境をシェルと呼ぶ
- 環境は、ユーザの好みに合わせて変更できる
- TSUBAME3でサポートしている(chsh可能な)シェルは以下のとおり
  - /bin/bash
  - /bin/csh
  - /bin/sh
  - /bin/ksh
  - /bin/tcsh
  - /bin/zsh
- 下記のようにフルパスで変更したいシェルを指定してください
  - $ chsh /bin/tcsh

変更の反映は5分程度かかります

環境

- ログインシェルはbash(/etc/passwdファイルを編集、または chsh)
- 起動時に確認する設定ファイルの順番

  /etc/profileおよび/etc/bashrc
  ~/.bash_profile
  ~/.bash_login(~/.bash_profileがなければ)
- 対話ログインは、~/.bashrcが読み込まれる
- 個人設定ファイル~/.bashrcの編集
  - PATH="/usr/games:$PATH“
  - export PATH
- 編集後に有効にするには再ログインか source .bashrcとする
- aliasもここに書いておくとよい。
- 変更する場合は、良く確認してから更新してください
- また、新しいターミナルで必ず確認してください

変数

- シェル変数
  - プロセス内でのみ有効な変数
- 環境変数
  - プロセス内に限らず子プロセスにも渡る変数

HISTSIZEシェル変数

- History機能
  - 今までに使用していたコマンド履歴を決められた数だけ保存。
  - コマンド実行時に有効に再利用できるようにする。
  - 保存したいコマンド履歴の数は、任意の数を選べる。
  - コマンド履歴は、この機能を有効にしてから保持。
- 設定例
  - $ export HISTSIZE=600 ⇒ コマンド履歴サイズを600にする
  - $ history 3 ⇒ ヒストリリストを3件表示する

```
534 cd
535 ls
536 history 3
```

PATH変数

- コマンド等の検索場所を設定
- 独自に作成したコマンド、プログラム等の保存場所を指定

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 Program]$ ls
a.out
[GSIC@t3support2 Program]$ a.out
If 'a.out' is not a typo you can use command-not-found to lookup the package that
contains it, like this:
    cnf a.out
[GSIC@t3support2 Program]$ PATH=$PATH:.
[GSIC@t3support2 Program]$ a.out
14digit@M_PI=3.14159265358979
[GSIC@t3support2 Program]$
```

※ .bashrc 等に記述しておけば毎回有効になる

TCP/IP

- Transmission Control Protocol,Internet Protocol
- インターネットやイントラネットで標準的に使われるプロトコル。
  - 米国防総省が、核攻撃で部分的に破壊されても全体が停止することのないコンピュータネットワークを開発する過程で生まれた。
- プロトコルとは
  - ネットワークを介してコンピュータ同士が通信を行なう上で、相互に決められた約束事の集合。
  - 通信手順、通信規約などと呼ばれることもある。

- TCPは通信相手を確認してからデータを送受信し、信頼性が高く、IPはデータをパケット分割し高速伝送するプロトコル

Linux上での代表的なサービス

- ファイル転送
- リモートログイン (p.17参照)
- X-Window
- 電子メール --- (以下の詳細は省略) ---
- Webクライアント
- NIS (Network Information Service)
- DNS (Domain Name System)
- NFS (Network File System)

ファイル転送

- ネットワークで接続されたシステムをホストと呼ぶ。
- このホスト間では、簡単に他のホストのファイルをコピーできる。
- この機能を使用するためのコマンド
  - ftp,rcp,rsync,sftp,scp
- 本学ではrsync,sftp,scpを導入

rsync/sftp/scpコマンドの例

GSIC@t3support2:~

File Edit View Help

```
#rsync
[GSIC@t3support2 ~]$ rsync -av --progress -e "ssh -i .ssh/id_ecdsa -l GSICUSER00"
login.t3.gsic.titech.ac.jp:/gs/hs0/soudan/UNIX/testfile ./
receiving incremental file list
testfile
         990 100%  966.80kB/s    0:00:00 (xfer#1, to-check=0/1)

sent 42 bytes  received 1078 bytes  2240.00 bytes/sec
total size is 990  speedup is 0.88
[GSIC@t3support2 ~]$

#sftp
[GSIC@t3support2 ~]$ sftp -i ~/.ssh/id_ecdsa GSICUSER00@login.t3.gsic.titech.ac.jp
Connected to login.t3.gsic.titech.ac.jp.
sftp> get /gs/hs0/soudan/UNIX/testfile
Fetching /gs/hs0/soudan/UNIX/testfile to testfile
/gs/hs0/soudan/UNIX/testfile
sftp> exit

#scp
[GSIC@t3support2 ~]$ scp -i ~/.ssh/id_ecdsa
GSICUSER00@login.t3.gsic.titech.ac.jp:/gs/hs0/soudan/UNIX/testfile .
testfile
```

sshコマンドの例

- ログイン名を指定してのログインGSIC→GSCIUSER00

GSIC@t3support2:~

File Edit View Help

```
[GSIC@t3support2 ~]$ ssh login.t3.gsic.titech.ac.jp -l GSICUSER00 -i ~/.ssh/id_ecdsa
Last login: Tue Oct  3 09:26:54 2017 from 131.112.3.100
GSICUSER00@login0:~>
GSICUSER00@login0:~> top
GSICUSER00@login0:~> exit
```

- 使用マシンのアカウントのログイン名がTSUBAMEと同じ場合のログイン
  GSCIUSER→GSCIUSER

GSIC@t3support2:~

File Edit View Help

```
[GSICUSER00@t3support2 ~]$ ssh login.t3.gsic.titech.ac.jp -i ~/.ssh/id_ecdsa
Last login: Tue Oct  4 09:26:54 2017 from 131.112.3.100
GSICUSER00@login0:~>
GSICUSER00@login0:~> top
GSICUSER00@login0:~> exit
```

X-Window (X)システムの特徴

- Linuxをはじめ多くのOSで使用
- GUIアプリケーションを構築
- X、X11などの別名あり
- ハードウェアに依存しない
  - Linux/Macintoshは標準装備
  - Windows用のソフトも多数
    - MobaXterm
    - PuTTy・Teraterm・RLogin + VcXsrv/X-ming
    - cygwin

クライアント/サーバ

- Xアプリケーションの動作
  → クライアントプログラム、サーバプログラム

- Xプロトコルという独自の通信手順
  → クライアント/サーバ間のデータのやり取り
    - サーバ側（表示される側）
      どのシステムからの表示要求を許すかを定義
      % xhost +クライアントホスト名

    - クライアント側（表示させる側）
      DISPLAY環境変数を指定
      % export DISPLAY=サーバホスト名

TSUBAME上でXを利用する

- 以下の手順にてXが利用可能です。
  - 別ターミナルを利用する方法

    $ qrsh -l f_node=1 -l 確保時間 -g TSUBAME3グループ

    確保時間はhh:mm:ss表記でも秒表記でもOK(24:00:00 や 600)

    例:

    qrsh -l f_node=1,h_rt=600 -g tga-hpe_group

    ユーザ名@**割り当てられた計算ノード**

    *別ターミナルを立ち上げ

    ssh login.t3.gsic.titech.ac.jp -YC

    ssh **割り当てられた計算ノード** -YC

  - qrshオプションで実行する方法

    $ qrsh –g TSUBAME3グループ -l *s_core=1*,h_rt=確保時間