Introdução Computação

Comandos Linux II

Professor Dr. Anderson Santos
santosardr@facom.ufu.br

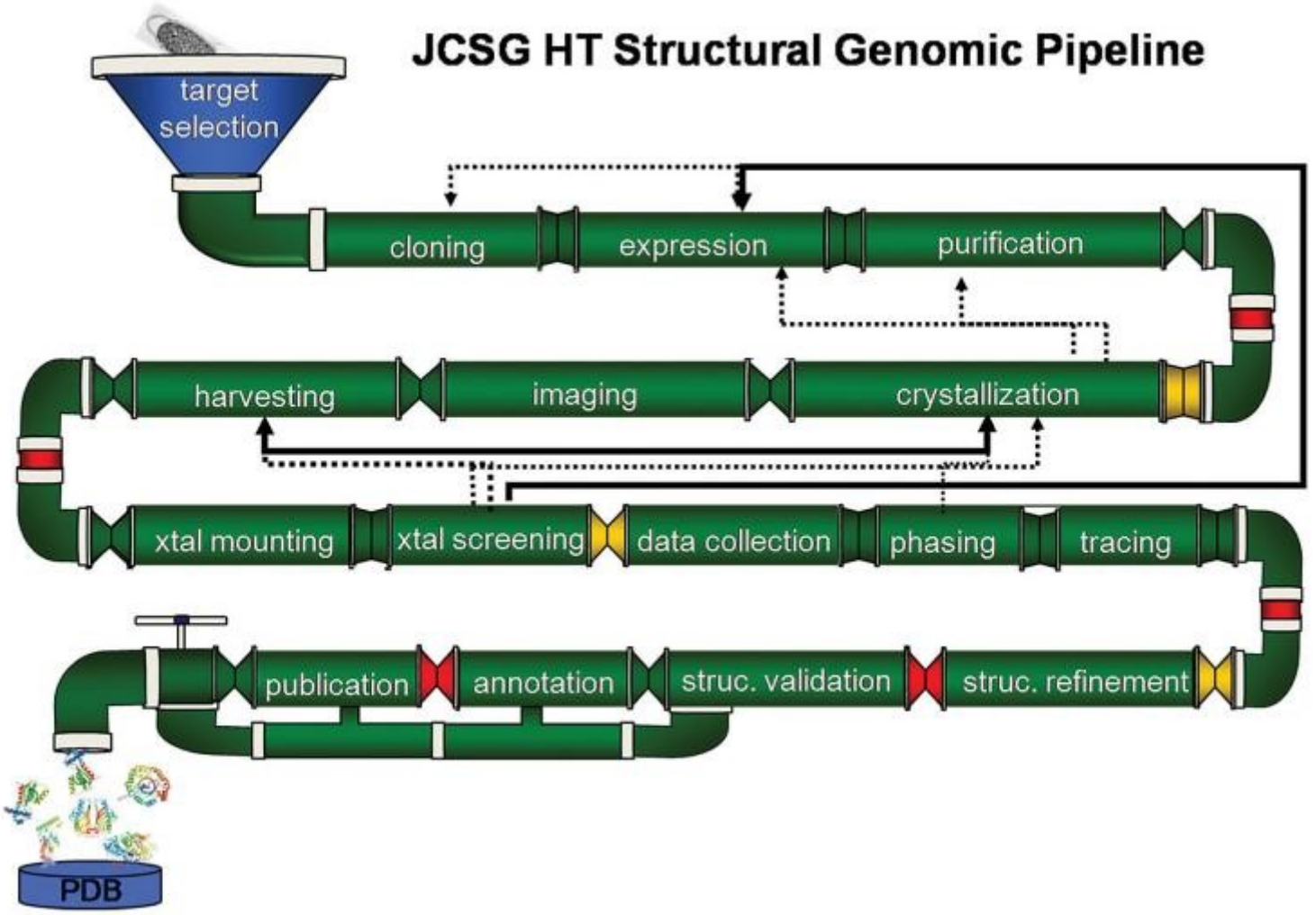
Comandos básicos

- Terminal
- man
- pwd
- ls
- cd
- clear
- alias
- ping
- wget
- tar xvf
- cat
- more & less
- gedit
- head
- tail
- cp
- mv
- rm & rm -rf
- mkdir
- rmdir
- which
- history
- exit

Comandos não tão básicos

- Pipeline
- grep
- find
- sed
- Expressões regulares:
  - em conjunto com grep e sed

Pipeline


JCSG HT Structural Genomic Pipeline
target selection
cloning
expression
purification
harvesting
imaging
crystallization
xtal mounting
xtal screening
data collection
phasing
tracing
publication
annotation
struc. validation
struc. refinement
PDB

Pipeline

- Pode-se criar pipelines com comandos simples no Linux
- Considere que cada comando é um pedaço de 'cano' de uma 'linha de canos' (*pipeline*):
  - ls = lista os arquivos de um diretório
  - wc = conta a quantidade de linhas, palavras e bytes de uma entrada

Pipeline

- Pode-se criar pipelines com comandos simples no Linux
- Considere que cada comando é um pedaço de 'cano' de uma 'linha de canos' (*pipeline*):
  - ls = lista os arquivos de um diretório
  - wc = conta a quantidade de linhas, palavras e bytes de uma entrada
- O símbolo conhecido como pipe '|' conecta a saída de um pedaço da linha à entrada da extremidade da linha:
  - ls | wc
  - Cada nome de arquivo retornado conta como uma linha

Pipeline

- Uma grande quantidade de comandos pode ser colocada em um pipeline:
  - cat $1.source | sed "s/>\([[:alpha:]]\+\)\([[:digit:]]\+\)/HEADERBEGIN\1\2HEADEREND/g" | tr '0-9' 'A-J' | tr '[a-z]' '[A-Z]' | tr -d '[:space:]' | sed "s/[[:punct:]]//g" | sed "s/HEADERBEGIN/#>/g" | sed "s/HEADEREND/#/g" | tr '#' '\n' > $1
  - O comando acima evitou a escrita de um código na linguagem C para diminuir o tamanho de um cabeçalho fasta, substituir números por letras, remover espaços em branco e sinais de pontuação de um arquivo texto que necessitava ser processado sem estes caracteres.

Pipeline

- Deve ser respeitada a compatibilidade de saída versus entrada de dados entre dois comandos que se deseje conectar em um pipeline
  - Um comando que retorne um arquivo junto com outro que manipule este arquivo;
  - O conteúdo de um arquivo junto com a contagem pela ocorrência de certas palavras esperadas
- Programas criados por você ou por terceiros também podem ser incluídos em um pipeline

grep

- Busca por padrões de formação de texto;
- Padrões construídos por texto exato ou por expressões regulares;
- O seu manual é uma oportunidade para aprender sobre expressões regulares;
- Identifica trechos de um texto que atendam ao padrão de construção fornecido;
- Opções para busca “invertida”, por exemplo.

grep

- Exemplos de grep:
  - Um determinado motivo proteico conservado AKAP foi descrito na literatura como importante para a instauração de uma infeção bacteriana. As proteínas do seu organismo de estudo possuem este motivo?
    - grep AKAP proteinas/*
  - Conte a quantidade de proteínas que existe no arquivo mt2.fasta.
    - grep -c '>' mt2.fasta

find

- Um dos programas mais úteis do linux;
- Ainda se fosse limitado a buscar arquivos seria muito interessante devido a:
  - grande reportório de opções de busca;
  - buscar pastas/arquivos filhas de pasta de início da busca;
- Vai além pela possibilidade de manipular livremente os arquivos encontrados em uma pesquisa qualquer;

find

- Algumas opções de busca:
  - Tipo de arquivo;
  - Tipo de permissão de acesso;
  - Tamanho do arquivo (>, >=, =, <=, <);
  - Data de criação;
  - Data de modificação;
  - Data de acesso;
  - entre outros ...

find

- Um parâmetro que faz toda a diferença:
  - exec
- Permite executar qualquer comando do sistema operacional ou programas de terceiros;
- Os comandos do OS ou programas executados podem receber os parâmetros adequados de execução;

find

- Exemplos de find:
  - A partir da pasta corrente, encontre os arquivos/diretórios filhos da pasta corrente que foram modificados a mais de dois dias e depois a menos de um dia;
    - find . -mtime +2
    - find . -mtime -1

find

- Exemplos de find:
  - A partir da pasta corrente, encontre os arquivos/diretórios filhos que possuem um tamanho maior que 500 kilobytes
    - find . -size +500k
  - Para arquivos nestas condições, liste as propriedades do arquivo;
    - find . -size +500k -ls
  - Encontre todos os arquivos cujo nome termine com pdf
    - find . -name '*.pdf' -ls
    - Neste exemplo também poderíamos ter utilizado o *ls* inclusive para buscar nos arquivos/diretórios filhos.

sed

- É um editor de fluxo de texto que funciona na linha de comando;
- Muito conhecido por sua habilidade de permitir editar arquivos sem a necessidade de editá-los manualmente linha a linha;
- sed -i “s/*procureisto*/*troqueporaquilo*/g” file
- Também utiliza expressões regulares, o que aumenta tremendamente seu potencial de edição;

sed

- sed -i “s/*procureisto*/*troqueporaquilo*/g” file
- Exemplos:
  - sed -i “s/ pld / PLD /g” experimentos/*.txt
  - sed -i “s/ > / maior /g” experimentos/*.txt

Expressões Regulares

- Uma padrão que descreve uma cadeia de caracteres;
- Similar a expressões aritméticas;
- O bloco fundamental das Expressões Regulares (ER's) são as ER's que permitem encontrar apenas um caractere:
  - *grep 'a' file.txt*
  - grep a file.txt
  - grep '>' file.fasta

Expressões Regulares

- O ponto '.' é um símbolo utilizado em ER's para significar qualquer caractere, dígito ou sinal de pontuação;
  - *grep a.a file.txt* retorna palavras como:
    - ata
    - asa
    - aba
    - ama
  - *grep at. file.txt* retorna palavras como:
    - ato
    - ata
    - até

Expressões regulares

- Uma expressão entre colchetes [ ] (*brackets*) é denominada uma lista de possibilidades para buscas;
- Quaisquer caracteres dentro da lista podem ser encontrados:
  - grep a[tm]a file.txt retorna:
    - ata
    - ama
  - Mas não retorna outras como:
    - asa
    - aba

Expressões regulares

- Se uma expressão entre colchetes [ ] (*brackets*) possui o sinal de '^' (acento circunflexo) em seu início, então ela funciona como uma lista invertida: somente os carecteres diferentes dos listados após o '^' serão exibidos;
  - grep a[^sb]a file.txt retorna:
    - ata
    - ama
  - Mas não retorna outras como:
    - asa
    - aba

Expressões regulares

- A expressão regular [0123456789] retorna qualquer digito;
  - grep ATA[0123456789][0123456789] file.txt retorna:
    - ATA01
    - ATA22
    - ATA00
- Também pode ser escrita como [0-9]:
  - grep ATA[0-9][0-9] file.txt

Expressões regulares

- O simbolo do operador menos '-' define um intervalo de quaisquer caracteres ou dígitos;
  - grep ATA[0-3][0-9] file.txt retorna:
    - ATA04
    - ATA39
    - ATA25
  - Mas não retorna:
    - ATA44
    - ATA91
    - ATA73
  - Também funciona com letras: [a-d], [A-Z], [x-w] ...

Expressões regulares

- Algumas classes de caracteres possuem identificadores predefinidos sob o formato [:nome_classe:], por exemplo:
  - [:alnum:] → [0-9a-zA-Z]
  - [:alpha:] → [a-zA-Z]
  - [:digit:] → [0-9]
  - [:punct:] → [,.;: *e outros*]
  - [:lower:] → [a-z]

Expressões regulares

- Alguns símbolos são considerados meta caracteres por possuírem significado especial em uma expressão regular, com por exemplo:
  - '[' e seu par ']'
  - '^'
  - '-'
  - '{' e seu par '}'
- Alternativas:
  - Incluir antes/depois da lista de colchetes;
  - Preceder por um caractere '\' que significa: busque pelo valor literal de um símbolo;

Expressões regulares

- Os operadores de repetição permitem que quaisquer padrões sejam encontradas sob quaisquer quantidades. São dispostos logo após a ER que se quer buscar pela repetição:
  - ATA[0-9]<operador de repetição>
- Atuam sempre na ER que estiver imediatamente à esquerda do operador de repetição;

Expressões regulares

- Os operadores de repetição:
  - ? → ER é opcional, podendo ocorrer só uma vez;
  - * → ER pode ocorrer zero ou mais vezes;
  - + → ER pode ocorrer uma ou mais vezes;
  - {n} → ER ocorre exatamente n vezes;
  - {n,} → ER ocorre n vezes ou mais;
  - {n,m} → ER ocorre pelo menos n vezes, mas não mais do que m vezes;

Expressões regulares

- Por exemplo:
  - ATA[0-9][0-9] encontra parte dos textos encontrados por:
    i. ATA[0-9]{2}
    ii. ATA[0-9]*
    iii. ATA[0-9]+
    iv. ATA[0-9]{2,}
  - Perceba aqui que apenas a ER i. é idêntica ao exemplo, as demais podem retornar quantidades muito mais diversificadas de textos, mas incluíndo sempre a possibilidade de retornar o exemplo.

- Os operadores de repetição:
  - ? → só uma vez;
  - * → zero ou mais vezes;
  - + → uma ou mais vezes;
  - {n} → n vezes;
  - {n,} → n vezes ou mais;
  - {n,m} → ocorre ≥ n e ≤ m vezes;

Expressões regulares

- Podem ser concatenadas:
  - [[a-z]{4}[0-9]+[A-Z]?]
  - [[:lower:]{4}[:digit:]+[:upper:]?]
- Podem ser alternadas:
  - [[a-z]{4}[0-9]+[A-Z]?] | [[:lower:]{4}[:digit:]+[:upper:]?]
- Alternação < Concatenação < Repetição;
- Podem ser separadas em grupos de parêntesis e chamadas pelo seu índice numérico sem a necessidade de repetição da ER;

Expressões regulares

- Podem ser separadas em grupos de parêntesis e chamadas pelo seu índice numérico sem a necessidade de repetição da ER;
  - [a-z]{4}[0-9]+[a-z]{4}
  - ([a-z]{4})[0-9]+\1

Expressões regulares

- ER's podem e são frequentemente utilizadas por outros programas, como por exemplo, o *sed*;
- ER's aliadas ao *sed* criam possibilidades ilimitadas para edição de arquivos em lotes (*batch processing*);
- Quando utilizadas no contexto de outros programas, como o *sed*, alguns símbolos podem necessitar de caracteres especiais:
  - \([a-z]{4}\)[0-9]\+\1
  - Caso não funcione como esperado, uma possibilidade é a necessidade de preceder com *backslash* (somente consultado o manual para saber).

Expressões regulares

- ER's podem e são frequentemente utilizadas por outros programas, como por exemplo, o programa *sed:*
  - Para que os arquivos com as proteínas de uma bactéria sejam reconhecidas por um determinado programa eles precisam incorporar no início do nome o prefixo Mt que representa a abreviatura do organismo. Modifique o cabeçalho de todos os arquivos para incluir este prefixo:
  - Lembre-se são 3.988 arquivos ...

Expressões regulares

- ER's podem e são frequentemente utilizadas por outros programas, como por exemplo, o programa *sed:*
  - Para que os arquivos com as proteínas de uma bactéria sejam reconhecidas por um determinado programa eles precisam incorporar no início do nome o prefixo Mt que representa a abreviatura do organismo. Modifique o cabeçalho de todos os arquivos para incluir este prefixo:
  - Lembre-se são 3.988 arquivos ...
  - sed -i "s/>\(Rv[0-9]\+c\?\)/>Mt_\1/g" proteinas/*