Paracel

Paracel BLAST User Manual

Revision 2.0.0 *May 2015*

Copyright

Feedback

Please direct any comments or suggestions about this document to:
support@paracel.com

Publication date and software version
Published 31 May 2015. Based on Paracel BLAST 2.0.0.

Contents

Chapter 1
Introduction

What is Paracel BLAST?

Paracel BLAST was developed to overcome the memory, sequence size, and efficiency limitations of NCBI BLAST. Paracel BLAST software is an enhancement of NCBI BLAST capable of executing searches on multiple non-shared-memory processors simultaneously. It is designed to run on Linux clusters and delivers superior performance for large-scale BLAST searching. Paracel BLAST incorporates a number of optimizations that facilitate searches with large query sequences, large databases, and large numbers of small query sequences. The Paracel BLAST software, when deployed on a multi-processor cluster, can solve biologically relevant large-scale problems faster, more cost-effectively, and more conveniently than competing software/hardware combinations.
The Paracel BLAST software runs variations of BLAST, MegaBLAST and PSI/PHI-BLAST. Efficiency is achieved by queuing search requests, scheduling searches across available processors, and query packing.

New features in Paracel BLAST 2.0.0

Paracel BLAST 2.0.0 introduces a web interface. This consists of two parts:

- A REST interface, which turns Paracel BLAST into a Web service, allowing jobs to be submitted, monitored and controlled using standard web technologies. See Chapter 7 REST Interface on p. 92.

- A Web UI, which allows jobs to be submitted, monitored and controlled using a browser. The Web UI makes use of the REST interface, providing a model for how this interface can be used. See Chapter 6 Web User Interface (Web UI) on p. 84.

The web interface makes Paracel BLAST more accessible:

- Searches can be submitted and controlled from platforms which do not have a Paracel BLAST client, such as iOS and Android.

- Customers can write scripts which directly access the API to control their searches.

Additional changes in version 2.0.0 include:

- NCBI version advanced to 2.2.26.

Search types

Paracel BLAST can perform the following types of searches:

- `blastp` Compares a protein query sequence against protein database sequences. This search type can use plain protein sequences or mass spectroscopy data (Mass Spec BLAST).
- `blastn` Compares a nucleotide query sequence against nucleotide database sequences.
- `blastx` Compares a nucleotide query sequence translated in all six reading frames against protein database sequences.
- `tblastn` Compares a protein query sequence against a translated nucleotide database.
- `tblastx` Compares the six reading frame translations of a nucleotide query sequence against the six reading frame translations of each of the nucleotide database sequences.
- `megablast` Performs a nucleotide sequence alignment search.
- `blastpgp` Performs gapped blastp searches and can be used to perform iterative searches in PSI-BLAST and PHI-BLAST mode.
- `psitblastn` Compares a single protein query sequence against a translated nucleotide database using a Position-Specific Scoring Matrix (PSSM) generated by a previous PSI-BLAST run (using `blastpgp`).

Text and notational conventions

The following text conventions are observed in this manual:

- Square brackets "[]" in the usage denote parameters that are optional.
- Commands and their arguments should be specified on a single line, though usages in this manual are sometimes shown on several lines for typographic reasons. A backslash "\" denotes the continuation of a command line.
- Terms in **`monospace`** are commands, parameters or arguments that should be entered exactly as they appear.
- Terms in *italics* indicate variables. When entering a command, parameter or argument, replace the italicized terms with the appropriate information. Italicized terms are usually enclosed in angle brackets "*<variable_name>*". Do not include the angle brackets in the command line input.
- Multiple databases or a mixture of databases may be specified on a single command line.
- For additional information on a command, type **`pb`** *<command>* **`--help`** .
- Arguments to the Paracel BLAST software are specified as **`--`***<flag>*, **`-`***<flag>* or **`--`***<parameter >*`=`*<value>*. The `blastall`, `formatdb` and `ls` commands support the parameter/value pair format (for example `-A 40`). The parameter and value are separated by a space.
- The pipe symbol '`|`' indicates alternation : only one out of a list of arguments may be selected: `optionA|optionB|optionC`.

Chapter 2
Installation and System Administration

Installation and Upgrading of Paracel BLAST

Note	Installation of Paracel BLAST software should be performed by a System Administrator or by a Paracel Customer Support Representative.

Prerequisites

Before installing Paracel BLAST, make sure you system has the following prerequisites:

- A designated manager node.
- Designated worker nodes.
- Manager and workers must be running either:
  - a recent Linux distribution which supports RPM for package management, or
  - the Rocks operating system.
- Manager and workers must have 64 bit architecture.
- Manager and workers must have at least 1 GB RAM per CPU.
- A shared filesystem, visible to both manager and workers (typically, this is exported from the manager, but that is not a requirement). This filesystem should be at least 100 GB, and it must support “large” files (> 4 GB). NFSv3 supports large files, while NFSv2 does not.
- The Paracel BLAST software.
- A valid license file for Paracel BLAST. If you have purchased Paracel BLAST, send email to support@paracel.com to obtain your license key.
- Root access on all nodes that will run the Paracel BLAST server daemons (pbd and pbwebd).
- It is convenient, but not required, to have ssh as root enabled on all worker nodes.

Upgrade instructions

To upgrade an existing Paracel BLAST installation, follow these instructions.

Access the installation directory

For CD-ROM installations, mount the CD-ROM and then `cd` into it:

```
mount /mnt/cdrom
cd /mnt/cdrom
```

For “tgz” installations, extract the `.tgz` file and `cd` to the directory it creates:

```
tar xzf ParacelBlast-<version>.tgz
cd ParacelBlast-<version>
```

Install the software

On the manager, for 32-bit systems:

```
cd <release>/RPMS
rpm -U pb-*i386.rpm pbd-*i386.rpm
```

and for 64-bit systems:

```
cd <release>/RPMS
rpm -U pb-*x86_64.rpm pbd-*x86_64.rpm
```

On the workers, for 32-bit systems:

```
cd <release>/RPMS
rpm -U pb-*i386.rpm
```

and for 64-bit systems:

```
cd <release>/RPMS
rpm -U pb-*x86_64.rpm
```

For clusters with local disks, install the "pbd-worker" RPM on each worker, but *not* on the manager node. For 32-bit systems:

```
cd <release>/RPMS
rpm -U pbd-worker*i386.rpm
```

and for 64-bit systems:

```
cd <release>/RPMS
rpm -U pbd-worker*x86_64.rpm
```

Restart the Paracel BLAST daemons

If you have set up a password-less root ssh to all the workers, then on manager (as root):

```
service pbd restartall
```

If not, then on manager (as root):

```
service pbd restart
```

and on each worker (as root):

```
service pbd restart
```

New installation instructions

Access the installation directory

For CD-ROM installations, mount the CD-ROM and then `cd` into it:

```
mount /mnt/cdrom
cd /mnt/cdrom
```

For "tgz" installations, extract the `.tgz` file and `cd` to the directory it creates:

```
tar xzf ParacelBlast-<version>.tgz
cd ParacelBlast-<version>
```

Configure the system

To configure the system for the first time:

```
./install
```

Answer the questions as the installation script prompts for them. You will need:

- The installation directory for Paracel BLAST. It must be on the NFS filesystem visible to the

manager and workers.

- The name of the manager node.
- The names of the worker nodes.

Copy the license file

Copy the license file obtained from Paracel to the `paracel/license` subdirectory of your Paracel BLAST installation directory. For example, if the installation directory is `/paracel` (the default), you would copy the license file to:

```
/paracel/paracel/license/pbd.lic
```

Install worker nodes (local disks)

For clusters with local disks, install the "`pbd-worker`" RPM on each worker, but *not* on the manager node. For 32-bit systems:

```
cd <release>/RPMS
rpm -U pbd-worker*i386.rpm
```

and for 64-bit systems:

```
cd <release>/RPMS
rpm -U pbd-worker*x86_64.rpm
```

Install worker nodes (Rocks)

For Rocks systems, you can do the entire install for the cluster nodes with the following two commands. On 32-bit systems (assuming the Paracel BLAST installation directory is `/paracel`):

```
cp pbd-worker*i386.rpm /paracel/paracel
cluster-fork rpm -i /paracel/paracel/pbd-worker*i386.rpm
```

and on 64-bit systems:

```
cp pbd-worker*x86_64.rpm /paracel/paracel
cluster-fork rpm -i /paracel/paracel/pbd-worker*x86_64.rpm
```

Start the Paracel BLAST daemons

If you have set up a password-less root ssh to all the workers, then on manager (as root):

```
service pbd restartall
```

If not, then on manager (as root):

```
service pbd restart
```

and on each worker (as root):

```
service pbd restart
```

For clusters with local disks, install the "pbd-worker" RPM on each worker, but *not* on the manager node:

Uninstall instructions

To uninstall Paracel BLAST, run the uninstall script (`./uninstall`) on the manager node. For systems with local disks on worker nodes, this script should also be run on all worker nodes.

Package Management

For details on package management using the Red Hat Package Manager (RPM), see http://rpm.redhat.com/RPM-HOWTO.

System Administration

Basic commands

Paracel BLAST can perform a variety of system administration tasks including:

- `chgrp` Changes the group ownership of files or directories.
- `chmod` Changes the permissions of a Paracel BLAST Filesystem (PBFS) file.
- `chown` Changes the ownership and, optionally, the group ownership of files and directories.
- `cp` Copies the specified PBFS file or directory.
- `df` Lists mounted filesystems and usage statistics.
- `formatdb` Formats a database to be used for BLAST searching.
- `killjob` Cancels the specified job.
- `ls` Lists the database files in the specified PBFS directory.
- `mkdir` Creates a PBFS directory.
- `mv` Moves or renames the specified PBFS file or directory.
- `reprioritize` Changes job priority.
- `rm` Removes the specified PBFS database file or directory.
- `shutdown` Shuts down the Paracel BLAST daemons.
- `status` Displays status information about the Paracel BLAST system.

Filesystem security

Limited security has been embedded in the handling of the `pbroot` folder. Paracel BLAST creates this folder with write privileges limited to root: no user will be allowed to write/delete directly in the `pbroot` folder. In order for a user to create personal files in the Paracel BLAST filesystem, the system administrator should proceed as he would for a normal Linux account. Then, to create a new folder for the user:

```
pb mkdir <new_user_name>
```

Then:

```
pb chown <new_user_name> <new_user_name>
```

User *<new_user_name>* will own the newly created folder and gain write permission in that folder. With this approach, users will be limited to their own folders when it comes to writing. They will have full access to all the files in the filesystem as long as the read permissions are set properly.

Note	The `root` user is always able to remove any file in the filesystem, no matter what the permissions are.

Job visibility

Users can not get *all* information on jobs running on a Paracel BLAST system. To get information on running jobs:

```
pb status
```

The user will see the list of all jobs, but only information on the jobs he is executing. The superuser, `root`, will always be allowed to look at all the information of all the jobs currently in the queue.

Remote access

While allowing remote access to users, Paracel BLAST will not allow the `root` user to perform any command remotely. This limitation stems from associated security issues. In order to perform such commands, `root` will have to explicitly log on the Paracel BLAST manager and issue those commands locally.

Controlling the Paracel BLAST daemons with `pbd` script

There are three types of daemon:

1. Manager Daemon that runs on the manager node (process name `pbd`).
2. Web Server Daemon that runs on the manager node (process name `pbwebd`).
3. Worker Daemons that run on worker nodes (process name pbd).

The daemons are started and stopped using variations of the `pbd` script. Note that the daemons are automatically spawned when Linux is rebooted. On rare occasions, the System Administrator may have to restart this process.

Before using the `pbd` function, the user must login to the manager or worker as `root` by typing:

```
ssh <hostname> -l root
```

where *hostname* is the name of the machine on which the daemons will be running.

Usage

```
/etc/init.d/pbd startall | restartall | stop
```

Description

The Manager Daemon (`pbd`) accepts requests from Paracel BLAST clients and distributes the processing load between the Worker Daemons (also `pbd`). This involves slicing the searches into search pieces, and distributing the search pieces to the workers. The Worker Daemons then return the results and error messages, if any, to the Manager Daemon. The Manager Daemon then returns the results to the Paracel BLAST client.

The Web Server Daemon (`pbwebd`) acts as a Paracel BLAST client, providing access to the Manager

Daemon via both a Web UI and a REST interface.

The `pbd` Manager Daemon and Web Server Daemon should be run as `root`. If it is run as a non-root user, no other users will be able to run searches or otherwise use the application. Some jobs submitted as `root` may fail with the error:

```
Exception (JOBDIED): Job exited abnormally ...
```

This occurs when a worker daemon attempts to access a filesystem to which it does not have root permission. To avoid this, run the job as a non-root user or export the filesystem with root permissions.

`pbd` daemon commands

`startall`	This command starts the daemons on the manager and the workers.
`restartall`	This command restarts the daemons on the manager and the workers. In order to do this, `restartall` stops any existing daemons and starts a new set of daemons.
`stop`	This command stops the daemons on the manager and all of the workers.

Paracel BLAST License File

Paracel BLAST can be run only with a valid license file. The default location for the license file is `/paracel/paracel/license/pbd.lic`.

Chapter 3
Command Line Usage

Introduction

This chapter contains the usages for the various types of Paracel BLAST searches as well as for the system administration utilities. Paracel BLAST commands are executed in the form

```
pb  <command> [ arguments... ]
```

The commands and arguments are described briefly in the next section and in detail in the pages that follow.

pb

`pb` is the Paracel BLAST client. This is the program used to access all Paracel BLAST functions.

Usage

```
pb  <command> [ <argument> <argument> ... ]
```

The following commands are supported:

`blastall`	`blastpgp`	`chgrp`
`chown`	`cp`	`chmod`
`dbinfo`	`df`	`fastacmd`
`formatdb`	`killjob`	`ls`
`megablast`	`mkdir`	`mv`
`reprioritize`	`rm`	`shutdown`
`status`		

The following options are applicable to most commands:

```
--config=<config_filename>
--dbpart=<number>
--help
--host=<hostname>:<port>
--priority=<priority>
--querypart=<number>
--quiet
--stats
```

Commands

The following are brief command descriptions. These commands are described in greater detail in subsequent sections.

Note	Do not use simple shell commands to manually create, rename, move or remove query or database files in the Paracel Blast Filesystem as the use of these commands can have unexpected side effects. Use only the equivalent `pb` commands for file management.

The commands:

`blastall`	Performs a BLAST search.	p. 20
`blastpgp`	Executes a PSI-BLAST and/or a PHI-BLAST search.	p. 31
`chgrp`	Changes the group ownership of files or directories.	p. 40
`chmod`	Changes the permissions of a PBFS file.	p. 41
`chown`	Changes the ownership and, optionally, the group ownership of files and directories.	p. 41
`cp`	Copies the specified Paracel BLAST File System (PBFS) file or directory.	p. 42
`dbinfo`	Gets information about a specified database.	p. 43
`df`	Lists mounted filesystems and usage statistics.	p. 43
`fastacmd`	Dumps out a formatted database, or part of one, in FASTA format.	p. 44
`formatdb`	Loads and formats a database to be used for BLAST searching. For DNA sequences, the process involves two-bit compression of the database (expressing the four canonical nucleotides in binary format) and creating an index file of BLAST words in the database.	p. 47
`killjob`	Cancels the specified job.	p. 50
`ls`	Lists the files in the specified PBFS directory.	p. 51
`megablast`	Executes a MegaBLAST search.	p. 52
`mkdir`	Creates a PBFS directory.	p. 61
`mv`	Moves or renames the specified PBFS file.	p. 61
`reprioritize`	Changes the priority of the job. For details, see the Glossary term "Job priority" on p. 106.	p. 62
`rm`	Removes the specified PBFS file or directory.	p. 63
`shutdown`	Shuts down the Paracel BLAST software.	p. 63
`status`	Displays status information about the Paracel BLAST system.	p. 64

Common arguments

These arguments apply to most commands.

`--config=`*<config file>*
Specifies a file containing additional arguments.

`--dbpart=`*<number>*
Override the Planner by telling it how many sub-jobs to create by splitting the database(s). The default value, 0, lets the Planner decide. If you specify a number greater than the number of sequences in all databases being searched, one sub-job will be generated for each sequence in each database.

To disable database splitting, set –`dbpart=1`.

Note	To force the search to run as a single job when there are multiple sequences in your query file, you must also set `--querypart=1`, otherwise, `blastpgp` will ignore the dbpart argument and will not do database splitting.

`--help`
Prints help and usage information.

`--host=`*<hostname>* [: *<port>*]
Specifies the *hostname* and *port*. If the `PB_HOST` environment variable is set, this value will be used if `--host` is not specified. Explicitly setting the `--host` argument on the command line will override any value specified by the `PB_HOST` environment variable.

`--priority=`*<integer>*
Sets an *integer* priority for a job. The valid range of values is -99 to 99. The default value is 0. For details, see the Glossary term "Job priority" on p. 106.

`--querypart=`*<number>*
Override the Planner by telling it how many sub-jobs to create by splitting the queryset file. The default value, 0, lets the Planner decide. If you specify a number greater than the number of query sequences in the file, each query sequence will be split into its own sub-job.

To disable queryset splitting, set –`querypart=1`.

Note	To force the search to run as a single job, you must also set `--dbpart=1`.

`megablast` will only do queryset splitting if the `-D` parameter is set to 2 (to generate traditional `blastn` style output). Otherwise, this argument will be ignored.

`--quiet`
Disables the display of status information to `stderr` .

`--stats`
Displays the resource usage statistics to `stderr`. These statistics include the sum of elapsed time of all subjobs, the sum of user (CPU) and system times for all subjobs, and the sum of maximum resident memory for all subjobs.

blastall

This is the main BLAST executable. It runs the most common BLAST programs: `blastp`, `blastn`, `blastx`, `tblastn`, `tblastx` and `psitblastn`.

Usage

```
pb blastall
```

Required arguments:

`-d` *<database_filename>*

`-p blastp|blastn|blastx|tblastn|tblastx|psitblastn`

Optional arguments:

`-A` *<multiple_hits_window_size>*

`-b` *<num_alignments>*

`-D` *<database_genetic_code>*

`-e` *<threshold E value>*

`-E` *<gap_extn_penalty>*

`-f` *<extn_threshold>*

`-F` *<filtering_type>*

`-g T|F`

`-G` *<gap_open_penalty>*

`-i` *<query_filename>*

`-I T|F`

`-J T|F`

`-l` *<gi_list>*

`-m` *<alignment_view_option>*

`-M` *<matrix_filename>*

`-o` *<output_filename>*

`-O` *<seqalign_filename>*

`-P` *<pass_options>*

`-q` *<mismatch_penalty>*

`-Q` *<query_genetic_code>*

`-r` *<match_reward>*

`-R` *<restart_input_filename>*

`-S 1|2|3`

`-T T|F`

`-t` *<max_intron_length>*

`-U T|F`

`-u` *<num_of_alignments>*

`-v` *<num_one_line_descriptions>*

`-W` *<word_size>*

`-w` *<frameshift_penalty>*

`-X` *<gap_x_dropoff>*

`-y` *<dropoff_second_pass>*

`-Y` *<searchspace_effective_size>*

`-z` *<database_effective_length>*

`-Z` *<gap_x_dropoff_final>*

Description

This command performs a BLAST search. The search types are specified on the command line. The search type specified must be consistent with query and database type. Unless otherwise specified, all numerical input values are integers.

Required arguments for `blastall`

Required Argument	Description

`-d`	Sets database name(s) which must be PBFS filename(s). Multiple databases may be specified in quotes, with each database filename separated by a space. For example: "*database1 database2 database3*" would specify to search *database1*, *database2* and *database3*.
	
	As in NCBI BLAST, the user may also set the environment variable `BLASTDB`, so as to specify the path to the database. For example, the following two C-shell command sequences are equivalent:
	
	`setenv BLASTDB ""`
	`pb blastall -d "disk1/database1 disk1/database2"`
	
	and
	
	`setenv BLASTDB "disk1"`
	`pb blastall -d "database1 database2"`
	
	For convenience, the user may specify PB_BLASTDB instead of BLASTDB . This will override anything in BLASTDB , and is primarily intended for users who switch between NCBI BLAST and Paracel BLAST. For example, if the environment variables are set as follows,
	
	`setenv BLASTDB "/home/ncbi/databases"`
	`setenv PB_BLASTDB "disk1/"`
	
	the NCBI `blastall` commands will get their database path from `BLASTDB`, and `pb blastall` commands will get their database path from `PB_BLASTDB`.
	
	Note In contrast to NCBI usage, the `-d` parameter is required. Paracel BLAST does not support the NCBI behavior of "no `-d` means `nr`".

Required Argument	Description	
`-p`	Specifies the search type:	
	`blastn`	Compares a nucleotide query sequence against nucleotide database sequences.
	`blastp`	Compares a protein query sequence against protein database sequences.
	`tblastn`	Compares a protein query sequence against a translated nucleotide database.
	`tblastx`	Compares the six reading frame translations of a nucleotide query sequence against the six reading frame translations of each of the nucleotide database sequences.
	`blastx`	Compares a nucleotide query sequence translated in all six reading frames against protein database sequences.
	`psitblastn`	Compares a single protein query sequence against a translated nucleotide database using a Position-Specific Scoring Matrix (PSSM) generated by a previous PSI-BLAST run (using `blastpgp`). See `psitblastn` on p. 29 for details.

Optional arguments for `blastall`

Optional Argument	Description
`-A`	Specifies the multiple hits window size. Note that this value is zero for single-hit algorithms. The default value is 40.
`-b`	Specifies the number of database sequences for which to show alignments. The default value is 250.
`-D`	Specifies the database genetic code. For `tblastn` and `tblastx` only. The default value is 1 (standard). See TABLE 4: blastall Genetic Codes on p. 77 for a listing of genetic codes.
`-e`	Specifies the Expectation Value (E value). Note that only hits with E values less than the value specified for this parameter will be returned. The default value is 10.0 (real).

Optional Argument	Description
-E	Sets the gap extension penalty. A value of zero invokes the default. For default values, see TABLE 2: “Hidden” Defaults on p. 30. Note that the -G and -E penalties are positive, while the -q penalty is negative. Also see TABLE 3: Matrix-Specific Overrides on p. 31.
-f	Sets the threshold for extending hits. If a value of zero is entered, the default behavior is invoked. For default values, see TABLE 2: “Hidden” Defaults on p. 30. This option is not allowed for blastn.
-F	Specifies whether or not to use the NCBI default filtering to filter the query sequence. The -F argument can take a string as input specifying that SEG should be run with certain values or that other non-standard filters should be used. • Setting -F F turns filtering off. The default value is -F T, which means to use DUST for blastn and SEG for all other search types with the default parameters. • The SEG options can be changed by using: -F "S 10 1.0 1.5" which specifies a window of 10, locut of 1.0 and hicut of 1.5. • Coiled-coiled filtering may be invoked by specifying: -F C. For details and default settings, see the Glossary term “Coiled-coiled filtering”on p. 106. • One may also run both SEG and coiled-coiled filtering together by separating the values with a semicolon, for example -F "C;S". • It is possible to specify that the masking should only be done during the process of building the initial words by starting the filtering command with **m**, e.g.: -F "m S" which specifies that SEG (with default arguments) should be used for masking, but that the masking should only be done when the words are being built. This masking option is available with all filters. • If the -U option (to mask any lower-case sequence in the input FASTA file) is used and one does not wish any other filtering, but does wish to mask when building the lookup tables, then one should specify: -F “m”. This is the only case where **m** should be specified alone. • Filtering by DUST may also be specified by: -F "D". DUST is used for blastn searches only.
-g	Specifies whether to perform a gapped alignment: -g T “gapped alignment” is **on** (default) -g F “gapped alignment” is **off** This parameter is not available for tblastx searches. The default value is T (on).

Optional Argument	Description
`-G`	Specifies the gap opening penalty. Values entered for this parameter should be positive. If a value of zero is entered, the default behavior is invoked. For default values, see TABLE 2: "Hidden" Defaults on p. 30. But see also TABLE 3: Matrix-Specific Overrides on p. 31. Note that the `-G` and `-E` penalties are positive, while the `-q` penalty is negative.
`-i`	Specifies the query file, which must be in FASTA format. If multiple FASTA entries are in a file, all entries will be searched and each will generate a complete BLAST report. If the query is extremely long, it will be chopped into pieces to allow processing to proceed. If not specified, defaults to `stdin`.
`-I`	Specifies whether to show the Genbank Indices (GI's) in the deflines: `-I T` "show GI's" is **on** `-I F` "show GI's" is **off** (default) By default, the GI's are not shown (`F`).
`-J`	Specifies whether to parse and interpret the query deflines: `-J T` "interpret deflines" is **on** `-J F` "interpret deflines" is **off** (default) The default value is `F`: BLAST treats the deflines merely as text. If set to `T`, the query deflines are parsed assuming an NCBI compliant defline format described at ftp://ftp.ncbi.nih.gov/blast/db/README. **Note** This option must be set to `T` if the `-O`, `-m 10`, or `-m 11` ASN output formats are used.
`-l`	Specifies the name of a file containing a list of GenBank Indices (GI's). This option restricts the search of the database to a list of GI's. Any databases used with this feature must be formatted with the `-o` flag set to `T` (see `pb formatdb` on p. 47). The input file for this option can be in text or in binary format. Binary files must be formatted properly using tools available from NCBI. The text file is simply formatted with one GI per line.

Optional Argument	Description
`-m`	Specifies how alignments are presented in the output file: `-m 0` Pairwise (default) `-m 1` Query-Anchored with Identities `-m 2` Query-Anchored without Identities `-m 3` Flat Query-Anchored with Identities `-m 4` Flat Query-Anchored without Identities `-m 5` Query-Anchored without Identities with Blunt Ends `-m 6` Flat Query-Anchored without Identities with Blunt Ends `-m 7` BLAST XML `-m 8` Tabular output without comments `-m 9` Tabular output with comments `-m 10` ASN (text) `-m 11` ASN (binary) The default value is "Pairwise" (`-m 0`). The `blastx` and `tblastx` search types for `pb blastall` support only alignment types `0` (default), `7`, `8`, or `9`. In other words, `-m` values of 1-6 are not allowed for these search types. For details on these alignment types, see Output Formats on p. 67.
`-M`	Specifies the name of the matrix to be used in a protein search. The default matrix is BLOSUM62. Some other matrices included in the Paracel BLAST distribution are: BLOSUM45, BLOSUM80, PAM30 and PAM70. Note that for gapped `blastp`, `blastx` and `tblastn` searches, only these matrices and BLOSUM50, BLOSUM62_20, BLOSUM90 and PAM250 (which are not included in the distribution) are allowed. **Note** Additional matrices may be installed by the system administrator by copying them to the correct place, namely *<pbroot>*`/data`. For ungapped `blastp`, `blastx`, `tblastn` and `tblastn` searches, any installed matrix can be used. Matrix names are case sensitive, so they must be entered in UPPER CASE.
`-o`	Specifies the BLAST report output file. The default is `stdout`.
`-O`	Creates a SeqAlign file with the specified filename. This file must be used with the `-J` parameter set to `T` (parse and interpret the defline). If this option is not specified, this file will not be created.

Optional Argument	Description
-P	The -P parameter takes one of the following values: 0 multiple hits 1-pass (default) 1 single hit 1-pass 2 2-pass The default is 0.
-q	Specifies the penalty for a nucleotide mismatch. Used only when a blastn search is specified. This value must be negative. **Note** The -q penalty is negative, while the -G and -E penalties are positive. Since release 1.8.0, only certain combinations of values for -q and -r are allowed: TABLE 1: Allowed Nucleotide Match Rewards and Mismatch Penalties -q: -1, -2, -3, -4, -5, -7 -r: 1, 1 3, 1 2, 1 3 5, 1 2 4, 2 Default value is -3.
-Q	Specifies the query genetic code to use for blastx and tblastx only. The default value is 1 (standard). See TABLE 4: blastall Genetic Codes on p. 77.
-r	Specifies the reward for a nucleotide match. Used only when a blastn search is specified. This value must be positive. The default value is 1. Since release 1.8.0, only certain combinations of -q and -r are allowed (see TABLE 1: Allowed Nucleotide Match Rewards and Mismatch Penalties on p. 27).
-R	Specifies the input filename for PSI-TBLASTN Restart. When using this option, it is required that the query specified on the command line match exactly the query in the restart file. See psitblastn on p. 29.

Optional Argument	Description
-S	Specifies the query strands to search against the database. Used for blastn, blastx and tblastx search types. The following values are defined: 1 forward complement 2 reverse complement 3 both (default) The default value is 3 (both).
-T	Specifies HTML output. -T T "HTML output" is **on** -T F "HTML output" is **off** (default) This option is valid for all output formats except BLAST XML (-m 7). The default is F (off).
-t	Specifies the maximum length of an intron to be allowed when linking multiple, distinct alignments (blastx, tblastn, psitblastn only). 0 (the default) invokes the default behavior. Negative values disable linking entirely. Positive values prevent linking on introns longer than the value given.
-u	Specifies the number of alignments to report per db subject. A value of 0 sets this flag to its default, meaning that there is no limit on the number of alignments for the unchopped query and 100 alignments for the chopped query. Note that -u 0 means to do default pruning, which generally does not prune except for very long sequences that are chopped while they are searched. In this case, the default is equivalent to -u 100. To disable pruning except for very long sequences, use something like -u 1000000000.
-U	Controls lower case filtering of a FASTA sequence: -U T "lower case filtering" is **on** -U F "lower case filtering" is **off** (default) Residues in low complexity areas are changed from upper to lower case. The default is F (off).
-v	Specifies the number of one-line descriptions desired. The default value is 500.
-W	Specifies the word size. A value of zero invokes default behavior. For default values, see TABLE 2: "Hidden" Defaults on p. 30. Whereas previously blastn supported word sizes only of 7 or greater, this search type now also accepts word sizes less than 7 as well; only word sizes of 2 or 3 are valid for other search types. Increasing the word size decreases the search time and the search sensitivity. Conversely, decreasing the word size increases both search time and sensitivity. The default values balance time and sensitivity.

Optional Argument	Description
-w	Specifies the frame-shift penalty for out-of-frame searches (blastx only).
-X	Specifies the X dropoff value for a gapped alignment in bits. If a value of zero (integer) is entered, the default behavior is invoked. For default values, see TABLE 2: "Hidden" Defaults on p. 30. For details, see the Glossary term "X dropoff value" on p. 109.
-y	Specifies the second pass dropoff for BLAST extensions in bits. Requires a real number. A value of zero (0.0) invokes default behavior. For default values, see TABLE 2: "Hidden" Defaults on p. 30. For non-blastn searches, if gapped extension is chosen, 7.0 will be used for both passes. If ungapped extension is chosen, 7.0 will be used on the first pass and 10.0 on the second pass. For non-blastn searches, if -y is specified < 7.0, the first pass dropoff value is also set to the specified value.
-Y	Specifies the *effective* size of the search space. This is a derived number. This number is provided in the output report, so if the user wants to re-run a search on part of a database and keep the statistics (E-values) unaltered, he would extract this number from the first run. Search space is the product of the number of characters in the database and the number of characters in the query. The *effective* length is the number of characters reduced by a statistically calculated adjustment value. If databases are split, the *effective* values used to compute E-values must be provided by either the -Y or -z parameter. The default values for these parameters are zero, meaning the *actual* size of the search space will be used. *Effective* search space may differ from *actual* search space when the database is split among several workers: each worker will operate on part of the database, but calculate E-values as if it had received the entire database.
-z	Specifies the *effective* length of the database. If the database is split, this will be the total length of the original (unsplit) database. The default value is zero, meaning the real length of the database will be used. For the distinction between *effective* and *real* length, see the preceding discussion, in the description of -Y.
-Z	Specifies the X dropoff value for final gapped alignment in bits. A value of zero (integer) invokes the default behavior. For default values, see TABLE 2: "Hidden" Defaults on p. 30. This parameter does not apply to tblastx, which does not support gapped alignment.

psitblastn

pb blastall -p psitblastn compares a single protein query sequence against a translated nucleotide database using a Position-Specific Scoring Matrix (PSSM) generated by a previous psiblast run.

Performing psitblastn involves two steps. In the first, the user should run PSI-Blast (blastpgp) to

create and save a PSSM given a single protein query sequence and a protein database and using the `-j` option with a value > 1 to set multi-pass to “on”:

```
pb blastpgp -i <protein_sequence> -j 2 \
     -C protein_sequence.pssm -F F -d <protein_database>
```

where the generated PSSM for the given protein query sequence is saved in *`protein_sequence.pssm`* (substitute with your own file name).

In the second step, the user runs `psitblastn` given the PSSM that was generated in the first step and the same protein query sequence that was used to generate the PSSM against a DNA database as follows:

```
pb blastall -p psitblastn -i <protein_sequence> \
     -R protein_sequence.pssm -F F -d <dna_database>
```

The `-R` option is required when using `pb blastall -p psitblastn`. All other options that apply when using `pb blastall -p tblastn` also apply when using `pb blastall -p psitblastn`, but the following restrictions should be observed:

- The query sequence used in the `psitblastn` search must be the same as the one used to generate the PSSM.

Note that the PSSM can be constructed using one database and then used to search a different database. Even if the two database names are the same, `pb blastpgp` uses the protein version while `pb blastall -p psitblastn` uses the DNA version.

As with NCBI Blast, performing PSI-Blast (`blastpgp`) or `psitblastn` searches with checkpoint recovery works only for queries that consist of a single protein sequence. It is an error if the query contains more than one sequence.

For convenience of reference, the “hidden” defaults for a number of frequently used parameters are tabulated below. The term “hidden” refers to cases where the command-line help documentation states that an argument value of “zero” invokes default behavior.

TABLE 2: “Hidden” Defaults

- By default, `pb blastpgp` has filtering off (`-F F`) while `pb blastall` has filtering on (`-F T`). To ensure consistency when using `blastpgp` and then `psitblastn`, the `-F` option should be set the same in both steps. So, it could either be set to `-F T` in the `blastpgp` step and not set in the `psitblastn` step (since it is set to `-F T` by default), or it should not be set in the `blastpgp` step (it is set to `-F F` by default) and set to `-F F` in the `psitblastn` step. It is best to set the `-F` option explicitly in both steps in order to avoid any ambiguity.

		Search Type				
Parameter	**symbol**	**blastn**	**blastp**	**blastx**	**tblastn**	**tblastx****
wordsize	-W	11	3	3	3	3
gap_open*	-G	5	11	11	11	-
gap_extend*	-E	2	1	1	1	-
gap_x_dropoff	-X	30	15	15	15	-
gap_x_dropoff_final	-X	50	25	25	25	-
dropoff_2nd_pass	-y	-	7	7	7	10

threshold_second*	-f	-	11	12	13	13

* Some of these values are overridden depending on the matrix.
** Note that tblastx is always ungapped.

TABLE 3: Matrix-Specific Overrides

Matrix	-G	-E	-f *
BLOSUM45	14	2	14
BLOSUM50	13	2	-
BLOSUM62	11	1	11
BLOSUM62_20	100	10	100
BLOSUM80	10	1	12
BLOSUM90	10	1	-
PAM30	9	1	16
PAM70	10	1	14
PAM250	14	2	-

* +1 for blastx, +2 for tblastn and tblastx

blastpgp

blastpgp performs gapped blastp searches and can be used to perform iterative searches in PSI-BLAST and PHI-BLAST mode.

Usage

```
pb blastpgp
```

Required arguments:

-d *<database_filename>*

Optional arguments:

-A *<window_size>*
-b *<num_alignments>*
-B *<input_align_filename>*
-c *<constant>*
-e *<E_value>*
-E *<gap_extn_penalty>*
-f *<extn_threshold>*
-F *<filtering_type>*
-g T|F

`-G` *<gap_open_penalty>*
`-h` *<E_value_threshold>*
`-H` *<end_req_qregion>*
`-i` *<query_filename>*
`-I T|F`
`-j` *<max_num_passes>*
`-J T|F`
`-k` *<pattern_filename>*
`-l` *<gi_list>*
`-m` *<align_view_option>*
`-M` *<matrix_filename>*
`-N` *<num_bits_for_gapping>*
`-o` *<output_filename>*
`-O` *<seqalign_filename>*
`-p` *<program_options>*
`-P` *<pass_options>*
`-q` *<scoremat_file_format>*
`-Q` *<matrix_output_filename>*
`-R` *<restart_input_filename>*
`-s` *<compute_S-W_aligns>*
`-S` *<start_req_qregion>*
`-T T|F` *(HTML output)*
`-t T|F` *(composition-based stats)*
`-U T|F` *(lower-case FASTA filtering)*
`-u 0|1|2` *(scoremat output format)*
`-v` *<num_db_seqs>*
`-W` *<word_size>*
`-X` *<gap_x_dropoff>*
`-y` *<dropoff_second_pass>*
`-Y` *<searchspace_effective_size>*
`-z` *<database_effective_length>*
`-Z` *<gap_x_dropoff_final>*

Description

`blastpgp` performs gapped `blastp` searches and can be used to perform iterative searches in PSI-BLAST and PHI-BLAST mode.

A Pattern-Hit Initiated BLAST (PHI-BLAST) search combines the matching of regular expressions with

local alignments surrounding the match. This allows the user to find database sequences that match a user-defined regular expression and are homologous in the area around the regular expression match. A PHI-BLAST search requires that the `-k` (pattern filename) and `-p` (selects PHI-BLAST rather than PSIBLAST) parameters be specified. See the descriptions in the Arguments section that follows for details on these parameters.

Position-Specific Iterated BLAST (PSI-BLAST) analysis is useful both for identifying the distant members of a protein family, whose relationship is not recognizable by straight sequence comparison, and also for deducing the function of hypothetical proteins that are unannotated in the database. The added sensitivity of this program over regular BLAST comes from the use of a profile that is constructed in the first iteration of searching by using a multiple alignment of the highest scoring hits. (Note that PHI-BLAST or `blastp` can be used for the first iteration of the search.) The profile is generated by calculating position-specific scores for every position in the alignment. A highly conserved position will receive a high score and weakly conserved positions receive scores near zero. The profile is then used to perform additional search iterations and the results of each iteration are used to refine the profile.

Some information in the preceding paragraphs was taken directly from NCBI's documentation. This documentation can be found at:

www.ncbi.nih.gov/blast/html/blastcgihelp.html

Required arguments for `blastpgp`

Required Argument	Description

`-d`	Sets database name(s) which must be PBFS filename(s). Multiple databases may be specified in quotes, with each database filename separated by a space. For example: "*database1 database2 database3*" would specify to search *database1*, *database2* and *database3*.
	
	Note Any databases used for this search type must be formatted with `formatdb` with the `-o` option set to T.
	
	As in NCBI BLAST, the user may also set the environment variable `BLASTDB`, so as to specify the path to the database. For example, the following two C-shell command sequences are equivalent:
	
	`setenv BLASTDB ""`
	`pb blastall -d "disk1/database1 disk1/database2"`
	
	and
	
	`setenv BLASTDB "disk1"`
	`pb blastall -d "database1 database2"`
	
	For convenience, the user may specify PB_BLASTDB instead of BLASTDB . This will override anything in BLASTDB , and is primarily intended for users who switch between NCBI BLAST and Paracel BLAST. For example, if the environment variables are set as follows,
	
	`setenv BLASTDB "/home/ncbi/databases"`
	`setenv PB_BLASTDB "disk1/"`
	
	the NCBI blastall commands will get their database path from `BLASTDB`, and `pb blastall` commands will get their database path from `PB_BLASTDB`.

Optional arguments for `blastpgp`

Optional Argument	Description
`-A`	Specifies the multiple hits window size. Note that this value is zero for single-hit algorithms. The default value is 40.
`-B`	Specifies the input alignment filename that is used for PSI-BLAST Restart. If this option is not specified, `blastpgp` will not look for this file.
`-b`	Specifies the number of database sequences for which to show alignments. The default value is 250.

Optional Argument	Description
`-c`	The constant in pseudo-counts for the multi-pass version is set using this parameter. The default is 9.
`-C`	Creates an the output file for PSI-BLAST check-pointing with the specified name. The file contains the query and the frequency count ratio matrix. Check-pointing allows a score model to be scored and later reused. See the -R parameter below. If this option is not specified, this file will not be created.
`-e`	Specifies the Expectation Value (E value). Only hits with E values less than the value specified for this parameter will be returned. The default value is 10.0 (real).
`-E`	Sets the gap extension penalty. A value of zero invokes the default. For default values, see TABLE 2: “Hidden” Defaults on p. 30. Note that the `-G` and `-E` penalties are positive, while the `-q` penalty is negative. Also see TABLE 3: Matrix-Specific Overrides on p. 31.
`-f`	Sets the threshold for extending hits. If a value of zero is entered, the default behavior is invoked. For default values, see TABLE 2: “Hidden” Defaults on p. 30. But also see TABLE 3: Matrix-Specific Overrides on p. 31.
`-F`	Specifies whether to use NCBI default filtering on the query sequence. If this parameter is set to `T` (default), the sequence will be filtered using SEG with the default parameters. For a full description of available options, see the discussion of the `-F` parameter in `blastall` arguments on p. 24.
`-g`	Specifies whether to perform a gapped alignment: `-g T` “gapped alignment” is **on** (default) `-g F` “gapped alignment” is **off** The default value is `T` (on).
`-G`	Specifies the gap opening penalty. Values entered for this parameter should be positive. If a value of zero is entered, the default behavior is invoked. For default values, see TABLE 2: “Hidden” Defaults on page 30. But see also TABLE 3: Matrix-Specific Overrides on page 31. Note that the `-G` and `-E` penalties are positive, while the `-q` penalty is negative.
`-h`	Sets the E value threshold for including sequences in the score matrix model. The default value is 0.005.
`-H`	Specifies the end of required region in query. A value of -1 (default) indicates end of query.

Optional Argument	Description
-i	Specifies the query file, which must be in FASTA format. If not specified, defaults to `stdin`.
-I	Specifies whether to show the Genbank Indices (GI's) in the deflines: `-I T` "show GI's" is **on** `-I F` "show GI's" is **off** (default) By default, the GI's are not shown (`F`).
-j	Sets the maximum number of passes to use in multi-pass version. The default value is 1 (i.e., multi-pass is off and a regular BLAST search is performed). For a PSI-BLAST search, this parameter must be set > 1.
-J	Specifies whether to parse and interpret the query deflines: `-J T` "interpret deflines" is **on** `-J F` "interpret deflines" is **off** (default) The default value is `F`: BLAST treats the deflines merely as text. If set to `T`, the query deflines are parsed assuming an NCBI compliant defline format described at ftp://ftp.ncbi.nih.gov/blast/db/README.
-k	The pattern file for PHI-BLAST is specified using this parameter. **Note** This option is required for PHI-BLAST.
-l	Specifies the name of a file containing a list of GenBank Indices (GI's). This option restricts the search of the database to a list of GI's. Any databases used with this feature must be formatted with the `-o` flag set to `T` (see formatdb on p. 47). The input file for this option can be in text or in binary format. Binary files must be formatted properly using tools available from NCBI. The text file is simply formatted with one GI per line.

Optional Argument	Description
`-m`	Specifies how alignments are presented in the output file: `-m 0` Pairwise (default) `-m 1` Query-Anchored with Identities `-m 2` Query-Anchored without Identities `-m 3` Flat Query-Anchored with Identities `-m 4` Flat Query-Anchored without Identities `-m 5` Query-Anchored without Identities with Blunt Ends `-m 6` Flat Query-Anchored without Identities with Blunt Ends `-m 7` BLAST XML `-m 8` Tabular output without comments `-m 9` Tabular output with comments `-m 10` ASN (text) `-m 11` ASN (binary) The default value is "Pairwise" (`-m 0`). For details on these alignment types, see Output Formats on p. 67.
`-M`	Specifies the name of the matrix to be used in a protein search. The default matrix is BLOSUM62. Some other matrices included in the Paracel BLAST distribution are: BLOSUM45, BLOSUM80, PAM30 and PAM70. Note that these matrices and BLOSUM50, BLOSUM62_20, BLOSUM90 and PAM250 (which are not included in the distribution) are allowed. **Note** Additional matrices may be installed by the system administrator by copying them to the correct place, namely *<pbroot>*`/data`. Matrix names are case sensitive, so they must be entered in UPPER CASE.
`-N`	Sets the number of bits to trigger gapping. The default is 22.
`-o`	Specifies the output filename for the alignment. The default is `stdout`.
`-O`	Creates a SeqAlign file with the specified filename. This file must be used with the `-J` parameter set to `T` (parse and interpret the defline). If this option is not specified, this file will not be created.

<table>
<tr><th>Optional Argument</th><th>Description</th></tr>
<tr><td>-p</td><td>Sets the program option for PHI-BLAST.

Note: This option is required for PHI-BLAST.

The most commonly used options are seedp and patseedp:
• seedp – restricts the search for local alignments to a subset of the pattern occurrences in the query. For details, see the Glossary terms “PHI-BLAST” and “PHI pattern” on p. 108.
• patseedp – searches the pattern (specified by -k) against the query sequence, then searches th same pattern against the database (specified by -d). Each of the occurrences in the query and each of the occurrences in the database form a match between the query and the database, These matches are called seeds. Each seed is extended with regular gapped blast extension code, and the best ones are reported.</td></tr>
<tr><td>-P</td><td>The -P parameter takes one of the following values:

0 multiple hits 1-pass (default)
1 single hit 1-pass
2 2-pass

The default is 0.</td></tr>
<tr><td>-q</td><td>Specifies the kind of ASN.1 input checkpoint file to read:

0 no scoremat
1 scoremat is in ASCII format
2 scoremat is in binary format

Requires -R option to specify restarting from a checkpoint file.</td></tr>
<tr><td>-Q</td><td>Specifies the output file for the PSI-BLAST matrix in ASCII format. If this option is not specified, this file will not be created.</td></tr>
<tr><td>-R</td><td>Specifies the input filename for PSI-BLAST Restart. When using this option, it is required that the query specified on the command line match exactly the query in the restart file.</td></tr>
<tr><td>-s</td><td>Specifies whether to compute locally optimal Smith-Waterman alignments:

-s T “compute Smith-Waterman alignments” is on
-s F “compute Smith-Waterman alignments” is off (default)

The default is F (off).</td></tr>
<tr><td>-S</td><td>Specifies the start of the required region in the query. The default is 1.</td></tr>
</table>

Optional Argument	Description
`-T`	Specifies HTML output. `-T T` "HTML output" is **on** `-T F` "HTML output" is **off** (default) This option is valid for all output formats except BLAST XML (`-m 7`). The default is F (off).
`-t`	Specifies whether to use composition-based statistics: `-t T` "composition-based stats" is **on** (default) `-t F` "composition-based stats" is **off** The default is T (on).
`-u`	Specifies the kind of ASN.1 input checkpoint file is output: `0` no scoremat output 1 ASCII scoremat output 2 binary scoremat output Requires creation of a checkpoint data file, via `-j` with a value greater than 1, `-J T`, and `-C`.
`-U`	Controls lower case filtering of a FASTA sequence: `-U T` "lower case filtering" is **on** `-U F` "lower case filtering" is **off** (default) Residues in low complexity areas are changed from upper to lower case. The default is F (off).
`-v`	Sets the number of database sequences for which to show one-line descriptions. The default is 500.
`-W`	Specifies the word size. The default is 3. See TABLE 2: "Hidden" Defaults on page 30. Only word sizes of 2 or 3 are valid for `blastpgp`.
`-X`	Specifies the X dropoff value for a gapped alignment (in bits). The default is 15. See the Glossary term "X dropoff value" on p. 109.
`-y`	Sets the dropoff (X) for BLAST extensions in bits. If not specified or set to 0, the default value of the dropoff is 7.
`-Y`	Specifies the *effective* size of the search space. The default value is 0, meaning use the real size of the search space. For an explanation of the distinction between effective and actual size of the search space, see the earlier discussion of the `-Y` parameter (see p. 29).

Optional Argument	Description
-z	Sets the effective length of the database. The default value is 0, meaning the actual size of the database is used. For an explanation of the distinction between effective and actual length of the database, see the earlier discussion of the -z parameter (page 29).
-Z	Sets the X dropoff value for final gapped alignment (in bits). The default value is 25.

chgrp

Usage

```
pb chgrp [-R] [-f] <group> [ <file> ... ]
```

Description

Changes the group ownership of PBFS files and directories. This command is available only to superusers.

Required arguments

Required Argument	Description
<group>	The group to which ownership is to be assigned. This may be a symbolic name, such as `root`, or an ID number such as 0.
<file>	Specifies the name of the PBFS file whose ownership status will be changed. Zero or more PBFS files may be specified. Wildcard characters are allowed.

Optional arguments

Optional Argument	Description
-R	Makes changes recursively down any directories named, changing ownership for every file contained therein, and so on for all subdirectories.
-f	Suppresses all error messages.

chmod

Usage

```
pb chmod [-R] [-f] <mode> [ <file> ... ]
```

Description

Changes the permissions of PBFS files.

Required arguments

Required Argument	Description
<mode>	Specifies the permissions mode to which the PBFS file will be changed. These modes are the same as those used in UNIX.
<file>	Specifies the name of the PBFS file whose permission mode will be changed. Zero or more PBFS files may be specified. Wildcard characters are allowed.

Optional arguments

Optional Argument	Description
`-R`	Makes changes recursively down any directories named, changing permission mode for every file contained therein, and so on for all subdirectories.
`-f`	Suppresses all error messages.

chown

Usage

```
pb chown [-R] [-f] <owner> [ :<group> ] [ <file> ... ]
```

Description

Changes the ownership and, optionally, the group ownership of files and directories. The owner and group may be specified as symbolic names as root or ID numbers like 0. This command is available only to superusers.

Required arguments

Required Argument	Description
<owner>	Specifies the new owner of the file.
<file>	Specifies the name of the PBFS file whose ownership will be changed. Zero or more PBFS files may be specified. Wildcard characters are allowed.

Optional arguments

Optional Argument	Description
<group>	The group to which ownership is to be assigned. This may be a symbolic name, such as `root`, or an ID number such as 0.
`-R`	Makes changes recursively down any directories named, changing ownership for every file contained therein, and so on for all subdirectories.
`-f`	Suppresses all error messages.

cp

Usage

`pb cp` *<orig_filename>* *<new_filename>*

Description

Copies the original PBFS database to the new PBFS database.

Required arguments

Required Argument	Description
<orig_filename>	Specifies the name of the Paracel BLAST file to be copied.
<new_filename>	Specifies the name of the Paracel BLAST file to which *<orig_filename>* is copied.

dbinfo

Usage

```
pb dbinfo <database_name> [-s] [-t] [-v]
```

Description

Returns information about a BLAST database such as database type, number of sequences, total length, maximum length of a sequence, database name and database creation date.

Required arguments

Required Argument	Description
<database_name>	Specifies the database name.

Optional arguments

Optional Argument	Description
`-s`	Prints the sizes of the database files.
`-t`	Prints the raw information in tab delimited format; not compatible with the `-v` option.
`-v`	Prints additional information about the database; not compatible with the `-t` option.

Example

```
> pb dbinfo gbpri2

Header of index file gbpri2.nbl:
 Protein = F
 Sequences = 41863
 Total Length = 135729565
 Max Sequence Length = 326663
 Title = "gbpri2"
 Creation Date = "Sep 19, 2000 3:35 PM"
```

df

Usage

```
pb df <filesystem> [ ... ]
```

Description

Lists mounted filesystems, along with some usage statistics such as free space. Without arguments, it lists all mounted filesystems. With arguments, it lists the information for the filesystem specified.

Required arguments

Required Argument	Description
<filesystem>	Specifies the path of the filesystem for which to list the usage statistics. Zero or more PBFS paths may be specified.

fastacmd

Usage

```
pb fastacmd
```

Required arguments:

`-d` *<database_filename>*

Optional arguments:

`-a T|F` (duplicate accessions)
`-c T|F` (^A defline separators)
`-D 1|2|3` (database format)
`-I T|F` (only print database info)
`-i` *<accession_filename>*
`-L` *<start>*,*<stop>*
`-l` *<line_length>*
`-o` *<output_filename>*
`-P` *<PIG>*
`-p T|F|G` (protein/nucleotide/guess)
`-S 1|2` (retrieve top/bottom nucleotide)
`-s` *<accession_list>*
`-T T|F` (print taxonomy information)
`-t T|F` (only target on GI deflines)

Description

`fastacmd` should be used to dump databases formatted by `pb formatdb` into FASTA files. Because Paracel BLAST uses a proprietary index format for better performance, it is necessary to run the Paracel BLAST version of `fastacmd`, not the NCBI version, on databases used by Paracel BLAST.

Required arguments

Required Argument	Description
`-d`	Specifies the database to dump. There is no default.

Optional arguments

Optional Argument	Description
`-a`	This option controls whether duplicate accessions are to be retrieved: `-a T` "retrieve duplicate accessions" is **on** `-a F` "retrieve duplicate accessions" is **off** (default) The default is F (off), meaning that only 1 copy of a given accession number's sequence will be retrieved.
`-c`	Controls whether to use Ctrl-A's as non-redundant defline separators: `-c T` "^A defline separators" is **on** `-c F` "^A defline separators" is **off** (default) The default is F (off), meaning that no special processing based on defline separators is to be performed.
`-D`	Specifies what to dump the database as: `0` dump nothing `1` FASTA format `2` GI list `3` Accession.version list The default is 0, meaning not to dump anything.

Optional Argument	Description
-I	Controls whether to print database information only, overriding all other options: -I T "print information only" is **on** -I F "print information only" is **off** (default) The default is F (off), meaning normal processing.
-i	Specifies an input file with GI's/accessions/loci for batch retrieval. There is no default value. Note that only one of -i and -s may be specified.
-L	Specifies a range of sequence to extract, in the format <*start*>,<*stop*>. For both ends, 0 means "all the way to the beginning/end". The default value is '0,0', meaning the whole sequence (from beginning to end).
-l	Specifies the line length for sequence data. The default value is 80.
-o	Specifies the output filename. The default is stdout.
-P	Retrieve sequences with this PIG (Protein Identification Group). There is no default.
-p	Controls symbol interpretation: T protein F nucleotide G guess The default value is G (guess), in which case it looks for protein first, then nucleotide.
-S	Specifies the strand to retrieve (nucleotide only): 1 top (uncomplemented) 2 bottom (complemented) The default is 1 (complemented).
-s	Specifies a comma-separated list of accessions and loci to retrieve ("search string"). There is no default. Note that only of -i and -s may be specified.
-T	Controls whether to print taxonomy information for the requested sequence(s): -T T "print taxonomy information" is **on** -T F "print taxonomy information" is **off** (default) The default is F (off), meaning don't print taxonomy information.

Optional Argument	Description
`-t`	Controls whether to print only the target GI on definition lines: `-t T` "print only target GI on deflines" is **on** `-t F` "print only target GI on deflines" is **off** (default) The default is F (off).

Example

```
pb fastacmd -d nt -D 1 -o nt.fa
```

This example shows the minimum amount of data that must be specified in order to extract a FASTA file from a formatted BLAST index. The sequence data from the `nt` database will go into the file `nt.fa`.

Another example:

```
pb fastacmd -d nt -I
```

This example shows how to gather a short summary of the statistics about the database `nt` (another way is the `pb dbinfo` command).

formatdb

Usage

```
pb formatdb
```

Required arguments:

`-n` *<database_name>*

`-p T|F` (protein database; this option is only required for DNA databases)

Optional arguments:

`-a T|F` (input file is ASN.1 format)

`-b T|F` (binary mode)

`-e T|F` (Seq-entry, rather than FASTA)

`-i` *<input_filename>*

`-l` *<log_filename>*

`-o` *<parse_options>*

`-s T|F` (index is limited to accessions)

`-S` (allows database shuffling)

`-V T|F` (verbose mode)

`-t` *<db_title>*

Description

`formatdb` should be used to load and format the FASTA databases for both protein and DNA databases for `blastall`. This must be done before `blastall` can be run. This saves disk space and improves performance, as the large FASTA file does not need to be accessed.

The input for `formatdb` may be either ASN.1 or FASTA format, the latter compressed (with gzip) or uncompressed. Use of the ASN.1 format is advantageous for users who might also wish to format the ASN.1 in different ways, such as a GenBank report. Usage of `formatdb` may be obtained by executing `pb formatdb --help`.

Because Paracel BLAST uses a proprietary index format for better performance, it is necessary to run the Paracel BLAST version of `formatdb`, not the NCBI version, on databases to be used by Paracel BLAST.

If `formatdb` is canceled by interrupting the client software or by using the `pb killjob` command during the 'job is running phase', an incomplete database will be created which can result in unexpected program behavior if such a database is searched. Such incomplete databases should be removed. If `formatdb` is canceled using `pb killjob` while it is transferring its input files, the client executable will not exit until it has completed the file transfer.

Required arguments

Required Argument	Description
`-n`	Specifies the Paracel BLAST database filename.
`-p`	This option controls whether the database is protein vs nucleotide: `-p T` "protein database" is **on** (default) `-p F` "protein database" is **off** The default value is T (protein). This option is only required for DNA databases.

Optional arguments

Optional Argument	Description
`-a`	Specifies that the input file is a database in ASN.1 format, otherwise FASTA format is expected: `-a T` "ASN.1 format" is **on** `-a F` "ASN.1 format" is **off** (default) The default is F (off), meaning that the database is in FASTA format.

Optional Argument	Description
`-b`	Specifies whether the ASN.1 database is binary mode; otherwise it is text mode: `-b T` "binary mode" is **on** `-b F` "binary mode" is **off** (default) The default is F (off), meaning the ASN.1 database is text mode.
`-e`	Specifies whether the input is a Seq-entry; otherwise FASTA format is expected: `-e T` "Seq-entry format" is **on** `-e F` "Seq-entry format" is **off** (default) The default is F (off), meaning that the database is in FASTA format.
`-i`	Specifies the input file for formatting. If not specified, defaults to `stdin`. If the file specified by the `-i` option doesn't exist, but a group of files formatted by NCBI with the same base name does exist, then `pb formatdb` will convert NCBI's index format to Paracel's proprietary index format. For example, if you have downloaded a nucleotide database from NCBI (`abc.nin`, `abc.nhr`, `abc.nsq`, ...) but you do not have access to the original FASTA file (`abc`), you can run: `pb formatdb -i abc -n xyz -p F` to convert NCBI's index format to Paracel's proprietary index format. Use `-p T` to convert an NCBI protein database. **Note** Due to incompatibilities between NCBI and Paracel index formats, you should not upload existing NCBI indices if you plan to key off GI numbers. Rather, you should obtain the FASTA formatted version of the database, and do a standard `formatdb`. If the system running the PB daemons is configured such that there are shared file systems between them and the client machine, the copy step of formatting may be skipped by giving a full path to the input file (that is, a name beginning with a slash '`/`').
`-l`	Sets the logfile name. The default value is `formatdb.log`.
`-o`	Controls whether to parse `SeqId`: `-o T` "parse `SeqId`" is **on** `-o F` "parse `SeqId`" is **off** (default) Note that all databases used for PSI/PHIBLAST should be created with this parameter set to T.
`-S`	Allows database shuffling.

Optional Argument	Description
`-s`	Controls whether to create indices limited only to accessions: `-s T` "indices limited to accessions" is **on** `-s F` "indices limited to accessions" is **off** (default) The default is F (off), meaning indices are not limited to accessions.
`-t`	Specifies the title for the database file. If a title is specified, it is used in the BLAST reports instead of the filename of the database.
`-V`	Controls whether to use verbose mode and check for non-unique string ID's in the database: `-V T` "verbose mode" is **on** `-V F` "verbose mode" is **off** (default) The default is F (off).

Example

```
pb formatdb -i data.fasta.nt -n disk2/data -p F
```

This example shows the minimum amount of data that must be specified in order to format a nucleotide database for BLAST searching. After the formatting command, `pb formatdb`, the input file (`data.fasta.nt`) is specified using the `-i` argument. The `-n` argument is used to specify the new PBFS filename and path: `disk2/data`. Finally, the database is specified as a nucleotide database with: `-p F`.

killjob

Usage

```
pb killjob <job_id>
```

Description

This command cancels the specified job. For caveats on canceling jobs during database formatting, see p. 48. If a job is canceled using `pb killjob` while it is transferring its input files, the client executable will not exit until it has completed the file transfer.

Required arguments

Required Argument	Description
<job_id>	This argument specifies the ID of the job to kill. The job ID can be found using the `pb status` command.

ls

Usage

```
pb ls [-a] [-F] [-l] [<PBFS_dir>] [<PBFS_file>]
```

Description

This command lists the files in the specified PBFS directory. By default, this command only shows the names of the databases.

Required arguments

Required Argument	Description
<PBFS_dir>	(Required only if *PBFS_file* is not specified) This argument specifies the Paracel BLAST directory whose files the user would like to list. Multiple directories may be specified.
<PBFS_file>	(Required only if *PBFS_dir* is not specified) This argument specifies the Paracel BLAST file. Multiple file names may be specified.

Optional arguments

Optional Argument	Description
`-a`	Shows hidden files, i.e., those beginning with a dot '`.`'.
`-F`	Puts a forward slash '`/`' after each file in the list that is a a directory, an asterisk '`*`' after each file that is executable, or an at sign '`@`' after each file that is a symbolic link.
`-l`	Shows permissions, owner, group, date of last modification, and filename with extension.

Example

In the following example, the command `ls -l` is used on a database named `database_dir1`:

```
ls -l database_dir1
```

These results were produced:

```
drwxrwxrwx root root Jun 28 09:11       4096 .
drwxr-xr-x root root Jul 10 09:17       4096 ..
drwxr-xr-x root root Jun 28 09:18       4096 Hsapiens
-rw-rw-rw- root root Jun 19 12:30  824167879 TurboDatabase1
-rw-rw-rw- root root Jun 17 08:27    5694187 chr23_100k
-rw-r--r-- root root Jun 22 11:37   62936904 chr23_50k
-rw-r--r-- root root Jun 22 11:45        175 formatdb.log
```

```
drwxr-xr-x root root May 17 15:11      16384 lost+found
-rw-rw-rw- root root Jun 15 18:28  350544138 nr
drwxr-xr-x root root Jun  1 11:33       4096 nt
drwxrwxrwx root root Jun  7 10:16       4096 nt7
drwxr-xr-x root root Jun 28 09:10       4096 protein
drwxr-xr-x root root Jun 28 09:10       4096 refseqs
drwxrwxrwx rism user Jun 12 09:23       4096 rism
```

megablast

MegaBLAST uses a Greedy algorithm (see Glossary, p. 106) developed by Webb Miller et al. for nucleotide sequence alignment search. It concatenates many queries at once to optimize the time spent scanning the database.

Usage

```
pb megablast
```

Required arguments:

`-d` *<database_filename>*

Optional arguments:

`-A` *<multiple_hits_window_size>*

`-b` *<num_db_seqs>*

`-D` *<detail_level>*

`-E` *<gap_extn_penalty>*

`-e` *<threshold_E_value>*

`-F` *<filtering_type>*

`-f T|F` (show full ID's in output)

`-G` *<gap_open_penalty>*

`-g T|F` (generate words for every base)

`-H` *<#HSP's>*

`-I T|F` (show GI's in the deflines)

`-i` *<query_filename>*

`-J T|F` (whether to "believe" the query defline)

`-l` *<gi_list>*

`-m` *<alignment_view_option>*

`-M` *<max_length>*

`-N` *<discontiguous_template>*

`-n T|F` (use dynamic programming)

`-o` *<output_filename>*

`-O` *<seqalign_filename>*

-P *<max_num_positions>*

-p *<identity_pcnt_cutoff>*

-Q *<mask_query_output_filename>*

-q *<mismatch_penalty>*

-r *<match_reward>*

-R T|F (report log information)

-S *<query_orientation>*

-s *<min_hit_score>*

-T T|F (HTML output)

-t *<discontiguous_word_length>*

-U T|F (lower-case FASTA filtering)

-V T|F (use old search engine)

-v *<num_db_seqs>*

-W *<word_size>*

-X *<x_dropoff_gapped_alignment>*

-y *<x_dropoff_ungapped_extension>*

-Z *<x_dropoff_gapped_extension>*

-z *<db_effective_length>*

Description

MegaBLAST uses a Greedy algorithm (see Glossary topic on p. 106) developed by Webb Miller et al. for nucleotide sequence alignment search. It concatenates many queries at once to optimize the time spent scanning the database. MegaBLAST is optimized for aligning sequences that differ slightly as a result of sequencing and other similar errors. This search method is up to 10x faster than some more common sequence similarity programs.

Required arguments for `megablast`

Required Argument	Description

`-d`	Sets database name(s) which must be PBFS filename(s). Multiple databases may be specified in quotes, with each database filename separated by a space. For example: "*database1 database2 database3*" would specify to search *database1*, *database2* and *database3*.
	
	As in NCBI BLAST, the user may also set the environment variable `BLASTDB`, so as to specify the path to the database. For example, the following two C-shell command sequences are equivalent:
	
	`setenv BLASTDB ""`
	`pb megablast -d "disk1/database1 disk1/database2"`
	
	and
	
	`setenv BLASTDB "disk1"`
	`pb megablast -d "database1 database2"`
	
	For convenience, the user may specify PB_BLASTDB instead of BLASTDB . This will override anything in BLASTDB , and is primarily intended for users who switch between NCBI BLAST and Paracel BLAST. For example, if the environment variables are set as follows,
	
	`setenv BLASTDB "/home/ncbi/databases"`
	`setenv PB_BLASTDB "disk1/"`
	
	the NCBI blastall commands will get their database path from `BLASTDB`, and `pb megablast` commands will get their database path from `PB_BLASTDB`.

Optional arguments for `megablast`

Optional Argument	Description
`-A`	Specifies the multiple hits window size (zero for a single-hit algorithm). The default value is 0.
`-b`	Specifies the number of database sequences for which to show alignments. This option is meaningful only when the `-D` parameter is set to 2 (generate traditional `blastn`-style output). The default value is 250.

Optional Argument	Description
-D	Specifies the level of detail in the alignment output:

0	(Default) Produces a one line output ID for each alignment in the form:
	`'subject-id'== '[=-] query-id' (s_off q_off s_end q_end) score`
	In this example, `query-id` is the either GI number, an accession number or some other type of identifier found in the FASTA definition line of the sequence in question. The + and - corresponds to the orientation of the query strand in the alignment. The score in this example refers to the score for non-affine gapping parameters. For non-affine searches, this means that the score is generated by adding the penalties for matches, mismatches and indels. Matches are scored as zero. Using this scoring scheme, a perfect match would generate a score of 0. Scores generated by selecting this option will generally be negative. Scoring for affine searches is the raw score of the alignment. The following is an example of this type of output:
	`'chr1-FRAG[990000,1989999]'=='-chr2-FRAG[102960000,103959999]' (815314 51560 815352 51523) 2`

1	Shows the same level of output as a value of 0, as well as the endpoints and percentage of identical nucleotides for each ungapped segment in the alignment. Note that this value uses the same scoring scheme as setting this value to 0. The following is an example of this type of output:
	`#'>chr1-FRAG[990000,1989999]'=='-chr2-FRAG[102960000,103959999]' (815314 51560 815352 51523) 2`
	`a {`
	`  s 2`
	`  b 815314 51560`
	`  e 815352 51523`
	`  l 815314 51560 815323 51551 (90)`
	`  l 815325 51550 815352 51523 (100)`
	`}`

2	Generates traditional `blastn` style output. Note that the scoring scheme for this value differs from all of the other values. The scoring scheme for this value rewards for matches and penalizes for mismatches and indels. Therefore, a positive or a negative score may be produced. For an example of this type of output, see Output Formats on p. 67.
3	Produces a one-line output for each alignment with the following fields separated by tabs: query ID, subject (database sequence) ID, percent identity, alignment length, number of mismatches (not including gaps), number of gap openings, start position of the alignment in the query, start position of the alignment in the subject, end of the alignment in the subject, expected value and bit score. Note that if the alignment is from the reverse strand, the subject start and subject end positions are printed in the reverse order. This is done to reflect the actual direction of the alignment. The scoring scheme for this value is the same as that for a value of 0. The following is an example of this type of output: `chr2-FRAG[102960000,103959999] chr1-FRAG[990000,1989999] 94.87 39 1 1 51523 51560 815352 815314 2.5e-04 59.96`

Optional Argument	Description
-E	Specifies the gap extension penalty. A value of 0 invokes the default behavior. For further details, see the discussion below for parameter -G.
-e	The E value is set using this parameter. The default is 1000000. By default this value is set to a very large number, i.e., effectively there is no expectation value cutoff.
-F	Specifies whether to use NCBI filtering to filter the query sequence. The -F option can take a string as input as described in the description of the blastall -F option on page 24. The string input can specify that DUST be run with certain values or that other nonstandard filters be used. • Setting to -F T (the default) means that the sequence will be filtered using DUST with the default parameters. • Setting to -F F, the sequence will not be filtered. • Setting to -F D specifies DUST filtering. • Setting to -F m: If the -U option (to mask any lower-case sequence in the input FASTA file) is used and one does not wish any other filtering, but does wish to mask when building the lookup tables, then one should specify: -F m. This is the only case where "m" should be specified alone.
-f	Show full IDs in the output. By default, for -D 0 and -D 1 outputs, the sequence ID's are reported as GI's or accession numbers (if GI's are not available). If -f is set to T, full ID's will be shown, unless -J option is set to F. In the latter case full deflines will be shown for the query sequences.
-G	Specifies the affine gap opening penalty. If -G and -E parameters are not set (both default to 0), then non-affine gapping is assumed with gap opening penalty 0 and gap extension penalty E, that can be computed from match reward r and mismatch penalty q by the formula: E = r/2 - q. The affine version of MegaBLAST requires significantly more memory, so it should be avoided, if possible, especially when some of the query or database sequences are very long. Setting the parameter to 0 invokes the default value.
-g	Specifies whether to generate words for every base of the database: -g T "generate words for every base" is **on** -g F "generate words for every base" is **off** (default) Both in blastn and traditional megablast, the database sequences are compressed 4:1, and words are looked up only at the beginning of each byte, i.e. at every 4th base. This option prescribes to lookup words starting at any arbitrary base of the database sequence.

Optional Argument	Description
-H	Specifies the maximum number of HSP's reported for each sequence, with the old engine (must use -V T). The default value is 0, meaning no limit on the number of HSP's reported.
-I	Specifies whether to show the Genbank Indices (GI's) in the deflines: -I T "show GI's" is **on** -I F "show GI's" is **off** (default) By default, the GI's are not shown (F).
-i	Specifies the query file. If not specified, defaults to stdin.
-J	Specifies how to "believe" the query defline. The default is T (for all types of output except -D 2 [blastn style]). In the latter case, the default is F, unless a SeqAlign ASN.1 output is required, as specified by the -O parameter. Note: the -J parameter must be set to F if the sequence ID's in the FASTA file are not unique.
-l	Restricts the search of the database to a list of GI's. Any databases used with this feature must be formatted with the -o flag set to T. The input file for this option can be in text or in binary format. Binary files must be formatted properly using tools freely available from NCBI. The text file is simply formatted with one GI per line.
-m	Specifies how alignments are presented in the output file: -m 0 Pairwise (default) -m 1 Query-Anchored with Identities -m 2 Query-Anchored without Identities -m 3 Flat Query-Anchored with Identities -m 4 Flat Query-Anchored without Identities -m 5 Query-Anchored without Identities with Blunt Ends -m 6 Flat Query-Anchored without Identities with Blunt Ends -m 7 BLAST XML -m 8 Tabular output without comments -m 9 Tabular output with comments The default value is "Pairwise" (-m 0). For details on these alignment types, see Output Formats on p. 67.
-M	Specifies the maximum total length of queries for a single MegaBLAST search. Setting this value smaller than the default can reduce the memory image of the program for large searches. The default value is 20,000,000 bases.

Optional Argument	Description

`-N`	Specifies the dis-contiguous template type:
	
	`0` coding
	`1` non-coding
	`2` both
	
	For each of the three template lengths, two dis-contiguous templates are supported. One of them, called coding, is based on the '110' pattern, the other is optimal, or close to optimal, based on the hit probability simulations for random sequences. The exact templates are:
	
	`W = 11, t = 16, coding: 1101101101101101`
	`W = 11, t = 16, non-coding: 1110010110110111`
	`W = 12, t = 16, coding: 1111101101101101`
	`W = 12, t = 16, non-coding: 1110110110110111`
	`W = 11, t = 18, coding: 1011011001011011 01`
	`W = 11, t = 18, non-coding: 1110100101100101 11`
	`W = 12, t = 18, coding: 1011011011011011 01`
	`W = 12, t = 18, non-coding: 1110101100101101 11`
	`W = 11, t = 21, coding: 1001011001011001011 01`
	`W = 11, t = 21, non-coding: 1110100101000100101 11`
	`W = 12, t = 21, coding: 1001011011011001011 01`
	`W = 12, t = 21, non-coding: 1110100101100100101 11`
	
	If 2 is specified (the 'both' option), then all initial matches satisfying either one of the two types of templates are extended.

`-n`	Specifies whether to use dynamic programming (T) or the greedy algorithm (F) to compute extensions for affine gaps: `-n T` "use dynamic programming" is **on** `-n F` "use dynamic programming" is **off** (default) The default is F (off).

`-O`	Sets the `ASN.1 SeqAlign` filename. This file must be used with the `-J` parameter set to `T` and the `-D` parameter set to 2. Using this option disallows database splitting. If this option is not specified, this file will not be created.
	
	The `ASN.1` will consist of separate `ASN.1` codes for each query sequence:
	
	`Seq-annot ::= {`
	All hits for first query
	`}`
	
	`Seq-annot ::= {`
	All hits for second query
	`}`
	
	etc.

Optional Argument	Description
-o	Sets the name for the BLAST report output file. The default is stdout.
-P	Sets the maximum number of positions for a hash value. This parameter provides for a very simple type of filtering if it is set to a non-zero value. Namely, any pattern of length 12 when word size is greater than or equal to 16 (8 for smaller word sizes) that appears in all of the query sequences together more than P times is masked and not included in the search look-up table. If such masking occurs, MegaBLAST shows a warning message on the standard output. This can be useful when running MegaBLAST for very long unmasked sequences, in which case the search might take a very long time if this parameter is not set. If not specified, the default value is 0.
-p	Sets the identity percentage cutoff (a real number). The alignments with an identity percentage below the value of this parameter are not reported in all output formats except when -D 0. (In this case, the traceback is not performed, so it is impossible to calculate the percentage of identical residues. If not specified, the default value is 0.
-Q	Sets the masked query output filename. All regions of the query sequences that were hit by any found alignment are masked by N's. The output is written to the file specified by this parameter. It can be used only in conjunction with -D 2 (blastn output). Using this parameter disallows database splitting. If this option is not specified, this file will not be created.
-q	Specifies the penalty for a nucleotide mismatch. This value must be negative. The default value is -3.
-R	Controls whether to report the log information at the end of output: -R T "report log information" is **on** -R F "report log information" is **off** (default) The default is F (off).
-r	Sets the reward for a nucleotide match. The default is 1.
-S	Specifies the query strands to search against the database: 1 forward 2 reverse 3 both (default) The default value is 3 (both).

Optional Argument	Description
-s	Specifies the minimum hit score to report. By default this value is set to W, where W is the word size (-W parameter), thus default behavior is to ignore the -s parameter since all found alignments are extended from an exact match of length at least W).
-T	Specifies HTML output. -T T "HTML output" is **on** -T F "HTML output" is **off** (default) This option is valid for all output formats except BLAST XML (-m 7). The default is F (off).
-t	If non-zero, specifies the dis-contiguous word approach. The supported template lengths are 16, 18, and 21. The word size (-W parameter) must be 11 or 12 in this case.
-U	Controls lower case filtering of a FASTA sequence: -U T "lower case filtering" is **on** -U F "lower case filtering" is **off** (default) As in blastall binary, this parameter makes it possible to treat lower case in the query sequences as masked residues. The default for this parameter is set to F (off), in which case lower case is treated identically to upper case.
-u	Specifies the number of alignments to report per db subject. The default value is zero. Note that MegaBLAST does not support long query chopping, so -u always means not to prune.
-V	Controls whether to use the old engine: -V T "use old engine" is **on** -V F "use old engine" is **off** (default) The default value is F , meaning to use the new engine.
-v	Specifies the maximum number of database sequences for which to show one-line descriptions. This option is meaningful only when the -D parameter is set to 2 (generate traditional blastn style output). If not specified, the default value is 500.
-W	Sets the word size, i.e. length, of best perfect match. When W is divisible by 4, this parameter guarantees that all perfect matches of length W + 3 will be found by the MegaBLAST search. However, perfect matches of length as low as W may also be found, although this is not guaranteed. Any value of W not divisible by 4 is equivalent to the nearest value divisible by 4. Values halfway between 4(n) and 4(n +1) are rounded down. Thus, 14 is equivalent to 12, as is 13, while 15 is equivalent to 16. If not specified, the default value is 28.

Optional Argument	Description
`-X`	Specifies the X dropoff value for gapped alignment (in bits). As in BLAST, this value provides a cutoff threshold for the extension algorithm tree exploration. When the score of a given branch drops below the current best score minus the X dropoff, exploration of this branch stops. If not specified, the default value is 20. For details, see the Glossary term "X dropoff value" on p. 109.
`-y`	Specifies the X dropoff value for ungapped extension (an integer). If not specified, the default value is 10.
`-Z`	Specifies the X dropoff value for dynamic programming gapped extension. If not specified, the default value is 50.
`-z`	Sets the effective length of the database. The default value is 0, meaning the actual size of the database is used. For an explanation of the distinction between effective and actual length of the database, see the earlier discussion of the `-z` parameter on page 29.

mkdir

Usage

```
pb mkdir <PBFS_directory_name>
```

Description

This command makes a single PBFS directory.

Required arguments

Required Argument	Description
<PBFS_directory_name>	The name of the PBFS directory to make.

mv

Usage

```
pb mv <PBFS_orig_filename> <PBFS_new_filename>
```

Description

This command specifies to move or rename the specified PBFS file or directory to the new file name.

Required arguments

Required Argument	Description
<PBFS_orig_filename>	This argument specifies the Paracel BLAST file name of the original file or directory. Multiple files or directories may be specified.
<PBFS_new_filename>	This arguments specifies the new Paracel BLAST file or directory name.
	Note An error will be issued if the user attempts to move a directory into a file.

reprioritize

Usage

```
pb reprioritize <job_id> --priority=<priority>
```

Description

This command sets the priority to the integer specified. Job sub-pieces that are on workers are not preempted, but pieces which are not yet allocated to workers will receive the new priority.

Note	Only the user who submitted the job or a superuser can reprioritize the job.

Required arguments

Required Argument	Description
<job_id>	This argument specifies the job ID number. To determine the job ID of a job, use the `pb status` command.
`--priority=`*<int>*	This argument sets the priority of the job specified. Valid values for users without superuser privileges are integers between -99 and 99. Valid values for users with superuser privileges are integers between -2 billion and 2 billion.

rm

Usage

```
pb rm [ <PBFS_filename> | <PBFS_directory_name> ]
```

Description

This command removes the specified PBFS file or directory. Use this command carefully, as the file or directory will be erased from the system.

Required arguments

Required Argument	Description
<PBFS_filename>	(Required if PBFS_directory_name is not specified) This argument specifies the Paracel BLAST file name. Multiple file names may be specified.
<PBFS_directory_name>	(Required if PBFS_directory_name is not specified) This argument specifies the Paracel BLAST directory to remove. Multiple directory names may be specified.

shutdown

Usage

```
pb shutdown
```

Description

This command can be performed only by a superuser and shuts down the Paracel BLAST software, not the host machines themselves. Be careful when using this command, as it will kill all jobs currently in progress.

Arguments

This command does not take any arguments.

status

Usage

```
pb status
```

Description

This command displays the status of the Paracel BLAST system. Note that complete details of jobs will only be shown to the job owner or a superuser. A non-root user can, however, check the percent completeness of other users' searches.

If a worker daemon dies, it may occur that the administrator is not alerted. Thus, it is recommended to run `pb status` periodically to verify that all worker connections are intact.

Arguments

This command does not take any arguments.

Example

The following are examples of status reports. Note how each report has the server version, date, server uptime, clients connected, workers connected, jobs queued and idle workers. If there are jobs running, the ID, priority, date started, query, database and current status of each job is given.

```
Server version: Paracel [2004-04-04] (protocol 20).
Server uptime: 17 seconds.
Client connections from:
    joe_user@desmodus
    joe_user@desmodus
    joe_user@desmodus
    joe_user@desmodus
    joe_user@desmodus
Worker connections from:
    desmodus
Jobs queued: 3.
Idle workers: 0.

Job 10 joe_user@desmodus priority 0 (82)
    Started: Tue April 4 17:24:38 2004 (idle 4s)
    Query (unsplit):
        /home/joe_user/data/querys/blastn/5.query
    Database (unsplit):
        databases/gbpri2
    Job is in job queue

Job 4 joe_user@desmodus priority 0 (85)
    Started: Tue April 4 17:24:37 2004 (idle 4s)
    Query (unsplit):
        /home/data/querys/blastn/2.query
    Database (unsplit):
        databases/dystrophin_dna
    Job is in job queue

Job 29 joe_user@desmodus priority 0 (90)
    Started: Tue April 4 17:24:42 2004 (idle 1s)
    Query (unsplit):
        /home/joe_user/data/querys/blastn/13.query
```

```
Database (unsplit):
    databases/dna.database.1.loose.hit.region.chromo.2.o.T
Job is in job queue
```

Updating while searching

Paracel BLAST incorporates an *update-while-searching* feature which makes it possible to execute write commands affecting a database even when a search is being executed on it. These commands include `pb formatdb`, `pb blastall` and `pb mv`. The write commands are not actually carried out until all previously queued searches of the database have finished. Any additional searches of the specified database are held until the write commands have been completed. In order to replace or update a database that is frequently searched, the following steps can be performed:

1. Submit searches against the database:

   `pb blastall -d db ...`

2. Load the new version of the database to a temporary location:

   `pb formatdb -n` *<temporary_name>* ...

3. Move the new version onto the old one. Note that this will block until all previously submitted searches against *<db>* have completed:

   `pb mv` *<temporary_name> <db>*

4. Additional searches can be submitted at any time:

   `pb blastall -d db` ...

Note that any searches submitted after the `pb mv` command will block until the `mv` is complete. In other words, these jobs will search the new copy of the database, not the old one.

Chapter 4
Input and Output Files

Paracel BLAST accepts and outputs various file formats as described in this chapter.

Input Format

FASTA Format

Paracel BLAST allows all query sequences to be in FASTA format. In this format, each query sequence is preceded by a description line (header string) that starts with a right angle bracket as the first character followed by the query index information.

```
>gi|532319|pir|TVFV2E|TVFV2E envelope protein
ELRLRYCAPAGFALLKCNDADYDGFKTNCSNVSVVHCTNLMNTTVTTGLLLNGSYSENRTQI
WQKHRTSNDSALILLNKHYNLTVTCKRPGNKTVLPVTIMAGLVFHSQKYNLRLRKLVEITPI
GFAPTEVRRYTGGHERQKRVPFVXXXXXXXXXXXXXXXXXXXXVQSQHQQMLKLTIWGVK
LLAGILQQQKNLLAAVEAQQQMLKLTIWGVKLLAGILQQQKNLLAAVEAQLLAAVEAQQQML
0LLAGILQQQKNLLAAVEAQQQMLKLTIWGVKLLAGILQQQKNLLAAVEAQQQMLKLTIWGV
KLLAGILQQQKNLLAAVEAQQQMLKLTIWGVKLLAGILQQQKNLLAAVEAQQQMLKLTIWGV
AGILQQQKNLLAAVEAQQQMLKLTIWGVKLLAGILQQQKN
```

Spaces, tabs, and carriage returns in the sequence are ignored. The start of a new sequence is indicated by the presence of another description line starting with a right angle bracket.

For a detailed description of this format, see:

http://www.ncbi.nlm.nih.gov/blast/html/search.html

Output Formats

Paracel BLAST report formats closely resemble those of NCBI BLAST. There are seven different alignment view formats available for BLAST searches. Each alignment view is preceded by the following:

- Header
- Summary Report
- Annotation and Alignment Statistics section

The alignment view formats are:

- Pairwise
- Query-Anchored with Identities
- Query-Anchored without Identities
- Flat Query-Anchored with Identities
- Flat Query-Anchored without Identities
- Query-Anchored without Identities with Blunt Ends
- Flat Query-Anchored without Identities with Blunt Ends

Additional output formats are also supported:

- BLAST XML
- Tabular Output without Comments
- Tabular Output with Comments
- ASN (text)
- ASN (binary)
- BLAST HTML

Header

The header portion contains:

- algorithm name and the date of the release
- annotation information on the query and the number of characters in the query
- database path and name, number of sequences and size

For example:

```
BLASTN 1.3.6-Paracel [2002-03-05]
Reference: Altschul, Stephen F., Thomas L. Madden, Alejandro A. Schaffer,
Jinghui Zhang, Zheng Zhang, Webb Miller, and David J. Lipman (1997),
"Gapped BLAST and PSI-BLAST: a new generation of protein database search
programs", Nucleic Acids Res. 25:3389-3402.
Query= emb Z12846 modified
(279 letters)
Database: someests
495 sequences; 100,322 total letters
Searching.....................................................done
```

Summary Report

The BLAST report also provides a summary of the high-scoring alignments, ranked by score. The description line in the summary is divided into the annotation for the sequence, the accession number, its abbreviated code and a one-line sequence description. The columns on the right report high score and E value.

For example:

```
Score E
Sequences producing significant alignments: (bits) Value
gi|24148|emb|Z12846.1|Z12846 HSA08C081 CLONTECH cDNA library CC... 466 e-133
gi|25272|emb|Z13398.1|Z13398 HSA45A081 CLONTECH cDNA library CC... 321 4e-90
gi|24570|emb|Z15500.1|Z15500 HSA23D012 CLONTECH cDNA library CC... 299 1e-83
gi|24679|emb|Z15555.1|Z15555 HSA27A032 CLONTECH cDNA library CC... 194 6e-52
gi|25595|emb|Z13562.1|Z13562 HSA57F051 CLONTECH cDNA library CC... 172 2e-45
```

Annotation and Alignment Statistics

The alignments and their statistics make up the third portion of the BLAST report. The contents of this section vary according to the option chosen in the Report Format pull-down list. In the following example, each high-scoring alignment lists a simple header comprising the database name, a description of the high scoring sequence, and the length of the database entry. The next few lines report the alignment's accumulated score and the E value (expected number of chance occurrences of a hit with a score greater than the given score in a database of a given size). It also lists the number of identities (exact matches), the number of positives (inexact matches), the number of gaps and the number of frame shifts.

For example:

```
>gi|24148|emb|Z12846.1|Z12846 HSA08C081 CLONTECH cDNA library CCRF-CEM,
           cat# HL1063g Homo sapiens cDNA clone 08C08
Length = 281
Score = 466 bits (235), Expect = e-133
Identities = 275/285 (96%), Gaps = 10/285 (3%)
Strand = Plus / Plus
```

Pairwise

This view shows the query sequence on the top and the database sequence on the bottom of the alignment. The numbers are the position of the section of the alignment in number of bases from the beginning of each sequence. These numbers appear at the beginning and end of each alignment. Identities are indicated by a pipe symbol '|' and mismatches are shown as whitespace.

For example:

```
Query: 1 ggcattaagttgggc-ggttctagtaatagttcaat-gcacagttttcacgtcaaatgct 58
         ||||||||||||||| |||||||||||||||||||| |||||||||||||||||||||||
Sbjct: 1 ggcattaagttgggctggttctagtaatagttcaatggcacagttttcacgtcaaatgct 60

Query: 59 tggtttagcaccagctatcgggccagttgttac----ttgggtggatttattaccaacgc 114
          |||||||||||||||||||||||||||||||| ||||||||||||||||||||||
Sbjct: 61 tggtttagcaccagctatcgggccagttgttactgctttgggtggatttattaccaacgc 120

...

Query: 115 tgggaagattactggactggtaaaaggtatgggttcagcagttattggtgctggtaaggt 174
           ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct: 121 tgggaagattactggactggtaaaaggtatgggttcagcagttattggtgctggtaagg- 179

Query: 175 gctgaatggttagttttgttgctagattgtttggaatggcagcaggtaacacagcggtag 234
           ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct: 180 ---gaatggttagttttgttgctagattgtttggaatggcagcaggtaacacagcggtag 236

Query: 235 caacatcatcaactgctgctgcaactggtacaaaagcagttgggg 279
           |||||||||||||||||||||||||||||||||||||||||||
Sbjct: 237 caacatcatcaactgctgctgcaactggtacaaaagcagttgggg 281
```

Query-Anchored with Identities

This view shows the database ID and position of the hit. The query sequence is shown in full. Identities are shown as dots. Thus, matches are shown as dots and mismatches are shown as characters. Insertions and deletions are shown below the alignment. Dashes in the sequence represent gaps.

For example:

```
QUERY 1 ggcattaagttgggcggttctagtaatagttcaatgcacagttttcacgtcaaatgcttg 60
0     1 ..................................................... 62
                       \                     \
                       |                     |
                       t                     g
1   200 ..................................................... 140
                       \                      \
                       |                       |
                       t                       g
3   134 ..................................................... 74
                         \                         \
                         |                         |
                         t                         g
```

```
QUERY 61 gtttagcaccagctatcgggccagttgttacttgggtggatttattaccaacgctgggaa 120
0     63 ..................................................... 126
                                    \
                                    |
                                  tgct
4     12                              ............................ 40
1    139 ..................................................... 76
                                     \
                                     |
                                   gctt
3     73 ..................................................... 10
                                      \
                                      |
                                    gctt

QUERY 121 gattactggactggtaaaaggtatgggttcagcagttattggtgctggtaaggtgctgaa 180
0     127 ...................................................----... 182
4      41 ...................................................----... 96
1      75 ...................................................----... 20
3       9 ......                                                     4
7       7                                                        ... 9

QUERY 181 tggttagttttgttgctagattgtttggaatggcagcaggtaacacagcggtagcaacat 240
0     183 ..................................................... 242
4      97 ..................................................-..... 155
1      19                                                ........... 4
7      10 ...........n.....................................-n..... 68

QUERY 241 catcaactgctgctgcaactggtacaaaagcagttgggg 279
0     243 ..................................... 281
4     156 .....................................  193
7      69 .....................................  106
```

Query-Anchored without Identities

This alignment view shows the same thing as the query-anchored with identities view except that both matches and mismatches are shown as characters. Insertions and deletions are shown below the alignment. Dashes in the sequence represent gaps.

For example, a multi-sequence alignment:

```
QUERY 1   ggcattaagttgggcggttctagtaatagttcaatgcacagttttcacgtcaaatgcttg 60
0     1   ggcattaagttgggcggttctagtaatagttcaatgcacagttttcacgtcaaatgcttg 62
                       \                    \
                       |                    |
                       t                    g
1     200 gcattaagttgggcggttctagtaatagttcaatgcacagttttcacgtcaaatgcttg 140
                       \                     \
                       |                      |
                       t                      g
3     134 gcattaagttgggcggttctagtaatagttcaatgcacagttttcacgtcaaatgcttg 74
                         \                          \
                         |                          |
                         t                          g
```

```
QUERY 61  gtttagcaccagctatcgggccagttgttacttgggtggatttattaccaacgctgggaa 120
0     63  gtttagcaccagctatcgggccagttgttacttgggtggatttattaccaacgctgggaa 126
                                       \
                                       |
                                     tgct
4     12                                ttgggtggatttattaccaacgctgggaa 40
1     139 gtttagcaccagctatcgggccagttgttacttgggtggatttattaccaacgctgggaa 76
                                        \
                                        |
                                      gctt
3     73  gtttagcaccagctatcgggccagttgttacttgggtggatttattaccaacgctgggaa 10
                                        \
                                        |
                                      gctt

QUERY 121 gattactggactggtaaaaggtatgggttcagcagttattggtgctggtaaggtgctgaa 180
0     127 gattactggactggtaaaaggtatgggttcagcagttattggtgctggtaagg----gaa 182
4     41  gattactggactggtaaaaggtatgggttcagcagttattggtgctggtaagg----gaa 96
1     75  gattactggactggtaaaaggtatgggttcagcagttattggtgctggtaagg----gaa 20
3     9   gattac                                                    4
7     7                                                       gaa 9

QUERY 181 tggttagttttgttgctagattgtttggaatggcagcaggtaacacagcggtagcaacat 240
0     183 tggttagttttgttgctagattgtttggaatggcagcaggtaacacagcggtagcaacat 242
4     97  tggttagttttgttgctagattgtttggaatggcagcaggtaacacagcggtag-aacat 155
1     19  tggttagttttgttgc                                          4
7     10  tggttagttttnttgctagattgtttggaatggcagcaggtaacacagcggta-naacat 68

QUERY 241 catcaactgctgctgcaactggtacaaaagcagttgggg 279
0     243 catcaactgctgctgcaactggtacaaaagcagttgggg 281
4     156 catcaactgctgctgcaactggtacaaaagcagttggg  193
7      69 catcaactgctgctgcaactggtacaaaagcagttggg  106
```

Flat Query-Anchored with Identities

This view shows insertions and deletions in the query sequence instead of showing them below the alignment. Matches are shown as dots and mismatches are shown as characters. This format is useful for rapid inspection of aligned sequences for conserved residues. Dashes in the sequence represent gaps.

For example, a multi-sequence alignment:

```
QUERY 1   ggcattaagttgggc-ggttctagtaatagttcaat-g-cacagttttcacgtcaaatgc 57
0     1   ..............t.....................g.-..................... 59
1     200  ..............t.....................-.g..................... 143
3     134  ..............t.....................-.g..................... 77

QUERY 58  ttggtttagcaccagctatcgggccagttgttac----t----tgggtggatttattacc 109
0     60  ..................................tgct.----.................. 115
4     12                                    ----.----.................. 29
1     142 ....................................----.gctt.................. 87
3     76  ....................................----.gctt.................. 21

QUERY 110 aacgctgggaagattactggactggtaaaaggtatgggttcagcagttattggtgctggt 169
0     116 ............................................................. 175
4     30  ............................................................. 89
1     86  ............................................................. 27
3     20  ..................                                          4

QUERY 170 aaggtgctgaatggttagttttgttgctagattgtttggaatggcagcaggtaacacagc 229
0     176 ...-----.................................................... 231
4     90  ...-----.................................................... 145
1     26  ...-----...................                                 4
7     7           ..............n..................................... 58
```

```
QUERY 230 ggtagcaacatcatcaactgctgctgcaactggtacaaaagcagttgggg 279
0     232 ............................................... 281
4     146 .....-.......................................... 193
7     59  ....-n.......................................... 106
```

Flat Query-Anchored without Identities

This alignment view shows insertions and deletions in the query sequence instead of showing them below the alignment. Matches and mismatches are shown as characters. Gaps are shown as dashes. Missing residues at the end of a sequence (blunt ends) are not indicated.

For example:

```
QUERY 1   ggcattaagttgggc-ggttctagtaatagttcaat-g-cacagttttcacgtcaaatgc 57
0     1   ggcattaagttgggctggttctagtaatagttcaatgg-cacagttttcacgtcaaatgc 59
1     200  gcattaagttgggctggttctagtaatagttcaat-ggcacagttttcacgtcaaatgc 143
3     134  gcattaagttgggctggttctagtaatagttcaat-ggcacagttttcacgtcaaatgc 77

QUERY 58  ttggtttagcaccagctatcgggccagttgttac----t----tgggtggatttattacc 109
0     60  ttggtttagcaccagctatcgggccagttgttactgctt----tgggtggatttattacc 115
4     12                                    ----t----tgggtggatttattacc 29
1     142 ttggtttagcaccagctatcgggccagttgttac----tgctttgggtggatttattacc 87
3     76  ttggtttagcaccagctatcgggccagttgttac----tgctttgggtggatttattacc 21

QUERY 110 aacgctgggaagattactggactggtaaaaggtatgggttcagcagttattggtgctggt 169
0     116 aacgctgggaagattactggactggtaaaaggtatgggttcagcagttattggtgctggt 175
4     30  aacgctgggaagattactggactggtaaaaggtatgggttcagcagttattggtgctggt 89
1     86  aacgctgggaagattactggactggtaaaaggtatgggttcagcagttattggtgctggt 27
3     20  aacgctgggaagattac                                          4

QUERY 170 aaggtgctgaatggttagttttgttgctagattgtttggaatggcagcaggtaacacagc 229
0     176 aagg----gaatggttagttttgttgctagattgtttggaatggcagcaggtaacacagc 231
4      90 aagg----gaatggttagttttgttgctagattgtttggaatggcagcaggtaacacagc 145
1      26 aagg----gaatggttagttttgttgc                               4
7       7         gaatggttagttttnttgctagattgtttggaatggcagcaggtaacacagc 58

QUERY 230 ggtagcaacatcatcaactgctgctgcaactggtacaaaagcagttgggg 279
0     232 ggtagcaacatcatcaactgctgctgcaactggtacaaaagcagttgggg 281
4     146 ggtag-aacatcatcaactgctgctgcaactggtacaaaagcagttggg  193
7     59  ggta-naacatcatcaactgctgctgcaactggtacaaaagcagttggg  106
```

Query-Anchored without Identities with Blunt Ends

This alignment view shows insertions and deletions below the alignment. Matches and mismatches are shown as characters. Missing residues at the end of a sequence (blunt ends) are indicated by dashes.

For example:

```
QUERY 1   ggcattaagttgggcggttctagtaatagttcaatgcacagttttcacgtcaaatgcttg 60
0     1   ggcattaagttgggcggttctagtaatagttcaatgcacagttttcacgtcaaatgcttg 62
                        \                    \
                        |                    |
                        t                    g
4         ------------------------------------------------------------
1     200 -gcattaagttgggcggttctagtaatagttcaatgcacagttttcacgtcaaatgcttg 140
                        \                     \
                        |                     |
                        t                     g
3     134 -gcattaagttgggcggttctagtaatagttcaatgcacagttttcacgtcaaatgcttg 74
                        \                     \
                        |                     |
                        t                     g
7         ------------------------------------------------------------

QUERY 61  gtttagcaccagctatcgggccagttgttacttgggtggatttattaccaacgctgggaa 120
0     63  gtttagcaccagctatcgggccagttgttacttgggtggatttattaccaacgctgggaa 126
                                     \
                                     |
                                   tgct
4     12  -------------------------------ttgggtggatttattaccaacgctgggaa 40
1     139 gtttagcaccagctatcgggccagttgttacttgggtggatttattaccaacgctgggaa 76
                                      \
                                      |
                                    gctt
3     73  gtttagcaccagctatcgggccagttgttacttgggtggatttattaccaacgctgggaa 10
                                      \
                                      |
                                    gctt
7         ------------------------------------------------------------

QUERY 121 gattactggactggtaaaaggtatgggttcagcagttattggtgctggtaaggtgctgaa 180
0     127 gattactggactggtaaaaggtatgggttcagcagttattggtgctggtaagg----gaa 182
4     41  gattactggactggtaaaaggtatgggttcagcagttattggtgctggtaagg----gaa 96
1     75  gattactggactggtaaaaggtatgggttcagcagttattggtgctggtaagg----gaa 20
3     9   gattac------------------------------------------------------ 4
7     7   ----------------------------------------------------------gaa 9

QUERY 181 tggttagttttgttgctagattgtttggaatggcagcaggtaacacagcggtagcaacat 240
0     183 tggttagttttgttgctagattgtttggaatggcagcaggtaacacagcggtagcaacat 242
4     97  tggttagttttgttgctagattgtttggaatggcagcaggtaacacagcggtag-aacat 155
1     19  tggttagttttgttgc-------------------------------------------- 4
3     3   ------------------------------------------------------------ 4
7     10  tggttagttttnttgctagattgtttggaatggcagcaggtaacacagcggta-naacat 68

QUERY 241 catcaactgctgctgcaactggtacaaaagcagttgggg 279
0     243 catcaactgctgctgcaactggtacaaaagcagttgggg 281
4     156 catcaactgctgctgcaactggtacaaaagcagttggg- 193
1     3   ---------------------------------------- 4
3     3   ---------------------------------------- 4
7     69  catcaactgctgctgcaactggtacaaaagcagttggg- 106
```

Flat Query-Anchored without Identities with Blunt Ends

This alignment view shows insertions and deletions in the sequence instead of below the alignment. Both matches and mismatches are represented as characters. Missing residues at the end of a sequence (blunt ends) are indicated by dashes.

For example:

```
QUERY 1   ggcattaagttgggc-ggttctagtaatagttcaat-g-cacagttttcacgtcaaatgc 57
0     1   ggcattaagttgggctggttctagtaatagttcaatgg-cacagttttcacgtcaaatgc 59
4         ------------------------------------------------------------
1     200 -gcattaagttgggctggttctagtaatagttcaat-ggcacagttttcacgtcaaatgc 143
3     134 -gcattaagttgggctggttctagtaatagttcaat-ggcacagttttcacgtcaaatgc 77
7         ------------------------------------------------------------

QUERY 58  ttggtttagcaccagctatcgggccagttgttac----t----tgggtggatttattacc 109
0     60  ttggtttagcaccagctatcgggccagttgttactgctt----tgggtggatttattacc 115
4     12  ----------------------------------t----tgggtggatttattacc 29
1     142 ttggtttagcaccagctatcgggccagttgttac----tgctttgggtggatttattacc 87
3     76  ttggtttagcaccagctatcgggccagttgttac----tgctttgggtggatttattacc 21
7         ------------------------------------------------------------

QUERY 110 aacgctgggaagattactggactggtaaaaggtatgggttcagcagttattggtgctggt 169
0     116 aacgctgggaagattactggactggtaaaaggtatgggttcagcagttattggtgctggt 175
4     30  aacgctgggaagattactggactggtaaaaggtatgggttcagcagttattggtgctggt 89
1     86  aacgctgggaagattactggactggtaaaaggtatgggttcagcagttattggtgctggt 27
3     20  aacgctgggaagattac---------------------------------------- 4
7         ------------------------------------------------------------

QUERY 170 aaggtgctgaatggttagttttgttgctagattgtttggaatggcagcaggtaacacagc 229
0     176 aagg----gaatggttagttttgttgctagattgtttggaatggcagcaggtaacacagc 231
4     90  aagg----gaatggttagttttgttgctagattgtttggaatggcagcaggtaacacagc 145
1     26  aagg----gaatggttagttttgttgc---------------------------------- 4
3     3   ------------------------------------------------------------ 4
7     7   --------gaatggttagttttnttgctagattgtttggaatggcagcaggtaacacagc 58

QUERY 230 ggtagcaacatcatcaactgctgctgcaactggtacaaaagcagttgggg 279
0     232 ggtagcaacatcatcaactgctgctgcaactggtacaaaagcagttgggg 281
4     146 ggtag-aacatcatcaactgctgctgcaactggtacaaaagcagttggg- 193
1     3   ------------------------------------------------- 4
3     3   ------------------------------------------------- 4
7     59  ggta-naacatcatcaactgctgctgcaactggtacaaaagcagttggg- 106
```

BLAST XML

Paracel BLAST can produce BLAST XML format output. The BLAST XML DTD is available at the NCBI website. This file is made available so that users who have developed applications that accept NCBI XML-formatted files can also have access to the output of Paracel BLAST.

For example:

```
<?xml version="1.0"?>
<!DOCTYPE BlastOutput PUBLIC "-//NCBI//NCBI BlastOutput/EN"
"NCBI_BlastOutput.dtd">
<BlastOutput>
  <BlastOutput_program>blastn</BlastOutput_program>
  <BlastOutput_version>blastn 1.5.0-Paracel [2003-03-27]</BlastOutput_version>
  <BlastOutput_reference>
    ~Reference: Altschul, Stephen F., Thomas L. Madden,
    Alejandro A. Schaffer, ~Jinghui Zhang, Zheng Zhang, Webb Miller, and David J.
    Lipman (1997), ~&amp;quot;Gapped BLAST and PSI-BLAST: a new generation of protein
    database search~programs&amp;quot;, Nucleic Acids Res. 25:3389-3402.
  </BlastOutput_reference>
  <BlastOutput_db>regression/dystrophin_dna </BlastOutput_db>
  <BlastOutput_query-ID>lcl|QUERY</BlastOutput_query-ID>
  <BlastOutput_query-def>Short piece of dystrophin</BlastOutput_query-def>
  <BlastOutput_query-len>685</BlastOutput_query-len>
  <BlastOutput_param>
    <Parameters>
      <Parameters_expect>10</Parameters_expect>
      <Parameters_sc-match>1</Parameters_sc-match>
      <Parameters_sc-mismatch>-3</Parameters_sc-mismatch>
      <Parameters_gap-open>5</Parameters_gap-open>
```

```
      <Parameters_gap-extend>4</Parameters_gap-extend>
      <Parameters_filter>D</Parameters_filter>
    </Parameters>
  </BlastOutput_param>
  <BlastOutput_iterations>
    <Iteration>
      <Iteration_iter-num>1</Iteration_iter-num>
      <Iteration_hits>
        <Hit>
          <Hit_num>1</Hit_num>
          <Hit_id>gi|5032314|ref|NM_004010.1|DMDe</Hit_id>
          <Hit_def>Homo sapiens dystrophin (muscular dystrophy,</Hit_def>
          <Hit_accession>NM_004010</Hit_accession>
          <Hit_len>14143</Hit_len>
          <Hit_hsps>
            <Hsp>
              <Hsp_num>1</Hsp_num>
              <Hsp_bit-score>264.146</Hsp_bit-score>
              <Hsp_score>133</Hsp_score>
              <Hsp_evalue>2.90317e-73</Hsp_evalue>
              <Hsp_query-from>262</Hsp_query-from>
              <Hsp_query-to>403</Hsp_query-to>
              <Hsp_hit-from>440</Hsp_hit-from>
              <Hsp_hit-to>582</Hsp_hit-to>
              <Hsp_query-frame>1</Hsp_query-frame>
              <Hsp_hit-frame>1</Hsp_hit-frame>
              <Hsp_identity>142</Hsp_identity>
              <Hsp_positive>142</Hsp_positive>
              <Hsp_gaps>1</Hsp_gaps>
              <Hsp_align-len>143</Hsp_align-len>
<Hsp_qseq>GTTCAAAAGAAAACATTCACAAAATGGGTAAATGCACAATTTTCTAAGTTTGGGAAGCAGCATATTGAGAA
CCTCTTCA-TGACCTACAGGATGGGAGGCGCCTCCTAGACCTCCTCGAAGGCCTGACAGGGCAAAAACT</Hsp_qseq>

<Hsp_hseq>GTTCAAAAGAAAACATTCACAAAATGGGTAAATGCACAATTTTCTAAGTTTGGGAAGCAGCATATTGAGAA
CCTCTTCAGTGACCTACAGGATGGGAGGCGCCTCCTAGACCTCCTCGAAGGCCTGACAGGGCAAAAACT</Hsp_hseq>

<Hsp_midline>||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||</Hsp_midline>
            </Hsp>
          </Hit_hsps>
        </Hit>
        ... // multiple hits omitted for the sake of brevity
      </Iteration_hits>
      <Iteration_stat>
        <Statistics>
          <Statistics_db-num>1</Statistics_db-num>
          <Statistics_db-len>14143</Statistics_db-len>
          <Statistics_hsp-len>0</Statistics_hsp-len>
          <Statistics_eff-space>1.58267e+07</Statistics_eff-space>
          <Statistics_kappa>0.710605</Statistics_kappa>
          <Statistics_lambda>1.37407</Statistics_lambda>
          <Statistics_entropy>1.30725</Statistics_entropy>
        </Statistics>
      </Iteration_stat>
    </Iteration>
  </BlastOutput_iterations>
</BlastOutput>
```

Tabular Output without Comments

This output report contains a synopsis of search results without depicting the alignments. The information is the same as that contained immediately below in Tabular Output with Comments, but without any annotation or table headings.

Tabular Output with Comments

This output report contains an annotated synopsis of search results without depicting the alignments.

```
# BLASTN 1.3.6-Paracel [2002-03-05]
# Database: someests
# Query: emb Z12846 modified
# Fields: Query id, Subject id, % identity, alignment length, mismatches, gap
openings, q. start, q. end, s. start, s. end, e-value, bit score
emb gi|24148|emb|Z12846.1|Z12846 96.49 285 0 4 1 279 1 281 1e-133 466.3
emb gi|25272|emb|Z13398.1|Z13398 97.33 187 0 2 92 278 12 193 3.8e-90 321.6
emb gi|24570|emb|Z15500.1|Z15500 95.02 201 0 4 2 196 200 4 1.4e-83 299.8
emb gi|24679|emb|Z15555.1|Z15555 95.42 131 0 3 2 126 134 4 5.9e-52 194.8
emb gi|25595|emb|Z13562.1|Z13562 97.03 101 2 1 178 278 7 106 2.2e-45 173.0
```

... multiple rows omitted for the sake of brevity

ASN (text)

This output report is in NCBI's SeqAnnot format, suitable for importation into NCBI toolkit programs. A segment of the text form appears below:

```
Seq-annot ::= {
  desc {
    user {
      type
      str "Hist Seqalign" ,
      data {
        {
          label
            str "Hist Seqalign" ,
          data
            bool TRUE } } } ,
    user {
      type
        str "Blast Type" ,
      data {
        {
          label
            str "BLASTN" ,
          data
            int 2 } } } } ,
  ...
```

ASN (binary)

This report format is also in NCBI's SeqAnnot format, suitable for importation into NCBI toolkit programs, except that it isn't in a human-readable form (and is usually considerably smaller).

BLAST HTML

This output report contains the header, summary and specified alignment view formatted in HTML so that it can be viewed in a web browser. All output formats with the exception of XML can be rendered into HTML

Genetic Codes

The following are descriptions for the different genetic code values that can be utilized during a `pb blastall` search. The values that should be specified after the `-Q` and `-D` options are in the left column and the descriptions are in the right column.

TABLE 4: `blastall` Genetic Codes	
Command Line Value	**Description**
1	Standard Nuclear Genetic Code
2	Vertebrate Mitochondrial
3	Yeast Mitochondrial
4	Mold Mitochondrial, Protozoan Mitochondrial, Coelenterate Mitochondrial and Spiroplasma
5	Invertebrate Mitochondrial
6	Ciliate Nuclear, Dasycladacean Nuclear, Hexamita Nuclear
9	Echinoderm Mitochondrial
10	Euplotid Nuclear
11	Bacterial and Plant Plastid
12	Alternative Yeast Nuclear
13	Ascidian Mitochondrial
14	Flatworm Mitochondrial
15	Blepharisma Macronuclear
16	Chlorophycean Mitochondrial
21	Trematode Mitochondrial
22	Scenedesmus Obliquus Mitochondrial
23	Thraustochytrium Mitochondrial

Log Message Files

Paracel BLAST log messages are sent to a separate file via the Linux standard `syslog()`. All worker logs go to the manager. The `syslog()` function takes care of rotating and purging log files.

Statistics are output using the `LOG_INFO` argument to `syslog()`.

Log lines are output by the manager daemon at search submission, cancellation, reprioritization, and completion, as well as for `formatdb` and file upload. A typical logline would be in the following format:

<date> <hostname> <process_name> [*<process_id>*]: ev=*<type>* [*<key>*=*<val>*]*

For example:

```
Jan 29 14:33:09 lavia pbd[1937]: ev=st id=234 ty=ba vr=ba
qC=23 qT=94589025 d1=gbbri1 d2=gbpri2 sC=5443
sT=9807431
```

Event types

Event	Description
`ev=st`	Job was submitted. The type st is also used to report statistics related to job start-up. Keys: `id` – Job id, for correlating with completion. `pr` – Job priority. `ty` – Job type (`ba` for `blastall`, `ft` for `formatdb`, `up` for `upload`, `mb` for `megablast`, `ps` for `blastpgp`, `pm` for `mergejob`). `vr` – `blastall` variant (`bn`, `bp`, `bx`, `tx`, `tn`). *Only blastall/megablast/blastpgp:* `qC` Number of queries submitted. `qT` – Number of bases in all queries. `qP` – Number of query pieces. `sC` – Number of data documents searched. `dC` – Number of sequences in the databases. `sT` – Number of bases in all databases. `sP` – Number of database pieces. `qK` – Query packing. *Upload only:* `fl` – Name of uploaded file. `sz` – File size. `cl` – Name of the user and client starting the job.

Event	Description
ev=fi	Job completion. Keys: rs – Job result: (fi)nished, (fa)iled, (ca)ncelled. id – Job id. cd – Error code. er – Error message. tm – Time to completion. pt – Number of bases in all queries. ef – Number of query pieces.
ev=rp	Reprioritization. Keys: id – Job id. pr – New priority.
ev=su	pbd started up. pbd is the PB Blast daemon. Keys: pt – Port. rt – Filesystem root. vs – Paracel BLAST version. pc – Protocol version.
ev=wo	Worker connection opened. Keys: hn – Host name of worker.
ev=wc	Worker connection closed. Keys: hn – Host name of worker.
ev=sd	pbd shut down.

The perl script which accomplishes this task can be found in *<install_dir>*/scripts/stats. This script assumes that the command line argument is the name of a file which looks like /var/log/pb.log. If no command line argument is given, the file /var/log/pb.log is used.

The `stats` script now computes the following:

- Manager uptime
- Duty cycle (total time of all tasks/Manager uptime)
- Number of jobs canceled
- Number of jobs failed

`blastall` jobs:

- Number of Blast jobs started,
- Number of Blast jobs finished
- Number of each variant run
- Min, max and average number of queries
- Query pieces
- Databases searched
- Number of sequences in the databases
- Size of databases
- Database pieces
- Time for completion

`formatdb` jobs:

- Number of `formatdb` jobs started
- Number of `formatdb` jobs finished
- Min, max and average size of file uploaded
- Time spend uploading
- Average transfer rate (MB/sec)
- Average formatting rate (MB/sec)

Chapter 5
Web Server

Paracel BLAST Web Server provides access to Paracel BLAST as a Web service. There are two parts to this:

1. A Web User Interface (Web UI) - see Chapter 6 Web User Interface (Web UI) on p. 84.
2. A scriptable REST Interface – see Chapter 7 REST Interface on p. 92.

The Web Server is implemented with the pbwebd daemon.

Note	For security reasons, search jobs which are submitted with the `pb` client are not accessible using Web Server. On the other hand, it is possible to control search jobs submitted with Web Server using the `pb` client. See pb on p. 17.

Using the pbwebd daemon

The Web Server daemon, `pbwebd`, is started and stopped along with the Paracel BLAST daemon, `pbd`. For instructions on starting and stopping the daemons, see Controlling the Paracel BLAST daemons with pbd script on p. 14.

Note	It is necessary that `pbd` and `pbwebd` both run on the same node – the manager node.

Configuring Web Server

Web Server behavior controlled with the configuration file `/etc/pbweb.conf`.

Note	For safety, Web Server defaults to being disabled. Configure `pbweb.conf` with suitable settings before enabling it.

After making changes to the configuration file, it is necessary to run the following, as root, on the manager node:

```
service pbd restart
```

The configuration contains the following directives:

Directive	Values	Default	Description
WebServerEnabled	`true` `false`	`false`	If `true`, Web Server is enabled. If `false`, Web Server is disabled. In this case, `pbwebd` still runs, but is inactive.
WebUiEnabled	`true` `false`	`true`	If `true`, and WebServerEnabled is `true`, the Web UI is enabled. If `false`, the Web UI is disabled, while leaving the REST interface available.

Directive	Values	Default	Description
Port	[1, 65535]	8080	Port on which Web Server serves HTTP requests.
SyslogLevel	DEBUG INFO NOTICE WARNING ERR	ERR	The minimum error level to be reported to syslog.

Chapter 6
Web User Interface (Web UI)

Paracel BLAST Web Server provides a Web User Interface (Web UI). With the Web UI, users can submit search jobs, then monitor and control them. Finally, results can be downloaded.

To access the WebUI, with `pbwebd` running with default settings, load into a browser a URL such as the following:

http://pbserver:8080

where *pbserver* is the name of the machine running `pbwebd`. If Web Server is configured to listen at port 80, then the Web UI could be accessed at:

http://pbserver

Pages are provided which allow you to submit a search job, monitor and control its progress, download its search results, and finally delete it.

Note	For security reasons, search jobs which are submitted with the `pb` client are not accessible using the Web UI. On the other hand, it is possible to control search jobs submitted with the Web UI using the `pb` client. See pb on p. 17.

Submit Job

Use the **Submit Job** page to submit a search job to Paracel BLAST.

The **Submit Job** page is accessed at `/pb/submit/blastp.html`, etc., depending on the search type.

Field	Description
Search type	Select the search type by clicking on one of the types listed in the left navigation bar: **blastp >>** **blastn >>** **blastx >>** **tblastn >>** **tblastx >>** **megablast >>** **blastpgp >>** **psitblastn >>**
FASTA Sequence	Paste the search query into this text box. Queries are in FASTA format. Multiple FASTA entries can be handled in a single search, in which case all entries will be searched and each will generate a complete BLAST report.
Job Title	This is an optional field. The job title can be helpful in identifying the search job. The job title can be changed later.

Field	Description
Contact	This is an optional field. Contact information can be helpful in identifying the search job. The contact can be changed later.
Databases	The available databases for the current query type are listed, each with a check box. Select one or more databases.
Search Options	Enter the Paracel BLAST search options for this search type. For a complete description of the options: • for `blastp`, `blastn`, `blastx`, `tblastn` and `tblastx`, see Optional arguments for blastall on p. 23. • for `megablast`, see Optional arguments for megablast on p. 54. • for `blastpgp`, see Optional arguments for blastpgp on p. 34. • for `psitblastn`, see psitblastn on p. 29.
Show Search Options	Click this button to see a quick summary of the available search options for the current query type. Click this button again to toggle the summary in and out of view.
Priority	Enter a priority for the search: a value from -100 to 100. The search priority can be changed later.
Submit	Press this button to submit the search. A pop-up alert box will appear if the search was rejected due to an error. Otherwise, the Web UI will take you to the **Jobs** page, highlighting your running job.

Jobs

Use the **Jobs** page to monitor and manage search jobs, both running and completed.

The **Jobs** page is accessed at `/pb/job.html`. The page is organized into the following sections:

Top section:	Selected Job	Shows fields for the currently selected job.
Middle section:	Button Row	Clicking a button on this row affects the currently selected job.
Bottom section:	Jobs Table	Table of all jobs. Click on a job to select it. Click the ▲ / ▼ on a column to sort the table in increasing / decreasing order on that column.

Top Section: Selected Job

Shows fields for the currently selected job.

Field	Description
Job Number	Unique sequential identifier for the job. This is different from the **Job ID** (described further down): • the **Job Number** is unique for all time, while the **Job ID** is not. • the **Job Number** increments sequentially, while the **Job ID** is semi-random.
Job Title	The job title can be helpful in identifying the job. To change the job title, enter a new title and press **Save**.
Contact	Contact information can be helpful in identifying the job. To change the contact, enter a new contact and press **Save**.
Priority	Current runtime priority for the search job. Jobs with the highest priority are run first. To change the priority, enter a new priority, from -100 to 100, and press **Save**.
Search Options	The Paracel BLAST search options for the current search job. See pb on p. 17.
State	Current state of the search job. **Running** Search job is active. Priority can be changed. **Complete** Search job is complete. Results can be downloaded. **Killed** A user terminated the search job by clicking the **Kill Search** button (described in the following section). **Error** There was an error during processing, terminating the search job. An explanatory message is in the **Error** field. **Failed** The search job failed to start. An explanatory message is in the **Error** field. **Abandoned** The search job was terminated due to `pbwebd` being restart - see Error: Reference source not found on p. Error: Reference source not found.

Field	Description
Job ID	The value to use for *<job_id>* in the following `pb` client commands: • reprioritize on p. 62 • killjob on p. 50 **Job ID** is only defined while the job is in the `running` state.
Started	Time and date the search job was submitted. The time is in local time coordinates.
Finished	Time and date the search job was completed or terminated. The time is in local time coordinates.
Errors	If the job terminated abnormally, this contains an explanatory message.

Middle Section: Button Row

Clicking one of these buttons affects the currently selected job.

Button	Description
Save	Click this button to save new values for the editable fields: • **Job Title** • **Contact** • **Priority**
Download Result	After a search job has completed, click this button to download the search result file.
View Result	After a search job has completed, click this button to view the search result on the browser screen. Note that the results can be large. If so, it may take several seconds for the results to appear. Click this button again to toggle the results in and out of view.
View Progress	Click this button to view a snapshot of the progress report for a search job. Click this button again to toggle the search progress out of view. Click again to view a fresh progress report.
View Query	Click this button to view the input search sequence for the search job. Click this button again to toggle the input search sequence in and out of view.

Button	Description
Kill Job	Click this button to terminate a running search job.
Delete	Click this button to remove the search job from the screen. Note that deleted jobs are not actually deleted from the file system, but are instead moved to the job-deleted subdirectory. Assuming the Paracel BLAST installation directory is `/paracel` (the default), this would be: `/paracel/paracel/web/job-deleted` It is necessary for a system administrator to periodically remove these deleted jobs in order to free disk space.

Bottom Section: Jobs Table

Table of all jobs. Click on a job to select it.

Click the ▲ / ▼ on a column to sort the table in increasing / decreasing order on that column.

Column	Description
Job Number	Sequential number identifying the job. See Job Number on p. 87.
Job Title	User settable title to identify the job. See Job Title on p. 87.
Contact	User settable contact for the job. See Contact on p. 87.
Priority	User settable priority for the job. See Priority on p. 87.
State	Current job state. See State on p. 87.
Cores	Number of CPU cores currently allocated to this job. This is affected by job priority.
% Done	Current percent completion of the job.
Started	Time and date the search job was submitted. The time is in local time coordinates.

Databases

Use the **Databases** page to monitor Paracel BLAST databases.

The **Databases** page is accessed at `/pb/database.html`.

The page is organized into the following sections:

Top section:	Selected	Shows fields for the currently selected database.

	Database	
Middle section:	Button Row	Clicking a button on this row affects the currently selected database.
Bottom section:	Databases Table	Table of all databases. Click on a database to select it. Click the ▲ / ▼ on a column to sort the table in increasing / decreasing order on that column.

Note	To add, remove or change a database, you can use the `pb` client, or make changes manually to the filesystem. The Web Server periodically scans the `pbroot` subdirectory for any changes. Assuming the Paracel BLAST installation directory is `/paracel` (the default), this would be: `/paracel/paracel/pbroot`

Field	Description
ID	Number which uniquely identifies the database.
Name	Name of the database files in `/paracel/paracel/pbroot`.
Type	Type of the database: either Protein or Nucleotide.
Size (GB)	Total size of the database in gigabytes.
State	Availability state of the database. **Ready** — Database is available for searching. **Forming** — Database has recently appeared in `/paracel/paracel/pbroot`, and is being evaluated. **Not searchable** — Database evaluation failed, meaning the database is not searchable. An explanatory message is in the **Error** field.
Created	Date the database was created. The time is in local time coordinates.
Description	The database description can be helpful in identifying the database. To change the description, type into the **Description** field, then press **Save**.
Errors	If the database has been determined to be malformed, this contains an explanatory message.

Deleted Jobs

The **Deleted Jobs** page helps you track deleted jobs. When a search job is deleted, it is moved into the `job-deleted` subdirectory. If the Paracel BLAST installation directory is `/paracel` (the default), deleted jobs directory would be:

```
/paracel/paracel/web/job-deleted
```

An administrator should periodically delete or archive these jobs, as needed.

Field	Description
Directory	Directory containing the deleted jobs.
Number	The number of deleted jobs.
Size	The number of bytes in the deleted jobs directory.

Chapter 7
REST Interface

Paracel BLAST Web Server provides a REST interface, turning Paracel BLAST into a Web service which can be controlled by customer applications using standard web technologies. See Chapter 5 Web Server on p. 81.

The REST interface makes use of JSON for message formatting: a message corresponds to a JSON object. A message is an object. An object contains a number of fields, each field with a value consisting of one of:

- number
- string
- boolean
- object
- array of values

Table: Information is organized into tables. There are two tables: jobs and databases.

Handle: Within each table, there are a number of rows. Each row is associated with a unique handle:

- Each job has a unique `job_handle`.
- Each database has a unique `database_handle`.

Revision: There is a global revision number which applies to both jobs and databases. Each change to a job or database corresponds to an increment of the global revision number.

Note	For security reasons, search jobs which are submitted with the `pb` client are not accessible using the REST interface. On the other hand, it is possible to control search jobs submitted with the REST interface using the `pb` client. See pb on p. 17.

The following REST methods are defined:

Method	URL	JSON Messages and Objects	Effect
Job Management			
POST	`/pb/job`	*Request:* Submit Job Message *Response:* Job Submitted Message	Starts a new search job.
GET	`/pb/job/`	*Response:* Get All Jobs Message	Returns a listing of all search jobs.
GET	`/pb/job/` *<handle>*	*Response:* Get Job Message	Returns information for a particular search job.
PUT	`/pb/job/` *<handle>*	*Request:* Update Job Message *Response:* Job Updated Message	Updates one or more fields of a search job, including: • changing priority • killing the search job

Method	URL	JSON Messages and Objects	Effect
DELETE	`/pb/job/<handle>`	*Response:* Job Deleted Message	Removes a search job.
GET	`/pb/job/<handle>` `/result.txt`		Retrieves a search job's query results.
GET	`/pb/job/<handle>` `/progress.txt`		Retrieves a search job's progress report.
GET	`/pb/job/<handle>` `/query.txt`		Retrieves a search job's FASTA query.
Database Management			
GET	`/pb/database/`	*Response:* Get All Databases Message	Returns a listing of all databases.
GET	`/pb/database/<handle>`	*Response:* Get Database Message	Returns information for a particular database.
PUT	`/pb/database/<handle>`	*Request:* Update Database Message *Response:* Database Updated Message	Updates one or more fields of a database.
Real-time Updates			
GET	`/pb/<table>/delta` `/<revision>`	*Response:* Delta Message	Returns an update when there has been a change to any job or database.
GET	`/pb/job/<handle>` `/delta/<revision>`	*Response:* *Delta Message*	Returns an update when there has been a change to a specified job.
Status			
GET	`/pb/about`	*Response:* Version Message	Returns Paracel BLAST version information.

Submit Job Message

To submit a new job, send the **Submit Job** message to `/pb/job` as a POST request.

On success, a Job Submitted Message is returned.

On failure, a Failure Message is returned.

Field	Contents
`type`	One of the supported BLAST search types: • `blastn` • `blastp` • `blastx` • `tblastn` • `tblastx` • `megablast` • `blastpgp` • `psitblastn`
`sequence`	The search sequence in FASTA format. The following sequence of two characters is used to represent a line break: `\n`
`title`	String containing a title for the search job.
`contact`	String containing contact information for the search job.
`options`	String containing Paracel BLAST search options. For a complete description of the available options, see: • blastall on p. 20 • megablast on p. 52
`priority`	Priority for the search job. This is a number in the range [-100, 100].
`database_handles`	An array of one or more database ID numbers.

Job Submitted Message

Returned in response to a **Submit Job** message, on success.

Field	Contents
`job`	Job Description Object – see below.
`revision`	New revision number resulting from addition of the new job.
`warnings`	Array of warning messages.

Job Description Object

Produced in the following circumstances:

- As part of the response to a Submit Job Message, when a new job has been created.
- As part of a Delta Message, produced after a new job has been created.
- As part of a
- As part of a Get Jobs message
- In response to a GET request to `/pb/job/`*<job_handle>*

Field	Contents
`job_handle`	Number which identifies the search job.
`title`	Title for the search job, as set by the user.
`contact`	Contact information for the search job, as set by the user.
`options`	String containing the Paracel BLAST search options for the search job.
`priority`	Priority for the search job, as set by the user.
`database_handles`	The database ID numbers which are being searched by this search job.
`state`	Current state of the search job. `running` – Search job is active. Priority can be changed. `complete` – Search job is complete. Results can be downloaded. `killed` – A user terminated the search job. `error` – There was an error during processing, terminating the search job. An explanatory message is in the **`error_messages`** field. `failed` – The search job failed to start. An explanatory message is in the **`error_messages`** field. `abandoned` – The search job was terminated due to `pbwebd` being restarted - see Restart the Paracel BLAST daemons on p. 10.
`job_id`	Number to use for *<job_id>* in the following `pb` client commands: • reprioritize on p. 62 • killjob on p. 50 **`job_id`** is only defined while the job is in the `running` state.
`num_pieces`	Number of pieces into which the search job has been divided.
`num_queued`	Number of search job pieces which are currently queued to be run.

Field	Contents
num_running	Number of search job pieces which are currently running.
num_done	Number of search job pieces which have been completed.
pcnt_done	Percentage of the search job which has been completed.
time_started	Time when the search job was submitted.
time_finished	Time when the search job finished.
error_messages	If the search job failed, an array of associated error messages.

Get All Jobs Message

Returned in response to a GET request to /pb/job/.

Field	Contents
jobs	Array containing a Job Description Object for each search job.
revision	Revision number corresponding to the returned values.

Get Job Message

Returned in response to a GET request to /pb/job/<*job_handle*>.

Field	Contents
job	Job Description Object for the job with the indicated job_handle.
revision	Revision number corresponding to the returned values.

Update Job Message

To change one or more fields in a search job, send the following message as a PUT request to /pb/job/<*job_handle*>. All fields are optional.

On success, a Delta Change Object is returned.

On failure, a Failure Message is returned.

Field	Contents
`job_handle`	Must be the same as *<job_handle>* in the PUT request. Optional.
`title`	String containing a new value for the job title. Optional.
`contact`	String containing a new value for the contact information. Optional.
`priority`	A new value for the job priority. This is a number in the range [-100, 100]. Optional.
`state`	A new value for the job state. Optional. If present, must be one of the following: `killed` – If search job state is `running`, the search job is killed, and the state is changed to `killed`.

Job Updated Message

Returned in response to a successful PUT request of Update Job Message to `/pb/job/`*<job_handle>*.

Field	Contents
`job`	Job Description Object with the updated values. Note that when `priority` is being changed, the new priority may not be reflected in the returned values, due to the fact that the change in priority may still be pending.
`revision`	Revision number corresponding to the returned values.
`warnings`	Array of warning messages.

Delete Job

To delete a search job, send a DELETE request to `/pb/job/`*<job_handle>*, where `job_handle` is the unique identifier for the search job.

On success, a Job Deleted Message is returned.

On failure, a Failure Message is returned.

Job Deleted Message

Returned in response to a successful DELETE request to `/pb/job/`*<job_handle>*.

Field	Contents
`update`	Delta Remove Object corresponding to the removed search job.
`revision`	Revision number corresponding to the returned values.

Get Database Message

Returned in response to a GET request to `/pb/database/`*<database_handle>*.

Field	Contents
`database`	Database Description Object corresponding to the database with the given `database_handle`.
`revision`	Revision number corresponding to the returned values.

Get All Databases Message

Returned in response to a GET request to `/pb/database/`.

Field	Contents
`databases`	Array containing a Database Description Object for each database.
`revision`	Revision number corresponding to the returned values.

Database Description Object

Contains all information for one Paracel BLAST database.

Field	Contents
`database_handle`	Number which uniquely identifies the database.
`name`	Name of database within the filesystem. A database consists of multiple files: `/paracel/paracel/pbroot/`*<name>.<ext>*
`type`	Database type: `nucleotide` – Nucleotide database. `protein` – Protein database.
`size_gb`	Total size of the database files in gigabytes.

Field	Contents
`state`	Current state of the search job. `ready` – Database is available for searching. `forming` – Database has recently appeared under `/paracel/paracel/pbroot`, and is being evaluated. `malformed` – Database evaluation failed. An explanatory message is in the **Error** field.
`date_created`	Date when the database was created.
`time_created`	Time of day when the database was created.
`description`	Description for the database, as set by the user.
`error_messages`	If the search job failed, an array of associated error messages.

Update Database Message

To change one or more fields in a database, send the following message as a PUT request to `/pb/database/`*<database_handle>*. All fields are optional.

On success, a Delta Change Object is returned.

On failure, a Failure Message is returned.

Note	To add, remove or change a database, you can use the `pb` client, or make changes manually to the filesystem. The Web Server periodically scans `/paracel/paracel/pbroot` for any changes.

Field	Contents
`database_handle`	Must be the same as *<database_handle>* in the PUT request. Optional.
`description`	String containing a new value for the database description. Optional.

Database Updated Message

Returned in response to a successful PUT request of Update Database Message to `/pb/database/`*<database_handle>*.

Field	Contents
`database`	Database Description Object with the updated values.
`revision`	Revision number corresponding to the returned values.
`warnings`	Array of warning messages.

Delta Message

To receive a real-time update, send a GET request to one of:

`/pb/`*<table>*`/delta/`*<revision>*
`/pb/job/`*<handle>*`/delta/`*<revision>*

where:

- **table** is one of:

`job`	The Job table.
`database`	The Database table.

- **revision** refers to the revision number. The first request should be for revision 0. The response to this request will be immediate. The Delta message includes a new revision number. The next request should be for this revision number, and so on. These later requests remain pending until there are any updates to report. As soon as updated information is available, a response is produced.
- **handle** is the optional handle of a particular table entry.

Field	Contents
`revision`	The new revision number, after applying the updates in this message.
`updates`	Heterogeneous array of one or more of the following: • Delta Reset Object • Delta Add Object • Delta Change Object • Delta Remove Object

Delta Reset Object

Indicates that the corresponding table should be cleared. This is always done at the beginning, but may also be done in preference to transmitting a large number of accumulated changes.

Field	Contents
`table`	String identifying one of the tables.
`reset`	The value 1.

Delta Add Object

Indicates the addition of a new row to the table.

Part of Delta Message.

Field	Contents
`table`	String identifying one of the tables.
`add`	One of the following, corresponding to the table: • Job Description Object • Database Description Object

Delta Change Object

Represents a change in the value of a field in a table.

Part of Delta Message.

Field	Contents
`table`	String identifying one of the tables.
`change`	Object describing a particular change corresponding to the table. One of: • Change Job Object • Change Database Object

Change Job Object

Represents a change in the value of one or more fields of a search job.

Part of Delta Change Object.

For a list of fields, see Job Description Object on 95.

Field	Contents
`job_handle`	Number identifying the search job.
<field 1>	New value for *<field 1>*.
<field 2>	New value for *<field 2>*.
…	...

Change Database Object

Represents a change in the value of one or more fields of a database.

Part of Delta Change Object.

For a list of fields, see Database Description Object on p. 99.

Field	Contents
`database_handle`	Number identifying the database.
<field 1>	New value for *<field 1>*.
<field 2>	New value for *<field 2>*.
…	...

Delta Remove Object

Indicates the removal of a row from a table.

Part of Delta Message.

Field	Contents
`table`	String identifying one of the tables - “`job`” or “`database`”.
`remove`	Number identifying the row to remove. This number corresponds to one of the following, corresponding to the table: • `job_handle` • `database_handle`

Failure Message

A **Failure** message can be returned in response to any submitted message. If a Failure message is returned, it indicates that no changes were made.

Field	Contents
`failure`	Array of error message strings.

Version Message

Returns Paracel BLAST version information.

Field	Contents
`version`	Version of Paracel BLAST.
`release_date`	Release date of the current version.
`ncbi_version`	Version of NCBI BLAST on which this version is based.
`source_revision`	Source code revision number of this version.
`product`	Paracel BLAST.
`company`	Paracel LLC.

Glossary

Term	Description
Coiled-coiled filtering	Coiled-coiled filtering is based on the work of Lupas et al. (Science , vol. 252, pp. 1162-4 (1991)) and written by John Kuzio (Wilson et al., J. Gen. Virol. , vol. 76, pp. 2923-32 (1995)). The three parameters for coiled-coiled filtering are listed below with their default values given in square brackets: • Window [22] • Cutoff (probability of a coil-coil) [40.0] • Linker (distance between two coiled-coiled regions that should be linked together) [32]. One may also change the coiled-coiled parameters in a manner analogous to that of SEG: `-F "C 28 40.0 32"` will change the window to 28.
Defline	Deflines are the lines that start with the right angle bracket '>' in a FASTA file.
GI	Short for Genbank Index.
Greedy algorithm	An algorithm that makes the locally optimum choice at each stage with the hope of finding the global optimum. Greedy algorithms rarely find the globally optimal solution consistently, since they usually don't operate exhaustively on all the data. Nevertheless, they often give good approximations to the optimum.
Job priority	Job priority is initially specified by the user at search submission time (with `--priority=`*<num>*), or according to search job size as determined by Paracel BLAST. Paracel BLAST assigns large search jobs a lower priority than small ones, improving interactive response on heavily loaded systems. The user can alter the priority of his search jobs, submitting time-critical jobs with high priorities and background or batch jobs with low priorities. Users or administrators can adjust job priority using the `reprioritize` command (see p. 62). Ordinary users can assign priorities for their own search jobs in the range from -99 to +99; administrators can set priorities for any search jobs in the range from -2 billion to +2 billion.

Term	Description
Job size	Sometimes a search may fail because the Paracel BLAST Planner has not divided the search job into small enough sub-jobs. If you have previously successfully run searches against larger databases or queries and then find that searches fail on smaller databases or queries, it is often an indication that the sub-job size is too large. You can override the PB Planner with `--dbpart` and `–querypart` command line arguments. These arguments force the Planner to partition the database or query using your values rather than the internal defaults which are determined based on database and query characteristics such as sequence length. Say a search fails and produces an error message such as: `Job was split into 10 pieces: 6 queued, 4 running, 0 done (0.0%)` In this case, kill the search job and resubmit, but specify a `--dbpart` value greater than the number of pieces reported in the error message: e.g., `--dbpart=20`. Alternatively, if the number of query sequences is very large, resubmitting the search with a larger value for `--querypart` may allow the search to complete successfully.
JSON	JavaScript Object Notation (JSON) is a standard text format commonly used in Web service applications. It is comparable to XML, but is more human-readable. The Paracel BLAST REST interface makes use of the JSON format. See: http://www.json.org/
LSF	Short for Load Sharing Facilities. Platform LSF can be used to schedule general purpose compute jobs on a system that is also running Paracel BLAST. For details on the interaction of LSF, BTK and `pbd`, see Semaphores below.
PBFS	Short for Paracel BLAST Filesystem. Paracel has created the PBFS so that the program can manage databases of extremely large size in a manner that is transparent to the user. There are a number of PBFS directory and file management commands with usage similar in those which apply to UNIX filesystems. These commands are described briefly in System Administration Functions starting on p. 13.
PHI-BLAST	Short for Pattern-Hit Initiated BLAST. The PHI-BLAST search combines the matching of regular expressions with local alignments surrounding the match. This allows the user to find database sequences that match a user-defined regular expression and are homologous in the area around the regular expression match.

Term	Description
PHI pattern	A regular expression pattern used in a PHI-BLAST search. The syntax for patterns in PHI-BLAST follows the PROSITE conventions. See the BioView Workbench User Manual or the BioView Workbench on-line help for a detailed description and examples of the syntax.
PSI-BLAST	Short for Position-Specific Iterated BLAST. PSI-BLAST performs an iterative search where sequences found in the first round of searching are used to build a score model for the next round. PHI-BLAST results can be used as the first iteration of a PSI-BLAST search. Otherwise, the default search type, `blastp`, is used. Later iterations use a special matrix that is developed from the previous iterations.
Query chopping, database splitting and scoring	In general, the scores and E-values in Paracel Blast are calculated in the same way as NCBI Blast. Details regarding NCBI's score calculation can be found at: http://www.ncbi.nlm.nih.gov/BLAST/tutorial/Altschul-1.html The optimized Paracel BLAST algorithms will occasionally produce scores that are slightly different than the same HSP's in NCBI. This is due to rounding differences and should have no effect on the overall quality of the results. The PB merge process is invoked for three cases: split querysets, split databases and "chopped" queries. • In a split queryset, separate query sequences in a query file may be distributed to different workers. The same database is used, however, so no adjustments need to be made to the score or the E-value. • Similarly, if the database is split, separate database sequences may be distributed to different workers. As before, no adjustments are made to the score. However, since the E-value calculation requires the combined length of all sequences in the database, the worker uses this combined length for the calculation even though the worker itself will only be using some of the sequences. • When a query is chopped, different portions of the same query sequence may be distributed to different workers. If an alignment extends across two different portions, it is impossible for the individual workers to calculate an accurate score. In this case, the query sequence is re-aligned using NCBI BLAST during the merge process.
REST	Representational State Transfer (REST) is a web services architecture which makes use of the HTTP protocol, providing implementations for PUT, GET, DELETE, etc. Paracel BLAST provides a REST interface for job submission and control. See: https://en.wikipedia.org/wiki/Representational_state_transfer

Term	Description
Semaphores	The semaphore principle is most clearly demonstrated by discussing how BTK and pbd interact on a GMBM worker box. Consider an installation which has only Paracel BLAST and BTK. The desire is to have no more than two daemon processes taking CPU at once since there are only 2 CPU's in each worker box. The processes could be two Aligners and zero `pbd` worker daemons, the reverse, or one of each. Both the BTK Aligner daemon and the `pbd` worker daemon are now compiled with the so-called "parashare" module. This module allows the daemons to communicate using an OS-level semaphore system. When no Aligner or blast jobs are being performed on the worker node, the semaphore equals the number of CPU's. If a job is about to be performed, the daemons will decrement the semaphore by one. If, after it is decremented, the semaphore is greater than one, then the job will proceed. If the semaphore less than 0, then there must be other active jobs, and the to-be-attempted job will block until such time as the semaphore is of the proper value. In the case of LSF, there is a configuration option (the default value) to make use of `with_sem`. That is a wrapper system of sorts, which LSF uses to look at available semaphores before a job is run. In this way, LSF jobs can be run in such a way that only two processes (be they BTK, PB, or LSF) use CPU at one time. LSF will try first to send jobs to unoccupied CPU's.
Web service	A Web service is a software function provided at a network address over the Web with the service always on as in the concept of utility computing. The purpose of the service is to manipulate representations of Web resources using a uniform set of stateless operations. Paracel BLAST is a Web service. See Chapter 7 REST Interface on p. 92.
Web UI	Web User Interface. A Web UI is an HTML-based application used to configure and manage a server appliance from a web browser. A Web UI is an example of a web application. Paracel BLAST provides a Web UI. See Error: Reference source not found on p. Error: Reference source not found.
X dropoff value	The X dropoff value is the maximum drop in the value of the score allowed from the highest point during the ungapped word extension phase.

Index

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

W

X

|