NAME

sem - semaphore for executing shell command lines in parallel

SYNOPSIS

sem [--fg] [--id <id>] [--semaphoretimeout <secs>] [-j <num>] [--wait] command

DESCRIPTION

GNU **sem** is an alias for GNU **parallel --semaphore**.

It works as a tool for executing shell commands in parallel. GNU **sem** acts as a counting semaphore. When GNU **sem** is called with command it will start the command in the background. When *num* number of commands are running in the background, GNU **sem** will wait for one of these to complete before starting another command.

Before looking at the options you may want to check out the examples after the list of options. That will give you an idea of what GNU **sem** is capable of.

OPTIONS

command

Command to execute. The command may be followed by arguments for the command.

--bg

Run command in background thus GNU **parallel** will not wait for completion of the command before exiting. This is the default.

See also: **--fg**

-j *N*

Run up to N commands in parallel. Default is 1 thus acting like a mutex.

--jobs *N*

-j *N*

--max-procs *N*

-P *N*

Run up to N commands in parallel. Default is 1 thus acting like a mutex.

--jobs *+N*

-j *+N*

--max-procs *+N*

-P *+N*

Add N to the number of CPU cores. Run up to this many jobs in parallel. For compute intensive jobs **-j** +0 is useful as it will run number-of-cpu-cores jobs simultaneously.

--jobs *-N*

-j *-N*

--max-procs *-N*

-P *-N*

Subtract N from the number of CPU cores. Run up to this many jobs in parallel. If the evaluated number is less than 1 then 1 will be used. See also **--use-cpus-instead-of-cores**.

--jobs *N*%

-j *N*%

--max-procs *N*%

-P *N*%

Multiply N% with the number of CPU cores. Run up to this many jobs in parallel. If the evaluated number is less than 1 then 1 will be used. See also **--use-cpus-instead-of-cores**.

--jobs *procfile*

-j *procfile*

--max-procs *procfile*

-P *procfile*

Read parameter from file. Use the content of *procfile* as parameter for *-j*. E.g. *procfile* could contain the string 100% or +2 or 10.

--semaphorename *name*

--id *name*

Use **name** as the name of the semaphore. Default is the name of the controlling tty (output from **tty**).

The default normally works as expected when used interactively, but when used in a script *name* should be set. *$$* or *my_task_name* are often a good value.

The semaphore is stored in ~/.parallel/semaphores/

--fg

Do not put command in background.

--semaphoretimeout *secs* (alpha testing)

--st *secs* (alpha testing)

If *secs* > 0: If the semaphore is not released within *secs* seconds, take it anyway.

If *secs* < 0: If the semaphore is not released within *secs* seconds, exit.

--wait

-w

Wait for all commands to complete.

EXAMPLE: Gzipping *.log

Run one gzip process per CPU core. Block until a CPU core becomes available.

```
for i in *.log ; do
  echo $i
  sem -j+0 gzip $i ";" echo done
done
sem --wait
```

EXAMPLE: Protecting pod2html from itself

pod2html creates two files: pod2htmd.tmp and pod2htmi.tmp which it does not clean up. It uses these two files for a short time. But if you run multiple pod2html in parallel (e.g. in a Makefile with make -j) you need to protect pod2html from running twice at the same time. **sem** running as a mutex will do just that:

```
sem --fg --id pod2html pod2html foo.pod > foo.html
sem --fg --id pod2html rm -f pod2htmd.tmp pod2htmi.tmp
```

BUGS

None known.

REPORTING BUGS

Report bugs to <bug-parallel@gnu.org>.

AUTHOR

LICENSE

Documentation license I

Documentation license II

Other Rights

In no way are any of the following rights affected by the license:

- Your fair dealing or fair use rights, or other applicable copyright exceptions and limitations;
- The author's moral rights;
- Rights other persons may have either in the work itself or in how the work is used, such as publicity or privacy rights.

Notice

For any reuse or distribution, you must make clear to others the license terms of this work.

A copy of the full license is included in the file as cc-by-sa.txt.

DEPENDENCIES

GNU **sem** uses Perl, and the Perl modules Getopt::Long, Symbol, Fcntl.

SEE ALSO

parallel(1)