Package para integrar los scripts Perl en SLURM

Objetivo

Comoquiera que el pipeline esta integrado en el *workload manager* se hecha en falta un paquete que permita la ejecucion simple de las tareas con sbatch. **(Si que hay todo tipo de modulos en CPAN pero termino antes si, en lugar de intentar entender uno, lo programo.)** La idea es bastante simples pues siempre hacemos todo de la misma forma.

Paquete

SLURM.pm

```
#!/usr/bin/perl

# Copyright 2021 O. Sotolongo <asqwerty@gmail.com>

# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation; either version 2 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.

use strict; use warnings;
package SLURM;
require Exporter;

our @ISA = qw(Exporter);
our @EXPORT = qw(send2slurm);
our @EXPORT_OK = qw(send2slurm);
our %EXPORT_TAGS =(all => qw(send2slurm), usual => qw(send2slurm));

our $VERSION = 0.1;

sub define_task{
# default values for any task
        my %task;
        $task{'mem_per_cpu'} = '4G';
        $task{'cpus'} = 1;
        $task{'time'} = '2:0:0';
```

```
        my $label = sprintf("%03d", rand(1000));
        $task{'filename'} = 'slurm_'.$label.'.sh';
        $task{'output'} = 'slurm_'.$label.'.out';
        $task{'order'} = 'sbatch --parsable '.$task{'filename'};
        $task{'job_name'} = 'myjob';
        $task{'mailtype'} = 'FAIL,TIME_LIMIT,STAGE_OUT';
        return %task;
}

sub send2slurm{
        my %task = %{$_[0]};
        my %dtask = define_task();
        my $content = '#!/bin/bash\n';
        $content.= '#SBATCH -J ';
        if(exists($task{'job_name'}) && $task{'job_name'}){
                $content.=$task{'job_name'}.'\n';
        }else{
                $content.=$dtask{'job_name'}.'\n';
        }
        if(exists($task{'cpus'}) && $task{'cpus'}){
                $content.= '#SBATCH -c '.$task{'cpus'}.'\n';
                $content.= '#SBATCH --mem-per-cpu=';
                if(exists($task{'mem_per_cpu'}) &&
$task{'mem_per_cpu'}){
                        $content.=$task{'mem_per_cpu'};
                }else{
                        $content.=$dtask{'mem_per_cpu'};
                }
        }
        if(exists($task{'time'}) && $task{'time'}){
                $content.= '#SBATCH --time='.$task{'time'}.'\n';
        }
        if(exists($task{'output'}) && $task{'output'}){
                $content.= '#SBATCH -o='.$task{'output'}.'-%j\n';
        }else{
                $content.= '#SBATCH -o='.$dtask{'output'}.'-%j\n';
        }
        $content.= '#SBATCH --mail-user='."$ENV{'USER'}\n";
        if(exists($task{'partition'}) && $task{'partition'}){
                $content.= '#SBATCH -p '.$task{'partition'}.'\n';
        }
        if(exists($task{'command'}) && $task{'command'}){
                if(exists($task{'mailtype'}) && $task{'mailtype'}){
                        $content.= '#SBATCH --mail-
type='.$task{'mailtype'}.'\n';
                }else{
                        $content.= '#SBATCH --mail-
type='.$dtask{'mailtype'}.'\n';
                }
                $content.= $task{'command'}.'\n';
        }else{
```

```
                $content.='#SBATCH --mail-type=END\n';
                $content.=':\n';
        }
        my $scriptfile;
        if(exists($task{'filename'}) && $task{'filename'}){
                $scriptfile = $task{'filename'};
        }else{
                $scriptfile = $dtask{'filename'};
        }
        open ESS, ">$task{'filename'}" or die 'Could not create slurm
script\n';
        print ESS "$content";
        close ESS;
        my $order;
        my $order;
        if(exists($task{'dependency'}) && $task{'dependency'}){
                $order = 'sbatch --parsable --
dependency='.$task{'dependency'}.' '.$scriptfile;
        }else{
                $order = 'sbatch --parsable '.$scriptfile;
        }
        return qx/$order/;
}
```

Uso

La idea es meter toda la info que se necesita para correr el script en un hash y pasarlo a la funcion *send2slurm*, que se encarga de ejecutarlo.