pytroll-schedule Documentation

Release 0.1.0

Martin Raspaud

Apr 07, 2017

Contents

Reception scheduling of polar weather satellites.

Contents:

CHAPTER 1

About PyTroll-Schedule

Case of One Receiving Station

In the case of a single station, the procedure of scheduling is quite straightforward. However, let us describe it in detail here, such that the background will be set for the more complex case of multiple reception station reception scheduling.

The first step to compute the schedule, is to know which satellites of interest are going to be rising above the horizon during the duration of the schedule. In order to find such cases, we retrieve the orbital information for each satellite of interest and apply orbit prediction using the aiaa sgp4 algorithm (ref). In practice, we use norad tle files (ref) as orbital elements, and the python implementation of the sgp4 algorithm provided in pyorbital (ref). From this, we then obtain a list of the coming overpasses for the station. We define an overpass as a risetime and fall time for a given satellite, during which it will be within reception reach of the station.

Now, we have to find the possible schedules for the station. The set of all overpasses gives us all the reception possibilities for the station. However, many of them will be in conflict with at least one other overpass and will be a concurrent to the reception race. We say that two overpasses conflict when the risetime dog one of them is comprised within the view time of the second. In case of conflicts, the scheduling algorithm has to choose one or the other overpass. However, in the case of several overpasses conflicting sequentially, we have to find the possible paths through the conflicting zone. In order to do that, we will use graph theory algorithms.

We define the graph of the conflicting zone with overpasses as vertices and create an edge between two conflicting overpasses. To find the possible non-conflicting combinations in this graph is actually searching for maximal cliques in the complementary graph, for which we use the Bron-Kerbosch algorithm. #illustration click

we obtain thus groups of passes that are not conflicting in the time frame. The next step is to find the optimal list of non conflicting passes under the duration on the schedule.

Cases of Connected Stations

There are several ways to computate schedules for connected stations, two are implemented in this program.

Several points should be considered: * Technical equipement, reception of L-band, Ku-band, X-band? * Geographic location, nearby or large distance between?

"Master-Slave" Operation

The mode of master-slave is best suited for two stations, located next to each other, with similar technical systems.

In this case a schedule for one, namely the "master" station, would be computed, as if it were only this one station.

In a second step this schedule plan is used as a substraction list when computing the schedule for the second, the "slave" station.

Co-operating Stations

A mode of co-operating stations can consider the distance between different geographical locations and differences in technical equipement, most notable different reception capabilities (X- & L-band vs. L-band).

In this case, each station defines a time span requirement for each pass. Then, if a connected station can fulfil this requirement and is scheduling the same pass, we can say that the stations are redundant. To avoid such redundancy, we can define ways to synchronise the schedule to optimise the intake of data and fulfil the pareto condition. A simple protocol can be used to perform this: both A and B provide alternatives and compute the enhanced score for the schedule including the others pass.

B can delegate the pass only if it can assure that the time span requirement of A is respected.

This operation can be extended to more than two stations, all receiving a single-operation schedule and an individual cooperating-schedule.

CHAPTER 2

Usage

Usage of the schedule script:

```
usage: schedule [-h] [-c CONFIG] [-t TLE] [-l LOG] [-m [MAIL [MAIL ...]]] [-v]
                [--lat LAT] [--lon LON] [--alt ALT] [-f FORWARD]
                [-s START_TIME] [-d DELAY] [-a AVOID] [--no-aqua-dump]
                [--multiproc] [-o OUTPUT_DIR] [-u OUTPUT_URL] [-x] [-r]
                [--scisys] [-p] [-g]

optional arguments:
  -h, --help            show this help message and exit
  -c CONFIG, --config CONFIG
                        configuration file to use
  -t TLE, --tle TLE     tle file to use
  -l LOG, --log LOG     File to log to (defaults to stdout)
  -m [MAIL [MAIL ...]], --mail [MAIL [MAIL ...]]
                        mail address(es) to send error messages to.
  -v, --verbose         print debug messages too

start-parameter:
  (or set values in the configuration file)

  --lat LAT             Latitude, degrees north
  --lon LON             Longitude, degrees east
  --alt ALT             Altitude, km
  -f FORWARD, --forward FORWARD
                        time ahead to compute the schedule
  -s START_TIME, --start-time START_TIME
                        start time of the schedule to compute
  -d DELAY, --delay DELAY
                        delay (in seconds) needed between two consecutive
                        passes (60 seconds by default)

special:
  (additional parameter changing behaviour)
```

```
  -a AVOID, --avoid AVOID
                        xml request file with passes to avoid
  --no-aqua-dump        do not consider Aqua-dumps
  --multiproc           use multiple parallel processes

output:
  (file pattern are taken from configuration file)

  -o OUTPUT_DIR, --output-dir OUTPUT_DIR
                        where to put generated files
  -u OUTPUT_URL, --output-url OUTPUT_URL
                        URL where to put generated schedule file(s), otherwise
                        use output-dir
  -x, --xml             generate an xml request file (schedule)
  -r, --report          generate an xml report file (schedule)
  --scisys              generate a SCISYS schedule file
  -p, --plot            generate plot images
  -g, --graph           save graph info
```

CHAPTER 3

Configuration

The configuration file is divided in several section, each titled with a name in square brackets. All sections are filled with key=value pairs.

Required sections are `default` and `pattern`, all others are referenced by the station list in section `default` and the satellites lists in the sections per station.

Main section "default"

```
[default]
station=nrk,ofb
forward=12
start=1
```

`forward` The timespan in hours the schedule should cover.

`start` Time offset between the time of computing and start of the schedule.

File- and directory pattern

Each of the keys in this section can be referenced from within other lines in this section.

Note: Be carefull not to create loops!

```
[pattern]
dir_output= {output_dir}/{date}-{time}
dir_plots= {dir_output}/plots.{station}
file_xml= {dir_output}/aquisition-schedule-{mode}_{station}.xml
file_sci= {dir_output}/scisys-schedule-{station}.txt
file_graph= {dir_output}/graph.{station}
```

time, date Time+date, when the computation started.

station Replaced with the station name. For co-operating stations "combined schedule" the string ".comb" is appended.

mode If a schedule file was created in *request* mode or in *report* mode. The first one is the format accepted by SciSYS software, while *report* mode is good for monitoring purposes.

output_dir This placeholder references the command-line argument `-o OUTPUT_DIR`.

dir_output This key is used only within this section.

dir_plots Where the plots (globe grafic files) are saved.

file_xml Path and filename format for xml-schedule (request and report).

file_sci Path and filename for schedule file in SciSYS format.

file_graph Graph files with information about the computation.

Stations

```
[ofb]
name=offenbach
longitude=8.747073
latitude=50.103095
altitude=0.140
area_file=/home/troll/etc/areas.def
area=euro4
satellites=metop-a,metop-b,noaa 19,noaa 18,noaa 15,aqua,terra,suomi npp
```

name Name of the station.

longitude Longitude in degrees east.

latitude Longitude in degrees north.

altitude Altitude above mean sea level, in km.

area_file, area File with area definitions, and the area referenced therein. This area is taken into computation, only satellite passes which swaths are cross-sectioning this area are considered for scheduling.

satellites List of satellites receivable from this station. The listed names refer to the satellite sections.

```
[nrk]
name=norrkoeping
longitude=16.148649
latitude=58.581844
altitude=0.052765
area_file=/home/troll/etc/areas.def
area=euron1
satellites=metop-a,metop-b,noaa 19,noaa 18,noaa 15,aqua,terra,suomi npp
```

While the above section contained values for the station Offenbach/Germany, this section has values for Norkoepping/Sweden.

Satellites

```
[metop-a]
night=0.1
day=0.6

[noaa 19]
night=0.05
day=0.3

[terra]
night=0.2
day=0.8

[suomi npp]
night=0.25
day=0.9
```

A few examples for satellite sections.

night Weight value for satellite swath parts on the night-side of the terminator.

day Weight value for satellite swath parts on the day-side of the terminator.

CHAPTER 4

Example

This Bash-script shows an example how to use the schedule script:

```
#!/usr/bin/bash
. $HOME/.bash_profile
bin=$HOME/.local/bin/schedule
conf=$PYTROLL_BASE/etc-schedule
logp=$PYTROLL_BASE/log/scheduler.log
logc=$PYTROLL_BASE/log/create.log
odir=$PYTROLL_BASE/schedules

cfgfile=$1
shift

# report-mode with plots
mode="-r -p"

# min time-diff btwn passes
delay="90"

# create output for scisys
sci="--scisys"

# dont include aqua-dumps
aqua="--no-aqua-dump"

# write gv-files for dot
graph="-g"

# exit if no new tle
(( `ls $PYTROLL_BASE/tle/*.tle 2>/dev/null|wc -l` > 0 )) || exit 0

# catch newest TLE-file, remove others
tle=`ls -t $PYTROLL_BASE/tle/*.tle|head -1`
mv $tle $tle.txt
rm -f $PYTROLL_BASE/tle/*.tle
```

```
# settings for TLE-check
satlst="NOAA 15|NOAA 18|NOAA 19|METOP-A|METOP-B|AQUA|SUOMI|TERRA "
satcnt=8
to_addr=pytroll@schedule

# check if TLE-file is complete
if (( `grep -P "$satlst" $tle.txt|tee .tle_grep|wc -l` != $satcnt )); then
    exit 0
else
    tle="-t $tle.txt"
fi

# start schedule script
$bin -v -l $logp -c $conf/$cfgfile $tle -d $delay -o $odir $graph $mode $sci $aqua $@
```

CHAPTER 5

Indices and tables

- genindex
- modindex
- search