Scheduller 2.1 用户手册

〇、运行环境

- Python版本：>=3.8
- 运行库：
  - astropy
  - matplotlib
  - numpy
  - Tkinter

  *默认的Anaconda环境包括以上所有所需的运行库。

一、源表编写

脚本的输入为源表，是一个二维的JSON格式。

1.1 JSON第一层：观测参数

参数名	变量名	格式	必须
观测开始时间	obs_start	YYYY.MM.DD hh:mm:ss	是
观测结束时间	obs_end	YYYY.MM.DD hh:mm:ss	是
望远镜位置	tele_loc	[Lat., Lon., Hei.]	是
观测高度范围*	elev_range	[min., max.]	是
最小太阳角度*	escape_sun	int. *(Deg. From Sun)*	是
观测源表	sources	*（见下一部分）*	是

*观测高度范围：指望远镜最大与最小仰角；

*最小太阳角度：指望远镜能运行观测的距离太阳的最小角度。

1.2 JSON第二层：观测源表

参数名	变量名	格式	必须
观测源名	identifier	源名	是
观测长度	dur	int.	是
权重	weight	float (0–1)	否（默认：1）
绝对优先	force	bool (False:0 True:1)	否（默认：0）

1.3 举例

```
{
    "obs_start": "2021.07.24 02:00:00", #Format: "YYYY.MM.DD hh:mm:ss"
    "obs_end": "2021.07.24 11:00:00", #Format: "YYYY.MM.DD hh:mm:ss"
    "tele_loc": [32.701500, -109.891284, 3185],
                #Format: [Lat., Lon., Hei.] (AZ Tele.)
    "elev_range": [30, 80], #Format: [min., max.] (AZ Tele.)
    "escape_sun": 10, #Format: [Deg. From Sun]
    "sources":
        [
            {"identifier": "G005.88-00.39", "dur": 1200, "weight": 1},
            {"identifier": "J1935+2154", "dur": 1200, "weight": 0.1},
            {"identifier": "J1935+2154", "dur": 1200, "force": 1},
            ...
            {"identifier": "G160.14+03.15", "dur": 1200}
            #源名(如："J1935+2154")           观测长度(如：1200)
        ]
} #脚本中输入的JSON文件中支持加入"#"开头的注释
```

二、源坐标离线数据库

源坐标数据库为解决Astropy包需联网获取脉冲星坐标的问题（部分电脑未联网）。和源表类似，源坐标数据库是一个JSON格式的数组文件，需要以“sources.db”文件名与脚本存放于同一目录下。

源坐标数据库结构如下：

```
[
    {
        "RA": "[坐标R.A.]",
        "DEC": "[坐标Dec.]",
        "IDENTIFIER": [
            "[源名1]", "[源名2]", ...
        ]
    },
    ...
    {
        "RA": "00:06:04.8",
        "DEC": "+18:34:59",
        "IDENTIFIER": [
            "J0006+1834"
        ]
    },
    {
        "RA": "00:07:01.7",
        "DEC": "+73:03:07.4",
        "IDENTIFIER": [
            "J1807-0847", "B1804-08"
```

```
        ]
    },
]
```

源坐标可以自动生成，具体方法请见本文档“使用技巧”部分”快速生成离线数据库“中的相关说明。

三、开始使用

1. 打开命令行，切换到脚本所在目录（cd [PATH]）
2. 运行python命令“python [脚本文件名] [源表（配置文件）]”
   举例：

```
python SchedullerVersion2.0.py psr_list.txt
```

3. 待脚本运行完毕之后会自动弹出最佳编排结果。 若没有找到可行的编排结果，脚本会在提示“[Warning] No valid schedule found.”后退出。

四、脚本中可使用的命令

【在脚本运行完毕后，当显示“[scheduller] >”时，即可开始输入命令（通常需要在关闭自动弹出的图表窗口后输入）】
【通常在窗口弹出后需关闭窗口后再输入命令。】

4.1 预览结果：preview

通常，输出结果会包含多种排列组合的可能性。通过preview命令可以预览不同的组合情况。

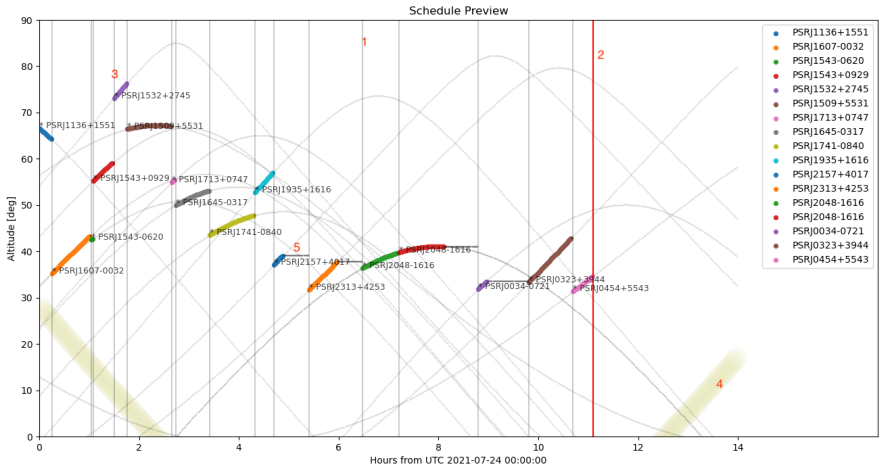
4.1.1 使用方法

```
[Scheduller] > preview [编号]
```

其中，“[编号]”为需要预览的组合编号，编号越靠前意味着组合越优（打分越高）。也就是说，如果需要预览最优组合需要使用命令：

```
[Scheduller] > preview 1
```

4.1.1 图表说明



上图是脚本生成的纲要预览图，其中：

- 数字“1”处灰色的竖线为前一个源观测截止、后一个源观测开始的分界处；
- 数字“2”处红色的竖线为整个纲要观测结束时间；
- 数字“3”处被加粗的线条为当前被观测的源；
- 数字“4”处为太阳高度；
- 数字“5”处的黑色细横线为源于源之间的Gap。

4.2 保存结果：save

通过save命令，可以保存上一次预览的编排结果。也就是说，如果上一次预览了编号为“1”的编排结果（”> preview 1”），使用save命令输出并保存的内容就是编号为“1”的编排结果。

4.2.1 使用方法

```
[Scheduller] > save [格式] [文件名]
```

其中，“[格式]”为编排结果输出格式（见下表），”[文件名]”为输出结果保存的文件名。

格式	命令	备注
Sched (*.key) 格式	”sched”, ”.key”	输出的结果仅为.key文件中编排观测的部分。

举例：

```
[Scheduller] > save sched sorted.txt
```

4.3 显示信息：show

通过show命令，可以显示上一次预览的编排结果的相关信息。也就是说，如果上一次预览了编号为“1”的编排结果（”> preview 1”），使用show命令查看的信息就是编号为“1”的编排信息。

4.3.1 scheduled/not scheduled

通过show命令后使用scheduled/not scheduled参数查看编排结果中编排上/未被编排上的源名。

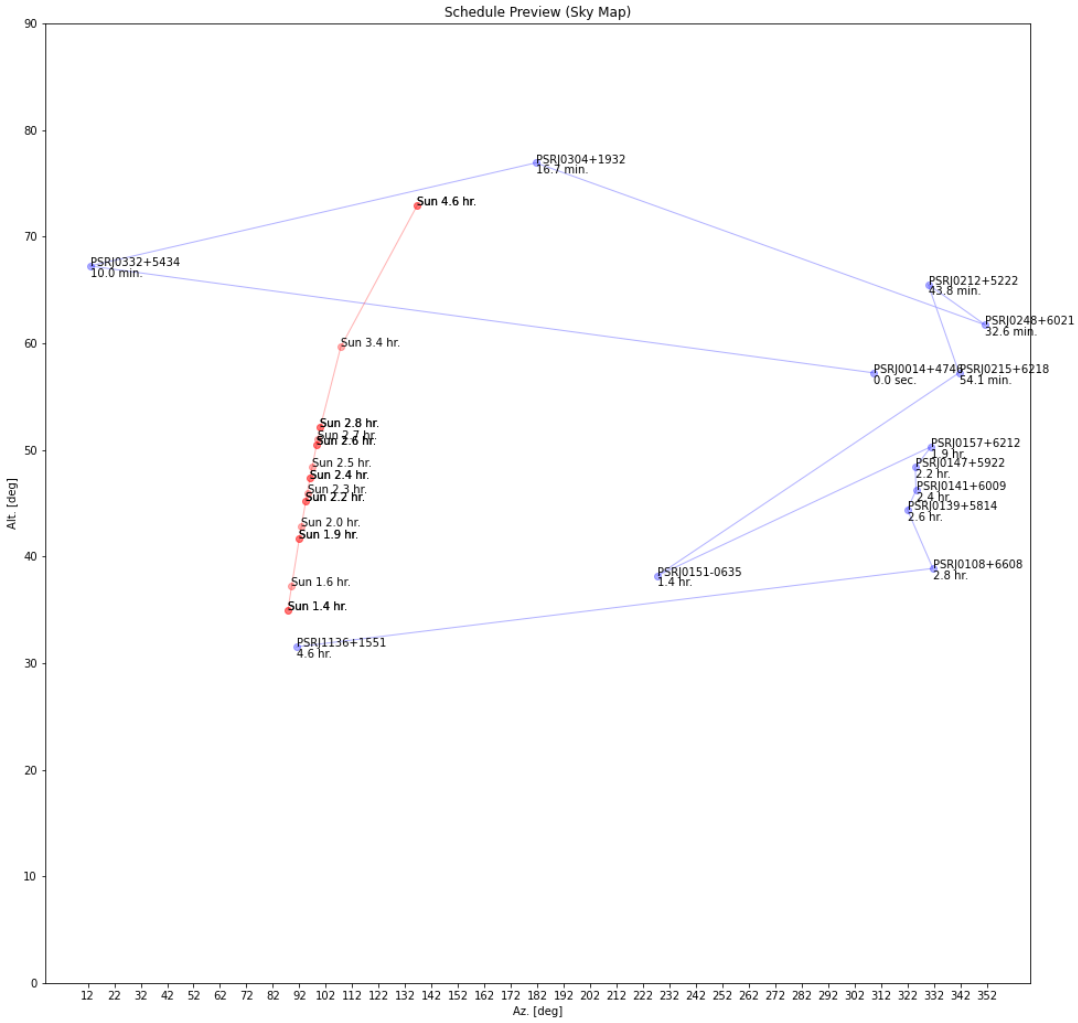
```
[Scheduller] > show scheduled
J0014+4746 J0332+5434 J0304+1932 J0248+6021 J0212+5222
[Scheduller] > show not scheduled
J0152-1637 J0055+5117 J0034-0721 J0134-2937 J1640+2224
```

4.3.1 skymap

通过show命令后使用skymap参数查看编排结果的在空中的方位、高度、距离太阳的距离。

```
[Scheduller] > show skymap
```

脚本会返回一张x轴为方位角、y轴位高度的图：



在源名的下方、太阳方位的右侧写出了其在此位置的时间。用过将源的观测时的太阳位置和源的位置对比可以得出距太阳距离；也可以通过该图看出望远镜的观测路径。当太阳位置超出观测范围将会被隐藏。

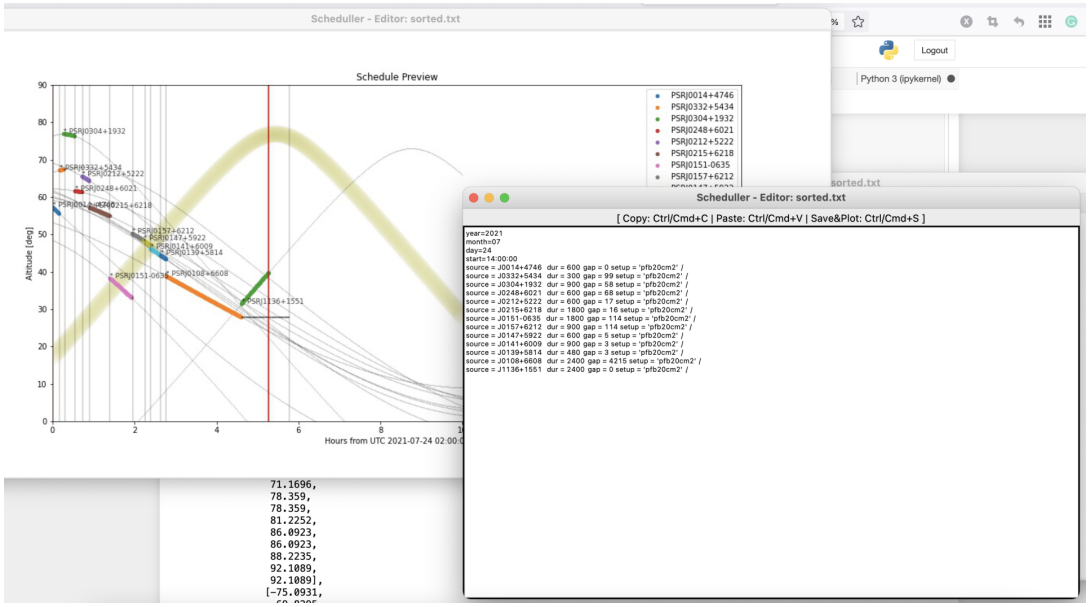
4.4 编辑纲要：edit

通过edit命令，可以编辑编排结果。**但是请注意，edit命令需要在保存sched格式的文件（save命令）之后编辑被保存的sched文件。**

```
[Scheduller] > save sched sorted.txt
[Scheduller] > edit sorted.txt
```

4.4.1 编辑器的使用

edit命令使用后稍等片刻，脚本会弹出Scheduller编辑器（如下图，编辑器启动时会生成纲要预览题）。



交换行与行的顺序

快捷方式为Alt/Cmd+上箭头或Alt/Cmd+上箭下，分别为将光标所在行向上移动或向下移动。

刷新纲要预览图（Save & Plot）

在编辑Sched文件后，可以使用编辑窗口左上角的“File > Save & Plot”刷新预览图。
（若系统支持，可以使用Ctrl/Cmd + S快捷键）

生成SkyMap（Save & Show SkyMap）

在编辑Sched文件后，可以使用编辑窗口左上角的“File > Save & Show SkyMap”查看SkyMap。
（若系统支持，可以使用Ctrl/Cmd + P快捷键）

计算Gap（Save & Compute Gaps）

在编辑Sched文件后，可以使用编辑窗口左上角的“File > Save & Compute Gaps”刷新源与源之间的Gap长度。**注意，此时的源与源Gap仅由望远镜旋转时长计算。**

重新加载纲要（reload）

将纲要撤回到上次保存时的版本。

窗口置顶（Always on Top）

启动编辑器后，可以使用编辑窗口左上角的“Window > Always on top”将窗口置顶。

4.4 退出脚本：exit

通过命令“exit”退出脚本（效果等同于Ctrl+C）。

```
[Scheduller] > exit
```

五、使用技巧

5.1 脚本分段运行

因为Python脚本通常只会占用一个CPU线程，可以通过脚本分段运行最大利用CPU。

5.1.1 使用场景

当我们有约300颗源需要尝试编排进12小时观测时，脚本需要长时间运行。我们可以通过分段运行减少脚本运行时间。
我们的算法意味着300颗源有300种组合的可能性（见附录），因此我们可以启动6个python进程，每个进程计算50种可能性（6*50=300）。
我们可以在脚本启动命令之后添加参数将脚本分段：

```
python SchedullerVersion2.0.py psr_list.txt [开始] [结束]
```

因此，

```
% 在脚本目录下启动新的Shell窗口，并运行命令：
python SchedullerVersion2.1.2.py psr_list.txt 1 50
% 在脚本目录下再次启动新的Shell窗口，并运行命令：
python SchedullerVersion2.1.2.py psr_list.txt 50 100
% 在脚本目录下再次启动新的Shell窗口，并运行命令：
python SchedullerVersion2.1.2.py psr_list.txt 100 150
% 在脚本目录下再次启动新的Shell窗口，并运行命令：
python SchedullerVersion2.1.2.py psr_list.txt 150 200
% 在脚本目录下再次启动新的Shell窗口，并运行命令：
python SchedullerVersion2.1.2.py psr_list.txt 200 250
% 在脚本目录下再次启动新的Shell窗口，并运行命令：
python SchedullerVersion2.1.2.py psr_list.txt 250 300
```

即可实现分段处理。
在每一段脚本运行结束后，可以通过分别查看每个窗口的最佳编排结果的得分评判哪一个是最佳纲要。

```
Sorting... Done    ] (4.2 hr. rem.) > [SORTING 100/327] Start \w J1543-0620
1627419558.6605692
Plotting the optimal schedule...# 0 Previewing [ Key 20 ]
Score: 9.604888041400415
Scheduled  59 out of 327
```

*分段运行脚本版本要求 >= 2.1.2

5.2 快速生成离线数据库

当脚本在联网计算机上运行时，若在离线数据库中没有找到源表中的源时，会联网在SIMBAD中下载源坐标（由Astropy实现）。坐标下载后会将其写入进离线数据库中，且下次遇到同样的源时不会再次联网下载。
我们可以利用这个功能快速生成离线数据库。将所有可能用到的源名按本文档第一部分的格式写进源表，并在联网计算机中运行脚本。脚本会将数据库中没有的源进行补充（若没有数据库，脚本会自动生成数据库文件）。直到脚本显示“Preprocessing... Done”位置，源表加载结束，数据库生成完成。

5.3 在预览图像的同时运行其他命令

可以注意到，当脚本在使用了“preview”或“show skymap”命令后，会弹出新窗口显示图像，同时命令行会卡住不能执行其他命令。这是因为Python单线程运行的原因。
脚本中内置了由_thread包实现的多线程运行模式。不过请注意，在实测中发现该模式在一些计算机上会出现”Dore dumped”的错误，并导致脚本退出。因此，请慎用此方法。
若需要多线程运行请将命令“preview”替换为“preview_”、“skymap”替换为“skymap_”。

```
[Scheduller] > preview_ 1
[Scheduller] > show skymap_
```

当然，在编辑器中（edit）也能够实现显示多张图像并同时编辑的功能，相比多线程模式更加稳定。

附录：算法原理

我们的目的是让计算机可以通过软件排序算法自动生成一个相对较实用的观测纲要。主要通过python编程实现，使用到的库包括astropy, matplotlib, numpy等。 要实现我们的算法，第一步我们先基于源与源之间的相对距离最短进行排序。在确定第一颗源后，通过三角函数来计算剩下的源中距离上一颗源最近的一颗排序。以这样的贪心算法遍历所有剩下的源来组成序列。这里会遇到一个问题就是我们无法确定在很多个源中，以哪一个源作为这个序列的第一个源是最优的。于是我们分别用这些源中的每一颗源来作为第一颗源并用上述的方法排序。这样我们会首先得到源的数量个不同的序列。例如有我们有30颗源需要排序，我们先以每一颗源做一次序列开头，然后用贪婪算法来寻找以它开头依次最短距离的下一刻源来排序。我们会得到30个可能的排列顺序。

接下来我们要对这些源数量个的序列进行处理来得到我们想要得到的最优序列。首先对于一个我们刚刚经过距离排序得到的序列，我们再次将它按照源升起落下的时间来排序，也就是源在地平线上的时间由短到长排序。如果把第一步按照距离排序的序列称为a的话，现在的操作是要按照源在天空中的时间长短对a序列再排序生成一个b序列。然后我们要确定我们的观测时间从而进行源的排序。首先对于第一步按照距离生成的序列a中的第一颗源安排观测时间，这时根据这颗源以及需要观测它的时间会有两种结果生成——即可以安排到时间和无法安排。

如果是第一种结果，可以安排到时间——我们需要继续判断刚才按照源在地平线上的时间由短到长排序的序列b中的第一颗源（也就是升起时间最短的源）在a的第一颗源被安排到时间之后是否还有时间被安排。也就是说我们需要判定a序列的这颗源被安排的时间是否占据了升起时间最短的这颗源的升起时间，从而导致升起时间最短的这颗源无法再被观测到。如果在a的第一颗源被安排到时间的情况下不会影响b序列中第一颗源的时间的话，我们就将这颗源放进观测计划中安排时间。如果影响了b序列中第一颗源的时间，我们就优先先将升起时间较短的这颗源先安排。

如果是第二种结果，无法被安排到时间——那我们就从序列a的第二颗源开始，与序列b的源的顺序做判定。看安排序列a的第二颗源之后是否会影响b序列中其他较短升起时间的源。

根据上面的方法，我们一次遍历序列a中的每一颗源来进行排序。并且让我们在第一步的按照距离排出的源数量个序列都进行一次这样的排序。经过上述程序，我们会得到源数量个新的序列，这时再对这每一个序列进行打分，得分最高的将是系统给出的最优排序。打分规则如下：

1. 基于时间利用率打分：

$$得分=\frac{被系统排上的观测时间}{总观测时间}$$

2. 基于望远镜旋转时间长度打分：

$$得分=\frac{1}{总\ Gap\ 望远镜转动所需时间)}$$

这里的gap指望远镜从一个源的位置转到下一个源的位置所需要的时间，对于天马来说，我们采用方位每秒转0.5度，俯仰每秒钟0.3度的速度来计算。

综合上面的两个打分公式，我们可以对这些源数量个序列按照分数高低排出一个降序的顺序，系统给出的第一个方案即为分数最高，也就是相对最优的方案。

权重：

在实际的观测情况中，我们有时候会有一些一定想要观测，或是需要长期重点观测的源。这时候权重的存在会帮助我们实现这个需求。在我们设计的系统中，我们从0到1来表示对应源的权重比例，越靠近1表示权重等级越高。在这个系统里，我们设置了两个用来计算权重的公式。第一个公式是在我们生成以源在地平线上的时间由短到长排序的时候加入，用 $\frac{升起时间\ 源在地平线上的持续时间)}{权重\ (0-1\ )}$ 来实现加入权重的排序。第二个公式在计算完源与源之间的gap（望远镜转动花的时间）后加入，用 $\frac{Gap\ 望远镜转动时间)}{权重\ (0-1\ )}$ 来加入权重排序。

躲避太阳：

当源被编排上的观测时间短与太阳的夹角小于一定设定的值，将放弃把源排进该时间段。