



中国科学技术大学

University of Science and Technology of China

模拟重建与Shell脚本入门

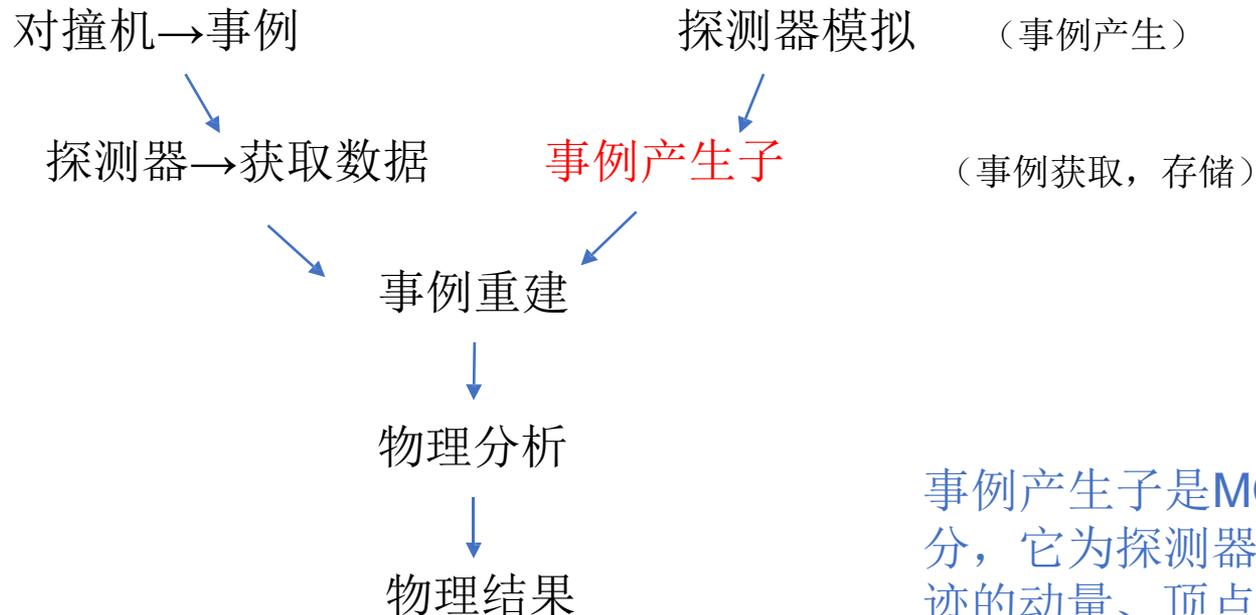
张越鹏

University of Science and Technology of China

2024/8/1

Monte Carlo (MC) 方法是按抽样调查法求取统计值来推定未知特性量的计算方法。基于此，可以对复杂的物理系统进行模拟，以近似得到真实的情形。

粒子物理实验中的一般流程



事例产生子是MC模拟的一个重要组成部分，它为探测器的事例模拟提供各条径迹的动量、顶点和粒子种类的信息

编写产生子（一般用现成的即可）



编写衰变卡（decay card）



.dec

模拟（simulation）



.rtraw

重建（reconstruction）



.dst

分析（analysis）.cxx



.root

\$TESTRELEASEROOT/run/jobOptions_sim.txt

```

1 //DENG Zi-yan 2008-03-17
2
3 #include "$OFFLINEEVENTLOOPMGRROOT/share/OfflineEventLoopMgr_Option.txt"
4
5 //*****job options for generator (KKMC)*****
6 #include "$KKMCROOT/share/jobOptions_KKMC.txt"
7 KKMC.CMSEnergy = 3.097;          束流质心能量
8 KKMC.BeamEnergySpread=0.0008;  束流能散
9 KKMC.NumberOfEventPrinted=1;
10 KKMC.GenerateJPsi=true;       共振态产生类型
11
12 //*****job options for EvtGen*****
13 #include "$BESEVTGENROOT/share/BesEvtGen.txt"
14 EvtDecay.userDecayTableName = "rhopi.dec";  衰变卡
15
16 //*****job options for random number*****
17 BesRndmGenSvc.RndmSeed = 100;  随机数种子
18
19 //*****job options for detector simulation*****
20 #include "$BESSIMROOT/share/G4Svc_BesSim.txt"
21
22 //configure for calibration constants
23 #include "$CALIBSVCROOT/share/calibConfig_sim.txt"
24
25 // run ID
26 RealizationSvc.RunIdList = {-9989};  Run号
27
28 #include "$ROOTIOROOT/share/jobOptions_Digi2Root.txt"
29 RootCnvSvc.digiRootOutputFile = "rhopi.rtraw";  产生文件的路径
30
31
32 // OUTPUT PRINTOUT LEVEL
33 // Set output level threshold (2=DEBUG, 3=INFO, 4=WARNING, 5=ERROR, 6=FATAL )
34 MessageSvc.OutputLevel = 5;  信息输出等级
35
36 // Number of events to be processed (default is 10)
37 ApplicationMgr.EvtMax = 50;  事例数
    
```

产生子

Decay 母粒子的名称

Br x1 x2 ..xn 衰变模型(参数);

Enddecay.....End

根据需要使用的数据，找到对应的run号填在{}中。

例如：{-9947,0,-10878}

查询网址：

<https://docbes3.ihep.ac.cn/~offlinesoftware/index.php/Production>

衰变卡 ($\$TESTRELEASEROOT/run/rhopi.dec$)

```
2 Decay J/psi
3   0.3333 rho0 pi0 HELAMP 1.0 0.0 0.0 0.0 -1.0 0.0;
4   0.3333 rho+ pi- HELAMP 1.0 0.0 0.0 0.0 -1.0 0.0;
5   0.3333 rho- pi+ HELAMP 1.0 0.0 0.0 0.0 -1.0 0.0;
6 Enddecay
7
8 End
```

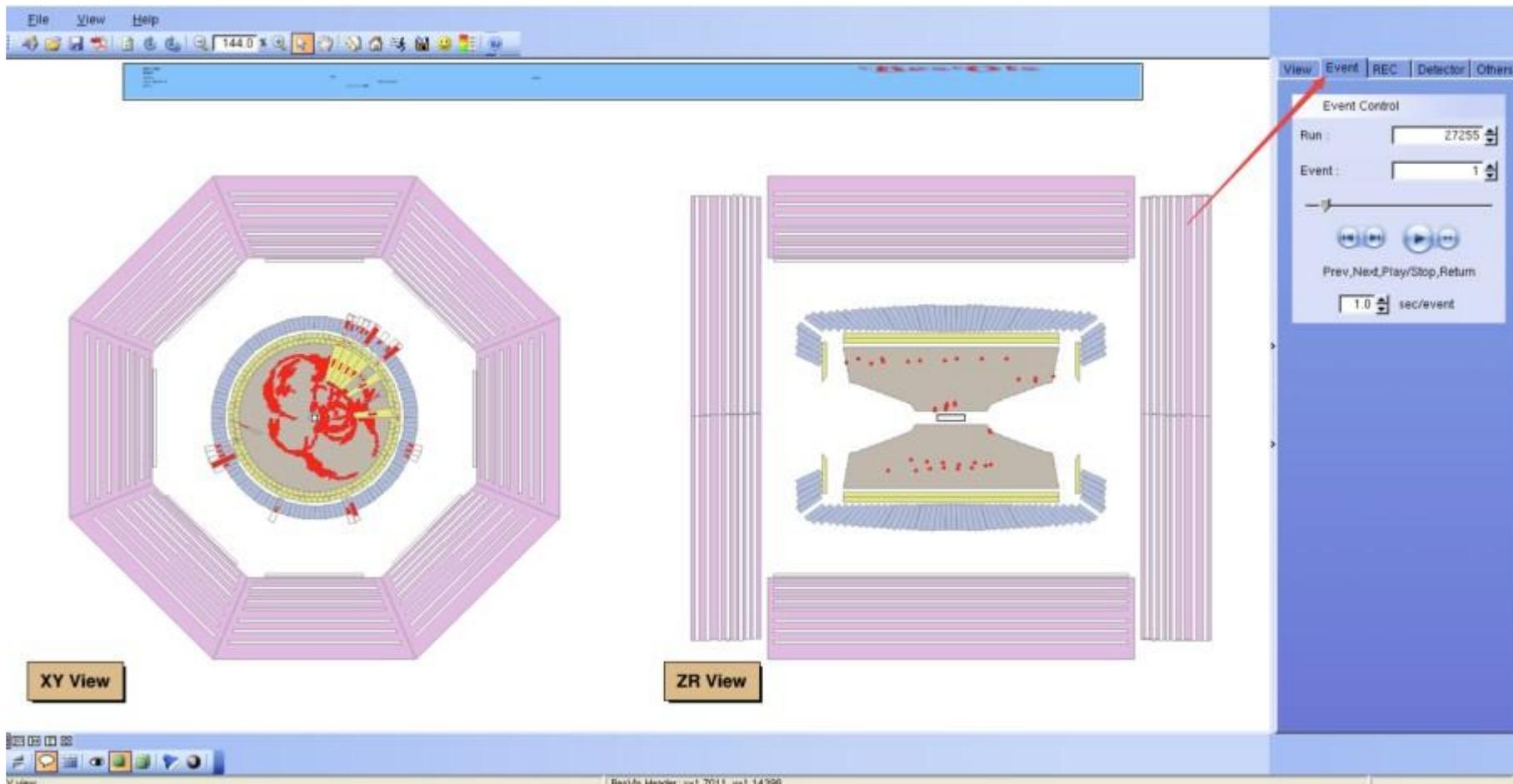
注: 1. 衰变卡片中粒子的名称必须按照EvtGen粒子表(可见:
 $\$BESEVTGENROOT/share/pdt.table$)中的定义填写。

2. 衰变模型必须是EvtGen中的注册模型(可见:
 $\$BESEVTGENROOT/share/DECAY.DEC$), 其引用及参数必须按照手册中的格式要求填写。

3. 如果某个母粒子的衰变道分支比之和不等于1, EvtGen平台将会对这些道的分支比重新归一。

4. 如果在DECAY.DEC中找不到的过程, 暂用PHSP模型。

模拟产生.rtraw文件。可用besvis.exe查看图像

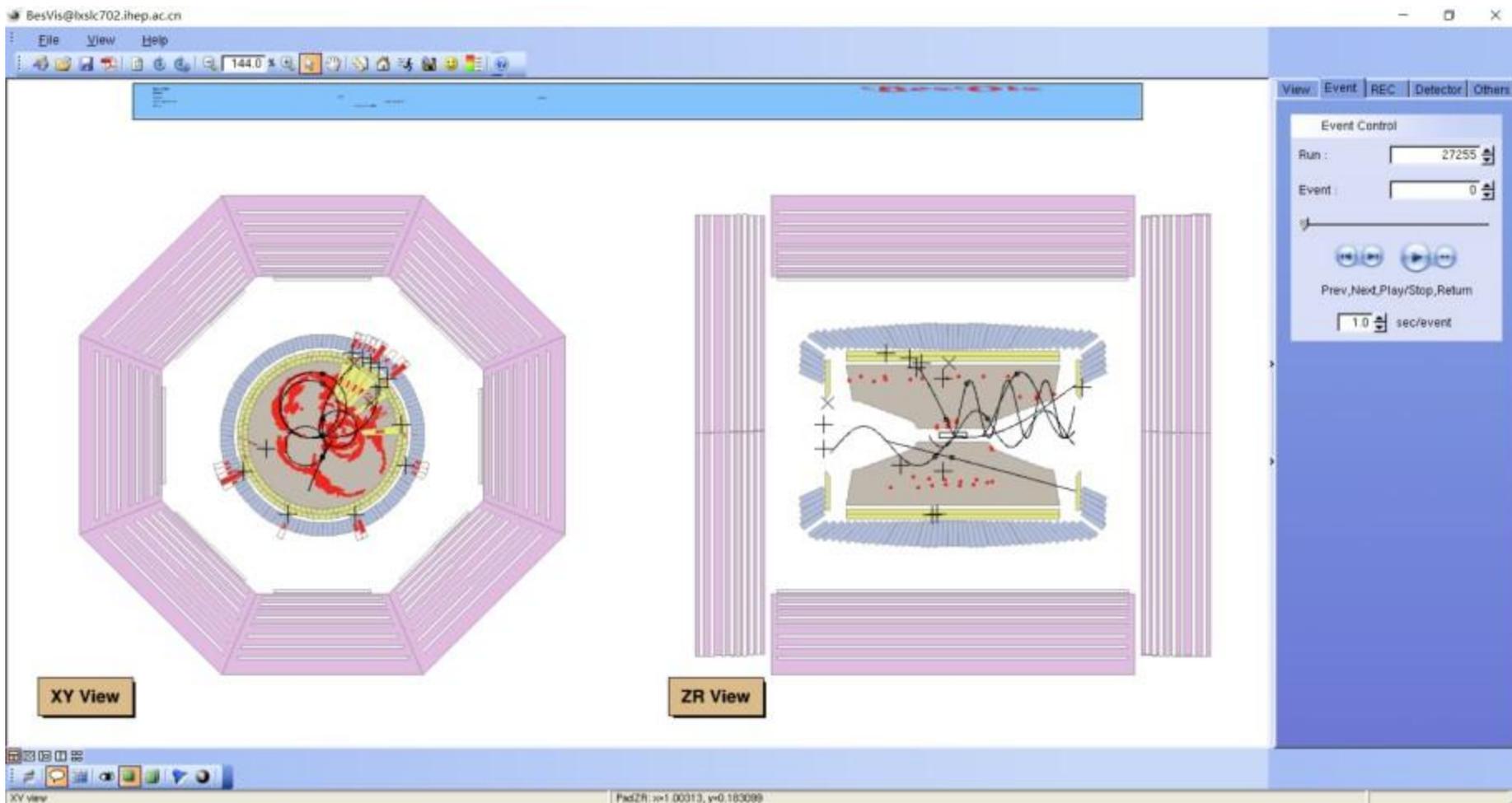


\$TESTRELEASEROOT/run/jobOptions_rec.txt

```
//*****job options for random number*****  
BesRndmGenSvc.RndmSeed = 100; 随机数种子, 与对应的sim文件保持一致  
  
//Set output level threshold (2=DEBUG, 3=INFO, 4=WARNING, 5=ERROR, 6=FATAL )  
MessageSvc.OutputLevel = 2; 信息输出等级, 越高信息量越少  
  
//ROOT input data file  
EventCnvSvc.digiRootInputFile = {"rhopi.rtraw"}; 输入文件, 即模拟的输出文件  
  
//ROOT output data file  
EventCnvSvc.digiRootOutputFile = "rhopi.dst"; 模拟的输出文件  
  
//Number of events to be processed (default is 10)  
ApplicationMgr.EvtMax = 50; 事例数
```

- 注:**
- 1.重建的事例数, 如果写-1则默认与模拟文件中的事例数一致。
 - 2.若不更改输出文件的路径, 则默认为程序运行的位置。

重建产生.dst文件。可用besvis.exe查看图像



\$TESTRELEASEROOT/run/jobOptions_rec.txt

```
1 #include "$ROOTIROOT/share/jobOptions_ReadRec.txt"
2 #include "$VERTEXFITROOT/share/jobOptions_VertexDbSvc.txt"
3 #include "$MAGNETICFIELDROOT/share/MagneticField.txt"
4 #include "$ABSCORROOT/share/jobOptions_AbsCor.txt"
5 #include "$RHOPIALGROOT/share/jobOptions_Rhopi.txt" 放入自己的分析算法
6
7 // Input REC or DST file name
8 EventCnvSvc.digiRootInputFile = {"rhopi.dst"}; 重建输出文件的路径
9
10 // Set output level threshold (2=DEBUG, 3=INFO, 4=WARNING, 5=ERROR, 6=FATAL )
11 MessageSvc.OutputLevel = 5;
12
13 // Number of events to be processed (default is 10)
14 ApplicationMgr.EvtMax = 50; 事例数, -1表示全部事例
15
16 ApplicationMgr.HistogramPersistency = "ROOT"; 输出文件的路径
17 NTupleSvc.Output = { "FILE1 DATAFILE='rhopi_ana.root' OPT='NEW' TYP='ROOT'"};
```

\$RHOPIALGROOT/share/jobOptions_Rhopi.txt

```
1 #include "$VERTEXFITROOT/share/jobOptions_VertexDbSvc.txt"
2 ApplicationMgr.DLLs += {"RhopiAlg"};  算法包的名称
3 ApplicationMgr.TopAlg += {"Rhopi"};  算法程序.cxx的名称
4
5 Rhopi.Vr0cut = 1.0;
6 Rhopi.Vz0cut = 5.0;
7
8 Rhopi.EnergyThreshold = 0.04;
9 Rhopi.GammaPhiCut = 20.0;
10 Rhopi.GammaThetaCut = 20.0;
11 Rhopi.GammaAngleCut = 20.0;
12
13 Rhopi.Test4C = 1;
14 Rhopi.Test5C = 1;
15 Rhopi.CheckDedx = 1;
16 Rhopi.CheckTof = 1;
```

“开关”

/cvmfs/bes3.ihep.ac.cn/bes3sw/Boss/7.0.8/Analysis/Physics/RhopiAlg/
RhopiAlg-00-00-23/src/Rhopi.cxx

注：如果在科大服务器上运行作业（包括模拟重建分析），需添加以下这句话：
DatabaseSvc.Host="10.1.2.12";

- 作业提交 `hep_sub jobscript`

`jobscript`: 作业脚本名，可以是绝对路径文件名也可以是相对路径文件名

- 作业状态查询 `hep_q -u`

`-u`: 指定查看某用户的作业，默认为当前用户。

例如: `hep_q -u` 可以查看自己的作业

如果提交标准的**boss**作业，可使用更简化的**boss.condor**命令

例: `boss.condor joboptions.txt`

- 作业删除 `hep_rm jobs`

`jobs`: 指定要删除的作业id，支持指定多个作业id同时删除。

例如: `hep_rm 12345 12345.6`

`hep_rm -a`: 删除当前用户所有作业

参考: [HTCondor作业](#)

<http://afsapply.ihep.ac.cn/cchelp/zh/local-cluster/jobs/HTCondor/>

- 查看作业时长限制:

`hep_clus -g bes --walltime`

实验	短作业(short)时长限制(小时)	普通作业时长限制(小时)	mid作业时长限制(小时)及资源使用量限制(百分比)
BES	<0.5	<40	<100:10%



HepJob涉及的所有命令都在以下目录，建议将该目录加入用户环境变量 `PATH` 中：

bash 用户

```
$ export PATH=/afs/ihep.ac.cn/soft/common/sysgroup/hep_job/bin:$PATH
```

tcsch 用户

```
$ setenv PATH /afs/ihep.ac.cn/soft/common/sysgroup/hep_job/bin:$PATH
```

在科大服务器需要用这句命令配置hepjob的环境

```
source /cvmfs/common.ihep.ac.cn/software/hepjob/setup_hepjob.csh ustrc
```

我们可能会遇到从大批量的文本中读取信息的情况，如果要一个个文件的检查可能会十分的复杂繁琐。

grep: 在目标文件中查找对应文本

```
grep -i(c L r) "word" filename
```

-i 不区分大小写 **-c** 输出匹配的行数 **-L** 输出不包含该文本的文件
-r 对文件下的所有文件进行查找

```
⊗ bash-4.2$ grep 'INFO Application Manager Terminated successfully' *bosslog
Hybrid_rec_30800_001.txt.bosslog:ApplicationMgr      INFO Application Manager Terminated
successfully
Hybrid_rec_30800_002.txt.bosslog:ApplicationMgr      INFO Application Manager Terminated
successfully
Hybrid_rec_30800_003.txt.bosslog:ApplicationMgr      INFO Application Manager Terminated
successfully
Hybrid_rec_30800_004.txt.bosslog:ApplicationMgr      INFO Application Manager Terminated
successfully
Hybrid_rec_30800_005.txt.bosslog:ApplicationMgr      INFO Application Manager Terminated
successfully
Hybrid_rec_30800_006.txt.bosslog:ApplicationMgr      INFO Application Manager Terminated
successfully
```

我们可能会遇到需要大批量编辑文本信息的情况，如果要一个个文件的编辑就会十分的繁琐。

sed: 流编辑器，可以对文本进行替换删除等操作

```
sed -i 's/old/new/g' filename
```

将文本中的所有‘old’字符替换成‘new’

s代表替换 p表示输出 i在行前插入 d删除

-e 多个编译操作 -i 直接修改文件内容 -n取消默认输出只输出经过编辑的行

awk: 可以处理较复杂的文本操作

```
awk '$3 > 50 {print $1, $2}' test
```

```
● bash-4.2$ cat test  
100 230 82 232  
11 13 31 110  
79 23 84 54
```

```
● bash-4.2$ awk '$3 >50 {print$1,$2}' test  
100 230  
79 23
```

你好，你能帮我写一个输出文本中每行的最后一个字符串的命令吗？



复制 重试 like dislike

当然可以！您可以使用 `awk` 命令来输出文本文件中每行的最后一个字符串。以下是一个示例命令：

```
awk '{print $NF}' yourfile.txt
```

在这个命令中：

- `awk` 是用于处理文本文件的命令。
- `{print $NF}` 表示打印每行的最后一个字段。
- `yourfile.txt` 是您要处理的文本文件的名称。

您只需将 `yourfile.txt` 替换为您实际的文件名，然后运行这个命令即可输出文本文件中每行的最后一个字符串。

希望这对您有所帮助！如果您有任何其他问题或需要进一步的帮助，请随时告诉我。我随时为您提供支持。

如果遇到shell命令的问题，可以问GPT或者师兄师姐



为了充分利用计算集群的资源，节约工作时间，实际工作中我们往往需要提交大量作业。而我们不可能一个个手动提交，因此可以借助脚本实现自动批量提交。

Shell作为Linux系统的命令解释器，提供了用户与内核进行交互操作的一种接口。它接收用户输入的命令并把它送入内核去执行。不仅如此，**Shell**有自己的编程语言用于对命令的编辑，可以编写脚本处理高复杂度工作。

考虑到我们只需要进行轻量化的工作，且基本所有Linux系统自带**Shell**，因此后续我们将介绍如何用**Shell**脚本批量提交作业。

后面的脚本文件是我自己用过的，仅供参考，大家需要根据自已的需求进行适当的修改

批量处理作业的原理:

与前面分析部分的文件基本一致

```
1 #include "$ROOTIOROOT/share/jobOptions_ReadRec.txt"
2 #include "$VERTEXFITROOT/share/jobOptions_VertexDbSvc.txt"
3 #include "$MAGNETICFIELDROOT/share/MagneticField.txt"
4 #include "$ABSCORROOT/share/jobOptions_AbsCor.txt"
5 #include "$RHOPIALGROOT/share/jobOptions_Rhopi.txt"
6
7 // Input REC or DST file name
8 EventCnvSvc.digiRootInputFile = {"rhopi.dst"};
9
10 // Set output level threshold (2=DEBUG, 3=INFO, 4=WARNING, 5=ERROR, 6=FATAL )
11 MessageSvc.OutputLevel = 5;
12
13 // Number of events to be processed (default is 10)
14 ApplicationMgr.EvtMax = 50;
15
16 ApplicationMgr.HistogramPersistency = "ROOT";
17 NTupleSvc.Output = { "FILE1 DATAFILE='rhopi_ana.root' OPT='NEW' TYP='ROOT'"};
```

run.head

run.foot

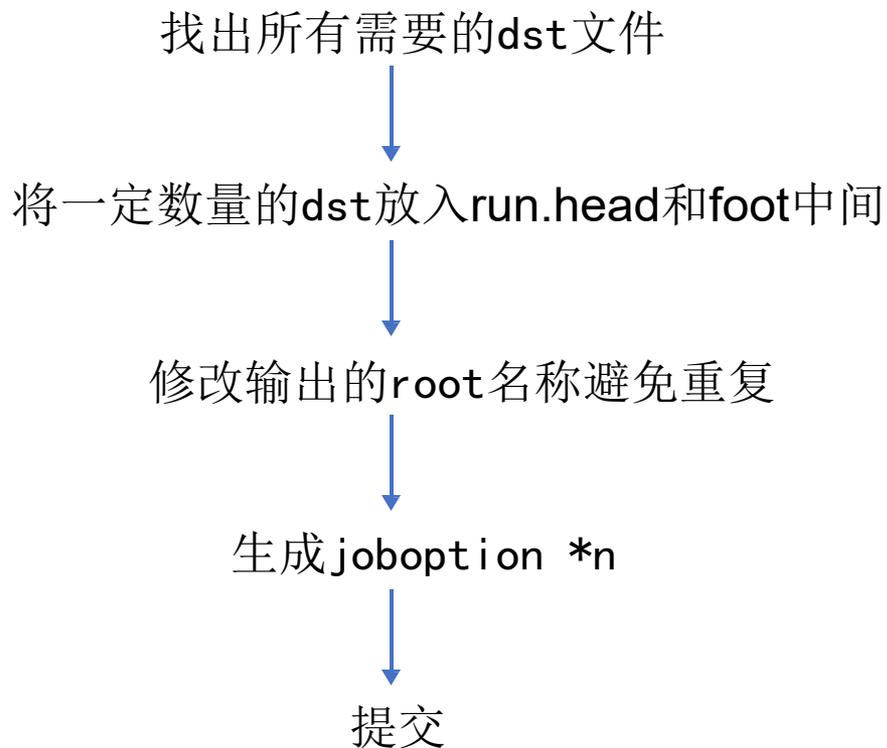
变动的只有input的dst文件以及output的位置

可以把joboption分成两部分 run.head run.foot

joboption = run.head + dst + run.foot



批量处理作业的原理：





高能所: /afs/ihep.ac.cn/users/z/zhangyuepeng/scratchfs/runjob/run.head
科大: /home/ypzhang/ustcfs/runjob/run.head

☰ run.head

```
1 #include "$ROOTIOROOT/share/jobOptions_ReadRec.txt"
2 #include "$VERTEXFITROOT/share/jobOptions_VertexDbSvc.txt"
3 #include "$MAGNETICFIELDROOT/share/MagneticField.txt"
4 #include "$ABSCORROOT/share/jobOptions_AbsCor.txt"
5
6 #include "$MCDECAYMODESVCROOT/share/GetDecayMode.txt"
7 #include "$DTAGALGROOT/share/jobOptions_dTag.txt"
8
9 #include "$HADRONSELALGROOT/share/jobOptions_HadronSel.txt"
10 // #include "$THREEPIONALGROOT/share/jobOptions_ThreePion.txt"
11 // #include "$FOURKAONALGROOT/share/jobOptions_FourKaon.txt"
12 // Input REC or DST file name
13
14 HadronSel.m_CheckLUARLW=true;
15 //HadronSel.m_CheckHybrid = true;
16 HadronSel.m_QQbarMC=true;
17 HadronSel.m_InputEcm = true;
18 HadronSel.m_Ecm=3.49;
19
20 EventCnvSvc.digiRootInputFile = {
21
```

分析程序的开关

后面接着dst文件



高能所: /afs/ihep.ac.cn/users/z/zhangyuepeng/scratchfs/runjob/run.foot
科大: /home/ypzhang/ustcfs/runjob/run.foot

run.foot

```
1  };
2
3  // Set output level threshold (2=DEBUG, 3=INFO, 4=WARNING, 5=ERROR, 6=FATAL )
4  MessageSvc.OutputLevel = 6;
5
6  // Number of events to be processed (default is 10)
7  ApplicationMgr.EvtMax = -1;
8
9  ApplicationMgr.HistogramPersistency = "ROOT";
10
11
12
13
14
15
16
17  //---
18
19
20
21  NTupleSvc.Output = { "FILE1 DATAFILE="/besfs5/isoff/349000/number.root' OPT='NEW' TYP='ROOT'"};
22
23
```

输出的root位置得自己修改

高能所: /afs/ihep.ac.cn/users/z/zhangyuepeng/scratchfs/runjob/run.sh

科大: /home/vpzhanaq/ustcfs/runjob/run.sh

run.sh

```
1  #!/bin/bash
2  set +x
3
4  ##date: produced dir for joboption
5  date=Loff349000
6
7  ##dpath: the dst file after rec
8  dpath=/besfs5/groups/tauqcd/leo591653959/FF/MC/LUARLW/NoISR/349000/SimRec/
9
10 dataFiles=`find $dpath -name "*.dst"|sort`
11 fileNumber=`find $dpath -name "*.dst" | wc -l`
12
13 ##the data dst file number in one joboption
14 let nDataFile=20
15
16 ##the job name with the format $myjob$iJob_run.txt
17 let iJob=1
18 myjob=test_`date`
19
20 ##change something below ONLY when you know what you are doing
21 let n=0
22 for recFile in $dataFiles
23 do
24     #echo $recFile
25     let m=n+1
26
27     if [[ ! -e $recFile ]]; then
28         echo "this data file $recFile doesn't exist! "
29     else
30
31         line="\`$recFile\`"
```

dst文件的位置

每个joboption存放的
dst个数



/afs/ihep.ac.cn/users/z/zhangyuepeng/scratchfs/runjob/run.sh

```
33 if [[ $n%$nDataFile -eq 0 ]]; then
34     if [[ -e ${myjob}"_0"${iJob}_run.txt ]]; then
35         sed -e s/number/"$date"_0"${iJob}/ ./run.foot >> ${myjob}"_0"${iJob}_run.txt
36         if [ -e ${date} ] ; then
37             cd ${date}
38             mv ../${myjob}"_0"${iJob}_run.txt ./
39             #boss -q ${myjob}"_0"${iJob}_run.txt
40             boss.condor ${myjob}"_0"${iJob}_run.txt
41             sleep 0
42             cd ..
43         else
44             mkdir ${date}
45             cd ${date}
46             mv ../${myjob}"_0"${iJob}_run.txt ./
47             #boss -q ${myjob}"_0"${iJob}_run.txt
48             boss.condor ${myjob}"_0"${iJob}_run.txt
49             sleep 0
50             cd ..
51         fi
52         # the number of pbs jobs is less than 30
53         let njob=`hep_q -u sunyk | wc -l`
54         let njob-=5
55         echo $njob
56         while [ $njob -gt 1000 ]
57         do
58             sleep 0
59             let njob=`hep_q -u sunyk | wc -l`
60             let njob-=5
61             echo $njob
62         done
```

提交作业



/afs/ihep.ac.cn/users/z/zhangyuepeng/scratchfs/runjob/run.sh

```
64     let iJob++
65     fi
66     touch ${myjob}"_0"${iJob}_run.txt
67     cat run.head >> ${myjob}"_0"${iJob}_run.txt
68     fi
69
70     ##if the last data file in this jobOption file
71     if [[ $m%nDataFile -eq 0 || $m -eq $fileNumber ]]; then
72     | echo ${line} >> ${myjob}"_0"${iJob}_run.txt
73     else
74     | echo $line, >> ${myjob}"_0"${iJob}_run.txt
75     fi
76
77     let n++
78     fi
79     done
80
81     ##for the last jobOption file
82     sed -e s/number/"$date"_0"${iJob}/ ./run.foot >> ${myjob}"_0"${iJob}_run.txt
83     if [ -e ${date} ]; then
84     cd ${date}
85     mv ../${myjob}"_0"${iJob}_run.txt ./
86     #boss -q ${myjob}"_0"${iJob}_run.txt
87     boss.condor ${myjob}"_0"${iJob}_run.txt
88     cd ..
89     else
90     mkdir ${date}
91     cd ${date}
92     mv ../${myjob}"_0"${iJob}_run.txt ./
93     #boss -q ${myjob}"_0"${iJob}_run.txt
94     boss.condor ${myjob}"_0"${iJob}_run.txt
95     cd ..
96     fi
97
98     exit 0
```

提交作业