POStER

Speed up evaluation by parallelization

November 2018

Michael Weiss – Bayer AG

POStER - Speed up evaluation by parallelization

- What is POStER?
- The Basics
  - Why to do parallelization?
  - How to parallelize SAS programs?
  - What is scheduling?
  - What are dependencies?
- On Top
  - Does the order matter?
  - Can we mix it?
  - Program Initialization
  - What was done?

POStER

What is POStER?

- **P**arallel **O**ptimized **St**atistical **E**xecution **R**untime
- SAS based Macro System
- Parallel execution of multiple SAS programs
- 29 macros (3 user relevant)
- ~ 4400 lines in 150KB
- Focus: easy to use

```
%add_job(
   programFile =
 , logFile     =
 , outFile     =
 , iniProg     = <default>
 , termProg    = <default>
 , options     = <default>
 , interpreter = <default>
 , weight      =
)
```

```
%sync_jobs()
```

```
%run_jobs(count = 4)
```

POStER – The Basics

Why to do parallelization?

- Runtime of programs affect timelines
  - Reduce timelines requires reduced runtime (wall clock time)

- Common PCs have multiple CPUs / Cores
  - Servers often have > 10 CPU Cores
  - Even Workstations often have 4 to 8 cores

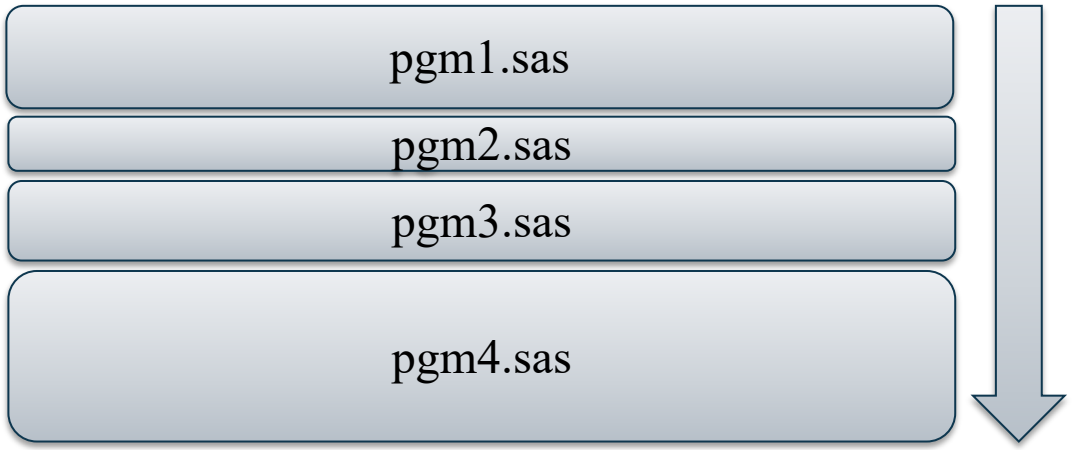
- SAS programs are by design linear
  - One PROC at a time
  - Either doing calculation or waiting for I/O
  - A study evaluation contains multiple (mostly) independent programs

POStER – The Basics

How to parallelize SAS programs?

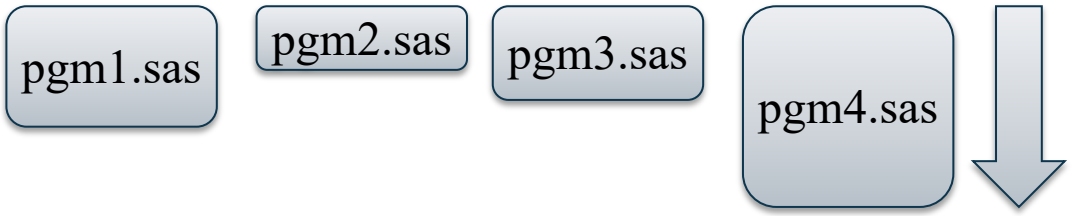
// Common practice is to use single “run-all” program with %INCLUDES → serial execution

```
%INCLUDE "&prgdir./pgm1.sas";
%INCLUDE "&prgdir./pgm2.sas";
%INCLUDE "&prgdir./pgm3.sas";
%INCLUDE "&prgdir./pgm4.sas";
```



// POStER provides similar API, but supports parallel execution

```
%add_job(programFile = &prgdir./pgm1.sas)
%add_job(programFile = &prgdir./pgm2.sas)
%add_job(programFile = &prgdir./pgm3.sas)
%add_job(programFile = &prgdir./pgm4.sas)
%run_jobs(count = 4)
```



POStER – The Basics

How to parallelize SAS programs?

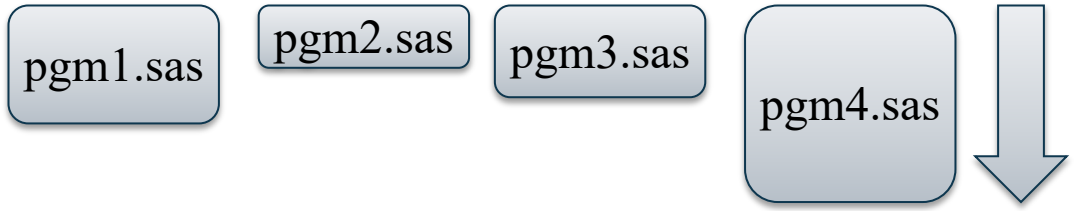
- SAS programs are linear ("*single threaded*")
  - Some procedures are multithreaded, but always execute just one "PROC" at a time
- Parallel execution can be performed by starting multiple SAS processes at the same time
  - Start SAS Process from SAS
  - Most Execution methods are meant to be used synchronized
    - `SYSTEM` Function or CALL Routine
    - `X '<command>'`
    - `%SYSEXEC` Macro Statement
    - `FILENAME ... PIPE '<command>'`
  - Synchronized execution is still linear ("*single threaded*")

POStER – The Basics

How to parallelize SAS programs?

// SYSTASK COMMAND … NOWAIT allows asynchronous execution

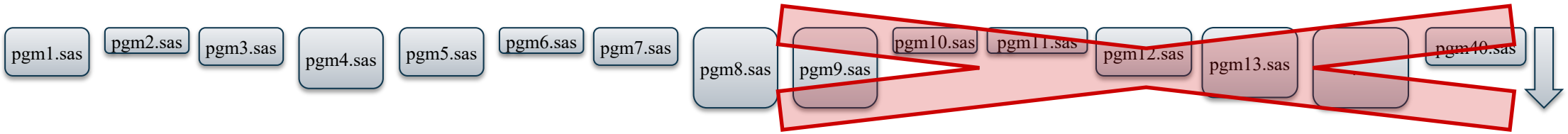
```
SYSTASK COMMAND "sas -sysin &prgdir./pgm1.sas" NOWAIT;
SYSTASK COMMAND "sas -sysin &prgdir./pgm2.sas" NOWAIT;
SYSTASK COMMAND "sas -sysin &prgdir./pgm3.sas" NOWAIT;
SYSTASK COMMAND "sas -sysin &prgdir./pgm4.sas" NOWAIT;
```



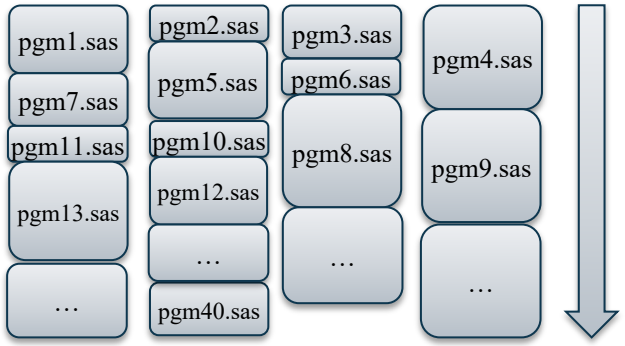
POStER – The Basics

What is scheduling?

// It is good to execute 4 or 8 programs at a time, but not 40, 80 or even more!



// Better would be a configurable number of programs to run in parallel at the same time:



// → This is scheduling

POStER – The Basics

What is scheduling?

// SAS Command WAITFOR → Either wait for _ALL_ or wait for _ANY_ process

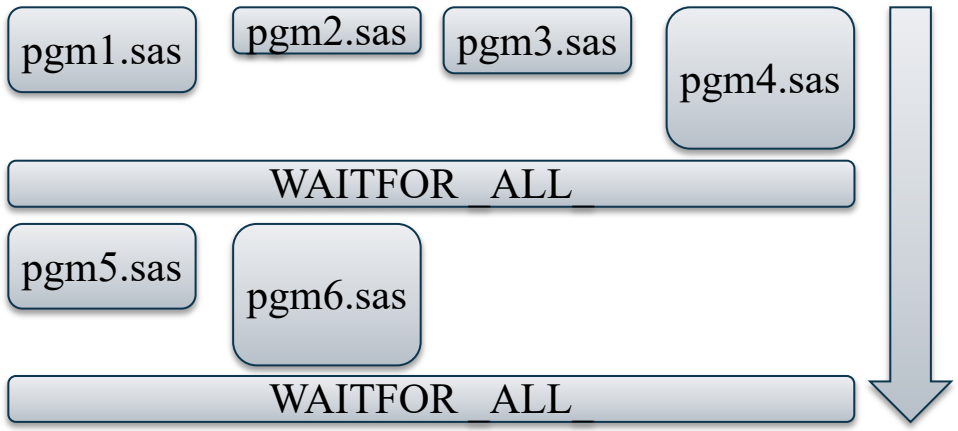
```
SYSTASK COMMAND "sas -sysin &prgdir./pgm1.sas" NOWAIT TASKNAME=_n1;
SYSTASK COMMAND "sas -sysin &prgdir./pgm2.sas" NOWAIT TASKNAME=_n2;
SYSTASK COMMAND "sas -sysin &prgdir./pgm3.sas" NOWAIT TASKNAME=_n3;
SYSTASK COMMAND "sas -sysin &prgdir./pgm4.sas" NOWAIT TASKNAME=_n4;
WAITFOR _ANY_ _n1 _n2 _n3 _n4;
SYSTASK COMMAND "sas -sysin &prgdir./pgm5.sas" NOWAIT TASKNAME=_n5;
WAITFOR _ANY_ &running_tasks;
SYSTASK COMMAND "sas -sysin &prgdir./pgm6.sas" NOWAIT TASKNAME=_n6;
WAITFOR _ALL_ &running_tasks;
```

POStER – The Basics

What are dependencies?

- Not all programs are independent of each other
  - Programs creating ADaM should be finished before programs start that use these data sets (TLF)!
  - Some programs require one ADaM data set to generate an other ADaM data set
  - …
  - → These are dependencies
- POStER implements synchronization request – `WAITFOR _ALL_`

```
%add_job(programFile = &prgdir./pgm1.sas)
%add_job(programFile = &prgdir./pgm2.sas)
%add_job(programFile = &prgdir./pgm3.sas)
%add_job(programFile = &prgdir./pgm4.sas)
%sync_jobs()
%add_job(programFile = &prgdir./pgm5.sas)
%add_job(programFile = &prgdir./pgm6.sas)
%run_jobs(count = 4)
```



POStER - Speed up evaluation by parallelization

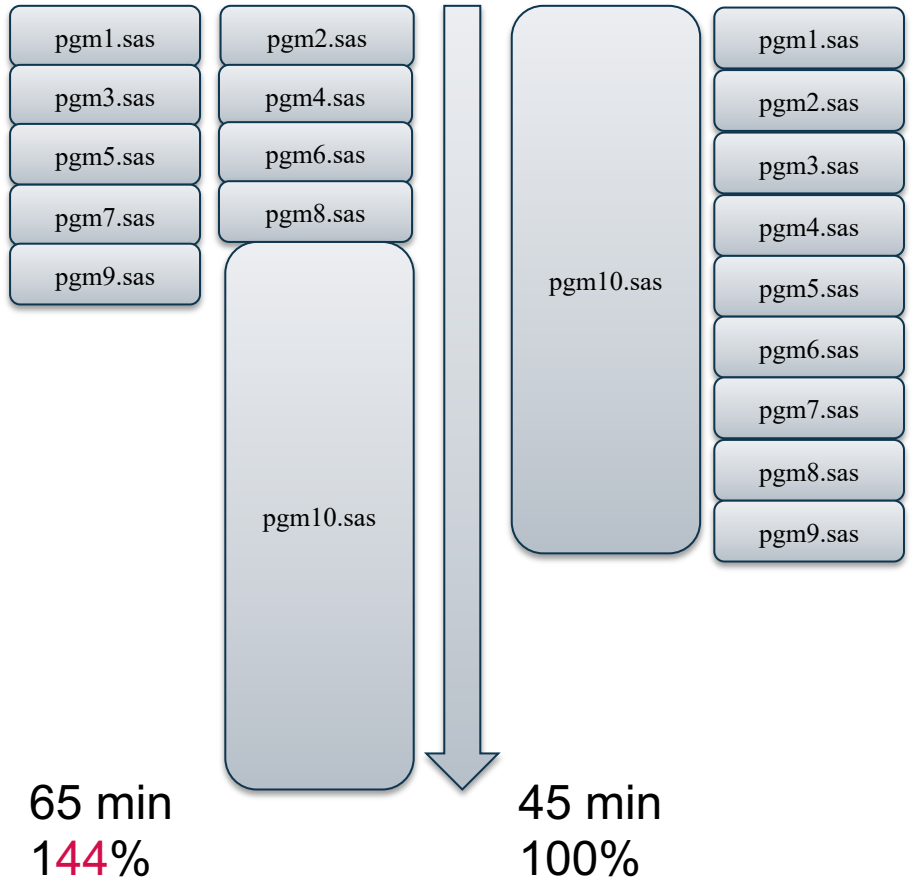
- What is POStER?
- The Basics
  - Why to do parallelization?
  - How to parallelize SAS programs?
  - What is scheduling?
  - What are dependencies?
- On Top
  - Does the order matter?
  - Can we mix it?
  - Program Initialization
  - What was done?

POStER – On Top

Does the order matter?

- Does the order of execution matter?
- Assume 10 programs to be run in 2 threads
  - 9 with 5min runtime each
  - 1 with 45min runtime

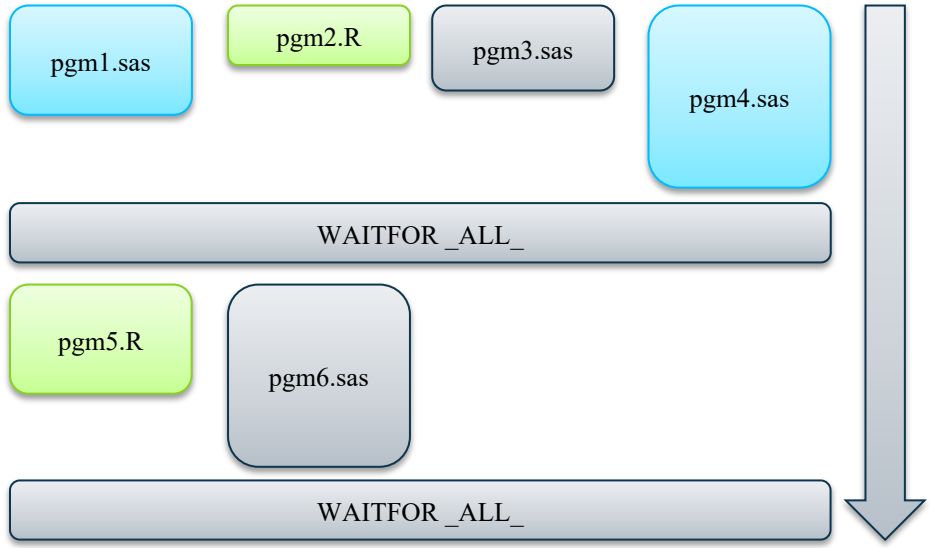
- POStER allows manual ordering and automated re-ordering on re-execution



POStER – On Top

Can we mix it?

```
%add_job(programFile = &prgdir./pgm1.sas
      , interpreter = SAS9.2)
%add_job(programFile = &prgdir./pgm2.R
      , interpreter = R3.1)
%add_job(programFile = &prgdir./pgm3.sas
      , interpreter = SAS9.2)
%add_job(programFile = &prgdir./pgm4.sas
      , interpreter = SAS9.4)
%sync_jobs()
%add_job(programFile = &prgdir./pgm5.R
      , interpreter = R3.1)
%add_job(programFile = &prgdir./pgm6.sas
      , interpreter = SAS9.4)
%run_jobs(count = 4)
```

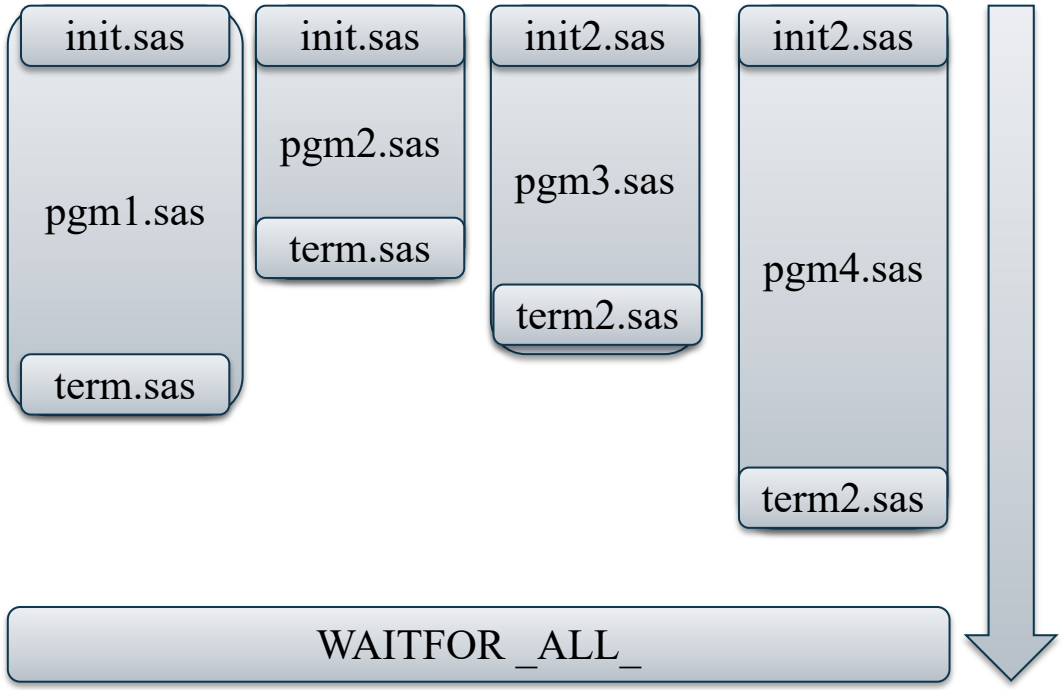


POStER – On Top

Program Initialization

// POStER supports INITSTMT and TERMSTMT SAS Options through separate files

```
%add_job(programFile = &prgdir./pgm1.sas
      , iniPROG     = &prgdir./init.sas
      , termPROG    = &prgdir./term.sas)
%add_job(programFile = &prgdir./pgm2.sas
      , iniPROG     = &prgdir./init.sas
      , termPROG    = &prgdir./term.sas)
%add_job(programFile = &prgdir./pgm3.sas
      , iniPROG     = &prgdir./init2.sas
      , termPROG    = &prgdir./term2.sas)
%add_job(programFile = &prgdir./pgm4.sas
      , iniPROG     = &prgdir./init2.sas
      , termPROG    = &prgdir./term2.sas)
%run_jobs(count = 4)
```



POStER – On Top

Program Initialization

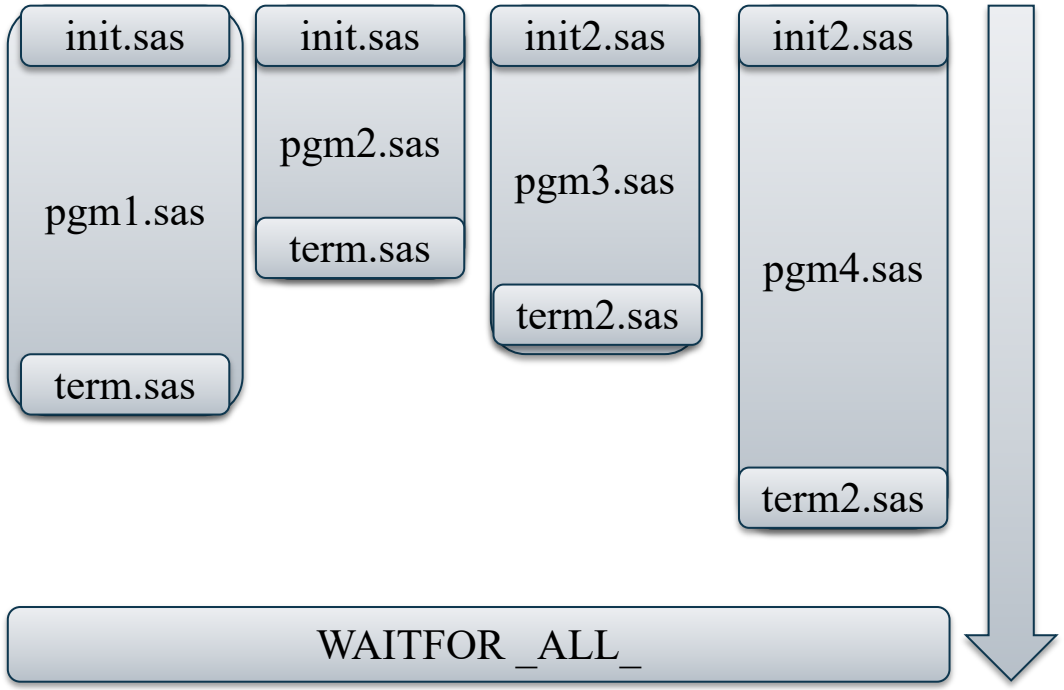
// POStER supports INITSTMT and TERMSTMT SAS Options through separate files

```
%LET POSTER_PARAM_INIPROG=&prgdir./init.sas;
%LET POSTER_PARAM_TERMPROG=&prgdir./term.sas;

%add_job(programFile = &prgdir./pgm1.sas)
%add_job(programFile = &prgdir./pgm2.sas)

%LET POSTER_PARAM_INIPROG=&prgdir./init2.sas;
%LET POSTER_PARAM_TERMPROG=&prgdir./term2.sas;

%add_job(programFile = &prgdir./pgm3.sas)
%add_job(programFile = &prgdir./pgm4.sas)
%run_jobs(count = 4)
```



POStER – On Top

What was done?

- Automated tracing – Depending on Operating System
  - SAS internal → RTRACE:
    RTRACE=ALL RTRACELOC="/path/to/trace.file"

- Linux / HP-UX
  - OS commands strace / ptrace / tusc:

```
strace -D -f -e trace=open,unlink,rename,stat -o "/path/to/trace.file"
```

POStER – On Top

What was done?

```
### STARTED:2018-08-13T09:32:04
### FINISHED:2018-08-13T09:33:27
### JOB-STATUS:0
### PGM:/var/swan/root/bhc/948862/16244/stat/csr/dev/pgms/t_adce.sas
### LOG:/var/swan/root/bhc/948862/16244/stat/csr/dev/logs/t_adce_fas.log
### RES:/var/swan/root/bhc/948862/16244/stat/csr/dev/results/t_adce_fas.lst
### INIPROG:/var/swan/root/bhc/948862/16244/stat/csr/dev/pgms/ini_14_shell_fas.sas
### TERMPROG:/var/swan/root/bhc/948862/16244/stat/csr/dev/pgms/term_14.sas
### INTERPRETER:sas9.4
### JOB-OPTIONS:
### LOG-STATUS:2
R   /var/swan/root/bhc/948862/16244/stat/csr/dev/pgms/t_adce.sas
 W  /var/swan/root/bhc/948862/16244/stat/csr/dev/logs/t_adce_fas.log
r   /var/swan/root/bhc/general/tools/eva/eva1/prod/macros/prepare_job.sas
R   /var/swan/root/bhc/general/tools/poster/poster1/dev/macros/prepare_job.sas
R   /var/swan/root/bhc/948862/16244/stat/csr/dev/pgms/ini_14_shell_fas.sas
...
R   /var/swan/root/bhc/948862/16244/stat/csr/dev/pgms/term_14.sas
W  /var/swan/root/bhc/948862/16244/stat/csr/dev/results/t_adce_fas.rtf
```

POStER

Speed up evaluation by parallelization

November 2018

Michael Weiss – Bayer AG

THANK YOU