The *fastr* package

Purpose of this document

This document describes the *fastr* package. *fastr* is a set of R functions for the analysis of data collected by the *FAST* performance tools. *fastr* concentrates on the analysis of GEANT4-based simulations run in the CMSSW framework.

This document describes:

1. The directory structure in which the raw data files from the profiling must be deployed.
2. How the data files must be *cleaned*.
3. The working environment relied upon by the analysis software.
4. The dataframes generated by the analysis software.
5. The files into which the analysis dataframes are written for use in later analysis.

Expected directory structure

We will call the top-level directory, under which the entire directory structure defined here is found, TOP.

Under TOP, each combination of Geant4 version and CMSSW version is given a directory, e.g.:

```
TOP/g4.9.3p01_cms_3_8_0
TOP/g4.9.3p02_cms_3_8_0
```

We call the set of output contained in each of these directories a CAMPAIGN. Under each compaign's directory there will be zero or more EXPERIMENT directories:

```
TOP/g4.9.3p01_cms_3_8_0/exp_7
TOP/g4.9.3p02_cms_3_8_0/exp_10
TOP/g4.9.3p02_cms_3_8_0/exp_11
```

In each experiment's directory are the *profdata_* and *txt* files from running the profiler.

Cleaning the data files

The *fastr* scripts expect to process only correctly-formed data files. Because various failures of the data collection process can result in defective data files, cleaning of the files must be done. Any trial for which any defect is found is to be removed in its entirety. A trial is defective if:

1. Its *names* file is of size zero. This indicates a failure to collect profiling results.
2. Its *eventdata* file does not contain 100 event time reports. This indicates a failure of the simulation program to run to completion.

This cleaning is done "by hand". Failure to do the cleaning can result in mismatching dataframes, and spurious results.

Expected working environment

The *fastr* scripts expect to be run with TOP as the current working directory.

Dataframes generated

The *fastr* code produces a set of dataframes for each campaign, and saves these dataframes to files. To compare campaigns, it is necessary to load the data for the compaigns to be compared from the several files in which they are stored. The independence of campaign data files helps to reduce the processing time and computer resources needed to process a new campaign, avoiding a linear expansion of time and memory needed to process a series of campaigns.

Run environments dataframe

The RUNENVS contains one row per trial in a campaign. It contains information about the environment in which each trial was run. It has columns:

```
exp        experiment id
run        run id
vendor     the manufacturer of the processor
family     the processor family
model      the processor model
stepping   the procesor stepping
name       the process name, as reported in /proc/cpuinfo
g4version  the version of Geant4 used
cmsversion the version of CMSSW used
modified   either "unmodified" or "modified"
```

The 'modified' column tells whether the Geant4 code being profiled has been locally modified or not.

Runmeta dataframe

The RUNMETA dataframe is a truncated form of the RUNENVS dataframe, suitable for using in *merge* commands with other dataframes carrying the *exp* and *run* columns. It includes the parts of the RUNENVS dataframe most important for identifying the conditions under which a trial was executed. It includes columns:

```
exp        experiment id
run        run id
vendor     the manufacturer of the processor
g4version  the version of Geant4 used
cmsversion the version of CMSSW used
modified   either "unmodified" or "modified"
```

Trials dataframe

The TRIALS dataframe contains one row per trial in a campaign. It has columns:

```
exp       experiment id
run       run id
trial.t   total time reported in the "trials_*.txt" data file
vendor     the manufacturer of the processor
g4version  the version of Geant4 used
cmsversion the version of CMSSW used
modified   either "unmodified" or "modified"
```

Note that trial.t includes program startup and shutdown time, not just the time taken for event processing itself.

For any campaign, the RUNENVS, RUNMETA and TRIALS dataframes should have equal row counts, one row for each trial in the campaign.

Events dataframe

The EVENTS dataframe contains now row per event per trial in a campaign. It has columns:

```
exp       experiment id
run       run id
event     the id of the event being timed
t         the time in seconds to process this event
vendor     the manufacturer of the processor
g4version  the version of Geant4 used
cmsversion the version of CMSSW used
modified   either "unmodified" or "modified"
```

There should be the same number of events for each trial (exp+run pair).

Functions dataframe

The FUNCTIONS dataframe contains one row per function per trial in a campaign. It has columns:

```
exp       experiment id
run       run id
leaf      function's leaf count
path      function's path count
leaf.frac function's leaf fraction
path.frac function's path fraction
lib       name of the library this function is in
mangled   mangled name of the function
name      unmangled name of the function
short     name of the function, stripped of argument list
vendor     the manufacturer of the processor
g4version  the version of Geant4 used
cmsversion the version of CMSSW used
modified   either "unmodified" or "modified"
```

There may be a different number of functions for each trial (exp+run pair), because infrequently-called functions may appear in some, but not other, trials.

Libraries dataframe

The LIBRARIES dataframe contains one row per library per trial in a campaign. It has columns:

```
exp       experiment id
run       run id
lib       library name
samples   sum of leaf counts for all functions in the library
vendor     the manufacturer of the processor
```

```
g4version  the version of Geant4 used
cmsversion the version of CMSSW used
modified   either "unmodified" or "modified"
```

There may be a different number of libraries for each trial (exp+run pair), because infrequently-called libraries may appear in some, but not other, trials.

Output data structure

Each dataframe from each campaign is written to its own binary data file in the directory for that campaign. The data file for each dataframe is named after the dataframe, e.g. the TRIALS dataframe is written to a file named "trials.rda".