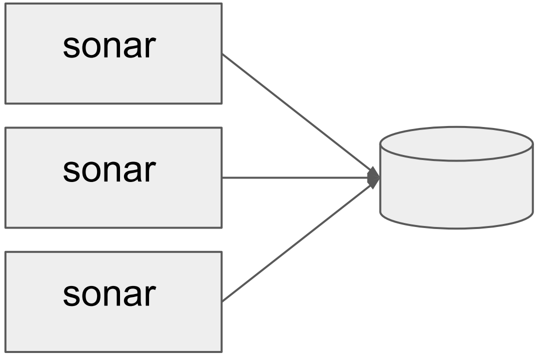
sonar, sonalyze, naicreport, and the pipeline

(technical, September 2023)

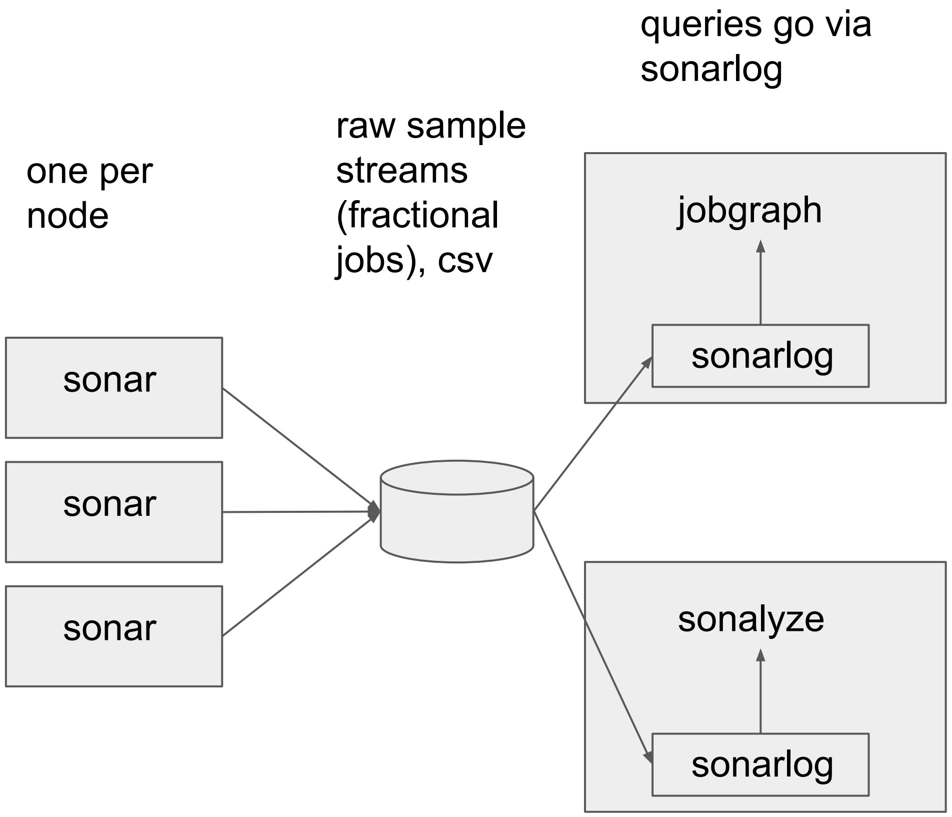
Architecture

one per node

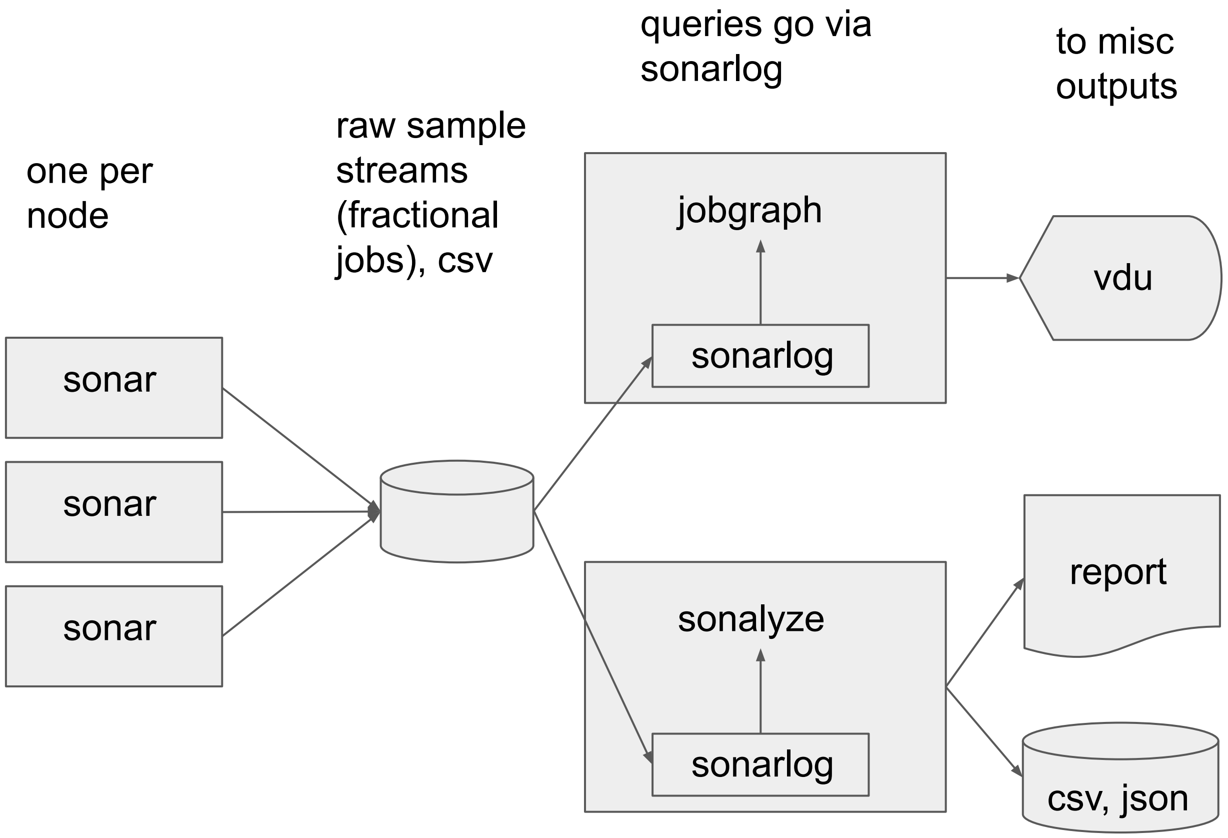
raw sample streams (fractional jobs), csv



Architecture



Architecture



Here's a job report from sonalyze

```
$ ./sonalyze jobs -u- --host ml7 --data-path data
jobm      user      duration  host  cpu-avg  cpu-peak  mem-avg  mem-peak  gpu-avg  gpu-peak  gpumem-avg  gpumem-peak  cmd
1817618>  saeedes   0d23h55m  ml7   102      138       26       83        29       68        9           11           python3
1818070>  saeedes   0d23h55m  ml7   103      141       32       76        28       66        10          11           python3
1818403>  saeedes   0d23h55m  ml7   104      179       32       60        27       57        9           11           python3
1818623>  saeedes   0d23h55m  ml7   104      160       24       50        31       56        10          11           python3
1925362>  saeedes   0d23h55m  ml7   4        5         3        3         0        0         0           0            python3
1925587>  saeedes   0d23h55m  ml7   4        5         3        3         0        0         0           0            python3
1925829>  saeedes   0d23h55m  ml7   4        5         3        3         0        0         0           0            python3
1926449>  saeedes   0d23h55m  ml7   4        5         3        3         0        0         0           0            python3
538368    tobiasao  0d 8h20m  ml7   531      1670      6        6         2        36        1           1            python
539554    tobiasao  0d 9h10m  ml7   523      1580      6        6         2        28        1           1            python
611949>   tobiasao  0d14h30m  ml7   1575     1900      4        4         0        0         0           0            python
623616    tobiasao  0d 0h15m  ml7   24       41        6        6         4        16        1           1            python
631575>   tobiasao  0d14h25m  ml7   1665     2079      4        4         0        0         0           0            python
640122>   tobiasao  0d14h20m  ml7   1663     1918      4        4         0        0         0           0            python
```

There are options for selection (time, host, user, utilization, component) and formatting (almost anything)

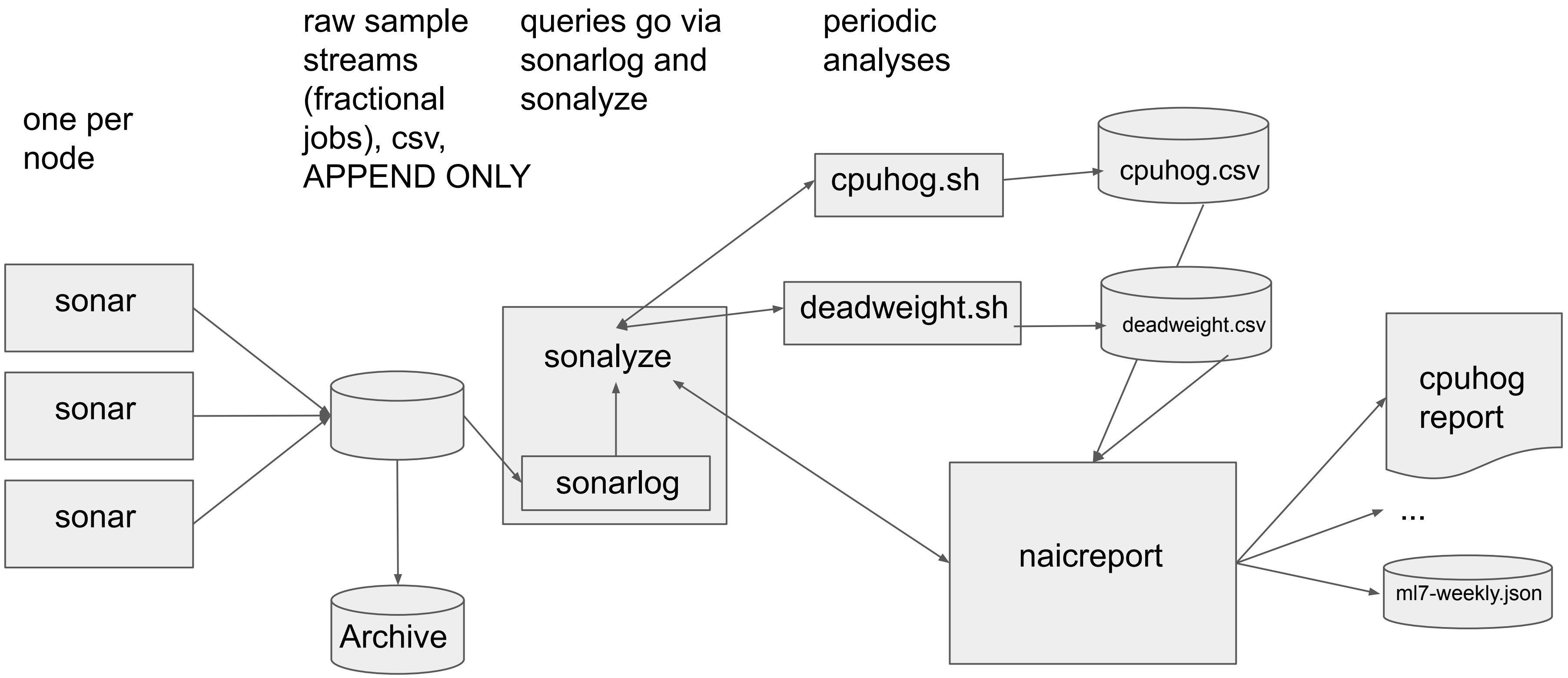
Here's a load report, too

```
[larstha@ml4 sonar]$ ./sonalyze load --host ml7 --data-path data
HOST: ml7.hpc.uio.no
date        time   cpu   mem  gpu  gpumem
2023-09-21  10:00  455   123  123  37
2023-09-21  11:00  449   124  140  35
2023-09-21  12:00  449   117  146  31
2023-09-21  13:00  449   117  142  36
2023-09-21  14:00  448   115  128  26
2023-09-21  15:00  447   115  133  34
2023-09-21  16:00  453   116  125  37
2023-09-21  17:00  448   121  120  39
2023-09-21  18:00  977   128  124  40
2023-09-21  19:00  5552  139  123  41
2023-09-21  20:00  5876  137  102  40
2023-09-21  21:00  5894  139  104  35
2023-09-21  22:00  5803  140  110  37
2023-09-21  23:00  5848  146  86   37
2023-09-22  00:00  5860  149  89   33
2023-09-22  01:00  5858  148  91   29
2023-09-22  02:00  5897  152  73   31
2023-09-22  03:00  5826  151  102  36
2023-09-22  04:00  5829  145  113  36
2023-09-22  05:00  5847  137  115  38
2023-09-22  06:00  5847  140  100  40
2023-09-22  07:00  5841  138  110  36
2023-09-22  08:00  5840  136  103  33
2023-09-22  09:00  5838  140  113  34
2023-09-22  10:00  5841  138  112  33
```

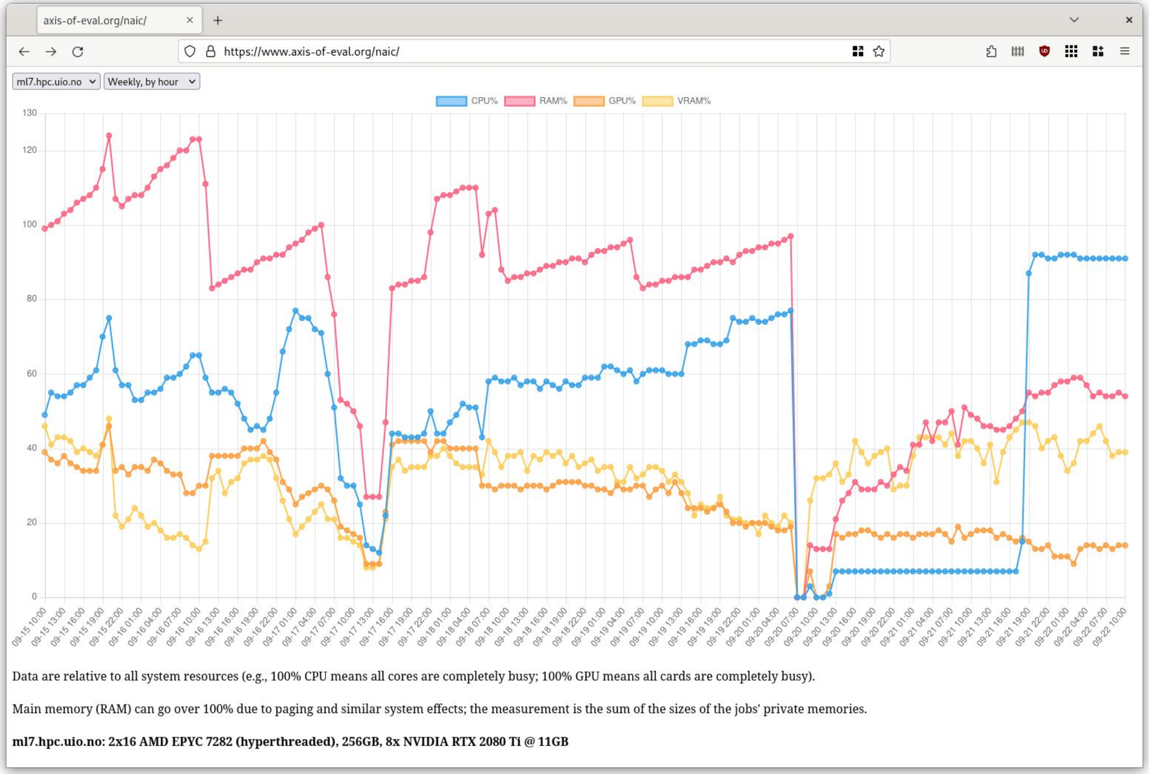
Load averaged hourly here
Daily also available

Many selection options
Many formatting options (also csv, json)

Architecture



Rendered view of ml7-weekly.json (load report)



Data are relative to all system resources (e.g., 100% CPU means all cores are completely busy; 100% GPU means all cards are completely busy).

Main memory (RAM) can go over 100% due to paging and similar system effects; the measurement is the sum of the sizes of the jobs' private memories.

ml7.hpc.uio.no: 2x16 AMD EPYC 7282 (hyperthreaded), 256GB, 8x NVIDIA RTX 2080 Ti @ 11GB

Cpuhog report (excerpt)

```
New CPU hog detected (uses a lot of CPU and no GPU) on host "ml6":
  Job#: 2712710
  User: hermanno
  Command: kited
  Started on or before: 2023-09-07 10:00
  Violation first detected: 2023-09-22 11:13
  Observed data:
       CPU peak = 37 cores
       CPU utilization avg/peak = 1%, 58%
       Memory utilization avg/peak = 5%, 6%

New CPU hog detected (uses a lot of CPU and no GPU) on host "ml6":
  Job#: 3043187
  User: poyenyt
  Command: python3.9
  Started on or before: 2023-09-07 07:50
  Violation first detected: 2023-09-22 11:13
  Observed data:
       CPU peak = 15 cores
       CPU utilization avg/peak = 5%, 24%
       Memory utilization avg/peak = 3%, 3%
```

Summary wall of text

- *sonar* runs on every node (by cron), fairly often (every 5 minutes on ml systems, about 2MB uncompressed text per day across all nodes), generates raw sample streams in csv form
- *sonarlog* ingests sonar data and and cleans them up for general utility:
    - add missing data
    - fixup questionable fields
    - filter records and jobs according to input arguments
    - segregate all sample streams by the job artifact key (host, job-id, cmd)
    - provide utilities to merge streams predictably and to correctly build aggregates. somewhat tricky.
    - sonarlog is pretty clean now and can probably be reused from other programs (eg jobgraph)
- *sonalyze* operates on the cleaned-up sample streams
    - aggregates data for jobs and hosts and prints these aggregates (multi-node jobs not 100% done probably)
    - command line switches for selecting input records, aggregating in different ways, printing in different ways
- cron jobs run sonalyze every 2 hrs for *cpuhog* and *deadweight* analysis, generates more csv
- sonalyze can also be run manually for ad-hoc queries, to better understand summarized data
- *naicreport* is run by cron occasionally to generate reports for the web front end (load, cpuhog, deadweight)
- naicreport can also be run manually as needed
- currently this pipeline is only on the ML nodes
- sonar, sonarlog and sonalyze written in Rust; naicreport written in Go; bash to glue everything together; HTML+JS for Web
- https://github.com/NAICNO/Jobanalyzer

Status + Future work

- sonar and sonarlog are very stable now, minor bug + feature work
  - definitely want to add some logging of communication work
- sonarlog is definitely shareable with jobgraph, probably a good idea
- sonalyze mostly stable but will see some more use-case driven work
- naicreport is growing daily, will continue with this
- NAIC
  - production deployment on ML nodes + hosted analysis & web is imminent
  - multi-node work: Fox, then the world!
  - flesh out NAIC use cases and requirements in more detail, then address them
  - probably a lot of experimentation to see what works
- data management:
  - compress and archive older data to keep the volume under control
  - allow sonarlog to work on the compressed data transparently: older data sometimes useful, long-term reports
- presentation
  - we have a UI intern, might try to make it look nice & more functional
  - lots of reports driven by NAIC use cases, TBD
- (more stuff i haven't thought about, most of it appears in the bug tracker anyhow)