



On sampling in traces

Jesus Labarta, BSC

EU H2020 Centre of Excellence (CoE)



Grant Agreement No 824080

1 December 2018 – 30 November 2021



- Sampling is often considered an alternative method to instrumentation ... and it shouldn't. It should be considered as complementary
- We will describe/demonstrate:
 - The complementarity between sampling and instrumentation
 - Focus on the use of Paraver to analyze sampled data
 - Refer to further presentations to elaborate on the combined use



Instrumentation and Sampling



- Instrumentation: performance/program data is captured when the program control flow hits specific predefined points in the source structure
- Sampling: performance/program data is captured at “random points” not directly related to a specific point in the control flow of the program
- Often used in different ways and for different purposes
 - sampling data is reduced online during the run to emit aggregated profile statistics at the end of the run.
 - **Order/time dimension lost**
 - Instrumented data is emitted “as is” for later visualization or post-processing
- But not always, not necessarily
 - Both types of acquisition used ... but for different purposes
- BSC tools (Extrae, Paraver,...) provide support for analyzing combined instrumentation and sampled data





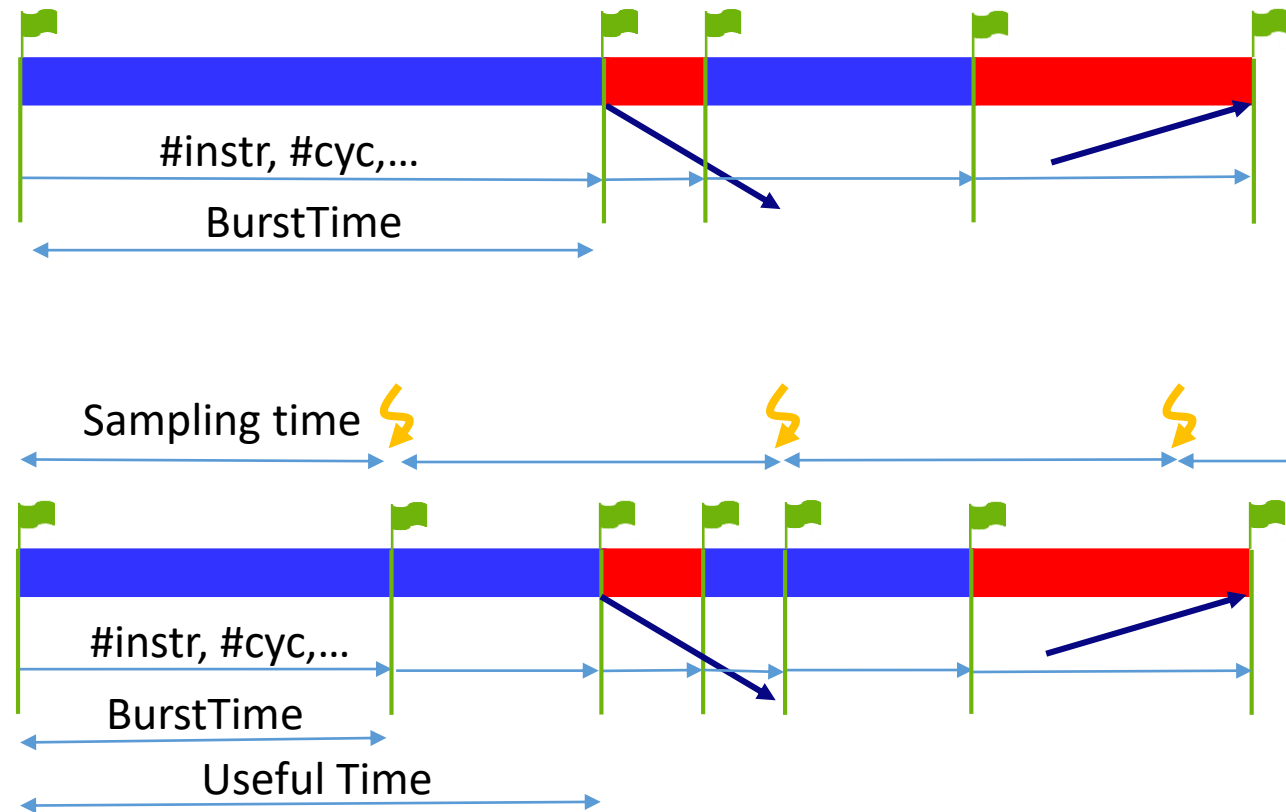
- Acquisition: through the xml control file (see other presentations on details)
 - Typical approach: “Periodic” sampling (focus of today)
 - Extrae xml specifies period (e.g. 6 ms)
 - some variability (e.g. +-1ms) to try and avoid possible correlations between samples and application structure.
 - Other options: correlated to hardware counter activity
- Views based on samples and statistical profiling
 - Per thread or aggregated
 - Visual correlation to instrumentation related view
- Beyond statistical profiling: (topic for another day)
 - Post processing of the trace to increase the precision of the data (e.g. provide sub ms detail with data sampled at several ms periodicity)



Sampled trace



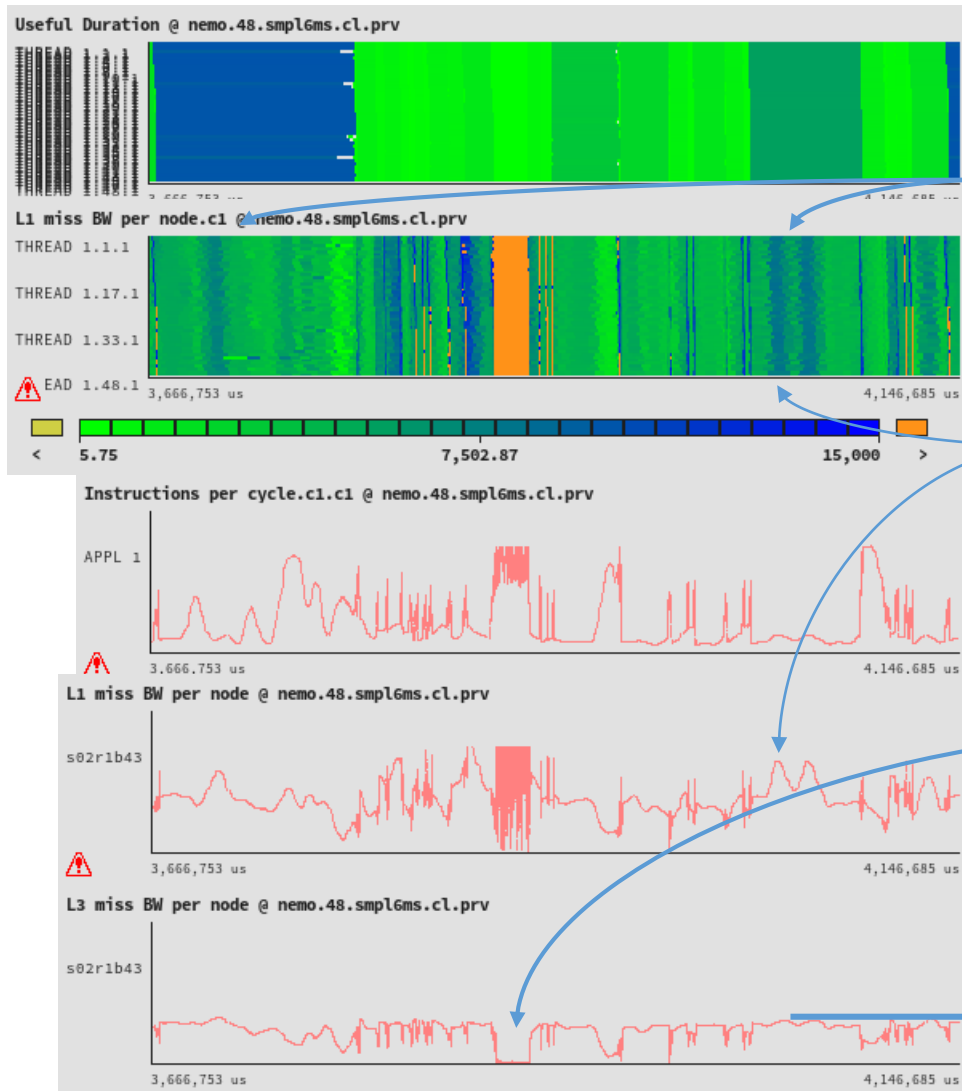
- Samples ... on top of instrumentation
 - Call stack
 - Hardware counters



Example: structure and performance



48 processes



Sub substructure

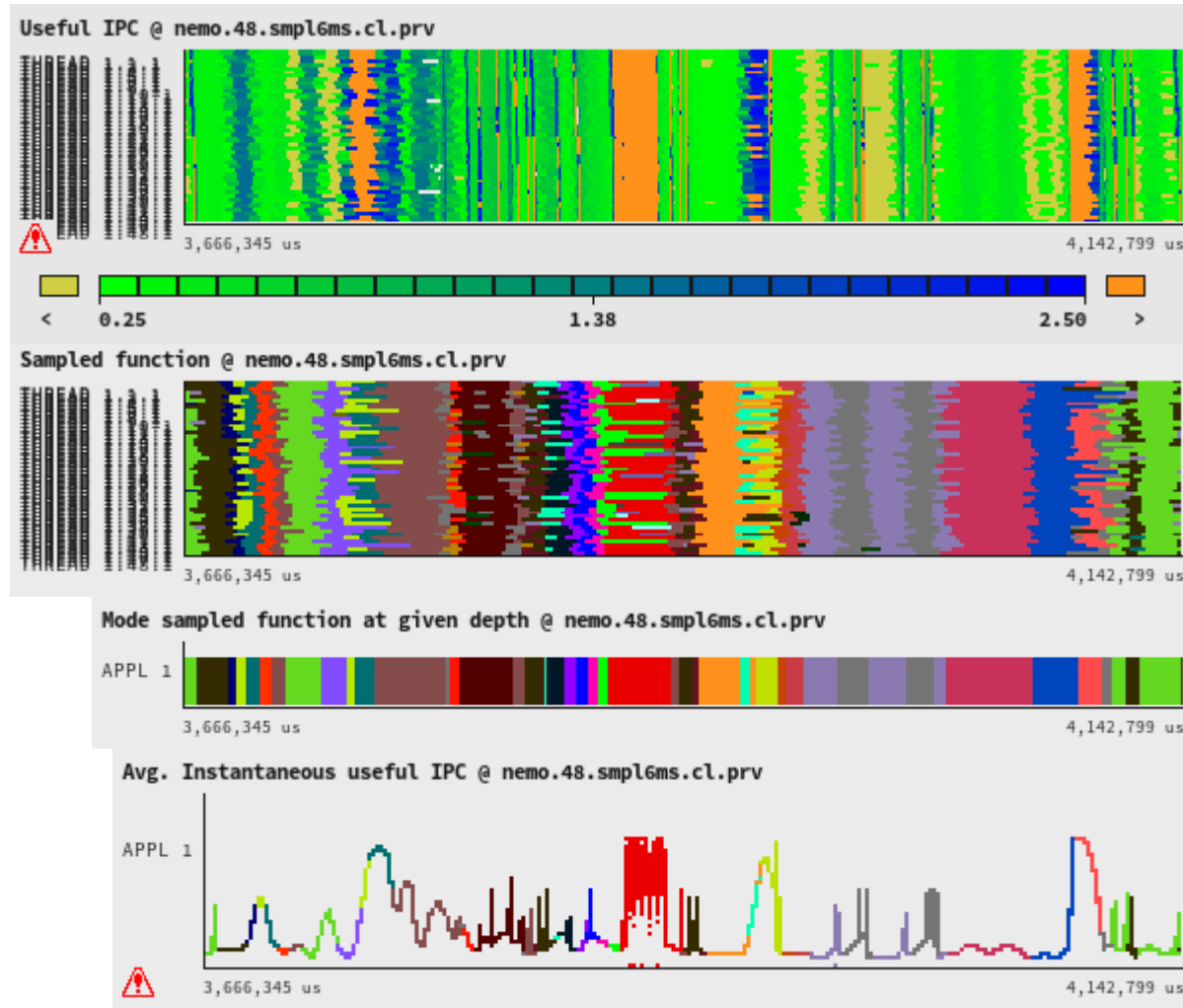
Synchronized BW demands.

Data fits in L3!

Limited Memory BW



Example: link to source



- lbcInk_m.._3d_ptr_ [lbcInk_mp_mpp_lnk_3d_ptr_]
- domvvl_m..nterpol_ [domvvl_mp_dom_vvl_interpol_]
- divhor_m..div_hor_ [divhor_mp_div_hor_]
- eosbn2_mp_bn2_
- zdfcke_m..tke_tke_ [zdfcke_mp_tke_tke_]
- zdfcke_m..tke_avn_ [zdfcke_mp_tke_avn_]
- zdfevd_m..zdf_evd_ [zdfevd_mp_zdf_evd_]
- zdfddm_m..zdf_ddm_ [zdfddm_mp_zdf_ddm_]
- zdfiwm_m..zdf_iwm_ [zdfiwm_mp_zdf_iwm_]
- ldfslp_m..ldf_slp_ [ldfslp_mp_ldf_slp_]
- domvvl_m.._sf_nxt_ [domvvl_mp_dom_vvl_sf_nxt_]
- dynkeg_m..dyn_keg_ [dynkeg_mp_dyn_keg_]
- dynzad_m..dyn_zad_ [dynzad_mp_dyn_zad_]
- dynvor_m..vor_eeen_ [dynvor_mp_vor_eeen_]
- dynldf_l..ldf_lap_ [dynldf_lap_blp_mp_dyn_ldf_lap_]
- dynhpg_m..hpg_sco_ [dynhpg_mp_hpg_sco_]
- dynspg_t.._spg_ts_ [dynspg_ts_mp_dyn_spg_ts_]
- dynzdf_m..dyn_zdf_ [dynzdf_mp_dyn_zdf_]
- sshwzv_mp_wzv_
- traqsr_m..tra_qsr_ [traqsr_mp_tra_qsr_]
- ldftra_m..eiv_trp_ [ldftra_mp_ldf_eiv_trp_]
- traadv_m..tra_adv_ [traadv_mp_tra_adv_]
- traadv_f..adv_fct_ [traadv_fct_mp_tra_adv_fct_]
- traadv_f.._nonosc_ [traadv_fct_mp_nonosc_]
- traldf_i..ldf_iso_ [traldf_iso_mp_tra_ldf_iso_]
- trazdf_m..zdf_imp_ [trazdf_mp_tra_zdf_imp_]
- dynnxt_m..dyn_nxt_ [dynnxt_mp_dyn_nxt_]
- domvvl_m.._sf_swp_ [domvvl_mp_dom_vvl_sf_swp_]
- stpctl_m..stp_ctl_ [stpctl_mp_stp_ctl_]
- eosbn2_mp_rab_3d_
- zdfsh2_m..zdf_sh2_ [zdfsh2_mp_zdf_sh2_]
- tra_nxt_vvl
- eosbn2_m.._insitu_ [eosbn2_mp_eos_insitu_]
- eosbn2_m..itu_pot_ [eosbn2_mp_eos_insitu_pot_]





Performance Optimisation and Productivity

A Centre of Excellence in HPC

Contact:

 <https://www.pop-coe.eu>

 pop@bsc.es

 [@POP_HPC](https://twitter.com/POP_HPC)

 youtube.com/POPHPC

