Management of the data processing & Run-by-Run simulations in KM3NeT

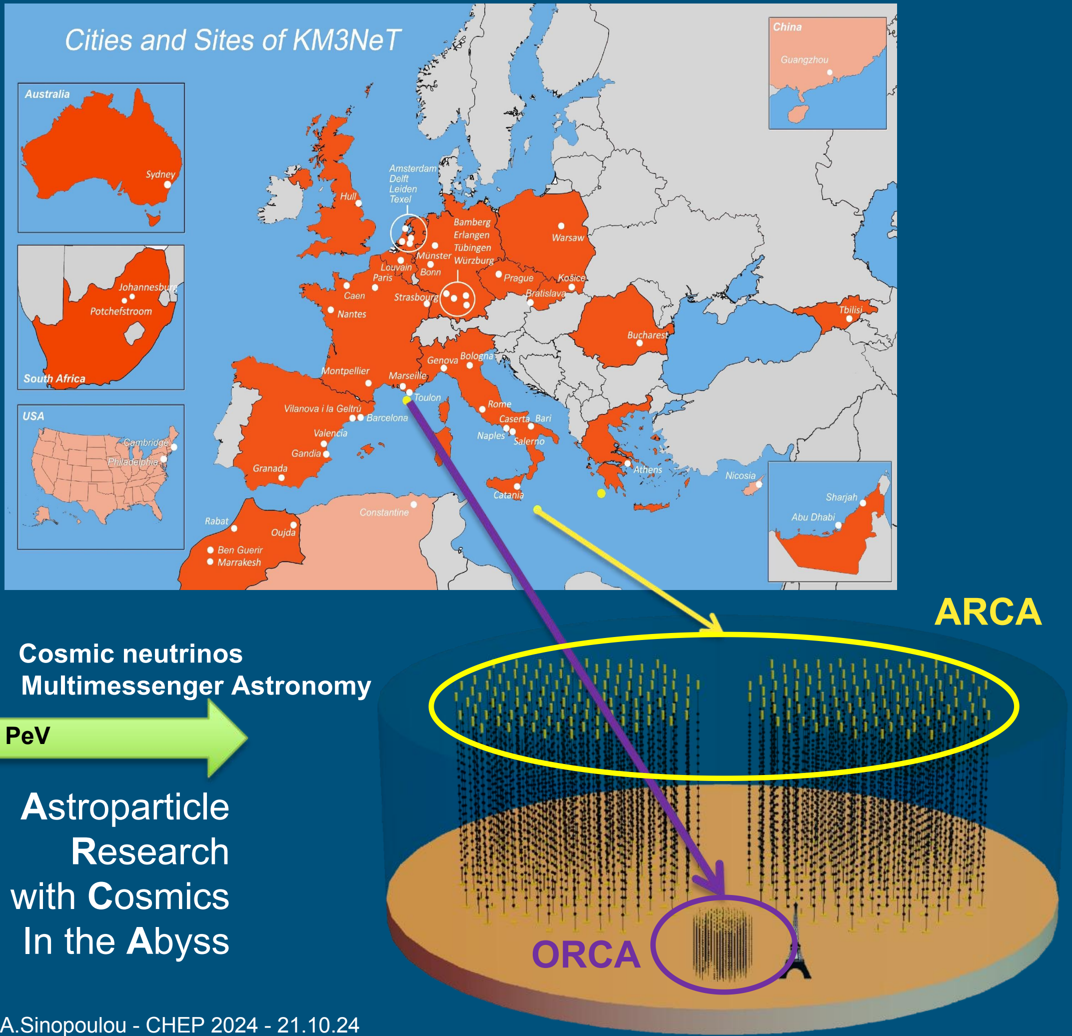
Anna Sinopoulou*, Mieke Bouwhuis, Carla Distefano, Luigi A. Fusco, Chiara Lastoria, Valentin Pestel, Benjamin Trocmé
on behalf of the KM3NeT collaboration

*Università di Catania, Istituto Nazionale di Fisica Nucleare (INFN-Sezione di Catania), Italy
anna.sinopoulou@ct.infn.it

Conference on Computing in High Energy & Nuclear Physics, CHEP 2024

KM3NeT

KM3NeT is an international collaboration that is building 2 underwater neutrino detectors in the Mediterranean Sea: the ORCA and ARCA detectors



Supernovae	ν Oscillations ν Mass Ordering	Dark Matter searches Exotics searches	Cosmic neutrinos Multimessenger Astronomy
MeV	**GeV**	**TeV**	**PeV**

Oscillation **R**esearch with **C**osmics In the **A**byss

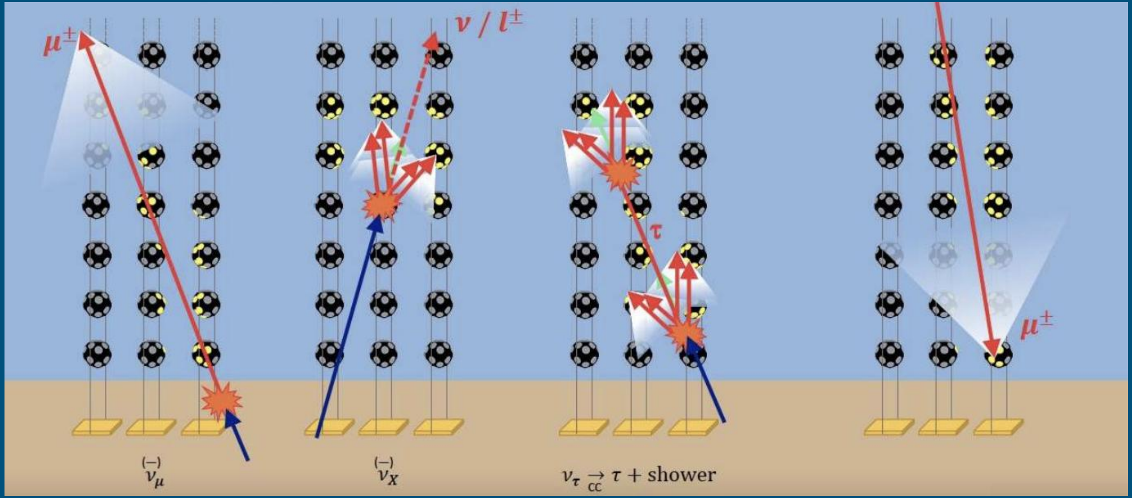
Astroparticle **R**esearch with **C**osmics In the **A**byss

Underwater Cherenkov detectors

Detection Unit (DU): a string of 18 **Digital Optical Modules (DOMs):** instrumented spheres hosting 12 upwards-pointing + 19 downward pointing 3” PMTs

↓

4π signal coverage

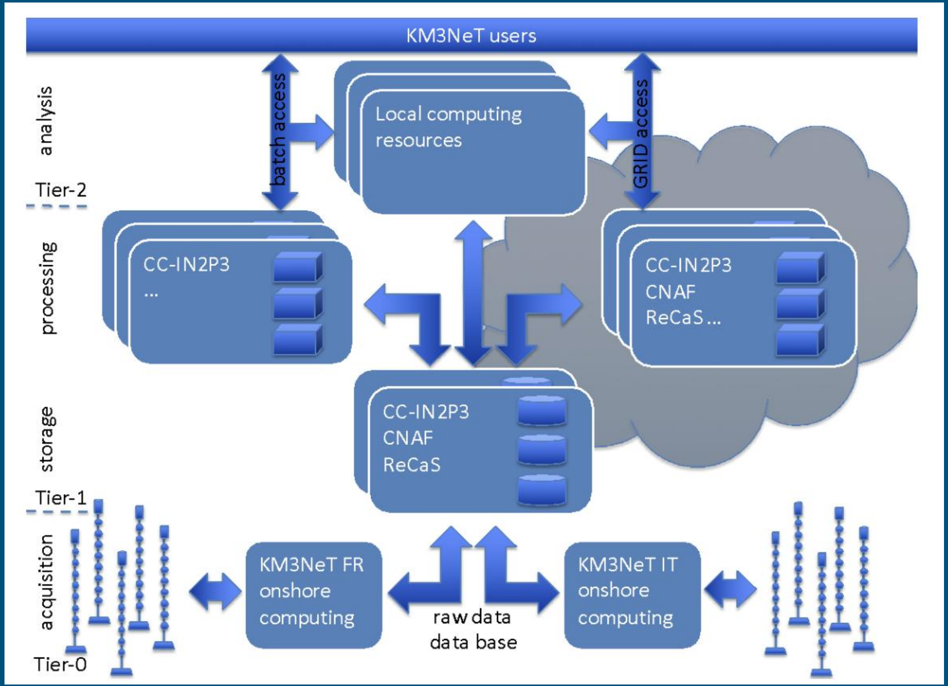


signal: neutrino events

bkg: atm. muon events

*[40]K decays, bioluminescence etc.

Dataflow - Tier-1 offline computing



Tier-2 local computing clusters for simulation and analysis.

Tier-1 computing centres for calibration and reconstruction, simulation.

Tier-0 at detector site for triggering, online-calibration, quasi-online reconstruction.

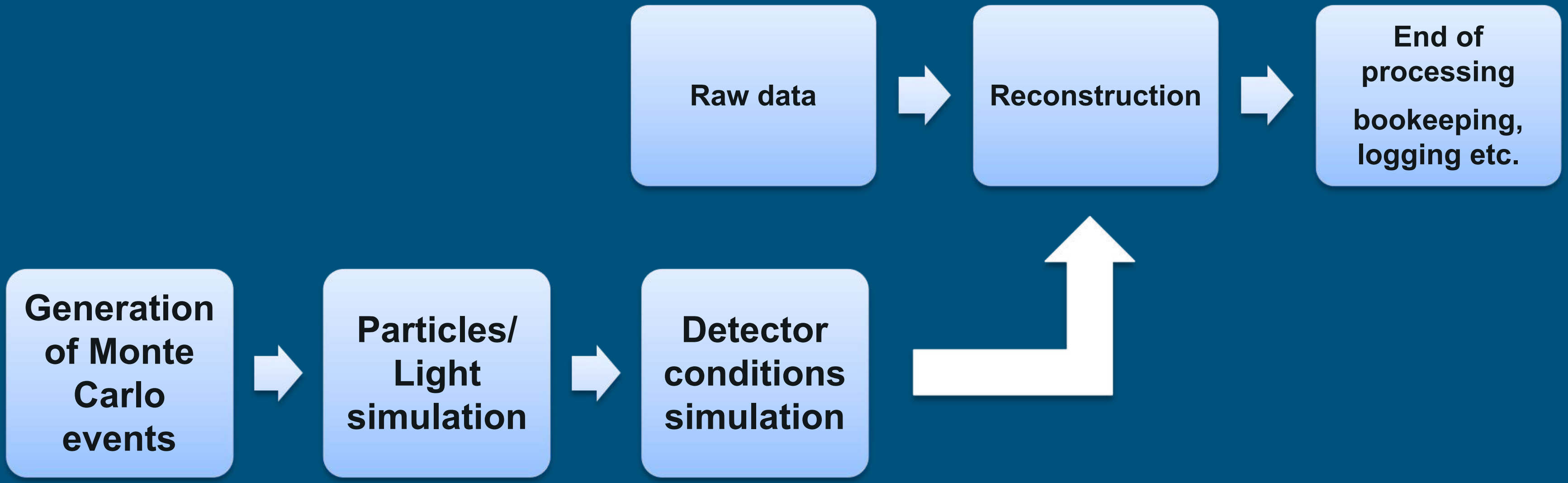
What is the type Tier-1 data ?

Data at the Tier-1 level can be divided into four types:

- **experimental raw data** (raw data in batches of time - runs),
- **acoustic data** (DAQ output - seperate channel - to be used for calibrations),
- **simulation data** (Monte Carlo simulations, neutrinos, atm.muons, pure noise events from ^{40}K decays in water),
- **condition data** (detector construction, calibration, logging, configuration, and environmental and run conditions - stored for experiment's lifetime).

*Mass data processing is organised in sets
(currently twice per year) of multiple data taking periods (runs) ~ O(5000).

Data processing in KM3NeT for analyses



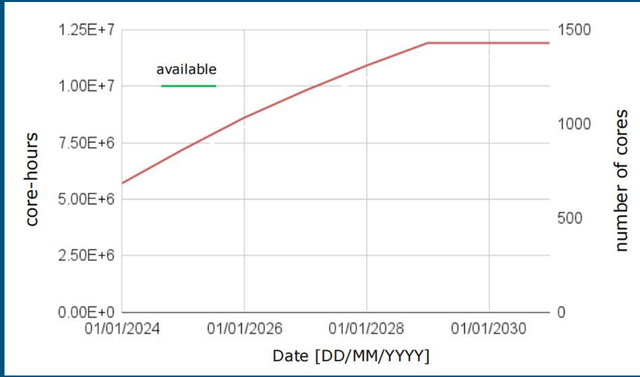
Experimental (raw) data

At the Tier-0 level, all data that are detected by the primary sensors (PMTs) are sent and filtered in the on-shore data acquisition systems (**all data-to-shore principle**).

During data processing, **the raw data are corrected for the calibration, and various models are fit to the data to determine the energy and direction of the neutrino candidate events.**

1. The **track reconstruction** fits the energy and direction of high-energetic muons. The muon trajectory can be described by 5 parameters: distance of closest approach muon track with PMT,PMT orientation angles,difference between and measured and expected hit time according to the Cherenkov hypothesis, energy of the muon.

2. The **shower reconstruction** also follows a two-step procedure: selects coincident hits on the same optical module within 20 ns and tests different direction hypotheses around the fitted vertex position. It can be improved when including timing information.



For 5% of a building block, the processing time of one data taking run (3h) was for ORCA: 200 CPU hours - single core and for ARCA: 125 CPU hours - single core

Monte Carlo simulations - Event generation

All flavour neutrino interactions simulated with the gSeaGen code.

- Efficiently generates high statistics samples of events in wide energy range, induced by neutrino interactions, detectable in a neutrino telescope.
- Considering topological differences between track-type and shower-like events.

Atmospheric muon simulation using the fast MUPAGE code.

- Large statistics can be produced with small CPU-time consumption.
- The flux of muon bundles at different depths and zenith angles, the lateral spread and the energy spectrum of the muons in the bundles are based on parametric formulas obtained according to a specific primary cosmic ray flux model and constrained by the measurements of the muon flux in the MACRO underground experiment.

Particles & Light simulation

Light tables from muons and electrons of given energies are provided by the KM3Sim and JSirene software package, using GEANT simulations, hit probabilities around a track segment, water and DOM properties. Multi-particle approximation for showers.

Light yield is provided as a weighted electron averaging particle properties.

Detector conditions simulation

Run-by-Run approach to the simulation;

to make a more reliable Monte Carlo simulation we are performing a “per run” data acquisition (DAQ) simulation . For each DAQ run corresponding MC files are produced!

With the TriggerEfficiency software we are simulating:

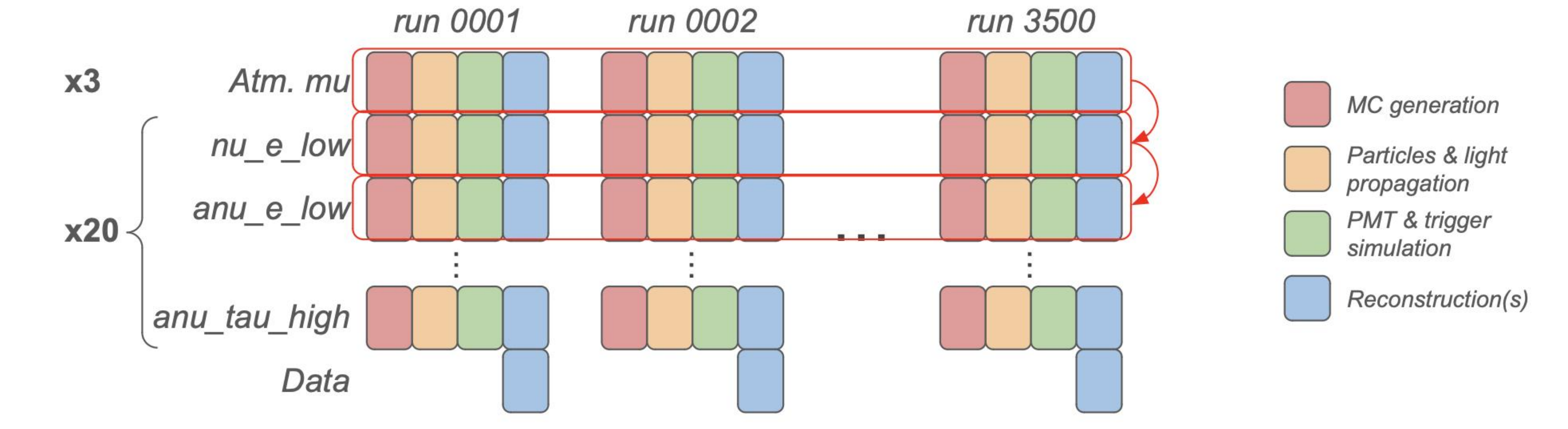
- realistic DOM and PMT behaviours (signal amplitude from lab measurements, TTS, etc.),
- the front-end electronics performance,
- the varying optical background (fixed rate or direct extraction from data files),
- diverse trigger setup conditítions, and
- singles rates

given from the given raw data file.

*Example: Bioluminescent creatures modifying the (stable) ^{40}K background on optical modules.

*Output in the same ROOT format as real data → directly processed with the reconstruction protocol

(old) Run-by-Run data processing workflow

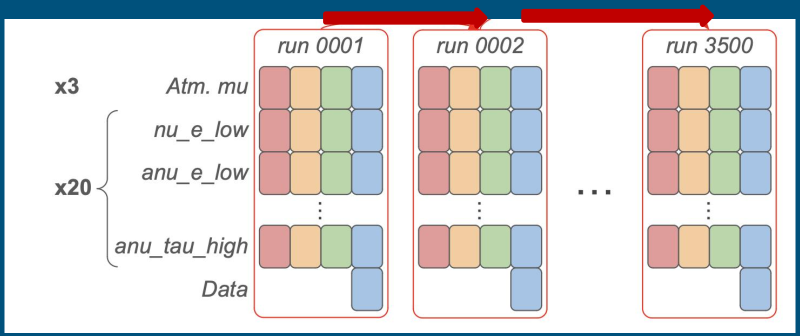


Past workflow example for ORCA processing: linear processing (run-by-run processing), 24 files per run needing merging after processing, lack of flexibility, difficulty to move to GRID etc.

Run-by-run data processing workflow

NOW: Run-based data processing with Snakemake*!

- Complete workflow management system.
- Containerized software.
- Singularity & Github based.
- “On-line” performance control.
- Mutualizing resources & disk space optimization during processing.
- Early file merging (not 24 files but 3!).
- Better logging, cleaner directories, easier error detection.




```
00000049/00007794
├── calibration
├── detector
│   └── logs
├── generator
│   ├── benchmarks
│   ├── logs
│   └── parameters
├── light
│   ├── benchmarks
│   ├── logs
│   └── parameters
├── reco
│   ├── benchmarks
│   └── logs
└── trigger
    ├── benchmarks
    └── logs
```

Run-based directory structure

*Mölder F, Jablonski KP, Letcher B et al. Sustainable data analysis with Snakemake [version 1; peer review: 1 approved, 1 approved with reservations]. F1000Research 2021, 10:33 (https://doi.org/10.12688/f1000research.29032.1)

Advantages of Snakemake workflow

Systematic logs collection.
Optimized demonstration of failures.
Possibility to restart the workflow from the step it crashed ⟹ optimizing resources usage.

Rules-wise summary

rule	Job in DAG	Planned	Waiting	Failed	Failed group	Failure [%]
benchmark_detid	1	1	1	0	0	-
benchmark_run	50	50	50	0	0	-
check_version_auxiliary_data	1	1	0	0	0	0.0
check_version_mupage_parameters_archive	1	1	0	0	0	0.0
do_detid	1	1	1	0	0	-
do_run	50	50	50	0	0	-
dst_production	200	200	200	0	0	-
export_configuration	1	1	0	0	0	0.0
gen_mc	1150	1150	0	75	0	6.521739
generation_timesort_Evt	1150	1150	918	0	75	0.0
get_calibrated_detector	50	50	0	0	0	0.0
get_calibration_archive	1	1	0	0	0	0.0
get_gdml_scheme	1	1	0	0	0	0.0
get_mupage_parameters_archive	1	1	0	0	0	0.0
get_raw_data_iRods	50	50	0	0	0	0.0
get_run_info	50	50	0	0	0	0.0
get_runs_info_for_detid	1	1	0	0	0	0.0
gseagen_generation	1100	1100	918	0	75	0.0
light_jsirene	750	750	586	0	75	0.0
light_km3sim	400	400	332	0	0	0.0
merge_files	50	50	50	0	0	-
mupage_generation	50	50	0	0	0	0.0
mupage_parameters	50	50	0	0	0	0.0
persistent_storage_local_run_level	400	400	400	0	0	-
process_all	1	1	1	0	0	-
prod_mupage_rate	50	50	0	0	0	0.0
reco_ORCA_JPPMuon_static	200	200	196	0	0	0.0
reco_ORCA_JPPShower_static	200	200	188	1	0	8.333333
reco_convert_to_offline	200	200	200	0	0	-
reco_merge_2_recos	200	200	200	0	0	-
trigger_jterbr	100	100	100	0	0	-
trigger_pure_noise_events	50	50	48	0	0	0.0

Mutualizing resources:
Each run processing is carried out in one place, allowing the use of a local cache directory. If one of the external input is missing, the processing stops before using too much resources.

Containerized software:
Ensuring software version control.
Management of 2nd order dependencies.
Easily integrate processing-ready scripts.

Processing at different computer sites & central data storage: CC-IN2P3, ECAP, Viper, CNAF, Nikhef

(User-Friendly) Snakemake workflow structure

- Steps are defined by **"rules"**, which denote how to generate a set of output files from a set of input files (e.g. using a shell command).
- Rules can be generated conditionally, arbitrary Python logic can be used to perform aggregations, configuration and metadata can be obtained and postprocessed in any required way.
- Dependencies between rules are determined automatically.
- Wildcards provide generalization.

```
rule check_input_detid:
    """
    For a given detid, check presense of per-run INPUTS_OK flags.
    """
    input:
        expand_on_run_for_detid("{{detid}}/{run}/KM3NeT.{{detid}}.{run}.INPUTS_OK")
    output:
        temp("{detid}/KM3NeT.{detid}.INPUTS_OK")
    shell: "touch {output}"
```

- By integration with the Conda package manager and containers, all software dependencies of each workflow step are automatically deployed upon execution.

```
envvars: # Required environement variables
    "KM3NET_SINGULARITY_DIR"

include: "modules/all.smk" # All smk submodules loaded at once

# Express global wildcard constraints
# ----------------------------------
wildcard_constraints:
    dtype="data|mc..*.jterbr",
    version="v[0-9].[0-99].*",
    run="\d{8}",
    detid="\d{8}",
    positioning="static|dynamic"

# Set workflow configuration
# --------------------------
set_singularity_args(workflow)
set_default_ressources(workflow)
set_config(workflow, config) # Check version, set runlist
set_default_cache_directory(workflow)

# Set onstart, onsucces and onerror functions
# -------------------------------------------
onstart: onstart_wrapper(workflow, config)
onsuccess: onsuccess_wrapper(workflow, config)
onerror: onerror_wrapper(workflow, config)
```

Further workflow applications

Profile files: .yaml files needed to define how snakemake interacts with batch system.

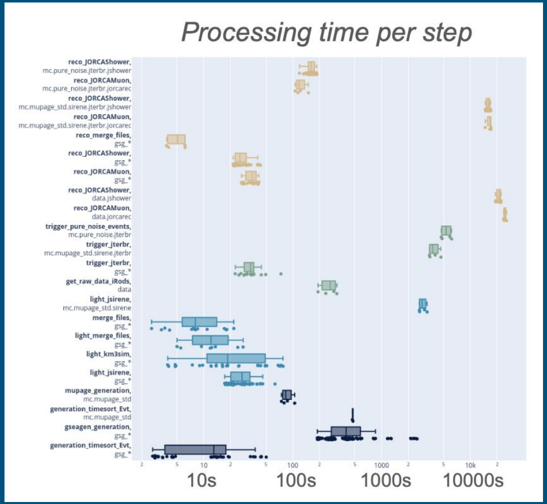
```yaml
cluster:
  mkdir -p logs/jobs; sleep $((5 + $RANDOM%30 )) &&
  sbatch
    --licenses=sps
    --time={resources.time}
    --mem={resources.mem_mb}
    --job-name=smk-{rule}-%j
    --output=logs/jobs/{rule}-%j.out
    -n {threads}
    --parsable

cluster-status:
  status_slurm.py

cluster-cancel:
  scancel

set-resources:
  - "light_km3sim:time=2-12:00:00"

default-resources:
  - time="2-00:00:00"
  - mem_mb=3050
  - disk_mb=3000
```

Snakemake files to define the container/software to be used, the desired parameters to be set and the input, output and dependencies.

With snakemake it is possible to **automatically generate detailed self-contained HTML reports** that encompass runtime statistics, provenance information, workflow topology and results.

```python
rule gseagen_generation:
    """
    Generate neutrino interaction using gseagen

    Output:
    --------
      - root: MC event file
      - binstat: TO COMPLETE I DON'T KNOW WHAT IT IS

    Input:
    -------
      - runinfo: contains run start and end time
      - can: text file that contains can defintion from detx file.

    Params:
    -------
      - conf: dictionnary corresponding to the type of neutrinos ({suffix})
      - theta: theta range for neutrino generation (-1,1 = full sky)
      - xsec: cross section used by GENIE
      - tune: water composition
      - flux: generation flux used for the neutrinos
      - nbin: TO COMPLETE ABSOLUTELY NO IDEA WHAT IT IS
      - weight: method to weight event. Default behavior in km3net is w=2.
      - seed: seed used for the generator.,
      - runinfo: dictionnary that contains runinfo (start,end of run)

    """
    output:
        root = temp("{detid}/{run}/generator/KM3NeT_{detid}_{run}.mc.gsg_{suffix}_UNSORTED.{version}.root"),
        binstat = temp("{detid}/{run}/generator/logs/KM3NeT_{detid}_{run}.mc.gsg_{suffix}_UNSORTED.{version}.binstat")
    input:
        runinfo = "{detid}/{run}/KM3NeT_{detid}_{run}.run_info",
        can = "{detid}/{run}/detector/parameters/can_definition_{detid}_{run}.txt"
    params:
        conf = gsg_config_parser,
        theta = "-1,1",
        flux = config['gseagen']['flux'],
        nbin = config['gseagen']['nbin'],
        weight = 2,
        seed = seed_from_outputs,
        propagator = "PROPOSAL"
    log:
        args = "{detid}/{run}/generator/parameters/KM3NeT_{detid}_{run}.mc.gsg_{suffix}.{version}.args",
        logs = "{detid}/{run}/generator/logs/KM3NeT_{detid}_{run}.mc.gsg_{suffix}_UNSORTED.{version}.logs"
    benchmark:
        "{detid}/{run}/generator/benchmarks/gseagen_generation/KM3NeT_{detid}_{run}.mc.gsg_{suffix}.{version}.tsv"
    container:
        config['gseagen']['container']
```

Run-by-Run mass processing in KM3NeT



Steps to be performed:

Input collection: Raw data (iRods), online/offline detector files (DB,Git), calibrations (Git).
Monte Carlo Generation: gSeaGen, MUPAGE
Light propagation: Km3sim, JSirene
Triggering: JTERBR, pure_noise MC
Reconstruction: JORCAMuon, JORCAShower, JARCAMuon, Aashower
Benchmarking steps

Different computing centers used and performance reports provided.

For the 2023-2024 mass processing:
1388.3 days of data livetime processed in some months(*).

- **DAQ failures: 2.2% of livetime to be processed**
- **Data_processing defects: 1.45% of initial livetime to be processed (<1%, corrupted metadata - <0.5%)**
- **Calibration dailures - 29.3 days, 1.9% of initial livetime to be processed**

Conclusions

KM3NeT is building 2 underwater neutrino detectors in the Mediterranen Sea.

Monte Carlo simulations consider the varying data acquisition conditions in the underwater environment. The KM3NeT run-by-run simulation system allows to describe most of the effects that are relevant to the data taking.

A new run-based data processing workflow using Snakemake has been introduced and used for the first time in the 2023-2024 mass processing.

Optimized performance has been achieved using containerized software, mutualization of the resources and multiple computing centers for the processing.

All the results shown since 2024 are based on the new processing chain!

Exciting KM3NeT news coming soon...

Stay tuned!

Thank you for your attention!