PROCEEDINGS OF SCIENCE

Management of Physics Simulations with Databases

Viktoria Pauw[a,1,*], David Číž[b], Vojtěch Mlýnský[c], Pavel Banáš[d], Michal Otyepka[b,d], Stephan Hachinger[a] and Jan Martinovič[b]

[a] *Leibniz Supercomputing Centre (LRZ) of the Bavarian Academy of Sciences and Humanities (BAdW), Boltzmannstr. 1, 85471 Garching b.M., Germany*

[b] *IT4Innovations National Supercomputing Center, VŠB – Technical University of Ostrava, 708 00 Ostrava, Czech Republic*

[c] *Institute of Biophysics of the Czech Academy of Sciences, Královopolská 135, Brno 61200, Czech Republic*

[d] *CATRIN, Šlechtitelů 241/27, 779 00 Olomouc, Czech Republic*

E-mail: pauw@lrz.de

The on-going work presented in this article explores different technical approaches and systems for management and analysis of data obtained from large physics simulations, optimising the respective data-driven workflows across Cloud-Computing (IaaS) and HPC systems. The work is carried out in the context of the EXA4MIND Horizon Europe project, which produces an Extreme Data processing platform, bringing together specialised data management systems and powerful computing infrastructures. We evaluate two typical use cases with physics simulations carried out on supercomputing systems at LRZ (Garching b.M./DE) and IT4Innovations (Ostrava/CZ). These use cases come from different areas of physics – they focus on the treatment of low energy many-body systems of molecules, and of high-energy (relativistic) elementary particles, respectively. Accordingly, molecular dynamics (MD, low energy) and plasma simulation methods (FDTD, Particle-in-Cell, high energy) are used. As often in computationally supported, data-driven science, a large fraction of the work then goes into postprocessing, visualising and re-assessing the data, often several times in an iterative process. A well-managed, integrated and efficient "next-generation" methodology to facilitate and manage such a (re-)use of the valuable data – in particular in the context of parameter studies – with an eye on FAIR research data management, is one of our final objectives.

International Symposium on Grids and Clouds (ISGC2024)
24 -29 March, 2024
Academia Sinica Computing Centre (ASGC), Institute of Physics, Academia Sinica
Taipei, Taiwan

[1]Speaker

PoS(ISGC2024)031

1. Introduction

Physics simulation modelling systems of atoms or subatomic particles are a typical application challenging the capabilities of high performance computing (HPC) systems. These simulations follow up to billions of particles, writing out their properties and location trajectories in each time step. The production of such simulation outputs can take up to hundreds of thousands CPU and GPU hours, and the particle and field data can occupy Gigabytes or Terabytes of storage for one simulation. These then have to be postprocessed and evaluated in order to reach conclusions that can be used for scientific advancement. Examples of such processing workflows include aggregation of fields or particles over the domain partitions, like patches, divided over several simulation processes, extraction of statistical information over particles or timesteps, calculations of spatial distributions over the domain, like plasma densities or the time evolution of distances between atoms in the molecule. Statistical evaluation or aggregation can happen on various levels – from ensembles of simulations, for example parameter studies, down to single particle trajectories or timesteps which are only a fraction of the whole output and have to be extracted from much larger dataset. Even for one study, the datasets are revisited typically several times, for example for visualisation, comparison against experimental data and evaluations of the validity of force field parameters via reweighting techniques [1][2]. For such purposes, an efficient and systematic data management is paramount.

While companies often use database management systems (DBMS) for dealing with data in frequent re-use, scientific supercomputing makes little use of them. This is at least in part due to a preference for direct file I/O without overheads and technical challenges in reaching DBMS from HPC cluster nodes. However, the increasing re-usage of scientific output data, for which we have given examples above, gives a strong motivation to re-visit the topic of proper data-management methods and systems, also keeping in mind the FAIR principles (Findable, Accessible, Interoperable, Reusable [3]) any modern research data management (RDM) should obey. The work we present here makes a contribution in this. It has been conducted in the context of the the Horizon Europe project EXA4MIND[2] on Extreme Data processing. EXA4MIND aims at bridging the ecosystems of DBMS, supercomputing, and European Data Spaces. It builds an extreme data platform with an "Extreme Data Database" (EDD) at its core. This EDD integrates specialised DBMS and storage systems, and interfaces them in an unified manner, facilitating optimised data-driven workflows across top-notch Cloud-Computing and HPC systems at IT4Innovations[3] (Ostrava, CZ) and LRZ[4] (Garching b.M./DE). It is to be connected to European Data Ecosystems such as EUDAT[5] and European Data Spaces such that FAIR RDM is facilitated as well.

In this contribution, we show first results on the way towards an optimal data management for our use cases. We test the performance of different data-management backends (in particular file-based storage vs. DBMS) with respect to data queries and iterative postprocessing steps. Clearly, the performance also depends on data access and parallelization schemes, a topic which we will briefly touch upon as well. In particular, row-based DBMS such as PostgreSQL will behave differently to column-based DBMS such as MonetDB or MariaDB Columnstore. Potential performance benefits may be reached with techniques like SciQL [4] which can operate using live queries on large array-based datasets in memory.

[2] "EXtreme Analytics for MINing Data spaces", https://cordis.europa.eu/project/id/101092944
[3] IT4niovations National Supercomputing Centre (cf. affiliations) – https://www.it4i.cz
[4] Leibniz Supercomputing Centre (cf. affiliations) – https://www.lrz.de
[5] https://eudat.eu

PoS(ISGC2024)031

The test data sets we center this work upon are, on the one hand, from Molecular-Dynamics (MD) simulations (Modelling for Nanotechnologies Lab, IT4Innovations, Ostrava/CZ), where dynamical movement and structural description of biomolecules is calculated (predicted) via empirical potentials (force fields, FF) [5]. On the other hand, we have outputs from the Plasma Simulation Code [6-8], simulating the fields and trajectories of charged particles in plasma physics (PP). Simulation-based research as at hand here typically includes multi-step postprocessing and evaluation pipelines to gain insights and knowledge about the properties of the physical systems represented. For example, plotting and visualization is an important step to verify correct set-up and disseminate results. These workflows typically involve a lot of manual labour and repeated adjustments requiring repetitive tasks and focused attention from the researcher to avoid mistakes. Automating these workflows and managing the data in systematic frameworks can potentially speed up these processes while reducing the risk of errors and simplifying re-use at a later point. Besides storage optimisation, clearly also systematic orchestration of tasks (e.g. via the LEXIS[6] workflow and data-management platform) will play an important role in increasing the efficiency of such research. FAIR RDM, finally, will hopefully help to create synergies across scientific groups and allow for efficient collaboration, also involving industry and SMEs. In this setting, EXA4MIND, LEXIS and the effort presented here are positioned as technological enablers for data-driven research across Europe.

This paper is organised as follows: section 2 introduces the use cases in PP and MD in some more detail. Section 3 gives some general considerations on data management, before we test – based on the use cases described – the performance of example storage schemes (DBMS, files) with representative data-analysis tasks. Section 4 describes aspects of FAIR data management to be considered in our use-cases in the near and further future, before we conclude in Section 5.

[6] "Large-scale EXecution for Industry and Society", https://lexis-project.eu, cf. H2020 project https://cordis.europa.eu/project/id/825532

PoS(ISGC2024)031

2. Applications ("Use Cases") Considered in this Work

2.1 Plasma Physics Use Case

The PSC [6-8] was used to run an large parameter study to determine which target and laser pulse properties produce the fastest ions in a simulation modelled after a Paul trap experiment carried out at J. Schreiber's group at LMU [9-13].

The modes of absorption of energy from electromagnetic fields to plasma are very complex and the interplay of different set-up parameters can drastically alter the dynamics of the production of fast particles leading to very varying outcomes in terms of particle spectrum.

The output files produced by these simulations contain the information of the particles position on the grid (x,y,z coordinates) and there velocities. They are written out by each process (MPI rank) separately and then later combined by postprocessing scripts. The sizes of these files (written in HDF5 format) vary in size from 10s of MB to TB per timestep, depending on the number of particles in the simulation, which ranges over many orders of magnitude from a few thousands to several billions.

When evaluating such a parameter study, it is of interest to both find the optimal parameter combination and understand the dynamic of the plasma movement that lead to the production of a high number of fast particles.

Time evolution of the particle density plotted as 2D slices or 3D iso-surfaces can give an intuitive overview of the dynamics, while the trajectory of particularly fast electrons and ions can give valuable insights into the acceleration dynamics producing the coveted ultra-fast ions.

The researcher may want to adjust the parameters of the visualization like the threshold at which particles are sorted into the "fast" category, to enhance the visibility of certain features. Thus, the plotting process may be repeated several times for each of the dozens of simulation runs.

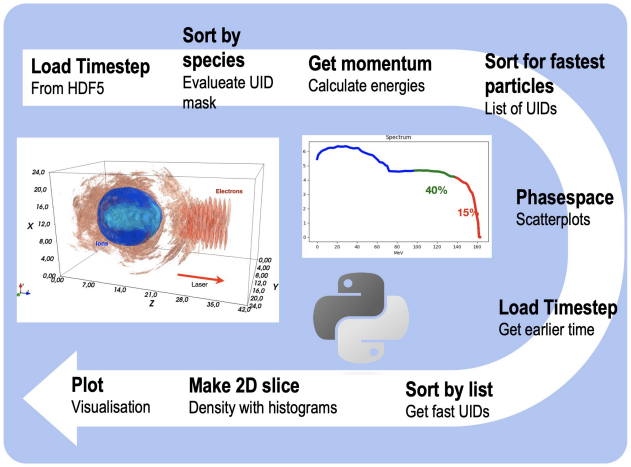

Figure 1: Sketch of Fast Particle Tracking workflow.

The main steps of this process (comp. Figure 1) are:

- Load the timestep at which particles have reached their maximum energy
- Sort by particle type and energy
- Store the fast particles' UID in list
- Load each earlier timestep
- Filter by fast particle UID list to generate coordinate and velocity information for these individual ions

PoS(ISGC2024)031

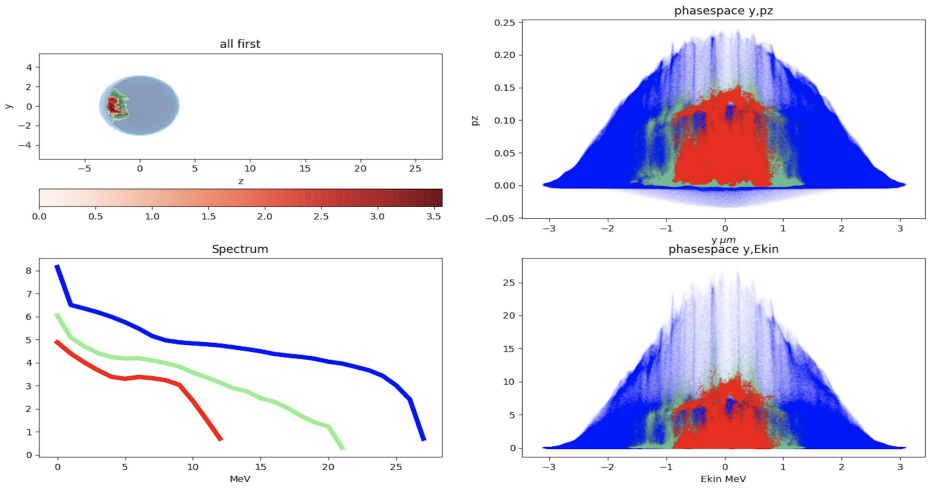

Figure 2: Example timestep showing the protons that will later on be accelerated to highest energies (red), medium energies (green) and low eneries (blue) at the beginning of the simulation, when the laser field starts impinging on the target. Plots show the grid position in x, z dimension (above left), phasespace in lateral (y) position and momentum (above right), early spectrum in number of particles (log10) vs. Kinetic energy (MeV), and phasespace in lateral (y) dimension and kin. Energy (bottom right)

This workflow requires the loading of large datasets several times. The time expensive parts of this task are particularly the loading of data from storage and the sorting/filtering by criteria such as matching UID. Therefore the performance of these process is of particular importance to streamlining the execution of these repetitive workflows.

Another important aspect, is finding the right simulation matching the selection criteria for evaluation, such as the same size but different densities of the target or the same target properties but different pulse shapes. In the planned database set-up, the simulations and their evaluation output are ordered in a hierarchical form, with general set-up parameters that only have a few possible values, like ion constellation or pulse polarization and metadata at the top level and the individual runs of the parameter study as an information table linking (via an id, path and name) to the actual files. Finished postprocessing output, such as rough statistics and existing plots can then be linked from this table to avoid unnecessary re-processing.

PoS(ISGC2024)031

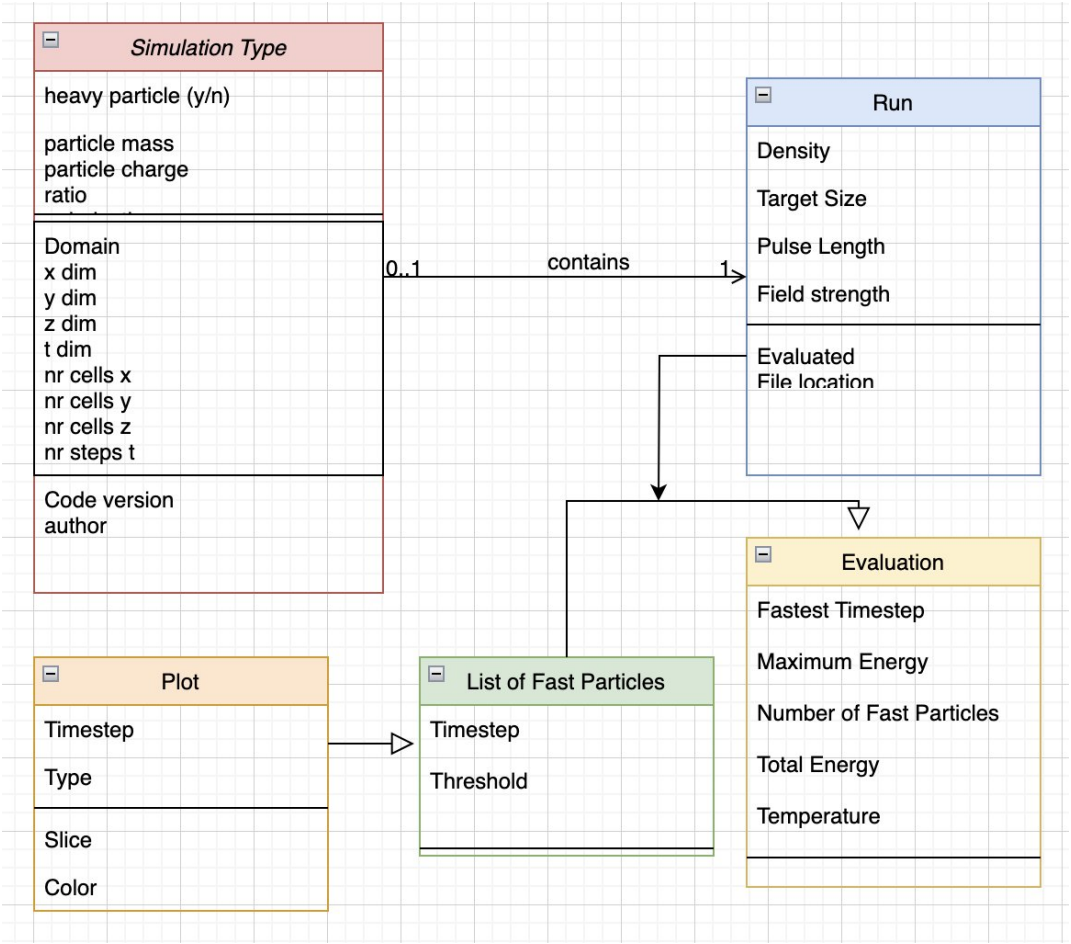

Figure 3: Sketch of table scheme for plasma physics DB set-up.

PoS(ISGC2024)031

2.2 Molecular Dynamics Use Case

Having important applications ranging from drug discovery over structure and function relationships to protein/nucleic acid design, MD simulations support a wide ranging field of physical chemistry research. The field has mainly been advanced by the development of empirical potentials, or force fields (FF), which can capture and predict many structural features of biologically relevant systems. One challenge is, that researchers often encounter problems in simulations, which stem from overparameterization and overfitting of the FF (artifacts). Current developments in the methods of force-field elicitation and improvement (e.g. through additional force-field terms) suggest that better alternatives of processing than manual or semi-automatic tuning/development of FF parameters are needed, as humans cannot capture the whole complexity involved in this process. We are certain that the next step of FF tuning must be automated and assisted by machine learning (ML) approaches on top of huge datasets acquired from HPC MD simulations. For this project work, MD simulation datasets, i.e. topologies and trajectories of simple RNA motifs (e.g., tetranucleotides and tetraloops) obtained by the state-of-the-art force field, are taken from [14], reference evaluations are from [15].

To facilitate the fully automatic processing of the assessment and re-weighting cycle, the EXA4MIND project partners are building a DBMS to store and access MD simulations. We execute benchmarks of typical involved queries, such as calculation of distances and dihedral angles of atoms within the simulated molecules to help design decisions for the storage structure in these DBs.
The main steps of a preliminary evaluation workflow is shown in Figure 4. The current implementation uses C++ based tools (cpptraj) and AWK based scripts. It is now being re-implemented in python using a postgres and hdf5 for storing metadata and simulation data respectively.

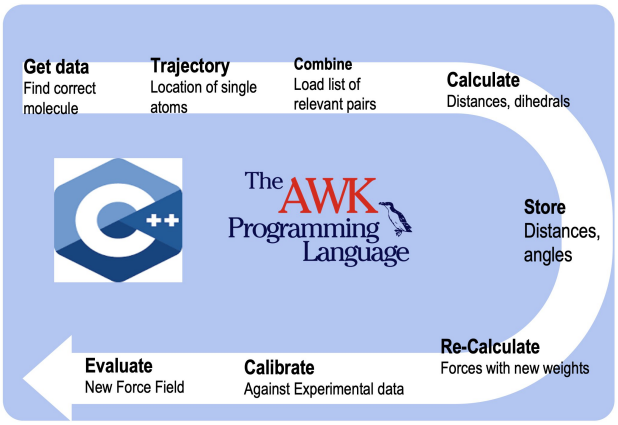

Figure 4: Rough sketch of evaluation steps for MD output.

3. Performance of Typical Data-Analysis Workflows Using Different Data Management Approaches

In this section, we discuss the performance of different data-management approaches, in particular when running a few typical data analysis and re-use tasks from our use cases. Subsection 3.1 first considers general trade-offs to be considered when devising a data management methodology for our use cases. The other subsections then discuss the backend-dependency of the performance of our sample data-evaluation tasks.

For an optimized set-up, the nature of the typical or most expensive queries should be evaluated to determine the optimal structure of the datasets in storage.

3.1 Trade-Offs to be Considered when Devising a Data Management Methodology

3.1.1 Flexibility vs. Efficiency

When designing data structure layout and analysis tools, one often encounters strong constraints related to the anticipated main usage. An example from the use cases in this study is the sorting of the data points by either timesteps or particle types in storage, when the main application is evaluating trajectories of particles. A trajectory of a single atom or electron can be plotted much faster, when the time ordered coordinates of this particle are stored consecutively rather than distributed over lists of tens to hundred thousands (as in MD) or billions (in PP) of particles lumped together by timestep.

3.1.2 Redundancy/Speed vs. Compactness

Naively-conceptualised data representations used by typical simulation codes often contain redundant information, which can actually accelerate data processing, depending on the circumstances. However, when such information is dumped to disc in the simulation output, it unnecessarily uses storage space and certain analytics on the output data might become more difficult: the volume of the entire dataset may be too large to fit reasonable portions into RAM, and convolved data structures can slow down data analytics. Well-understood data organisation is thus a prerequisite for efficient re-usage of scientific outputs in the longer term.

3.1.3 Performance of DBMS vs. File Storage

In HPC, input and output data are traditionally stored in files, with an increasing use of container file formats (e.g. HDF5 [16]) for which parallelized read/write access has been

PoS(ISGC2024)031

implemented in special libraries. The EXA4MIND project aims to evaluate in which cases the use of DBMS instead of files, or mixed with files (e.g. for the largest portions of input/output data) is beneficial to overall performance. The general idea to be confirmed by tests is, that DBMS have their strengths when the data retrieval is preceded by a complex query, which is the correct data portion to be loaded. A higher performance can then be achieved when the data is in a format for which the respective database software and access pattern is optimised.

However, in any case, when intending to use DBMS for high-performance data analytics one has to consider that the DBMS is an additional layer between the file system and the data analytics or simulation software, coming with overhead. Thus, DBMS can generally only accelerate data-driven workflows when these overheads are minimised.

Considering this, and the fact that the data organisation has to be intuitively understandable and accessible for the domain researcher, a typical use case requires design decisions following tests and benchmarks. These design choices include the structure of the tables in the DBMS. In addition, it must be understood whether it is beneficial to store large datasets (e.g. complete simulation data) in the database or whether it is preferable to restrict the database to metadata and simulation parameters, while keeping most of the raw data in files (which can be referenced in the database).

When choosing either DB tables or files, the hierarchy and arrangement of data can make several orders of magnitude difference to query performance. Also, parallel performance depends strongly on the data management backend and its parallel-access capabilities, where HPC centres have built most of their experience with direct file access and not with DBMS.

3.2 Data Retrieval Evaluation on Row vs. Column DBMS

The layout of the data in storage can have a big influence on data retrieval time. If the queries are suitable for array-like execution, cache misses can be significantly reduced. For a simulation dataset of 2 GB, for example, we identified a significant improvement of runtime with MariaDB Columnstore as compared to the regular row based storage engine of MariaDB (see Table 1).

Two benchmark queries were executed in this example: one obtaining a simple value typical for first time evaluation of plasma physics runs - the total energy of the particles, and another calculating average distances of particles from the origin. These aggregations were calculated within the database, fetching all values from a large single table for one simulation (second and third column of Table 1, for regular/row and Columnstore storage). For a HDF5/NumPy[7]-based reference calculation (first column), the same data was stored in a hierarchical HDF5 file using groups for different chunks of timesteps and datasets for particle types. Comparing the runtimes shows that the assumption that DBMS overhead will in all cases mean a slower execution time than classical file access is not necessarily true. In the case here, datasets from the HDF5 files had to be re-aggregated in memory by the script, making it slower for the distance queries than the direct calculation in the database.

[7] https://numpy.org/

PoS(ISGC2024)031

The picture changes, however, for a different data layout (compare Table 3 for MD use c

ase), where entire atom trajectories are stored in one dataset of the HDF5 files. Comparing the load times in this scenario, we see a considerable overhead compared to HDF5/NumPy (at least for small datasets of 300MB) for fetching data from the database by queries on large tables containing join statements. It is then not relevant, whether the calculation is executed inside the database with a SQL query or using NumPy. The processing in NumPy is very fast if the necessary coordinates can be retrieved in full directly from a few datasets within the HDF5, but less so, if many chunks have to be looped over to be opened and the contained data reassembled in an NumPy array in memory.

	hdf5 & numpy	row DB	column DB
Distance small	5.2 (0.2) s	0.09	0.11
Energy small	3.5(2.6) s	0.04s	0.06s
Distance Large	110 (73)s	46s	13s
Energy Large	89 (51) s	247 (46) s	15s

Table 1 HDF5 re-assembly vs. RDBMS vs. CDBMS; numbers in brackets are the execution time when the data is already in cache, after prior work on the same data chunk.

3.3 Impact of General Dataset Structure and Parallelization Approaches

To evaluate the impact of general data structure on query speeds in an example, we compare the performance of two data structures for storing MD data in a PostgreSQL database (Table 2): All trajectory points for all atoms in one table vs. separate tables for each atom. For atom-distance calculations from MD examples, we find that the lookup-time for the coordinates of an atom from a PostgreSQL database, using the Python API, is reduced by a factor of 10-50 when separate tables are used for each atom instead of one table for the complete simulation (second vs. third data column of Table 2). With respect to a query evaluating the distance for all atoms in the dataset (the example dataset used for this test contained 149 atoms) which are in the combined table (Table 2, first column), pre-filtering for a single atom with a "where" clause and doing the calculation only for this atom is faster (column 2), which is not a surprise. However, we observed that for an atom-to-fixed-point distance evaluation, there is only a factor ~2 difference between calculating that distance for all atoms vs. one atom (first vs. second column of first/second row of Table 2). This suggests that, in case of a combined table, lots of time is actually spent in

	Combined table all	Combined table Filter for atoms	Separated table per atoms
Distance 1 atom short	3.565 s	1.215 s	32 **ms**
Distance 1 atom long	6.470 s	2.602 s	50 ms
Distance 2 atom short	142.486 s	1.253 s	85 **ms**
Distance 2 atom long	630.277 s	2.717s	124 ms

Table 2: Runtime of query (with data evaluation) showing the difference in look-up time of MD data for either 2008 timesteps (short), or 1000000 timesteps (long) caculating the distance between either an atom and a fixed point (1 atom) or two (2) atoms in these scenarios (columns): 1) c all atoms in the simulation in one table, 2) as 1, retrieving an atom with a "where" clause, 3) with retrieving atoms from separate tables for each atom type.

PoS(ISGC2024)031

loading the table (and rather little time in the actual computation), were the DBMS apparently succeeds in avoiding a re-load when evaluating 149 distances instead of one. The "caching benefit" however vanishes (third/fourth row of Table 2 and also Table 3) when access patterns are too complicated (dihedral comptutation requiring several fields from 4 atoms) or datasets do not fit completely into memory.

The measurements we show in Table 3 again touch upon the question whether it is faster to evaluate file-based or database data. Calculating a single dihedral angle for four atoms takes about the same time in the both HDF5/NumPy set-up and accessing PostgreSQL via the python API (see Table 3, second row). Also a mixed approach, where the data is fetched via a query and the calculation is done with NumPy yields approximately the same execution time. While the distance calculation for two atoms is faster in PostgreSQL for a single query (Table 3, first row), repeated serial accesses to the database lead to long runtimes (Table 3, last two rows). In contrast, accessing the HDF5 many times, when it is already loaded to memory remains fast. Testing a typical MD script to evaluate a molecule, requiring 40 distance calculations among critically positioned atoms, we measure 11 seconds for the 40 consecutive HDF5 accesses compared to vs 3.7 seconds for the calculation of a single pair, and 62 seconds vs 1.6 seconds for 40 consecutive DB queries. This shows that multiple repeated accesses to the database have probably not been accelerated by any caching in our set-up. This points to a likely efficiency issue when using DBs for large datasets and should be addressed when planning such a set-up. Retrieval of small data portions however can be very performant in this setting.

	hdf5 & numpy	PosgresSQL	PosgresSQL + numpy
Distance pair	3.7s	1.6s	-
Dihedral pair	3.7s	3.7s	3.7s
Distance all	11 s	62s	-
Dihedral all	66 s	95s	103s

Table 3: MD evaluation example runtime comparision - details in text

Typical memory and cache sizes must be considered in optimal data division, because, as seen in the above tests, re-loads significantly decrease performance. A single table or file object (e.g. HDF5 dataset) should not be larger than the typically available memory divided by the number of concurrent processes times the number of datasets concurrently in use by the calculation. For example in a distance calculation in MD, these are the 3 space coordinates for 2 atoms. If several distances are calculated from the same atom, it is beneficial to order the execution to keep the one atom in memory for all calculations. Here, shared memory parallelization (such as OpenMP or python multiprocessing) can be helpful, to avoid loading a dataset several times by separate processes.

In the plasma physics use case, in contrast, most steps of the workflow can be performed concurrently in an embarassingly parallel fashion. Evaluation scripts or plotting functions are typically performed separately for each timestep. If a comparison of values over several timesteps, e.g maximum energy, is required, it is beneficial to store it in a metafile after the first processing run (see table scheme in Figure 3). Parallelisation can be approached in scripts by classical parallelisation via python modules like mpi4py, subprocess or dask, where each process loads data from one timestep.

4. FAIR Research Data Management Aspects, Treatment of Metadata

As stated above, our work – in addition to the focus on performance – has the ambition of implementing FAIR [6] RDM in the data lifecycle to an adequate degree. Core requirements in

PoS(ISGC2024)031

FAIR RDM center on the assignment of persistent identifiers and appropriate metadata to datasets. Our final data products of general interest shall be published with Digital Object Identifiers (DOIs, [17]). However, already at the data-production and data-management stages, the scene has to be set for making our outputs intrinsically FAIR and reproducible. Our approach to this foresees the assignment of persistent identifiers (PIDs, e.g. B2HANDLE handles [18]) to intermediate results, and – most importantly – the annotation of intermediate and final results with technical, structural and descriptive metadata. Examples on this, in the context of the use cases presented above, are given below.

4.1 Structural and Technical Metadata

The use of DBMS implies the creation of a proper database schema, and also when using file based storage, we devise an adequate structure in our use cases. We document the relation of the storied entities; a glimpse on this has been given in Figure 1. In addition to this structural information, technical metadata facilitate a proper understanding of the data held.

In the plasma physics use cases, technical parameters that need to be part of the annotation include *(i)* atom- or particle types contained in the target, *(ii)* the initial temperature and density distribution, including target shape and size, *(iii)* pulse parameters like focal size, pulse shape and length, field-strength, polarisation (linear or circular) and wavelength, and *(iv)* spatio-temporal domain and resolution parameters, such as the size of the simulation domain, the grid resolution, the number of simulation timesteps and output frames, grid coordinates, and the particle resolution.

4.2 Descriptive Metadata

FAIR research data management relies to a large part on descriptive metadata, including standard elements such as the dataset creators with their affiliations, the dataset title and a high-level description of the dataset as well as the usage license or conditions. The assignment of certain flavours of PIDs, in particular of DOIs via registration agencies such as DataCite[8], already requires the submission of descriptive metadata to a central registry.

4.3 Implementation: Storing Metadata

A standard method to store the metadata elements outlined above is adding them to a data catalogue for one or more research projects. However, when having a DBMS at hand to store actual scientific data in a database, adding metadata into a separate table in the database has clearly an appeal, in particular when using relational databases. Detailed metadata, e.g. describing only parts of the dataset or even single data points, can then be collected and technically related to these data points with standard relational-database methods.

5. Conclusion and Outlook

During the ongoing EXA4MIND project we aim to show that modern data storage concepts involving DBMS can be employed to improve performance, long term data use and enable data sharing and systematic metadata management for data on the 1-100 TB scale generated by HPC simulations in (bio-)physics. We aim at facilitating a research data management scheme that helps scientists leverage the full potential of their simulation results over time frames, and envision incorporating the FAIR principles from the start to sharpen the researchers awareness of their benefits and foster in-built sustainability in data evaluation processes.

To this end, our work will continue with extensive benchmarks pertaining to data querying and retrieval on HPC systems, considering file- and DBMS-based back-ends for data management.

[8] https://datacite.org

PoS(ISGC2024)031

Acknowledgements

A warm thank you to the team from the project partners (IT4Innovations, VSB, METU) and colleagues at LRZ (M.Hayek, A.Batsaikhan) for the good collaboration and lively discussions. Plasma simulations were carried out on SuperMUC and SuperMUC-NG under GCS project pr74si, the Linux Cluster and Compute Cloud at LRZ; systems at IT4Innovations were used for MD simulations. This work was furthermore supported in parts by the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90254).

Parts of this research received the support of the EXA4MIND project, funded by the European Union´s Horizon Europe Research and Innovation Programme, under Grant Agreement N° 101092944. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the granting authority can be held responsible for them.

References

[1] T. Shen and D. Hamelberg, *"A statistical analysis of the precision of reweighting-based simulations"*, J. Chem. Phys. 129(3) (2008) 034103, https://doi.org/10.1063/1.2944250

[2] V. Mlynsky, P. Kuhrova, P. Stadlbauer, M. Krepl, M. Otyepka, P. Banas, and J. Sponer, *"Simple Adjustment of Intranucleotide Base-Phosphate Interaction in the OL3 AMBER Force Field Improves RNA Simulations"*, J. Chem. Theory Comput. 19(22) (2023) 8423, https://doi.org/10.1021/acs.jctc.3c00990

[3] M.D. Wilkinson et al., *"The FAIR Guiding Principles for scientific data management and stewardship"*, Scientific Data 3 (2016) 160018, https://doi.org/10.1038/sdata.2016.18

[4] H. Lustosa, F. Porto, P. Blanco and P. Valduriez, *"Database System Support of Simulation Data"*. Proceedings of the VLDB Endowment (PVLDB), VLDB Endowment 9(13) (2016) 1329

[5] J. Sponer et al. (2018), *"RNA structural dynamics as captured by molecular simulations: a comprehensive overview."*, Chem Rev 118(8) 4177, https://doi.org/10.1021/acs.chemrev.7b00427

[6] H. Ruhl, https://www.plasma-simulation-code.net, Cited 5 May 2024

[7] K. Germaschewski, https://github.com/psc-code/psc, Cited 5 May 2024

[8] K. Germaschewski, W. Fox, S. Abbott, N. Ahmadi, K. Maynard, L. Wang, H. Ruhl and A. Bhattacharjee, *The Plasma Simulation Code: A modern particle-in-cell code with patch-based load-balancing*, Journal of Computational Physics 318 (2016) 305, https://doi.org/10.1016/j.jcp.2016.05.013

[9] B. Liu, K.U. Bamberg, N. Moschüring and V. Pauw, *"PSC Simulation Support for Novel Accelerator Concepts"*, in : P. Bastian, D. Kranzlmüller, H. Brüchle and M. Brehm (eds.), *"High Performance Computing in Science and Engineering"*, Garching/Munich, Germany (2018) ISBN: 978-3-9816675-2-3

[10] T.M. Ostermayr et al., *"Proton acceleration by irradiation of isolated spheres with an intense laser pulse"*, Physical Review E 94(3) (2016) 033208, https://doi.org/10.1103/PhysRevE.94.033208

[11] T.M. Ostermayr et al., *A transportable Paul-trap for levitation and accurate positioning of micron-scale particles in vacuum for laser-plasma experiments.,* Rev. Sci. Instrum. 89(1) (2018) 013302, https://doi.org/10.1063/1.4995955

[12] P. Hilz et al., "Isolated proton bunch acceleration by a petawatt laser pulse", Nature Communications. 423 (2018), https://www.nature.com/articles/s41467-017-02663-1, https://doi.org/10.1038/s41467-017-02663-1

PoS(ISGC2024)031

[13] V. Pauw et al., *Particle-In-Cell simulation of laser irradiated two-component microspheres in 2 and 3 dimensions*, Nucl. Instrum. Methods Phys. Res. A 829 (2016) 372, https://doi.org/10.1016/j.nima.2016.02.012

[14] V. Mlynsky, M. Janecek, P. Kuhrova, T. Frohlking, M. Otyepka, G. Bussi, P. Banas and J. Sponer, "*Toward Convergence in Folding Simulations of RNA Tetraloops: Comparison of Enhanced Sampling Techniques and Effects of Force Field Modifications*" . J. Chem. Theory Comput. 18(4) (2022) 2642, https://doi.org/10.1021/acs.jctc.1c01222

[15] T. Cheatham et al., https://amberhub.chpc.utah.edu/cpptraj/ , Cited 5 Mai 2024

[16] M. Folk, G. Heber, Q. Koziol, E. Pourmal and D. Robinson, *"An overview of the HDF5 technology suite and its applications."* In: *Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases (AD '11)*, Association for Computing Machinery, New York, NY, USA (2011), https://doi.org/10.1145/1966895.1966900

[17] N. Paskin, *"Digital Object Identifiers for scientific data"*, Data Science Journal, 4(0) (2006) 12, https://doi.org/10.2481/dsj.4.12

[18] D. Lecarpentier, P. Wittenburg, W. Elbers, A. Michelini, R. Kanso, P. V. Coveney and R. Baxter, *"EUDAT: A New Cross-Disciplinary Data Infrastructure for Science*", Int. J. Digit. Curation 8 (2013) 279, https://doi.org/10.2218/ijdc.v8i1.260

[19] H. Daido, M. Nishiuchi and A.S. Pirozhkov, *"Review of laser-driven ion sources and their applications."*, Rep. Prog. Phys. 75(5) (2012) 056401, https://doi.org/10.1088/0034-4885/75/5/056401

[20] A. Macchi M. Borghesi and M. Passoni, *"Ion acceleration by superintense laser-plasma interaction."*, Rev. Mod. Phys. 85(2) (2013) 751, https://doi.org/10.1103/RevModPhys.85.751

[21] D. Tskhakaya et al., *"The Particle-In-Cell Method"*, Contrib. Plasma Phys. 47 (2007) 563, https://doi.org/10.1002/ctpp.200710072

[22] S. Bottaro, G. Bussi, S.D. Kennedy, D.H. Turner and K. Lindorff-Larsen, "*Conformational ensembles of RNA oligonucleotides from integrating NMR and molecular simulations*", Sci. Adv. 4(5) (2018) eaar8521, https://doi.org/10.1126/sciadv.aar8521

PoS(ISGC2024)031