Raising the treasure of SPDMs – How data compression and automatized event detection support engineers

D. Borsotto, V. Krishnappa, S. Müller, F. Natter, T. Roth, K. Schreiner, H. Talaat, C.-A. Thole, T. Weinert

SIDACT GmbH, Germany

1 Introduction

To cope up with the ever growing amount of simulation runs being performed, tools and techniques are needed to store the huge amount of simulation data and to make use of data being stored. While current Simulation Data Management systems allows managing and accessing datasets and would facilitate putting this into action for analysis, the demand on bandwidth and storage increases.
Even with SPDMs, the users usually only had tools and time to make rather straight forward model to model comparisons, between current model versions and their immediate predecessors. To take analysis capabilities and model development a leap forward, it is necessary to also make use of whole model development branches to learn from the gathered simulation information. With the availability of such tools, the value of past simulation data increases.
This gives rise to two challenges. The first challenge is to implement an efficient storage mechanism keeping as many simulation results as possible. This challenge can be met by data compression. It has proven in many application fields that specialized compression tools like FEMZIP [1] for simulation results outperforms general compression tools like Winzip [2].
The second challenge is to introduce a tool capable of analyzing Terabytes of simulation result files and supporting the engineer in his task of creating crashworthiness design.
Model order reduction allows building a database meeting the second challenge. Continuously being fed with new simulation runs, a database makes it in reasonable time possible to automatically detect unknown behavior in the most recent simulation runs compared to all predecessors at a time. To achieve this, the database does not only need to store and detect every new deformation pattern, but in addition needs to handle geometric changes and being able to detect local effects.
Such databases allow to automatically detecting anomalies within the crash deformation behavior, pinpointing exactly to the location in space and time where the model is showing unknown or unwanted deformation patterns. While in daily work the engineer often only has time to compare single simulations with each other, this approach shows how to compare the current simulation with hundreds of predecessors at a time.

2 Benchmark Data set

In this paper, we investigate our strategies on the LS-DYNA™ [3] model of the Chevrolet Silverado modeling a frontal crash. It is provided free of charge by the National Highway Traffic Safety Administration (NHTSA), see [4]. The model consists of 929.181 elements, 682 parts divided into 34 1D elements, 567 2D elements and 81 3D elements. The model was simulated 50 times with sheet thickness variations without geometric changes. 152 time steps with a sampling rate of 1ms were written out. Thus, the temporal resolution is twice as high in comparison to simulation results, which are generated nowadays by automobile manufacturers. On the other hand, the number of elements is smaller by a factor of 5. The uncompressed size is for the array of simulation results is 452.7 GB.

3 Data compression

For our compression approach, we follow three ideas.
The first strategy is motivated by the Rate-Distortion theory [5]. Loosely speaking, it says „The greater the permissible reconstruction error, the higher the compression rate." Since numerically simulated data contains numerical and modelling errors anyways, a lossy compression seems reasonable.
The second idea is to remove as many dependencies in the data as possible by predicting values with already processed values. We perform this step to improve the statistical properties for compression.

The third strategy is to apply an asymmetric compression with spending more time for compression than for decompression. Since the compression can be performed directly on an HPC system with high CPU performance for a simulation that usually need several hours up to days to complete, it is reasonable, if the on-the-fly decompression, e.g. when loading the data set into a postprocessor, is accelerated.
In the following section, we address the main components of our proposed compression strategy. We focus on the lossy compression of numerically simulated floating point data since it is generally dominating the size of simulation results.
The main steps of the proposed compression strategy are:

1. Quantization
2. Prediction
3. Encoding

In the quantization step, we map a floating point number to an integer, considering a user specified precision, the so-called quantum *q*. Hereby, we divide the floating point number *f* by the quantum *q* and round it to the next integer number *i*:

$$i = round\left(\frac{f}{q}\right).$$

This procedure is called absolute quantization [2]. At time of decompression, the original floating point value *f* can be reconstructed by

$$\tilde{f} = i \cdot q.$$

With the quantization, we introduce a loss in precision for the floating point value *f* bounded by the reconstruction error

$$e_f = \left|f - \tilde{f}\right| \leq \frac{q}{2}.$$

After quantization, the resulting integer values can be predicted exploiting the knowledge regarding the finite element model used to simulate the data. The main idea is to eliminate redundancies in time and space and between similar simulation results. This reduces the entropy of data sets which can be utilized by several encoders, e.g. entropy encoders [4].
In the last compression stage, we encode the residuals of prediction. If the data was predicted well, the range of different numbers will be much smaller than the number of all possible values in the 32 Bit range. Therefore, we can provide shorter bit representations for frequently occurring values. This leads to the application of an entropy encoder like Huffman coding or arithmetic coding [6] or specialized mesh-based encoders, like the induced Markov chain encoder [7].

The applications of the compression strategies for the use cases of compression of single simulation results and compression of an array of simulation results follow.

3.1 FEMZIP

Data compression has become an integral part of today's computer-aided engineering (CAE). Already 18 years ago at the 5th European LS-DYNA Conference in Birmingham, the tool FEMZIP-L for the compression of LS-DYNA results was presented [8] and has since established itself as a unique compression tool on the market. The ability to return to the original d3plot output format eliminates a lock-in effect, allowing companies to remove FEMZIP from their processes without leaving any residue.
As part of a new development of FEMZIP-L, the compression and decompression algorithms have been parallelized to achieve significantly faster processing times. For the compression of a model, the runtime could be reduced from about 100 seconds for FEMZIP-L 13.02 with the compression option "-L4" to just under 40 seconds. For loading all post values into GNS Animator [9], the runtime could be accelerated from 143 seconds for FEMZIP-L 13.02 to 131 seconds for FEMZIP-L 15.
The compression efficiency is very high for the Silverado datasets, since post values that are not needed for the analysis are stored very imprecisely. These include the velocities and accelerations as node variables and the shell history element variables. Using the precisions as in [10], a compression

ratio of 63.9 on average over all simulation results can be achieved, so that the data set can be reduced from an initial size of 452.7 GB to only 7.1 GB.

3.2 SDMZIP

In this section we investigate how similarities between simulation models can be exploited to improve compression compared to FEMZIP compression. For this purpose, we consider the Predictive Principal Component Analysis (PPCA) method [10, 11, 12], which is the core of the SDMZIP compression tool [13].

The compression is performed as follows. First, all available simulation models are decomposed into components and the geometry of the models is stored in a database (db). Subsequently, the time-dependent post values for each part (PID) are extracted from all result files and PCA [14] is applied to the respective data matrices. Here, the data matrix is first centered by subtracting the mean value of the row sum from each column. Subsequently, a singular value decomposition (SVD) is applied. The results of the SVD act as a dimension reduction method, where only the leading left-hand singular vectors (principal components) and right-hand singular vectors (coefficients), as well as the corresponding singular values, are further processed. For this purpose, the principal components are stored in the database and the coefficients in a simulation-specific file (ssf). The difference between PPCA and other methods using PCA for compression is that the reconstruction, i.e. the multiplication of the principal components with the singular values and the coefficients is used as a prediction for the data matrix. The error of the prediction is stored in the ssf-file. Thus, a given accuracy is maintained during decompression. The number of singular values and thus the dimension of the reduction is determined by an optimization procedure with regard to the compression rate to be achieved.

If all 50 LS-DYNA results of the Silverado model are already available for the proposed offline step, a compressed size of 1.4 GB for all simulation results can be achieved using the identical precisions as for FEMZIP-L, see Section 3.1. This is a factor of 5.1 more efficient compression than FEMZIP-L and a compression ratio of 326.4 is achieved, which means that the size has been shrunk to significantly less than 0.5% of the original size.

Usually, not all simulation results are available in an SPDM system, but they are calculated and added gradually during the development process. For this purpose, the Online procedure of the PPCA was developed. In this procedure, the data matrices for the post values of one part are predicted with the principal components already in the db file. If the crash behaviour of the newly added result is similar, the prediction error is small. If a new crash behavior occurs or new components are added, this data is written to a database update file (udb). With this procedure, the event space is learned little by little. Therefore, the compression rate for the Online use case, where one simulation result is added after the other, is worse than in the Offline case with a total compression rate of 235.3. However, the achieved compression rate is higher by a factor of 3.7 compared to FEMZIP-L and the compressed size is less than 0.5% of the original size as in the Offline case.

4 Model Order Reduction

With the data compression tools already presented, it is possible to hold a large number of models. But it is necessary to make a profit out of the data treasure. For this purpose we apply model order reduction (MOR) to compare new simulation results not only against single data sets but against a whole development tree. With this strategy, all simulations undertaken remain valuable and engineers benefit all the more from the work of their colleagues.

In the remainder of this section, we describe the implementation of MOR in the tool FEMALYST. First, data sets, should they have geometric changes, are mapped to each other. This achieves that models with geometric adjustments can also be compared. The comparison between the simulation results is then carried out in the FEMALYST Event Detection. For this purpose, individual simulation results are usually added to the database in an initialization phase. All subsequent results are then compared with the database. The comparison is based on the node positions as well as on the post values. Using the FEMALYST Event Score, the behavior in the simulation is evaluated for each component and if it behaves unexpectedly, it is marked as an anomaly.

In the following, we will go into more detail about the process and the techniques used and conclude this section with a short evaluation.

4.1 Geometric Mapping

One of the main challenges in making simulation results comparable over a complete development tree is the handling of geometric changes. We address this challenge with a geometric mapping. The user has the possibility to specify part ID (PID) ranges before starting the FEMALYST analysis. Within the PID ranges, the grid entities are mapped using a nearest neighbor method. For the elements, the topology of the finite element mesh is also taken into account. For strong geometric changes, there is the possibility to influence the mapping in such a way that the number of grid entities that are mapped to an element or node is limited. With these strategies it is possible to make models with geometric changes comparable.

4.2 FEMALYST Event Detection

The techniques being developed focus both on the possibility to automatically spot unknown behaviour in new simulation runs, as well as being able to track known behaviour which engineers flag as being important. It allows detecting new behaviour/events within the current simulation compared to all predecessors and automatically raises this to the engineer.
The basic idea is similar to that of the Online procedure of SDMZIP, see Section 3.2. Using dimensionality reduction, in this case Principal Component Analysis (PCA) [14], a low-dimensional database is built. When a new dataset is added, it is examined how well the new dataset fits into the event space of the already processed datasets. If the data set is similar, a representation is well possible and no event is raised. If the behavior in individual components or subcomponents differs substantially from the previously learned event space, an event is raised and this is communicated to the user. Since event detection runs automatically as a batch process in the workflow, the user does not have to take any action and is warned that the crash behavior has changed even before opening the model in a postprocessor. The newly learned behavior is added to the database and is considered already known for future simulations. Since this can also include an unwanted behavior, the so-called tracked events were introduced, which offer the possibility to warn or inform the engineer when a certain behavior occurs.

4.3 Tracking Events

Another important and useful feature that FEMALYST offers is the possibility to search similar events. Many times there are behaviors (desired/undesired) which the Engineer vividly remembers but cannot recall. In such situations, FEMALYST can act as a “search engine” for searching events of interest in the database. Such a search can reveal the simulations which show a behavior similar to the event searched and could also spot time-shifted behaviors. Event searches are computationally inexpensive and can be performed interactively, thanks to the precise characterization and representation of behavioral patterns in the database.

4.4 FEMALYST Event Score

The before mentioned detection of new events is based on a PCA related outlier score computation. This results in an event score in the range from [0;2] for every part and time-step, defining how much of an unknown behaviour this event represents. This also gives feedback to the engineer right away about how strong the outlier is, depending on the value of the event score. A score below 1 represents known behaviour, while scores above 1 represents unknown behaviour. The higher the FEMALYST score, the stronger the outlier.
With respect to the challenge of analysing crash structures it is mandatory to not only highlight complete parts which are conspicuous, but also the local segment of the part which showed the behaviour, which is why internally parts are being split into fragments, see Fig. 1, to also support a high geometrical resolution and detect local events. Additionally, next to the geometry also all post-values, as e.g. strains and stresses, as well as failure are being analysed.

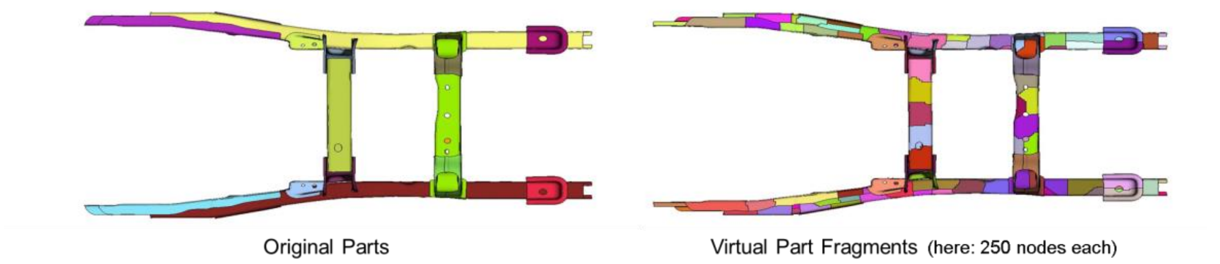

Fig.1: Decomposition of parts to detect local events. Geometry taken from the example case: Chevrolet Silverado [4]

To not only detect single part differences the event computation incorporates an event clustering. Based on the neighbourhood and the propagation of the deformation, parts are clustered together, forming events which also represent the propagation over time and space. Especially with respect to finding possible root causes for the deviation, this allows to get a first idea about the causal chain.

4.5 Evaluation

A detailed robustness analysis [15] [16] of a set of simulation runs from the Chevrolet Silverado [4], based up on thickness variations in the range of [-3;3] % , showed a clear bifurcation. While for some simulation runs the break-booster hooked up to the suspension, being pushed into the firewall, for others there was no such hook-up. Therefore the test scenario shown here consists of a set of simulation runs with similar behavior of no hook-up and a newly analyzed simulation containing the hook-up, see Fig. 2.

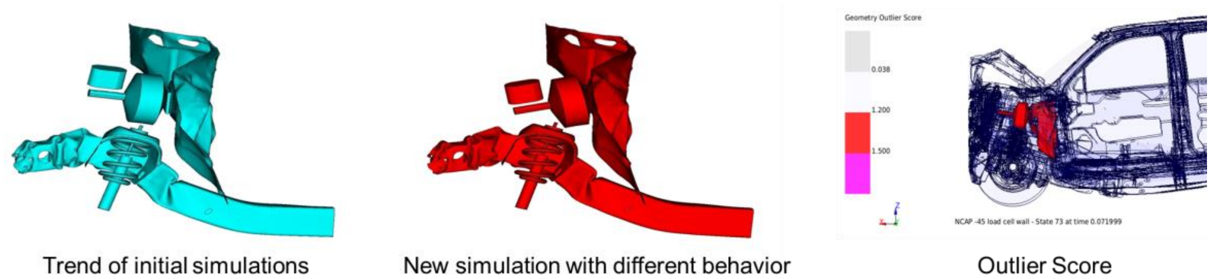

Fig.2: Detection of different deformation patterns in a crash scenario. The outlier score automatically detects and highlights the area with new behavior

5 Summary

SPDM systems make it possible to store simulation results in a structured way and to provide them with meta-information. By using efficient data compression, which focuses on compression efficiency as well as on the speed of decompression, SPDM systems become significantly more powerful, since more simulation results can be stored and processed faster. In addition, there is the possibility to apply novel compression algorithms, which on the one hand promise higher compression algorithms, and on the other hand provide the basis for automated analysis of new simulation results. This automated analysis supports the engineer by marking anomalies and evaluating them with the help of the FEMALYST score. With this, the engineer can already make statements as to whether a crash-relevant adjustment was successful, even before he opens the model in the postprocessor. On the one hand, this accelerates the development and, on the other hand, it relieves the engineer, since the complete model no longer has to be analysed and unexpected side effects are automatically reported.

6 Literature

[1] SIDACT GmbH, FEMZIP, https://www.sidact.de/femzip (accessed 08/23/2023).
[2] Corel Corporation, *Winzip*, https://winzip.com (accessed 08/25/2023).

[3] ANSYS Inc., “ANSYS LS-DYNA”, https://www.ansys.com/de-de/products/structures/ansys-ls-dyna (accessed 08/23/2023).
[4] NHTSA, Chevrolet Silverado, https://www.nhtsa.gov/sites/nhtsa.gov/files/mass-reduction-feasibility-2014-silverado.zip (accessed 08/23/2023).
[5] T. M. Cover and J. A. Thomas, Elements of information theory. Hoboken, N.J.: Wiley-Interscience, 2006.
[6] D. Salomon, Data Compression: The Complete Reference, Springer-Verlag New York, Inc., Secaucus, NJ, 2006.
[7] S. Müller, “Induced markov chains for the encoding of graph-based data”, In Signal Processing and Information Technology (ISSPIT), 2014 IEEE International Symposium, pp. 143–148.
[8] C.-A. Thole, Compression of LS-DYNA Simulation Results using FEMZIP, presented at the 5th European LS-DYNA User Conference, Birmingham, 2005.
[9] GNS, “Animator”, https://www.gns-mbh.com/products/animator4/ (accessed 08/23/2023).
[10] S. Müller, “Compression of an array of similar crash test simulation results“, Logos Verlag, Berlin, 2022.
[11] S. Mertler, S. Müller, C.-A. Thole, “Predictive principal component analysis as a data compression core in a simulation data management system”. In Data Compression Conference (DCC), 2015, pp. 173–182.
[12] S. Mertler, S. Müller, C.-A. Thole, “EP2924585A1 Method for compressing observations of a plurality of test procedures“, EP Patent, 2015.
[13] SIDACT GmbH, SDMZIP, https://www.sidact.de/sdmzip (accessed 08/23/2023).
[14] J. Lee, M. Verleysen, “Nonlinear dimensionality reduction”. Springer, New York, 2007.
[15] D. Borsotto, R. Strickstrock, C. A. Thole, Robustness analysis - Significant reduction of scatter occurrence, presented at NAFEMS Seminar: Optimization and Robust Design, March 23-24, 2015.
[16] SIDACT GmbH, DIFFCRASH, https://www.sidact.de/diffcrash (accessed 08/23/2023).