Key Time Steps Selection for Large-Scale Time-Varying Datasets Using an Information-Theoretic Storyboard*†

Bo Zhou Yi-Jen Chiang

As the size of data generated from simulations and data acquisitions continues to grow exponentially, scientific visualization of time-varying datasets has constantly posted a big challenge. The sheer size of the data often makes the task of interactive exploration impossible, as only a small portion of the data can fit into main memory, and the computation cost is often too high for typical algorithms to run in real-time. To facilitate effective data visualization, it is no longer feasible for the user to perform a trial-and-error process; rather, it is essential to first perform some data analysis to identify salient features, to guide the user through the data exploration process. In this paper, we employ this paradigm to address the issues of large-scale time-varying data visualization, by developing a novel algorithm for key time step selection.

As tens of thousands of time steps become common in time-varying simulations, where tiny changes between consecutive time steps are usually not very informative, selecting a subset of the most salient time steps to visualize becomes crucial in effective and efficient visual analysis of the evolution of the data. However, this crucial task is highly non-trivial, since the evolution of important events can have complex patterns and occur at unknown frequencies across the time steps. In this paper, we develop an *information-theoretic* approach for the evaluation and selection of key time steps from time-varying scalar fields over 3D regular-grid volumetric meshes.

Our approach is based on the following idea: given two selected time steps i and j, if the intermediate time steps that are skipped can be reconstructed from i and j using linear interpolation[1] with a small amount of *information difference* from the original data, then the selected steps i and j can well represent the skipped time steps and thus are "salient" or "representative". Specifically, we apply information theory to quantify the amount of information difference using *variation of information (VI)*, which compares the reconstructed/interpolated time steps against the original data. Then, what if the information difference is too large and i and j are not representative enough? A natural choice would be to add another selected time step between i and j that is most under-represented by this pair, to lower the information difference. In fact, this is the method used in [2] for selecting representative isovalues from scalar fields (albeit under a different measure of representativeness). However, such approach is just a greedy method and can be sub-optimal, since we would never try "backtracked" solutions that do not select i or j. To achieve *globally optimal solutions*, we devise a *dynamic programming* algorithm for extracting the subset of time steps that optimally minimize the information difference.

It is important to point out that our dynamic programming algorithm is *fully automatic* without the need of any user interaction. Moreover, it is performed in the *preprocessing* phase, which is only done *once* with the results stored in a file. For example, we can let it run once overnight without supervision, and the results are ready to be used for user interaction. In the run-time phase, the user can either provide a desired number of time steps k, or give a threshold ϵ in percentage (%) for the information difference[2] and let our algorithm choose a minimum k satisfying ϵ; in either case our algorithm will select the best k time steps very efficiently. To guide the user on what values of k or ϵ to choose, we use a novel chart to present the dynamic programming results to the user, which shows a selection table (for each value of k, what are the best k time steps selected) and a curve of information difference (for each value of k, what is the corresponding amount (in %) of information difference for the selected k time steps); see Fig. 1 for an example. Together, they serve as a storyboard of the data to guide the user through the time-step selection process.

Our run-time algorithm works equally efficient in the out-of-core setting, where there are too many time steps to fit in-core and they must be kept on disk. However, our basic preprocessing algorithm described so far are in-core,

*The full paper has been submitted to *IEEE Transactions on Visualization and Computer Graphics*, August 2016, and is available on-line at `http://cse.poly.edu/chiang/Submitted/TVCG16.pdf`. This work was supported in part by DOE Grant DE-SC0004874.

†The authors are with Department of Computer Science and Engineering, Tandon School of Engineering, New York University, Brooklyn, NY. Email: bz387@nyu.edu; chiang@nyu.edu.

[1] In our scheme other interpolation methods can be used; in this paper we focus on using linear interpolation as a simple and intuitive method.

[2] This percentage is with respect to the maximum total information difference.

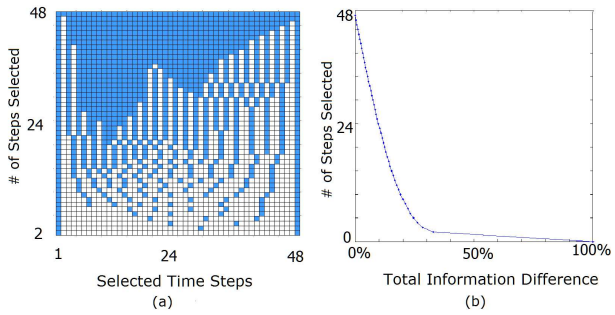

Figure 1: The Results of Isabel-TC Dataset: (a) the Selection Table; (b) the curve of Total Information Difference.

requiring all time steps to fit in main memory at the same time. When there are too many time steps to fit, a straightforward adaptation would need to read, for all pairs of time steps i and j, *every intermediate time steps* of the original data between i and j, for computing the information difference (between the interpolated and original data), which is prohibitively expensive. We extend our basic approach into a novel out-of-core approximate algorithm to achieve *optimal I/O cost* (where the number of I/O operations is no more than reading the whole dataset *twice*), which also significantly reduces the in-core computing time and exhibits a nice trade-off between computing speed and accuracy.

We remark that previously most results for selecting representative time steps are based on local considerations and are suboptimal. In the *dynamic time warping (DTW)* technique [1], a dynamic programming is applied to achieve global optimal when selecting k time steps for a user-specified k. However, the user is not provided with information on how to choose a suitable k. Also, the reconstruction of the skipped time steps is different under the DTW scheme (i.e., if i and j are two consecutive selected time step and r is a skipped step in between, then r is mapped to and represented by either i or j; in other words, r is "reconstructed" as either i or j, rather than linearly interpolated between i and j as in our approach). Moreover, their approach is in-core and does not consider the out-of-core setting.

Contributions: To discuss the computing complexity, suppose the given time-varying dataset has T time steps, where each time step has data size N (i.e., N vertices in the underlying grid mesh) — the total dataset size is TN. We can summarize the contributions of this paper as follows:

(1) We give a fully automatic preprocessing algorithm based on information theory and dynamic programming, to achieve global-optimal solutions for key time steps selection with minimum information difference. The (in-core) computing cost of this preprocessing is $O(T^3N + T^3)$. Note that $O(T^3)$ is for the actual dynamic programming, which is dominated by $O(T^3N)$.

(2) We provide an information-theoretic storyboard of the data to guide the user through the key-time-step selection process in the run-time phase, which can be performed extremely fast (in optimal time $O(\log T + k)$ to report the k time steps if a query threshold ϵ is given, or in optimal time $O(k)$ if a query k is given; in both cases the query takes almost no time).

(3) We extend our preprocessing approach to a novel out-of-core approximate algorithm, using **sliding window** and **multi-pass dynamic programming**, to achieve optimal I/O cost and much reduced in-core computing time with a nice speed-accuracy trade-off. Specifically, the I/O cost is optimal $O(T\frac{N}{B})$ where B is the number of data items fitting in one disk block, and the in-core computing cost is $O(t^2TN+T^3)$ where $t < T$ is the number of time steps to fit in main memory at once (t is a parameter we can adjust, for how much main memory we want to use). Typically we can achieve a good accuracy with a small constant value of t (such as 12). This essentially makes the in-core computing cost to be $O(TN + T^3) = O(TN)$ (i.e., linear in the dataset size TN) when N is asymptotically larger than T^2, which is mostly true in practice.

The experiments demonstrate the efficacy of our new techniques. In particular, the experiments showed that for our out-of-core approximate algorithm, the in-core computing time was much higher than the I/O time. For the former, the dynamic programming time corresponds to the $O(T^3)$ term, which is negligible compared to the $O(t^2TN)$ term. The latter is linear in TN (the dataset size) when t is fixed ($t = 12$); the experiments confirmed that the total time and I/O time were **both roughly linear in the dataset size** TN. More importantly, compared to a direct adaptation of DTW in the out-of-core setting, our out-of-core approximate algorithm achieves **significant run-time advantages** (e.g., 2.1 hours vs. 21.6 hours, and 19.5 hours vs. at least 1024 hours (extrapolated)) while maintaining similar qualities.

References

[1] X. Tong, T.-Y. Lee, and H.-W. Shen. Salient time steps selection from large scale time-varying data sets with dynamic time warping. In *Proc. IEEE Symp. Large Data Analysis and Visualization (LDAV '12)*, pages 49–56, 2012.

[2] T.-H. Wei, T.-Y. Lee, and H.-W. Shen. Evaluating isosurfaces with level-set-based information maps. *Computer Graphics Forum (Special Issue for EuroVis '13)*, 32(3):1–10, 2013.