PIVE: Per-Iteration Visualization Environment for Supporting Real-time Interactions with Computational Methods

Jaegul Choo*, Changhyun Lee†, Hannah Kim*, Hanseung Lee‡, Chandan K. Reddy∨, Barry L. Drake∧, Haesun Park*
*Georgia Institute of Technology, †Google Inc., ‡University of Maryland, ∨Wayne State University, ∧Georgia Tech Research Institute

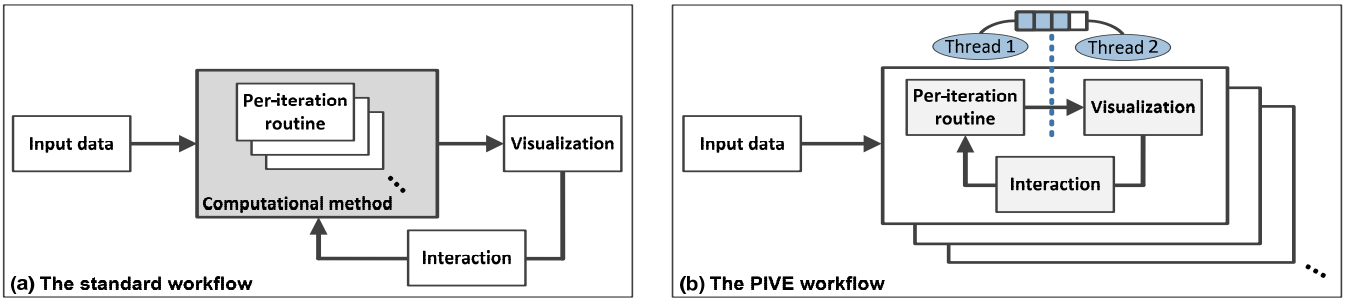


Figure 1: PIVE workflow (in comparison to a standard one). In the standard approach (a), the interactive visualization with a computational method is performed only after algorithm iterations finish. In contrast, PIVE (b) breaks up the computational method into per-iteration routines, allowing real-time interactive visualization with the intermediate output from algorithm iterations. To avoid overhead computations, PIVE divides the entire process into two concurrent threads (blue ellipses) linked with a message queue (blue rectangles).

ABSTRACT

A main bottleneck in integrating computational methods with visual analytics is their significant computational cost, which hinders real-time interactive visualization with them. To solve this, we present PIVE (**P**er-**I**teration **V**isualization **E**nvironment), which visualizes intermediate results from algorithm iterations, thus allowing users to efficiently perform multiple interactions in real time.

Keywords: Real-time interaction, large-scale visualization, multithreading, clustering, dimension reduction, visual analytics.

1 INTRODUCTION

Interacting with computational methods plays an important role in visual analytics [2]. However, their significant computational cost is a serious concern in achieving real-time interactive visualization with them in visual analytics. Previous studies proposed various approaches for this problem, e.g., using sub-samples [1] and multithreading [3, 4]. However, they treated the computational methods as a black box and did not exploit the important characteristics of most computational methods that these methods solve the problem via iterative refinement steps. In response, we propose a novel approach called PIVE (**P**er-**I**teration **V**isualization **E**nvironment). The main idea of PIVE is to visualize the intermediate results from algorithm iterations and to provide efficient user interactions in real time. In the following, we describe PIVE in detail and discuss its issues and solutions. Afterwards, we present several usage scenarios using dimension reduction and clustering methods.

2 PIVE (PER-ITERATION VISUALIZATION ENVIRONMENT)

Overall Workflow. As shown in Fig. 1, the main idea of PIVE is to immediately deliver the intermediate results from algorithm iterations to a visualization module. In this manner, these intermediate results are visualized to users much more quickly, rather than having to wait for the converged solutions. Accordingly, users can efficiently initiate interactions with the computational method, which then affects its subsequent iterations immediately. In this sense, user interactions are performed at an individual iteration level instead of requiring a converged set of multiple iterations, which is the key to real-time interactions.

Interaction Support. Both basic and advanced interactions with computational methods can benefit from PIVE. First, users can dynamically adjust algorithm parameters in real time, e.g., the number of clusters in clustering methods. In addition, users might want to select/filter out data subsets based on their interest, which would accelerate the subsequent computations by reducing the size of the data on which to run the computational methods.

Second, algorithmically sophisticated but semantically meaningful interactions can be smoothly performed under PIVE. In order to support them in real-time via PIVE, we should still avoid significant computations that might be caused by such interactions. Thus, we developed an efficient, widely-applicable interaction methodologies called *soft* and *hard replacements*. The main idea is to replace the (partial) intermediate results with those modified by users during their interactions. Afterwards, when the algorithm continues its iterations, soft replacement allows the algorithm to update such user-modified results while they remain the same during the rest of the iterations in the case of hard replacement. Note that, although not changing, the fixed part of the results still contributes to updating the rest of the results. In this sense, one can view the former as restarting the algorithm with new initializations but the latter as a simple form of constrained algorithms.

3 FURTHER CONSIDERATIONS

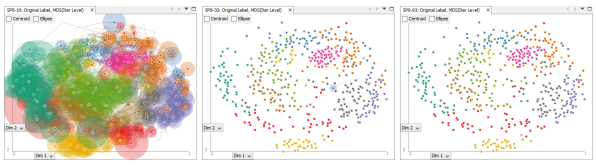
Although PIVE offers a promising framework for real-time visual analytics with computational methods, it poses several challenges.

Visual Stability and Trustworthiness. An obvious challenge with PIVE is the difficulty in deciding when users can initiate interactions during algorithm iterations. One issue is the visual stability due to constantly updated visualization. This may prevent users from analyzing the visualized results and perform interactions consistently. Another issue is whether intermediate outputs are trustworthy (or high quality) enough for users to be willing to spend time on analyzing and interacting with them.

In this respect, it is crucial for users to be able to determine whether an intermediate result is sufficiently stable and close to the final solution. One approach for this is to explicitly provide the stability and trustworthiness information in both a quantitative and a qualitative manners. In terms of quantitative information, the

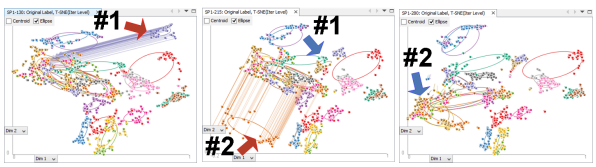
*e-mail: jaegul.choo@cc.gatech.edu

IEEE Symposium on Visual Analytics Science and Technology 2014
November 9-14, Paris, France
978-1-4799-6227-3/14/$31.00 ©2014 IEEE



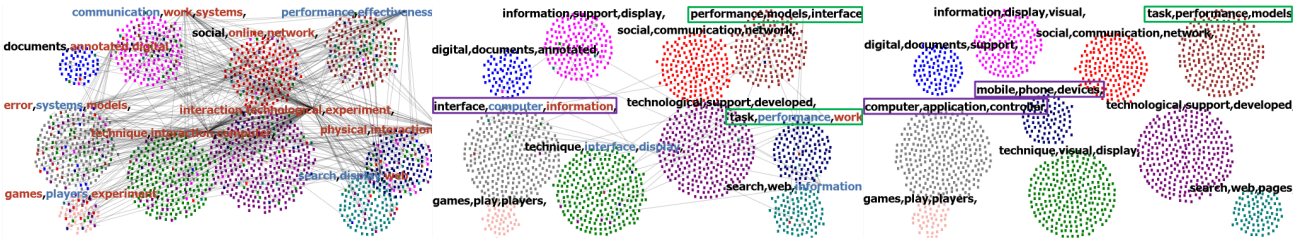
(a) Iteration 10 (b) Iteration 30 (c) Iteration 93 (final)

Figure 2: The visualization snapshot of MDS at different iterations. 500 handwritten pendigit data represented in 16 dimensions have been used.



(a) Iteration 130 (b) Iteration 215 (c) Iteration 280

Figure 3: The PIVE integration of t-SNE. Red (and blue) arrows point to interactions (and results). 1,558 spoken letter data represented in 618 dimensions have been used.



(a) Iteration 1 (b) Iteration 6 (c) Iteration 26 (final)

Figure 4: The splitting/merging interactions of the *k*-means algorithm. At the sixth iteration (b), we perform cluster merging (green rectangles) and splitting (a purple rectangle) interactions, and the final result due to these interactions is shown in (c). The CHI conference papers published between 1999 and 2010 have been used.

system can show the trend of various evaluation measures over iterations along with the dynamically updated visualization snapshots. Such information may also be visually encoded (in a qualitative manner) into the original visualization itself. In the following section, we will describe the details of our approach for a particular set of computational methods we chose.

Computational Overhead. In PIVE, additional computations are incurred since it has to process and visualize the intermediate outputs repetitively. To handle this, we adopt a multi-threaded approach. That is, we separate out the processing and visualization module of intermediate outputs into another thread (the blue ellipse labeled as Thread 2 in Fig. 1(b)). Since modern commodity computers are usually equipped with a multi-core CPU, these two threads can be executed virtually in parallel without any performance loss compared to the standard (non-PIVE) approach.

4 User Interactions under PIVE

Dimension Reduction.[1] Fig. 2 shows the visualization outputs of multidimensional scaling (MDS) under PIVE. In each snapshot, lines and circles represent the trajectories and the movement amount of data points, respectively, for the last five iterations. At an early iteration (Fig. 2(a)), one can see that the visualization is not stable yet, but at the 30th iteration (Fig. 2(b)), it becomes stable, which is virtually the same as the converged result at the 93rd iteration (Fig. 2(c)). This shows that, instead of performing a fairly large number of iterations until the MDS algorithm completely converges, PIVE quickly provides sufficiently good visualization.

Now we present the user interaction scenarios for spoken letter data (26 clusters corresponding to different letters) using t-distributed stochastic neighbor embedding (t-SNE). Given an initial visualization (Fig. 3(a)) showing many overlapping clusters, we performed a hard replacement interaction by moving the cluster 'q' (the red arrow labeled as '#1') to separate it out of the overlapping clusters. As a result, t-SNE separates out another cluster 'u' (the blue arrow labeled as '#1'), revealing its close similarity to the cluster 'q'. In fact, this makes sense in that these two letters sound similar, compared to the rest of the overlapping clusters, 'b', 'd', 'e', 'g', 't', and 'p', which sound different. Next, we perform another interaction by moving the cluster 'p' (the red arrow labeled as '#2'). Correspondingly, the yellow- (g) and brown-colored (t) clusters are pulled from the overlapping clusters towards this cluster (the blue arrow labeled as '#2').

Clustering. We customized the *k*-means clustering algorithm in Jigsaw visual analytics system under PIVE (Fig. 4). For the visual stability issue, we represent document cluster membership changes as grey lines. Out of the entire set of 26 iterations, the visualization at the 6th iteration already shows sufficiently stable result. At this point, we merge multiple small or semantically related clusters (green rectangles) and split large or unclear clusters (a purple rectangle) in Fig. 4(b). Consequently, the subsequent iterations show the clusters are properly merged and the splitting interaction produces a new cluster labeled as 'mobile, phone, device' (Figs. 4(c)), which would not have been found without the interaction.

5 Conclusion

We presented a novel framework that offers immediate visualization during algorithm iterations and allows real-time interactions with computational methods in visual analytics. In the future, we plan to extend PIVE to support other types of semantically meaningful interactions.

Acknowledgments. The work was supported in part by NSF grants CCF-0808863, IIS-1242304, and IIS-1231742, NIH grant R21CA175974, and DARPA XDATA grant FA8750-12-2-0309.

References

[1] D. Fisher, I. Popov, S. Drucker, and m. schraefel. Trust me, I'm partially right: incremental visualization lets analysts explore large datasets faster. In *CHI*, pages 1673–1682, 2012.

[2] J. D. Mulder, J. J. van Wijk, and R. van Liere. A survey of computational steering environments. *Future Generation Computer Systems*, 15(1):119 – 129, 1999.

[3] H. Piringer, C. Tominski, P. Muigg, and W. Berger. A multi-threading architecture to support interactive visual exploration. *IEEE TVCG*, 15(6):1113–1120, 2009.

[4] H. Yu, C. Wang, R. Grout, J. Chen, and K.-L. Ma. In situ visualization for large-scale combustion simulations. *IEEE CG&A*, 30(3):45–57, 2010.

[1] A demo video: `http://tinyurl.com/pive2014`.