TGE-viz : Mixed Initiative Plan Visualization

Sriram Gopalakrishnan, Subbarao Kambhampati
Computing, Informatics, and Decision Systems Engineering
Arizona State University
699 S Mill Ave, Tempe, AZ 85281

Abstract

Existing work for plan trace visualization in automated planning uses pipeline-style visualizations, similar to plans in Gantt charts. Such visualization do not capture the domain structure or dependencies between the various fluents and actions. Additionally, plan traces in such visualizations cannot be easily altered without tracking the dependencies across actions, and that imposes a higher cognitive load. We introduce TGE-viz, a technique to visualize plan traces using an embedding of all the fluents and actions in a domain in low dimensional space. TGE-viz allows users in mixed-initiative planning to visualize and criticize plans more intuitively. It also allows users to visually extract the structure of domains and the dependencies in plans.

Introduction and Motivation

One of the barriers to the adoption of automated planners is their usability. This is due to the amount of time and knowledge needed to interpret any output, and interact with the planner. An area in which we can improve usability is plan trace and domain visualization. Current plan trace visualizations such as SPIFe (Clement et al. 2010), Fresco (Chakraborti et al. 2017), Conductor (Bryce et al. 2017), and Webplanner (Magnaguagno et al. 2017) represent plans in a pipeline or linear sequence as seen in Figure 1 If a plan has no complete ordering of actions, adjacent actions may have no immediate relationship or dependencies. So with existing visualizations, the user would have to remember the effects of actions, and connect it with future action(s) to realize the need or relationships with the prior action. Consequently, the user may have to painstakingly parse and remember dependencies across the entire plan, before beginning to think about other possible plans. We think that high cognitive load can lead to mental fatigue in the user, which reduces the quality of plan criticism in mixed-initiative planning. Thus, it is important to present information in a visual and easy-to-parse (and recall) format. This would allow users to quickly conceptualize about alternate plans, modify existing ones or compare plans. Indeed, the Ecological Interface Design Principles (Vicente and Rasmussen 1992), which helped set the standards for design in complex human-machine systems, require that the correct affordances (actions) are easily inferable to the operator. To this end, we introduce TGE-viz, a visualization approach that uses graph vertex embeddings to display all the fluents and actions of a domain in 2 dimensions. The embeddings are learned using the relationships between the fluents and actions. Embedding graphs in lower dimensional spaces are a popular method to organize and visualize the vast amount of information in graphs (Cai, Zheng, and Chang 2018), and we apply it to planning. Lower dimensional embeddings allow humans to be involved in the analysis. We can see structures and relative distances, which can be used to augment any automated analysis. It is this human insight that we hope to bring into mixed-initiative planning. We present some of the technical details involved in learning the embeddings, followed by the features and functionality of our user interface. For more details on the embeddings, experiments, and analysis, please refer to our full paper at the link in the footnote. [1]

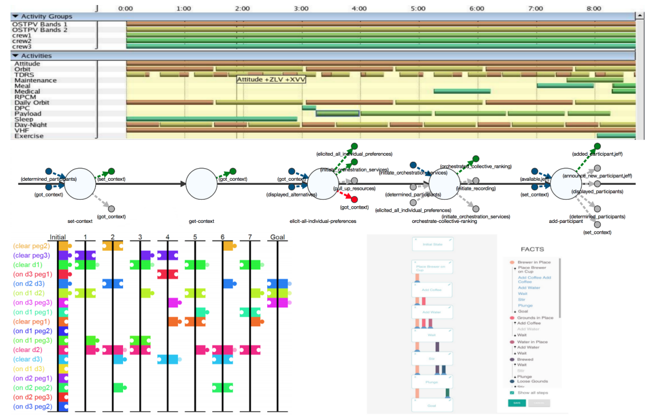

Figure 1: Existing Visualization For Planners. Clockwise: SPIFe, Fresco, Conductor, WEBPLANNER

Technical Details

For demonstrating TGE-viz we chose to modify the standard IPC logistics domain to allow for different types of dependency structures in the domain. The only change needed was that airplanes cannot fly to any city, but to specific cities en-

[1] https://arxiv.org/abs/1811.09900

forced through a ”flies_to” proposition. This allows for diverse plans and structures in the domain. The first step is to embed the grounded actions and fluents in a 2-d space for visualization. We implemented a variant of Force (spring) embedders (Fruchterman and Reingold 1991). For embedding algorithm details and pseudocode, please see our full paper at footnote 1. To visualize how the embeddings update occurs, see the start of the video of our demo at the link in the second footnote.[2] Once the embeddings have been determined, TGE-viz then plots them in a 2-d space using PyGame (Shinners 2011). TGE-viz is implemented in python, and uses PyGame for the user interface. In the background, TGE-viz calls Fast-downward (Helmert 2006) to do the planning when instructed to do so by the user. We now present the details and functionality available in TGE-viz.

Functionality and Features

In addition to the embeddings data, TGE-viz also takes in a problem and domain file. The problem file helps color the fluents that are true at the start state in red. All other possible fluents (from the embeddings data) are $blue$, and all actions are colored $green$. The colors help separate the different components of the planning problem, and makes it easier to extract information. This is in-line with the ”Part-Whole” paradigm of representations from the Ecological Interface Design (Vicente and Rasmussen 1992). EID recommends that a good UI allows the display of the different parts of the system separately. TGE-viz allows the user to see either only the fluents, or fluents and actions by toggling the $DisplayActions$ button. Since a majority of the nodes are actions, it can be hard to separate them from fluents. Additionally, if the user would only like to focus on planning in a particular area of the domain, we let the user zoom in and out (Figure 3) using the scroll wheel of the mouse. Another important criterion emphasized in EID is the ease of knowledge acquisition. When user hovers over a node we display the textual information describing it in the top left. If it is an action, we display the precondtions, as well as the positive and negative effects. We also draw a red line to fluents that are consumed (deleted), a black line to fluents that are generated (positive effect), and a yellow line to fluents that are just preconditions. Since the embeddings were learned based on cooccurring fluents and actions, the related fluents and actions are grouped together. This makes clearly separated groups of embeddings with semantic meaning to the clustering. For example, in our logistics domain, all the fluents and actions related to transportation within a city are in a group, and those actions for transportation between two specific cities are closer together and in between the clusters of the two cities. When users wants to generate a plan, they can direct the Fast-downward planner by clicking the appropriate goal fluent. A plan is generated and displayed both on the user interface and in the terminal as a list of numbered actions. The numbers line up between the terminal display and the user interface display, and represent the order of actions. Additionally, we draw the red,black, and yellow lines for each action in the plan to show the associated fluents. Such a plan trace display helps the user visualize the trajectories and dependencies in the plan as illustrated in Figure 2. This is much harder in other pipeline visualization methods as the user would have to remember and track effects, and dependencies. In TGE-viz, one can also easily see alternative plans, as illustrated in Figure 2. Finally, the user can extend a plan by clicking other goal fluents, or restart from the initial state by clicking $Restart\_Planning$. This allows the user to direct the planner through specific parts of the problem space, by an of ordering sub-goals.

[2]Video of Full Demo - https://youtu.be/48au9idWZEA

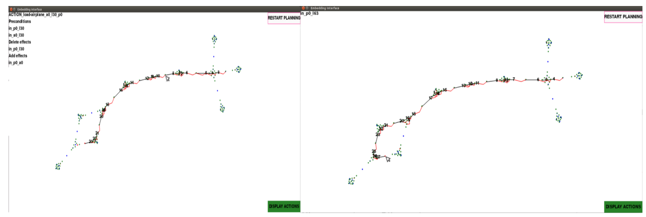

Figure 2: Visualization of two alternate plans for delivering a package to the same location

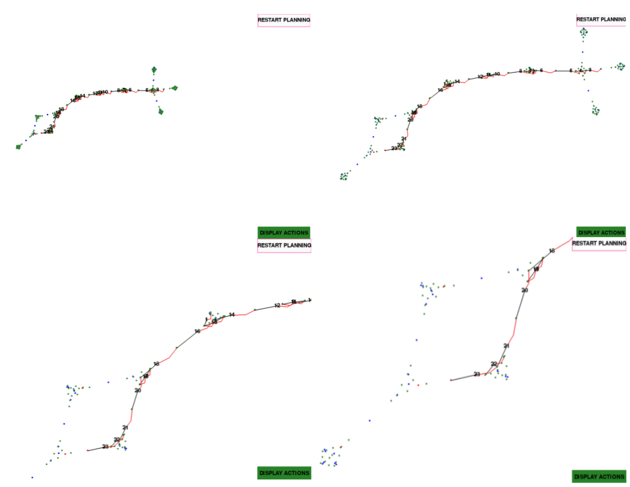

Figure 3: Magnification in TGE-viz interface

Summary

TGE-viz allows visualization of plan traces in the context of all the fluents and actions using embeddings that have a semantic clustering of fluents and actions. In our demo (video in footnote 2), we will first display the animation of how the embeddings are learned. Then, we show how information can be extracted quickly by hovering over nodes to display details. We also demo the ability to turn actions on/off and zoom to reduce the density of nodes. After this, we will demonstrate setting goal fluents by simply clicking them, and how the resultant plan is displayed. Finally, we demo extending a plan by selecting more goals. Then we open it up to the audience to play with the software. We hope to get feedback from the ICAPS community on usability, cognitive load, and software enhancements.

Acknowledgements

This research is supported in part by the ONR grants N00014-16-1-2892, N00014-18-1-2442, N00014-18-1-2840, the AFOSR grant FA9550-18-1-0067, and the NASA grant NNX17AD06G.

References

Bryce, D.; Bonasso, P.; Adil, K.; Bell, S.; and Kortenkamp, D. 2017. In-situ domain modeling with fact routes. In *Proceedings of the Workshop on User Interfaces and Scheduling and Planning, UISP*, 15–22.

Cai, H.; Zheng, V. W.; and Chang, K. 2018. A comprehensive survey of graph embedding: problems, techniques and applications. *IEEE Transactions on Knowledge and Data Engineering*.

Chakraborti, T.; Fadnis, K. P.; Talamadupula, K.; Dholakia, M.; Srivastava, B.; Kephart, J. O.; and Bellamy, R. K. 2017. Visualizations for an explainable planning agent. *arXiv preprint arXiv:1709.04517*.

Clement, B. J.; Barreiro, J.; Iatauro, M. J.; Knight, R. L.; and Frank, J. D. 2010. Spatial planning for international space station crew operations. In *Proceedings of the International Symposium on Artificial Intelligence, Robotics and Automation in Space*. Citeseer.

Fruchterman, T. M., and Reingold, E. M. 1991. Graph drawing by force-directed placement. *Software: Practice and experience* 21(11):1129–1164.

Helmert, M. 2006. The fast downward planning system. *Journal of Artificial Intelligence Research* 26:191–246.

Magnaguagno, M. C.; Pereira, R. F.; Móre, M. D.; and Meneguzzi, F. 2017. Web planner: A tool to develop classical planning domains and visualize heuristic state-space search. In *Proceedings of the Workshop on User Interfaces and Scheduling and Planning, UISP*, 32–38.

Shinners, P. 2011. Pygame. `http://pygame.org/`.

Vicente, K. J., and Rasmussen, J. 1992. Ecological interface design: Theoretical foundations. *IEEE Transactions on systems, man, and cybernetics* 22(4):589–606.