# – Introduction

## Motivation

## Goals

## Research Questions

## Contributions

This paper introduces new perspectives on software engineering learning, leveraging the current state of the art, based on game, to a level where the game provenance can produce and consolidate knowledge. This knowledge can help on (1) confirming the hypotheses formulated by students, (2) supporting tutors for a better guidance, (3) motivating group dynamics around some case studies, and (4) extracting behavior patterns from individual sessions or groups of sessions.

The provenance visualization can occur both on-the-fly or in post-mortem sessions. It allows the discovery of issues that contributed to specific game flows and results achieved throughout the gaming session. This analysis can be used on games to improve understanding of the game flow and identifying actions that influenced the outcome, aiding the player to understand why they happened the way they did. It can also be used to analyze a game story development, how it was generated, and which events affected it.

Currently, we do not make inferences to the user, but let the user decide what he wants to infer. Studies in this area can be made in order to identify information that can be omitted from the user without affecting the overall analysis. Another interesting research is to automatically identify patterns in the game flow. Lastly, we are working on different graph visualization layouts and running experimental studies on the usage of provenance in educational games to evaluate the aspects of learnability.

## Organization

# – Conclusion

## Contributions

This paper introduces new perspectives on gameplay modeling and analysis, leveraging the current state of the art, based on gameplay, to a level where the game provenance can aid the detection of gameplay issues. This knowledge can help on (1) confirming the hypotheses formulated by the beta tester, (2) supporting developers for a better gameplay design, (3) identifying issues not reported by testers, and (4) data-mining behavior patterns from individual sessions or groups of sessions.

The provenance visualization can happen on either on-the-fly or post-mortem sessions. It allows the discovery of issues that contributed to specific game flows and results achieved throughout the gaming session. This analysis can be used on games to improve understanding of the game flow and identifying actions that influenced the outcome, aiding developers to understand why events happened the way they did. It can also be used to analyze a game story development, how it was generated, and which events affected it.

Currently, we do not make inferences to the user, but let the user or developers decide what needs to be inferred. However, we provide the necessary tools to create inference rules, like filters and collapses (both for vertices and edges). Studies in this area can be made in order to identify information that can be omitted from the user without affecting the overall analysis. Another interesting research is to automatically identify patterns in the game flow. Lastly, we are working on different graph visualization layouts and also studying the possibility of using game provenance in educational digital games to aid in the understanding of the concepts taught in the game.

## Limitations

## Future Work

### ACCESSIBILITY

In relation to accessibility, we propose to make *Prov Viewer* less context sensitive, allowing the user to customize filters (edge filters and attribute status visualization) without the need of hard coding it in the application. For example, allowing the user to provide a configuration file that specifies the type of each filter. Thus, this would make *Prov Viewer* compatible with other games or provenance applications without the need of tinkering with the source code.

Another change would be related to *Prov Viewer’s* input file format. Currently, the *game flow log* is a simple tab separated value text file. However, there are plans to modify the structure to use some semi-structured format such as JSON or XML for greater compatibility with other applications. Thus, it would allow *Prov Viewer* to be more accessible by other applications due to the usage of semi-structured format.

One hypothesis for improving accessibility for *Prov Viewer* would be in creating an extension for existing game engines (i.e. Unity3D). This extension, when enabled, would automatically capture gameplay information when executing events, thus generating a *game flow log* without the need of changing the game’s source code.

### Inferences

One of the current draw-backs of *Prov Viewer* is related to inferences. Currently, we do not make inferences to the user, but let the user or developers decide what needs to be inferred. However, we provide the necessary tools to create inference rules, like filters and collapses (both for vertices and edges). Studies in this area can be made in order to identify information that can be omitted from the user without affecting the overall analysis, while at the same time not being context dependable. Thus it would provide generic inferences rules that can be used in any game.

### Scalability

Another area to be worked upon is related to the graph scalability. Depending on the game style, a game session might take several hours to complete, or even days in case of RPGs. This makes the size of the provenance graph to be overwhelming to the user, even when doing a pre-filtering during the generation of the *game flow log*. One way to avoid such situations is to show the provenance graph with some filters selected instead of its full extension. For example, before showing the graph to the user, it is possible to use collapses to reduce the graph’s size. Combats can be identified and collapsed into a single vertex for each instance. Places visited in the game can also be collapsed into a single vertex, containing all interactions made in that location, even combats. It is also possible to have collapses inside collapses. In this case, a collapsed combat inside a collapsed area visited by the player may contain other actions aside from the combat, such as interactions with the ambient. This gives an impression of a map from the player’s journey, showing vertices for each location visited by the player, while allowing the player to expand only the situations he desires to analyze. It is similar to *google maps*, where it shows the entire world and allows the user to zoom into specific locations. However in this case, it shows instances of the journey taken by the player.

It is also possible to go beyond that. Instead of collapsing all combats and locations, filters can be used to decide which combats or locations were not relevant to the story, or had no noticeable impact in the player’s journey, while keeping important events visible to the player. This is possible because provenance is analyzed from the present to the past. This way, combats outcomes are known and can be used to decide if it they are relevant or not. If the player was victorious with minor challenge, did not suffer severe wounds, or barely used any resources at his disposal, then the entire combat can be simplified into just one vertex representing the combat with the enemy. However, if the combat was challenging or the player lost, it is interesting to display all actions in it for analysis, allowing the player to identify important facts that influenced the combat outcome.

Another problem related to scalability involves performance. Algorithms in *Prov Viewer* are not implemented optimized for computing a graph with thousands vertices and edges.