Using Edge Bundle Views for Clone Visualization

Benedikt Hauptmann, Veronika Bauer, Maximilian Junker
Technische Universität München
Garching b. München, Germany
{*hauptmab, bauerv, junkerm*}*@in.tum.de*

***Abstract*—Clone detection results are often voluminous and difficult to present. Most clone presentations focus on the quantitative clone results but do not relate them to the structure of the analyzed system. This makes it difficult to interpret the results and draw conclusions. We suggest using edge bundle views to interrelate the system's structure with the clone detection results. Using this technique, it is easier to interpret the clone results and direct further analysis effort.**

***Keywords*-Clone detection, clone presentation, edge bundle views**

I. Introduction

Clone results for real systems are often overwhelming and difficult to interpret. They often consist of detailed lists of clones and clone classes, as well as a few aggregated metrics (e.g., the overall clone coverage or the amount of redundancy free code) or visualizations. However, these types of presentation don't provide sufficient information to interpret the data. Especially at the beginning of a clone assessment, it is important to know which other parts of the system a certain source file shares clones with.

For example, cloning is often used as an informal reuse strategy [1]. To uncover this practice, quantitative clone results are not sufficient. By relating the structure of the code in the clone interpretation directly, identifying cohering clones is easier. Furthermore, it helps to uncover misconfigurations of the clone detection. Having the knowledge that some clone classes are, for example, locally restricted makes it easier to identify generated code which can then be excluded from the analysis. Additionally, during inspections of industrial software systems, we noticed that in some cases the code base accidentally contained the same files in different, mostly outdated, versions. Knowing at first glance that some system parts are strongly cloned with others helps to exclude these special cases code from the analysis.

However, most clone presentations focus just on the actual clone results. In this paper, we suggest to use edge bundle views to include the structure of the analyzed system into the presentation of clone detection results.

II. Visualizations of Clone Detection Results

Several visualization concepts exist to illustrate the results of clone detections. Rieger et al. [2] propose multiple polymetric views for clone visualization: The *Duplication Web* represents the strength of clone relationships between files. It arranges them on a circle, but does not capture any system structure at all.

Clone Scatterplots, *Duplication Aggregation Tree Maps*, and *Clone Class Family Enumerations* show further detailed cloning information, such as proportions of internal vs. external cloning on a class level. *Dotplots* [3] mark cloned code between classes as a dot in the diagram. This representation provides a very detailed insight, however, it quickly becomes overwhelming for large systems. *Hasse Diagrams* [4] visualize only the cloned parts of a system, arranging them in a partial order. None of these visualizations, however, relates the cloning information to the system structure.

Other approaches visualize clone information by projecting it on the system decomposition in terms of components or directory structures. *Tree-maps* [5] show the measured metric values over the entire system decomposition. The system's structure is flattened on a rectangular area, preserving the hierarchy. However, tree-maps do not show any information about the system elements participating in the clone relationships.

System Model Views [2] order source files according to their directory structure, focusing on the physical location of files and duplicates. They show the amount of cloned code between different files or system parts, but due to its forest structure, it does not scale well for large systems.

Bundle Edge Views [6] overcome these deficits: they are structured according to the system decomposition and they display the cloning relationships between the different parts of the systems. They also alleviate the problem of visualizing deeply nested system structures. To reduce the number of edges accross the circle, edges with similar orgins and destinations are bundled together.

III. Preliminary Experiment

We created a prototypical implementation to visualize clone dependencies as edge bundle view on top of the open source quality assessment toolkit ConQAT[1] [7]. Figure 1 shows an example of a bundle edge view we created for a real system. The structural decomposition of the analyzed system, in this example the structure of the folders in which the source files are stored, is visualized as a multilayered circle. Starting from the outmost layer, every inner segment of

[1] www.conqat.org

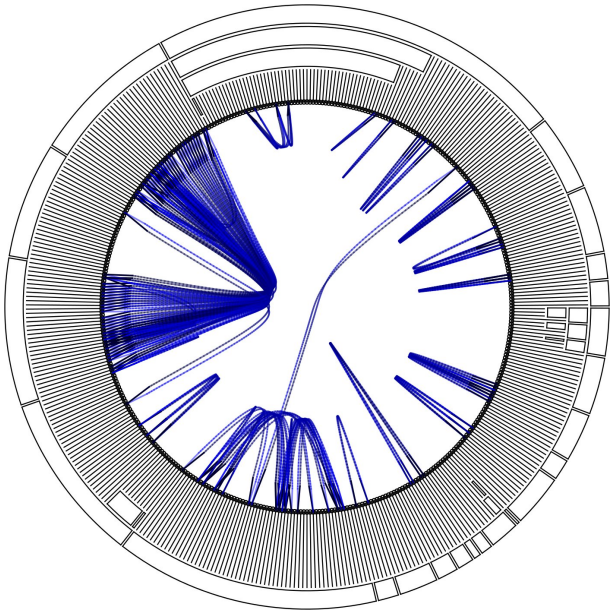


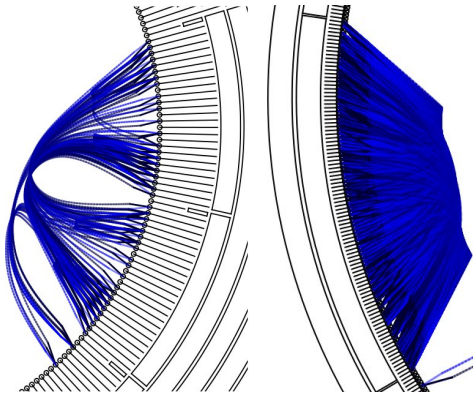
Figure 1: Edge bundle view visualizing clone relationships.

the circle represents one underlying directory of the source tree. Source files are represented by pins connected with segments of the innermost circle layer. An edge between two files exists if there is at least one clone class in which the corresponding files are involved in. To provide additional orientation while browsing the visualization interactively, we display a tool tip showing detailed clone information while hovering over the pins with the cursor. We performed clone detections and created edge bundle views for open source projects and for industrial projects from previous studies [8]. So far we made the following experiences:

Duplicated Source Code: In figure 1, there is a considerable amount of clone relations between files of two source folders having the same size. Manual inspections revealed that, during the data preparation, a part of the source code had accidently been duplicated. The edge bundle view uncovered the data corruption by integrating cloning information with the system structure.

Cloning as Reuse: In figure 2a, there are also two directories of similar size and structure with many clone relations between them. In manual inspections, we found out that the affected parts were not identical. We found a lot of gapped clones as well as some files which were not affected by cloning at all. It turned out that the code was duplicated on purpose as a way of code reuse. In this case, the information provided by the edge bundle view helped us to focus further manual inspections.

Generated Code: In figure 2b, there is a noticeable amount of clone relations within files in one particular directory. During manual inspections, we found out that this directory contained generated code with stereotypical source code.



(a) A cloned component (b) Generated code

Figure 2: Examples of findings.

Since we performed the clone detection to uncover potential maintenance problems, we excluded the generated code from the analysis.

So far, edge bundle views provided beneficial indications to uncover errors in the setup of the clone detection, helped to make conclusions about the clone results and to focus further manual inspections. The edge bundle views we are generating so far are quite rudimentary. Therefore, in future work, we plan to improve the quality of the presentation and the expressiveness of the visualization.

Acknowledgments

We thank Elmar Juergens for providing the necessary clone detection data and Oleksandr Panchenko for suggesting to use this technique to visualize code clones.

References

[1] C. Kapser and M. W. Godfrey, ""cloning considered harmful" considered harmful," in *WCRE*, 2006.

[2] M. Rieger, S. Ducasse, and M. Lanza, "Insights into system–wide code duplication," in *WCRE*, 2004.

[3] K. W. Church and J. I. Helfman, "Dotplot: a program for exploring self-similarity in millions of lines of text and code," in *INTERFACE*, 1992.

[4] J. H. Johnson, "Visualizing textual redundancy in legacy source," in *CASCON*, 1994.

[5] B. Johnson and B. Shneiderman, "Tree-maps: A space-filling approach to the visualization of hierarchical information structures," in *Visualization*, 1991.

[6] D. Holten, "Hierarchical edge bundles: visualization of adjacency relations in hierarchical data." *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, 2006.

[7] E. Juergens, F. Deissenboeck, and B. Hummel, "Clonedetective - a workbench for clone detection research," in *ICSE*, 2009.

[8] E. Juergens, F. Deissenboeck, B. Hummel, and S. Wagner, "Do code clones matter?" in *ICSE*, 2009.