Clone Detection for Max/MSP Patch Libraries

Nicolas Gold[1], Jens Krinke[1], Mark Harman[1], David Binkley[2]

[1]King's College London, CREST
{nicolas.gold, jens.krinke, mark.harman}@kcl.ac.uk

[2]Computer Science Department, Loyola University
binkley@cs.loyola.edu

Finding content-based similarities in Max/MSP [1] patches may have a number of useful applications e.g. supporting patch construction and composition or the navigation of large patch collections. We present our current work on detecting similarities in patches written in Max/MSP. The technique we propose is based on clone detection, a well-known approach in software engineering to finding similar and identical pieces of source code within large software systems [2-4]. Dataflow languages such as Max/MSP present particular challenges to existing clone detection technology because of the absence of explicit control flow in the source language. Recent work has addressed this problem for Simulink models [5, 6] but these algorithms are unsuitable for application directly to Max/MSP. Control flow in Max/MSP is dependent on the spatial relationships of the objects used in a patch, thus graph isomorphic approaches such as that of Pham et al. [5] cannot be applied.

Our algorithm has three steps. First, patches (in JSON format) are pre-processed to extract information about the Max/MSP objects therein. Sub-patches are not recursively parsed in our current implementation. Second, clone candidates are generated by following paths from the first patch-line found to all linearly-reachable boxes from that path. Patch-lines in cyclic sub-patches are considered only once (i.e. cycles are not permitted in clone candidates). Once the pool of clone candidates has been generated the final stage can take place. The third and final step is clone detection where every member of the pool is compared to every other same-sized member of the pool (excluding itself and any path that overlaps itself). Clones are classified according to a clone classification scheme we have previously presented [7].

The algorithm was implemented and executed on the set of Max patches supplied as part of the Max/MSP 5 distribution. This is a corpus of 43 patches of varying complexity and purpose. In total there are 1881 top-level boxes (i.e. not including nested patches) and 1790 individual lines in the corpus. The investigation found 349 DF0 clone pairs (identical fragments), 1151 DF1 clone pairs (near-identical fragments where allowing non-semantics-affecting layout variation and comment changes are permitted), and 4451 DF2 clone pairs (near-identical fragments where non-semantics-affecting variation in layout, comments, and literal values are permitted). The proportion of unique code elements in cloning relationships varied between 16% (DF0 lines) to nearly 70% (DF2 boxes). The largest found clone-pair involved seven boxes.

Future work will include improving the efficiency of the algorithm, implementing clone-pair merging to form larger candidate fragments from linear clone-pairs, implementing recursive parsing to extract information from nested sub-patches, developing better user interface support for navigating clone-pairs, and evaluation on a wider range of patches.

References

1 http://www.cycling74.com

2 Krinke, J.: 'Identifying Similar Code with Program Dependence Graphs'. Proc. Eighth Working Conference on Reverse Engineering, Stuttgart, Germany, 2001

3 Jia, Y., Binkley, D., Harman, M., Krinke, J., and Matsushita, M.: 'KClone: A Proposed Approach to Fast Precise Clone Detection'. Proc. 3rd International Workshop on Software Clones, Kaiserslautern, Germany, 24th March 2009.

4 Stefan, B., Koschke, R., Antoniol, G., Krinke, J., and Merlo, E.: 'Comparison and Evaluation of Clone Detection Tools', IEEE Trans. Softw. Eng., 2007, 33, (9), pp. 577-591.

5 Pham, N.M., Nguyen, H.A., Nguyen, T.T., Al-Kofahi, J.M., and Nguyen, T.N.: 'Complete and Accurate Clone Detection in Graph-based Models'. Proc. IEEE International Conference on Software Engineering, Vancouver, Canada, May 2009.

6 Deissenboeck, F., Hummel, B., Juergens, E., Schatz, B., Wagner, S., Girard, J.-F., and Teuchert, S.: 'Clone detection in automotive model based development'. Proc. Proceedings of the 30th International Conference on Software Engineering, Leipzig, Germany, 2008.

7 N.E. Gold, J. Krinke, M. Harman, D. Binkley: "Issues in Clone Classification for Dataflow Languages," Proceedings of the 4th International Workshop on Software Clones, Cape Town, SA, 2010.