Incremental Clone Detection

Exposé

Nils Göde

University of Bremen

2008-04-11

1 Tasks

The task is to extend the existing clone detection tool *clones* in order to analyze two revisions of a given software. Information about what happens with the clones from one revision to the other is to be extracted. The tasks in detail are as follows:

1. Implement an algorithm, that takes results from the analysis of one revision and uses these results to analyze the other revision. Additional input to the algorithm is information about which files changed between the revisions.
2. Interpret the changes made to the tokens and the suffix tree in order to answer the question which clones are
   - *ADDED* (in the new, but not in the old revision)
   - *REMOVED* (in the old, but not in the new revision)
   - *MOVED* (in both revisions, but changed location)
   - *UNCHANGED* (identical in both revisions)

   Optional: Provide information about what happened to the clones in more detail.

2 Procedure

Task 1 requires the adaption of the *clones* implementation in Ada. Having available the tokens of the old revision and the files which are changed, the new token-stream does not need to be built from scratch. Instead, only the files which did change are newly scanned and the old token-stream updated at the respective positions in order to retrieve the token-stream for the new revision.

Knowing which parts changed in the token-stream, the old suffix tree can be updated according to the algorithm (or a modified version) presented in [1].

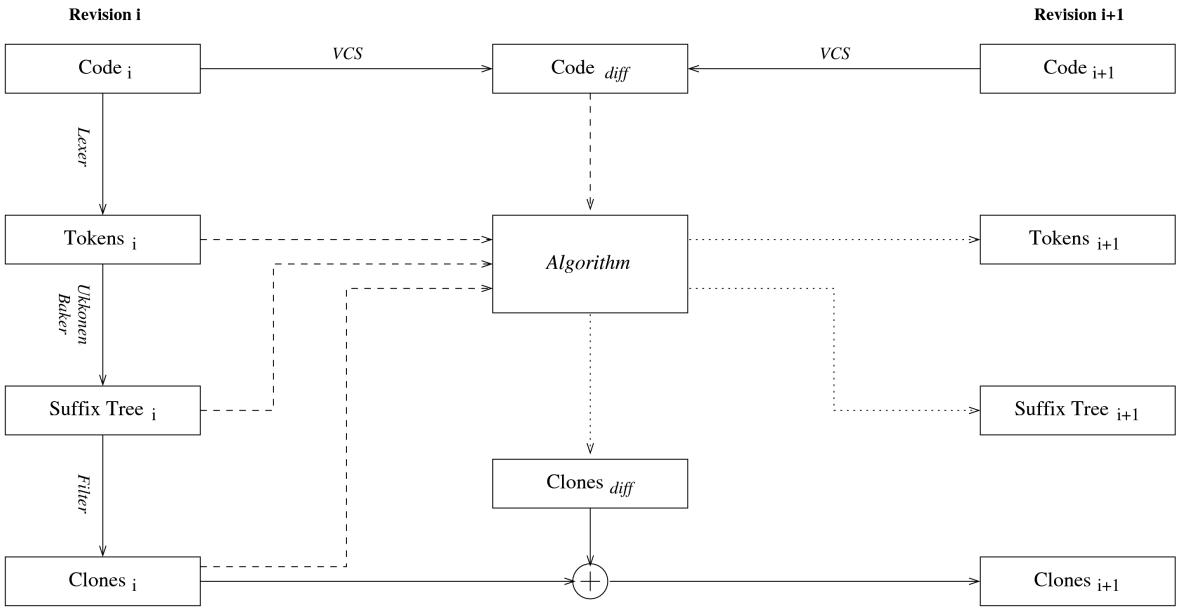

Figure 1: The diagram shows how the algorithm integrates into the analysis of two revisions of a system. Dashed lines indicate input to the algorithm, dotted lines its output.

The retrieval of the information to answer task 2 needs to be incorporated into the modification of the suffix tree. As soon as the tree is modified, the clones represented by the modified parts need to be updated.

3 Expected results and evaluation

The following things are expected after fulfilling the tasks.

- The modification of the old token-stream is faster than building the new token-stream from scratch. This can be measured by comparing the new algorithm to individual runs of the usual clone detection. Attention needs to be payed to the choice of parameters to make results comparable.
- The modification of the old suffix tree has a similar time consumption compared to building the new token-stream from scratch.
- Concerning task 2, information about clones is assumed to be complete and correct.

4 Milestones

The following milestones have been identified for the diploma thesis.

1. *End of planning stage.* **2008-04-18**
   (a) Preliminary title set.
   (b) Contents set.

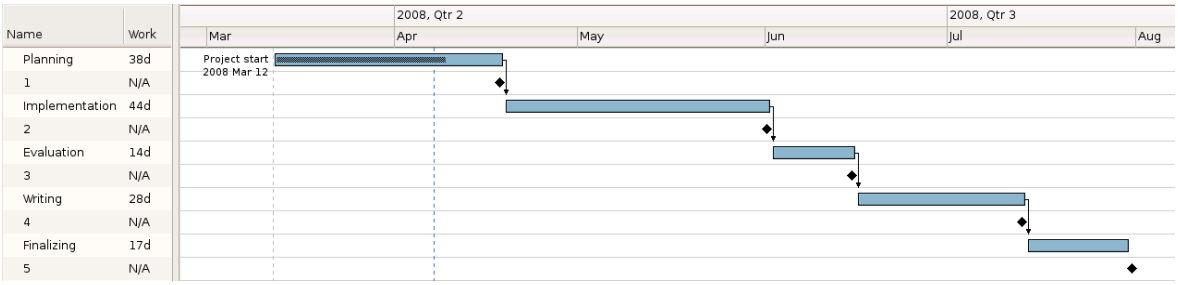

Figure 2: Gantt chart which shows the different phases and milestones.

(c) Exposé done.
(d) Feasibility checked.
(e) Registration of thesis done.

2. *End of implementation stage* **2008-06-1**
   (a) *diff* input conversion implemented.
   (b) Token-stream modification implemented.
   (c) Suffix tree modification implemented.
   (d) Dumping of results implemented.
   (e) Integration into clones completed.
3. *End of evaluation stage* **2008-06-15**
   (a) Evaulation of task 1 done.
   (b) Evaulation of task 2 done.
4. *End of writing stage* **2008-07-13**
   (a) Content completely written down.
   (b) Read by someone else.
5. *Thesis completion* **2008-07-31**
   (a) Spell/grammar-checked.
   (b) Printed.
   (c) Thesis handed in.

5 Risks

The following problems may occur.

- *Insufficient knowledge of the Ada programming language.* Though sophisticated programming skills are present, the somewhat atypical way of formulating things in Ada might be a problem. This does affect the time needed to implement the algorithm.

- *Feasibility of the algorithm.* Despite careful preparation, a small detail might make the algorithm unusable or not even implementable.
- *Unacceptable time and space consumption of the algorithm.* As the data which is processed is quite huge, little variations in time and/or space consumption of a single step might have an undesirable effect on the whole algorithm.

References

[1] Manuela Montangero Paolo Ferragina, Roberto Grossi. A note on updating suffix tree labels, 1997.