Intelligent Code Merge using Abstract Trees

William Cao, Jared Le

```
$ git status
# On branch cse-403-18sp
# You have unmerged paths
#   (fix conflicts and run git commit)
```

Line-based Conflict Merge is Dumb

Advantages of line based:

- Simple
- Easy to implement

Disadvantages:

- Requires user input to do things that should be **TRIVIAL**

Original

```
1  void function(int x) {
2    for(int i = 0; i < x; i++) {
3      doOtherThing(i);
4    }
5  }
6  void otherFunction() {
7    // TODO: implement
8  }
```

You

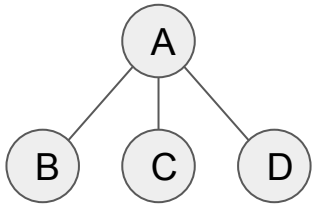
```
1  void function(int x) {
2    // deprecated
3  }
4  void otherFunction() {
5    for(int i = 0; i < x; i++) {
6      doOtherThing(i);
7    }
8  }
```

Your coworker

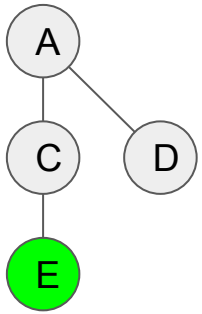
```
1  void function(int x) {
2    for(int i = 0; i < x; i++) {
3      if (x <= 0) return;
4      doOtherThing(i);
5    }
6  }
7  void otherFunction() {
8    // TODO: implement
9  }
```

Tree-based Conflict Merge is Smart!

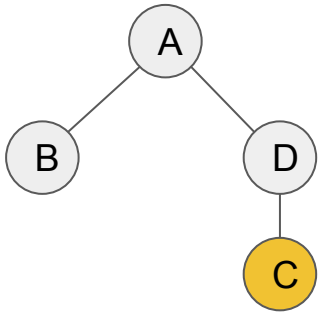
Code is just syntax trees, so use these to identify similar statements across files



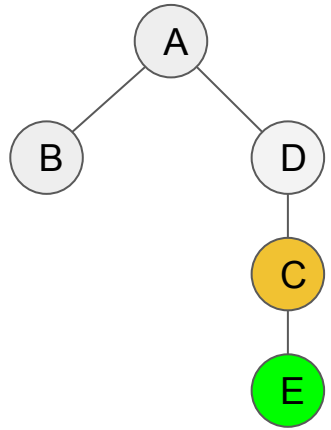
Base



Rev. A



Rev. B



Final

A: Function header
B: If statement
C: Procedure Call

D: For Loop
E: Assignment

Considerations, Limitations, Previous Work

- Must be able to decompose the language into grammatical parts (**BNF!**)
- Extra care must be taken to preserve custom formatting, deal with non-syntactic code

Potential Improvements: use “wildcard” expression type to deal with non-syntactic code, add support for other languages (python, ruby, etc.)

Possible Metric: number of merge conflicts compared to normal merge

Further Reading:

Precise Version Control of Trees with Line-based Version Control Systems, Dimitar Asenov, Balz Guenat, Peter Müller, and Martin Otth