This paper is a very broad “systems” style paper. We will try to zero in on the parts that are relevant to our work.

(1) We have approached a few systems using an analysis technique that breaks things down into language, denotational semantics, and algebraic laws. To what parts of today’s paper can this analytical technique be applied, and what does the analysis look like?
If you find gaps in the presentation, identify them, and fill them in as best you can.

(2) Words like “merge”, “merge conflict”, and “commutativity” are not featured concepts in this paper. But I believe these problems are there under the surface. What can you identify that is related to merging, merge conflicts, and commutativity? What are the relationships?

(3) I first heard about this work at a talk given in 2009, at the same meeting that took me to Annapolis this year. At that time I thought that the work would be better described as “syntactic patching.” Maybe I should reconsider. What information or techniques do you find in the paper that would justify the name “semantic patch?”

(4) The project described in this paper aims to ease the development of device drivers for Linux, a system whose source code is maintained using distributed source-code control. How do you think the authors imagine that their tool fits into the Linux ecosystem in general and the source-code control system in particular? How do you imagine that the tool fits in?

(5) Some people are really interested in a richer model of patches that goes beyond simple line-based patching. What lessons should these people learn from this paper?