1 Method

The syntax of most programming languages is context-free. Our proposed method is simple. We construct a context-free grammar representing the intersection between the langauge syntax and an automaton recognizing the Levenshtein ball of a given radius. Since CFLs are closed under intersection with regular languages, this is admissible. Three outcomes are possible:

1. $\mathcal{G}_\cap$ is empty, in which case there is no repair within the given radius. In this case, we simply increase the radius and try again.
2. $\mathcal{L}(\mathcal{G}_\cap)$ is small, in which case we simply enumerate all possible repairs. Enumeration is tractable for $\sim 80\%$ of the Python dataset in ≤ 90s.
3. $\mathcal{L}(\mathcal{G}_\cap)$ is too large to enumerate, so we sample from the intersection grammar $\mathcal{G}_\cap$. Sampling is necessary for $\sim 20\%$ of the Python dataset.

When ambiguous, we use an n-gram model to rank and return the top-k results by likelihood. This procedure is depicted in the flowchart below:

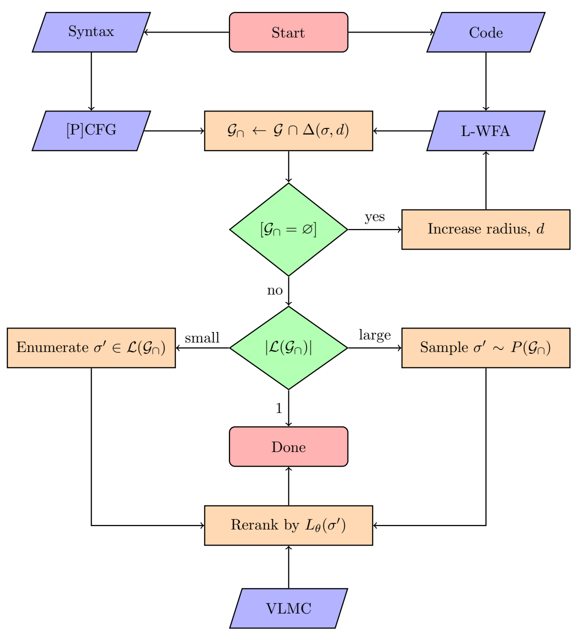

Figure 1: Flowchart of our proposed method.