Poetry finds two ways into computer science, according to Manurung: in Artificial Intelligence, ever since Deep Blue conquered chess, “more artistic-based tasks --- are often proposed to be the defining benchmark”; and in Natural Language Generation, we want flexibility to generate “fluent and natural-sounding texts” or texts “where the communicative goal is vague.”

It is a sizable task to specify a poetry generation process and what parts might be aided by computation. First, of the many features of poetry, such as sounds and themes, we choose to delve into ambiguity. Traditionally, this seems like a separation point between computer science and literature, as programming languages have aimed for deterministic interpretations. However, higher level languages (e.g. Python) and commercial AI aim more and more to guess user intentions, while probabilistic algorithms and data-dependent programs give a numerical perspective on ambiguity. Second, of the many available tools and approaches, we focus on the linguistic formalism of Tree Adjoining Grammar, which gives us a rich structural representation of sentences.

In particular, we employ a neural-network parser for Tree Adjoining Grammar to detect structural ambiguity. We present a method and tools to split a sentence in multiple ways into well-formed phrases (as judged by our parser). In future work, we aim to expand our tools to illustrate these ambiguous readings, and convert our detection process into a generative process.

---