Speak Well or Be Still: Solving Conversational AI with Weighted Attribute Grammars

(Extended Abstract)

Vadim Zaytsev

Formal Methods and Tools, University of Twente, The Netherlands

There is a growing need to specify models of possible conversations with non-human entities. Such models have a difficult task to set the bar for correctness yet tolerate conversations with only partial conformance to it, and accommodate computations that non-human entities perform during the conversation on several possibly independent emergent models with unrelated flows of information in them. As it turns out, this is possible to specify formally in a fairly concise way with weighted attribute grammars [6]. In this paper, a variant of those is presented, called **WAGIoT**, that combines the power of analytic, generative, attribute, weighted and probabilistic grammars in one DSML.

Conversation entities are an essential part of smart IoT systems. They come in many forms, largely synonymous: conversational AI, virtual digital assistants, interactive agents, smart bots, chatbots, etc. In the presence of the Turing test [17] as the ultimate goal of artificial intelligence, conversation programs became an iconic example of an AI system very early. Notable milestones shaping and eventually commoditising the trend, were ELIZA [19], PARRY [4], A.L.I.C.E. [18], Wolfram Alpha [20], IBM Watson [9], Siri [3], Cortana [12], Alexa [2], Google Assistant [7], Alisa [22] and Bixby [15]. Conversation agents are needed in many areas from smart homes to game design. We refer to the excellent recent overview of this research direction by Adamopoulou and Moussiades [1] and focus now on the relation between the linguistic component and the operational logic of the conversation entity.

Looking at the problem linguistically, the conversation can be encoded as an automaton with states representing internal states of the conversation component, and transitions annotated with inputs (coming from the user or an edge device) and outputs (being sent to the user or to actuator). In the computation theory such automata are called Mealy machines [11] if they are deterministic and have a finite number of states. Both limitations are unfortunately too crippling, so all the substantial body of research on Mealy machines cannot be applied directly. What might be theoretically more feasible, is an input/output extension of pushdown transition systems or process rewrite systems [10], that can handle both infinite/uncountable number of states or transitions, have enough memory to handle complex tasks intelligently, and still represent a strict subclass of Turing machines such that reachability is decidable.

The lack of available theories pushed people to consider hybrid setups. For instance, actual derivations can be handled by a grammar, but the grammar rules that get applied, must follow an associated coloured graph [21] or a path in a Petri net [5]. The contemporary systems used in the industry, like RiveScript [13], are also hybrid in nature: the conversation is specified as request-response pairs (akin to the event-based paradigm in grammarware [23]) with ad hoc computations on global data. Powerful off-the-shelf AI packages like Dialogflow [8], lifting natural language understanding tasks to intents and their fulfillment, and using a smart knowledge connector to incorporate existing data, allow these computations to be arbitrarily complex. For example, Salvi et al build Jamura [14], a conversational smart home assistant using Telegram API for collecting user input, Dialoglflow Agents for processing it and the ThingSpeak platform to connect to the server and the clients.

One of the heterogeneous approaches is binary context-free grammars [16] which combine two generative grammars into one mathematical object. Essentially our proposal is to combine attribute grammars (that can propagate and share data in a very controlled fashion), weighted grammars (that provide highly controllable nondeterminism), analytic grammars (for parsing user inputs) and generative grammars (for producing answers).

Our domain-specific language for conversation models is called WAGIoT, and is in fact a tailored implementation of Weighted Attribute Grammars [6]. It allows conversation designers to specify interactions with actuators, sensors and users. A typical WAGIoT model can be seen on Figure 1. There are many model transformations that infer enough information from this provided model in order to make it formally executable. For instance:

- ⋄ a synthesized attribute $\underset{\sim}{n}$ in `getname` and an inherited attribute $\dot{n}$ in `time` have the same name; the given grammar is represented as a directed graph of nonterminals, in which WAGIoT normaliser finds the shortest path between these two places and makes sure the information is properly propagated;
- ⋄ analytic nonterminals such as `greet` and `stop` have multiple branches, which get assigned counters; these are increased each time a branch is chosen, and this information is always available elsewhere—wherever there is a need to build a model of the human user's behaviour/speech patterns;
- ⋄ all the unassigned weights are calculated based on available information; for instance, `time` branch in rule 10 gets a probability $[1 : 1 + w]$ because w is a dynamic value, so the only option is to add the default 1; if rule 11 had the weight $[1 : 3]$ instead of $[w]$, then rule 10 would get a weight $[2 : 3]$ to compensate for the missing static part;
- ⋄ types of attributes are inferred: n is determined to be a string due to a built-in nonterminal `Id`, and w is an integer because it is used as weight;
- ⋄ code blocks between ⟦ and ⟧ are expected to conform to a special interface with methods responding to their use in generative and analytic contexts, and otherwise are lowered to strings, which is what happens in rules 10–11.

Our preliminary experiments show that these transformations hide details that otherwise overwhelm those who are supposed to write such models. Although we claim that the simplicity of RiveScript [13] pushes its users to combine it with informal hacks, exposing too much of the complexity at once is just as damaging. More work is required to find the best balance, as well as to pursue the obvious enhancements like replacing fixed terminals with signals from a NLP model. Our ongoing endeavours are made public at https://github.com/grammarware/wagl.

$$
\begin{array}{rcll}
S & \leftarrow & \texttt{setup activity}^{*}\ \texttt{stop.} & (1) \\
\texttt{setup} & \leftarrow & \texttt{greet?\,getname.} & (2) \\
\texttt{greet} & \leftarrow & \texttt{'hello'.} & (3) \\
\texttt{greet} & \leftarrow & \texttt{'good morning'.} & (4) \\
\texttt{greet} & \rightarrow & \texttt{'greetings, human!'\ 'what is your name?'.} & (5) \\
\texttt{getname} & \leftarrow & \texttt{'I am'}\ n := \mathsf{Id}. & (6) \\
\texttt{getname} & \rightarrow & \texttt{'nice to meet you,'}\ \dot{n}. & (7) \\
\texttt{activity} & \leftarrow & \texttt{time}\,\big|\cdots. & (8) \\
\texttt{time} & \leftarrow & \texttt{'what time is it?'.} & (9) \\
\texttt{time} & \rightarrow & \texttt{'it is'}\ [\![DateTime.Now]\!]\ \texttt{','}\ \dot{n}\ \blacktriangleright w := 5. & (10) \\
\texttt{time} & \rightarrow & [w]\texttt{'it is'}\ [\![DateTime.Now.Hour]\!]\ \texttt{'o'clock'}\ \blacktriangleright w := w - 1. & (11) \\
\texttt{stop} & \leftarrow & \texttt{'stop'}\,|\,\texttt{'off'.} & (12)
\end{array}
$$

Figure 1: A simplified WAGIoT grammar for a holistic example showcasing the notation. All ←-rules are analytic, all →-rules are generative. Having multiple analytic rules means that any of them can be applied—as in (3–4). Having multiple generative rules means one of them is chosen at random with probabilities following rule weights—as in (10–11). Rule (1) contains a regular right part (a Kleene star). Rule (12) contains a shorthand notation for multiple rules of the same kind for the same nonterminal (a Backus bar). The ellipsis in rule (8) designates that this is but a fragment of a larger grammar. := in rules (6) and (10–11) denotes attribute assignment. $\dot{x}$ are inherited attributes (propagated downwards in the tree); x are synthesized attributes (propagated upwards). Weight is printed in square brackets.

References

[1] E. Adamopoulou, L. Moussiades, An Overview of Chatbot Technology, in: AIAI, Springer, 2020, pp. 373–383.

[2] Amazon, Amazon Alexa Voice AI, https://developer.amazon.com/en-US/alexa, 2014.

[3] Apple, Siri, https://www.apple.com/siri/, 2011.

[4] K. M. Colby, S. Weber, F. D. Hilf, Artificial Paranoia, Artificial Intelligence 2 (1971) 1–25.

[5] J. Dassow, S. Turaev, k-Petri Net Controlled Grammars, in: LATA, LNCS 5196, Springer, 2008, pp. 209–220.

[6] M. Gerhold, V. Zaytsev, Towards Weighted Attribute Grammars, Under review, 2022.

[7] Google, Google Assistant, your own personal Google, https://assistant.google.com, 2016.

[8] Google, Google Cloud: Dialogflow, https://cloud.google.com/dialogflow/docs/, 2017.

[9] IBM, IBM Watson, https://www.ibm.com/watson, 2011.

[10] R. Mayr, Process Rewrite Systems, Information and Computation 156 (2000) 264–286.

[11] G. H. Mealy, A Method for Synthesizing Sequential Circuits, Bell System Tech Journal 34 (1955) 1045–1079.

[12] Microsoft, Your Personal Productivity Assistant, https://www.microsoft.com/en-us/cortana, 2014.

[13] N. Petherbridge, RiveScript, https://www.rivescript.com, 2018.

[14] S. Salvi, Geetha V, Sowmya Kamath S, Jamura: A Conversational Smart Home Assistant Built on Telegram and Google Dialogflow, in: TENCON, IEEE, 2019, pp. 1564–1571.

[15] Samsung, Bixby | Apps & Services, http://bixby.samsung.com/, 2017.

[16] S. Turaev, R. Abdulghafor, A. A. Alwan, A. A. Almisreb, Y. Gulzar, Binary Context-Free Grammars (2020).

[17] A. M. Turing, Computing Machinery and Intelligence, Mind 59 (1950) 433–460.

[18] R. S. Wallace, The Anatomy of A.L.I.C.E., in: Parsing the Turing Test, Springer, 2009, pp. 181–210.

[19] J. Weizenbaum, ELIZA — A Computer Program for the Study of Natural Language Communication between Man and Machine, CACM 9 (1966) 36–45. doi:10.1145/365153.365168.

[20] Wolfram Research, Wolfram|Alpha: Computational Intelligence, https://www.wolframalpha.com, 2009.

[21] D. Wood, A Note on Bicolored Digraph Grammar Systems, IJCM 3 (1973) 301–308.

[22] Yandex, Alisa — Voice Assistant from the Yandex company, https://yandex.ru/alice, 2017.

[23] V. Zaytsev, Event-Based Parsing, in: REBLS, 2019. doi:10.1145/3358503.3361275.