Experimental AI in Games: Papers from the AIIDE Workshop
AAAI Technical Report WS-16-22

Juke Joint: Characters Who Are Moved By Music

James Ryan, Tyler Brothers, Michael Mateas, and Noah Wardrip-Fruin
Expressive Intelligence Studio
University of California, Santa Cruz
{jor, michaelm, nwf}@soe.ucsc.edu, tgbrothe@ucsc.edu

Abstract

We present *Juke Joint*, a small work of interactive storytelling that demonstrates an extension to the *Talk of the Town* framework by which characters form thoughts, expressed in natural language, that are elicited by environmental stimuli. *Juke Joint* takes place in a procedurally generated American small town, in a bar with a haunted jukebox and two patrons facing personal dilemmas; the player is a ghost whose only action is to select which song from the jukebox will play. As the lyrics of the song emanate from the machine, thoughts are elicited in the minds of the patrons, constituting streams of consciousness that may eventually lead them to resolutions of their respective dilemmas. In this paper, we outline the game and also the AI architecture that makes it possible; the latter combines a light simulation of stimulus processing with a novel approach to natural language generation.

Introduction

So much of human phenomenology is rooted in the quiet streams of consciousness that guide us toward the decisions we make in life, and a familiar and effective storytelling device puts character thoughts into the purview of the audience. Curiously, not many AI systems have explored the generation of character streams of consciousness, though a few have. DAYDREAMER is an early system that generates daydreams by reflecting on past experiences encoded in a knowledgebase (Mueller 1990). More recently, GRIOT uses conceptual blending to generate poetry that proceeds from evocative user input (Harrell 2005). Follow-up work extended the system to produce daydream-like output that generates character thoughts, outlined in the third person, that are also guided by user input (Zhu and Harrell 2008). V-Daydreamer has similar aims to these systems, but produces visual output (Pérez y Pérez, Sosa, and Lemaitre 2007).

In this paper, we present our own exploration of the generation of character thoughts, though as a response to (and a way of processing) environmental stimuli, rather than user inputs. Specifically, our contribution centers on *Juke Joint*, a small work of interactive storytelling that demonstrates an extension to the *Talk of the Town* AI framework (Ryan et al. 2015) that enables characters to now form thoughts that are expressed in generated natural language.

Related Work

At a high level, our project is about the autonomous processing of ongoing subjective experience. Very many earlier projects have explored agent real-time perception (often visual processing) in real or simulated environments (Horswill 1993; Scheier and Lambrinos 1996; Ferber 1999; Weyns, Steegmans, and Holvoet 2004), including perception systems for NPCs in combat and stealth games (Leonard 2003; Diller et al. 2004; Orkin 2006; Isla 2013). More relatedly, projects on autonomous characters—from *Tale-Spin* (Meehan 1977) to the Oz Project (Bates, Loyall, and Reilly 1992) to *Façade* (Mateas and Stern 2003)—have modeled realtime perception for expressive purposes. The Oz Project work of Loyall and Bates (1997) is especially related, since it too integrates agent perception and natural language generation (NLG) technologies. While these projects aim to generate optimal or believable agent behavior in particular environmental contexts, in *Juke Joint* we aim to model how an agent's subjective *inner world* may shift subtly as a largely unconscious response to evocative stimuli, like song lyrics.

Given this difference, let us turn then to the expressive modeling of agent subjectivity. While elsewhere we have outlined the literature on subjective belief (Ryan et al. 2015), here we will consider systems that have modeled memory, thought, and other subjective phenomena. A classical system in this area is CYRUS, which models agent reconstructive memory (Kolodner 1983). Later systems include the aforementioned ones that model daydreaming, which *Juke Joint* departs from primarily by *situating* its daydreamers, both socially (they have carried out simulated lives) and environmentally (they think as a way of processing a modeled environment). Also worthy of mention here is the *autobiographic agents* project, where agents dynamically reconstruct themselves in response to ongoing subjective experience, as a way of indexing memories of that experience (Nehaniv and Dautenhahn 1998; Ho, Dautenhahn, and Nehaniv 2008). Relative to these agents, ours are more socially situated and more characterized, and we focus additionally on NLG. A more recent related project is Ian Horswill's *MKULTRA*, an experimental game built around the mechanic of *belief injection*: players solve puzzles by

forcing thoughts into the minds of NPCs (Horswill 2014; 2015). While *MKULTRA* and *Juke Joint* are both examples of AI-based game design (Eladhari et al. 2011; Cook et al. 2015), they appear to utilize different design patterns (Treanor et al. 2015): both use the *AI is Visualized* pattern, but belief injection in *MKULTRA* matches *AI is Guided* (though differently than in other games), whereas *Juke Joint*, with its spare player input, relies more on *AI as Spectacle*. This comparison also holds for the LabLabLab trilogy, whose conversational puzzles are solved by using unconstrained natural language to shift NPC hidden states (Lessard 2016). Finally, another quite related project is *The Ice-Bound Concordance* (Reed et al. 2014), which also recombines content that was annotated with metadata capturing evocative symbolic concerns (Garbe et al. 2014). Again, a comparison can be made in terms of Treanor et al. (2015) design patterns, with their project opting for the *AI as Co-creator* pattern.

For a deep comparison of our method for NLG to earlier approaches, see our recent publications that provide more detail and context on that aspect of the project (Ryan et al. 2016; Ryan, Mateas, and Wardrip-Fruin 2016a).

Juke Joint

Juke Joint is set in the fall of 1987, in a bar in a procedurally generated American small town. There is a jukebox there, and two patrons who are each facing personal dilemmas; the player's only action is to select which song from the jukebox will play. As the lyrics of the selected song emanate from the machine, thoughts are elicited in the minds of the characters, constituting streams of consciousness that may eventually lead them to resolutions of their respective dilemmas. The tone of the game is light and playful. While we are currently developing a graphical version that is playable in a browser, *Juke Joint* is at the time of writing a text-based game that is executed by a Python interpreter. In this section, we broadly outline the experience.

Generating a Town

Prior to gameplay, the *Talk of the Town* world generation procedure is enacted to produce the American small town in which gameplay will take place. Specifically, the town is simulated from its founding in 1839, when a handful of families converge on an empty townscape to establish farms, up to the fall of 1987, when *Juke Joint* takes place. While this procedure has been used in *Talk of the Town* and *Bad News* (Samuel et al. 2016) to produce towns brimming with hundreds of socially situated characters, here we employ it to gain access, as authors, to the raw material of *two* character's simulated lives—their personal histories, social networks, family situations, work situations, love lives, beliefs, and more. As we explain later, our process of generating a thought for a character deeply relies on the various social contexts of that person's life in the simulation. Due to space, we will refrain from further explanation of this procedure, but the interested reader may consult another publication that describes it in more depth (Ryan et al. 2015), as well as one that explains the generation of character social networks in particular (Ryan, Mateas, and Wardrip-Fruin 2016b).

Setting the Scene

Once a town has been generated, we select its oldest bar as the scene in which gameplay will take place. From here, we must simply ensure that two characters are in the bar, since that is the cast size that we target.[1] Currently, we do not intelligently pick which characters will be at the bar, but it would be possible to select ones whose life contexts best match the concerns captured in our pool of generable thought content.

A Ghost in the Machine

The player is a ghost who haunts the jukebox, selecting a single song that the machine will appear to play on its own; the apparition can also read characters' minds. This framing is hinted at in a brief expository scene that plays out at the beginning of gameplay.

Character Dilemmas

The characters in the bar have dilemmas that dominate their streams of consciousness—critically, these are deliberated over and eventually resolved in ways that are guided by the lyrics of the song that the player selects to play. Initially, we planned to support a wide array of character dilemmas that pertain to activities that are actually simulated in the *Talk of the Town* framework—considerations of marriage proposal, divorce, leaving a job, hiring an employee, and more—but due to authorial burden (in specifying generable thoughts, which we explain below) we settled on one: whether the character will depart the town that gameplay takes place in. Because there is only one possible character dilemma, both characters in the bar are deliberating over the same personal conflict. While we would prefer to support a number of possible dilemmas in the future, we found that this actually showcased our AI architecture quite nicely. While both characters are respectively grappling with the same issue, they tend to deliberate quite differently (because what thoughts can be generated for a character depends on various contexts in her life) and often come to opposite resolutions (or the same ones, but for different reasons).

Songs and Themes

Jukeboxes currently feature the same three songs across all playthroughs: "Coal Miner's Daughter", by Loretta Lynn (1969); "The River", by Bruce Springsteen (1981); and "Never Gonna Give You Up", by Rick Astley (1987). These songs were chosen for not being anachronistic (they were all released prior to the fall of 1987) and for having themes that could serve as authorial scaffolding for specifying a generator of character thoughts—issues of family, economic struggle, and love could believably guide deliberation over whether to leave a town for somewhere else.

Songs are treated as data, specifically, collections of *stanzas*[2] that were each annotated by us for their respective lyri-

[1] We decided on this number through playtesting. With only a single character, the breadth of generable thought content was not properly showcased; with more than two characters, it became hard to track all the unraveling streams of consciousness.

[2] More precisely, any number of contiguous lines that we as authors decided to cluster together.

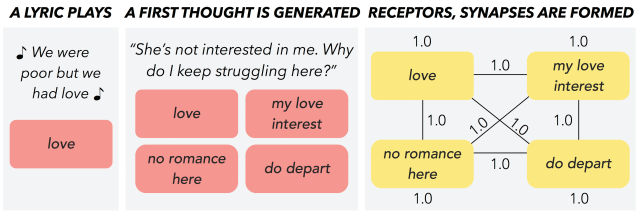

Figure 1: A song lyric emitting a *love* signal elicits a first thought in the mind of a character, causing an initial set of signal receptors and synapses to be formed.

cal themes. Here, a *theme* is a notion (represented as a string, *e.g.*, 'commitment') that in our system functions as an environmental stimulus that elicits character thoughts. More specifically, a theme is a type of *signal*, which is a critical notion that we discuss in more detail in a section below. Our annotation process was simple: we read over the lyrics of our chosen songs and brainstormed specific thoughts that they could believably evoke that would pertain to whether or not to leave a town; we then isolated the themes that could connect stanzas to such thoughts and annotated the former accordingly (later we discuss how such themes are also attached to the generable thought content). To prevent authorial blowup, we settled on an initial limited set of three total lyrical themes: *love*, *commitment*, and *disappointment*. Figures 1 and 2 show examples of lyrics and associated themes.

Core Loop

Juke Joint is a small work of interactive storytelling, and its core loop is intentionally simple: as a ghost haunting the jukebox, the player chooses which of its three songs will play; the song begins to play; the character presses a key to proceed from the current stanza to the character thoughts that it evokes; the song continues; once a character's dilemma is resolved, she leaves the bar; once the song has finished playing, the game ends. The playthroughs we like best end with the last few song stanzas tapering off in an empty bar (since both characters have already left to take action in their lives). Later, we discuss how exactly song lyrics evoke character thoughts, as well as how character thoughts bring about resolutions to their respective dilemmas.

Stimulus Processing

In this section, we will describe the mechanisms by which environmental stimuli are processed to elicit character thoughts. While we loosely operationalize concepts from neurobiology, this is only for the sake of conveniently structuring our architecture according to elegant and familiar notions like neurons and synapses; we make no claims about biological accuracy, and such realism is not a design goal.

Signals

The simulated game environment emits *signals*, which may evoke character thoughts. These are represented as strings, and their primary function is to structure content requests that are sent to the content generator that produces character thoughts (which we discuss in the next section). The main signals that we implemented in *Juke Joint* are the lyrical themes that we discussed above, though there are other kinds, too, as we discuss below. Moreover, it is worth emphasizing that the architecture we have built is agnostic to the nature or origin of signal. As we discuss below, this paves the way for future extensions by which characters may form thoughts in response to all different types of signals—for instance, ones pertaining to specific kinds of events or artifacts, or aspects of characters or places, or even literal characters and places.

Receptors

When a character encounters a new signal in the world, a dedicated *receptor* for that signal is formed in the character's mind. Such a receptor then handles any subsequent exposure to its associated signal, and it will also adapt according to the character's ongoing subjective experience. This adaptation is characterized by the receptor's *voltage*, which is a simple positive floating-point number that represents the salience of the associated signal to its character. The stronger a receptor's voltage, the more likely its signal will be to elicit associated character thoughts; in turn, the more such thoughts are elicited, the stronger the receptor's voltage becomes. This simple mechanism for updating signal voltages supports an elegant feedback loop: as a signal becomes more salient to a character, it will elicit more thoughts in her mind, which in turn will make the signal even more salient to her, and so forth. In *Juke Joint*, this specifically makes lyrical themes that have already elicited thoughts more likely to elicit more thoughts, which elegantly enables *trains of thought* to emerge in ways that are guided by the lyrics. Contrastingly, as time goes by without a character encountering a given signal, the voltage of the associated receptor will decay by a set rate. This is implemented in our architecture, but it is not at play in *Juke Joint*, since not enough time will pass during a given playthrough. However, because characters will think several thoughts over the course of a song, character receptor voltages change drastically across playthroughs. As we discuss below, this is a core part of how characters come to resolve their dilemmas.

Synapses

Each character thought constructed by our content generator (in response to environmental stimuli) comes packaged with a set of associated signals, which is then operated over to form *synapses* between the corresponding receptors. Specifically, each possible pair of signals packaged up with the character thought will be enumerated to form a synapse between their corresponding receptors; this is shown in Figure 1. If such a synapse already exists, its *weight* is increased; this is illustrated in Figure 2. Synapse weights are like receptor voltages in that they represent salience and allow for feedback loops by which signals that elicit thoughts become more likely to elicit further thoughts—with synapses, the feedback loop is powered by synaptic transmission, which we discuss next. Generally, the more a pair of signals co-occur in the environment or in a character's thought, the more associated they become in the character's mind. This means that certain signals will come to remind characters of

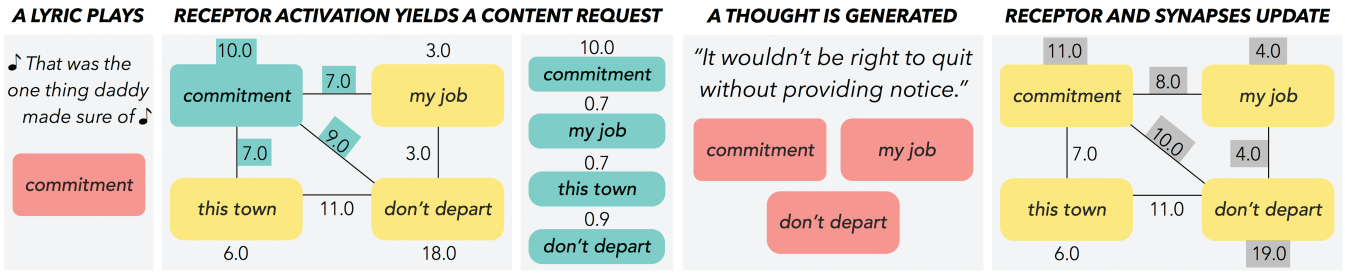

Figure 2: A *commitment* signal emitted by a song lyric activates the corresponding receptor in a character's (already developed) neural machinery, triggering synaptic transmission, which in turn produces a content request. This request is then processed by our NLG system, where its signal weights are used to drive the system's heuristic approach (not pictured). After a thought has been generated by the system, the signal tags packaged with it are used to update the thinker's receptors and synapses.

certain other signals, allowing them to think thoughts that are not directly elicited by environment stimuli, but rather by virtue of synaptic pathways. For instance, a *commitment* receptor could form a synapse with a *my job* receptor, making exposure to *commitment* signals evoke thoughts about one's job (as in Figure 2). This is a loose operationalization of *long-term potentiation* (Malenka and Nicoll 1999), a concept in neurobiology described by a familiar phrase—*fire together, wire together*. Synapse weights also decay, but again this does not occur during the short duration (in terms of simulated game time) of a *Juke Joint* playthrough.

Synaptic Transmission

When a set of environmental signals are encountered by a character, her mind's corresponding receptors are activated, which in turn causes activity to propagate across the synaptic pathways that connect the activated receptors to other receptors (see Figure 2). This causes the connected receptors to also become activated, though less so.[3] The result of this *synaptic transmission* process is a set of receptors each having one of two activation levels: the base level of activation for receptors that were activated by environmental signals, and a lower level for ones that were activated by synaptic transmission. As a final step, we modulate the activation levels of the environmentally activated receptors according to their voltages (higher voltages increase activation) and modulate the levels of the synaptically activated receptors according to the weights of the pertinent synapses (higher weights increase activation). The result of this is an array of *(receptor signal, activation level)* tuples, which are the very content requests that are sent to the thought generator, as we discuss next. Figure 2 illustrates how synaptic transmission works to produce content requests.

Character Thoughts

In this section, we will describe how character thoughts are elicited as a result of receptors being activated in the way described in the last section. This is an NLG task, and the subsystem that handles it is used as a case study in another paper (Ryan et al. 2016), so we invite the reader to consult that publication for supplementary details.

[3]To prevent things from getting too unwieldy, activity does not also propagate across the synapses of the connected receptors.

Expressionist

Character thought generation is made possible by a tool that we have developed called Expressionist (Ryan et al. 2016). Using Expressionist, an author specifies a *context-free grammar* (CFG)—that is, *nonterminal symbols* and the *production rules* that may exhaustively expand them to produce *terminal derivations* composed fully of *terminal symbols* (*i.e.*, strings). Crucially, nonterminal symbols may be annotated using arbitrary *tagsets* and *tags* that are defined by the author. When a terminal derivation is produced by any given CFG, it will have expanded a set of nonterminal symbols along the way—in Expressionist, such a derivation accumulates all of the markup that an author has attributed to all of the symbols in this set. This allows an author to modularly specify capsules that contain both symbolic markup (for our purposes here, annotations for *signals*, described above) and the rules for producing variations of the linguistic expression of that markup. After a session with Expressionist, an author exports her grammar as a JSON file that a content generator may operate over to produce content on the fly.

Our Grammar

We used Expressionist to author a grammar that has two tagsets: *preconditions* and *signals*. Specifically, the nonterminal symbols in the grammar have been attributed zero or more tags specifying preconditions for expanding them, as well as zero or more tags that specify which signals their expansions may be used to express (more on this later). The former tags allow us to author thoughts that may only occur in the minds of characters with certain life contexts, a prospect we alluded to above. For instance, a nonterminal symbol whose expansions pertain to thoughts about a character's job may be tagged with a precondition that the character for whom a thought is currently being generated actually has a job. In fact, we can specify arbitrary preconditions that may be checked against the current game state—these are simply raw snippets of Python code that are evaluated at runtime (during content generation). The *Juke Joint* Expressionist grammar currently features 292 nonterminals, 801

production rules, and a massive possibility space comprising 6.3 quintillion unique character thoughts.

Content Requests

As alluded to above, thought generation in our architecture begins with a *content request*—a structured set of data that drives the generation process. These are specifically arrays of *(receptor signal, activation level)* tuples that are output by the stimuli-processing pipeline outlined above; Figure 2 shows an example content request. Additionally, the content generator always maintains access to the current game state (including the specific character for whom the thought is being generated).

Generation Task

The generation task is as follows: operate over the Expressionist grammar to find a path through it whose nonterminal symbols all have their preconditions met (we will use the term *viable* for this) and are highly associated with the signals appearing in the content request (as indicated by corresponding tags). For any such path, executing all of the production rules on it (and concatenating the results) will produce a surface-level thought that is associated with the signals that appeared in the content request.

Symbol and Rule Evaluation

Given our articulation of the task, a core issue is being able to evaluate how associated a given nonterminal symbol is with the signals included in the content request. To do this evaluation, we employ a simple heuristic: the *association score* for a nonterminal symbol is the sum of the activation levels (specified in the content request) for all signals appearing in both the request and the signal tags attributed to the nonterminal symbol. Similarly, production rules are scored by summing the association scores of all of the symbols appearing on both their left- and righthand sides. Note that this scoring procedure does not penalize for the appearance of tags cueing signals that are *not* in the content request. This is because we want to allow characters to have thoughts that are not solely connected to the environmental signals—*e.g.*, as in Figure 2, the lyrical theme *commitment* should be able to evoke a thought about commitment to one's job, which would also have the signal tag *my job*—to allow both for more subtle kinds of association, and also for rich synaptic phenomena. In fact, the resolution of character dilemmas in *Juke Joint* critically relies on this, as we discuss below.

Generation Procedure

To actually generate thoughts at runtime, we employ a greedy procedure that we call *heuristic expansion* (Ryan et al. 2016). First, all viable nonterminal symbols in the grammar are ranked according to their association scores. Next, the procedure selects the first nonterminal in the ranking and ranks all of its production rules by their association scores. It then attempts to expand the selected symbol by executing the top-ranked production rule, which is only successful if all of the symbols on both sides of the rule are viable. From here, the process is recursive: do the same for each choice between nested nonterminal symbols (or nested production rules), expanding (and firing) them until a set of terminal symbols (*i.e.*, a string) has been produced. While this will greedily produce an expansion of the nonterminal symbol that was initially targeted, the result may only be a fragment of a complete thought—this is because the target symbol may not have been a top-level symbol in the grammar. In this case, the procedure then carries out the same procedure in the *other direction*. That is, instead of chaining from symbols on the lefthand side of production rules to symbols on the righthand side, it chains in the opposite direction. We call this unusual usage of CFGs *middle-out expansion*, and detail it at length elsewhere (Ryan, Mateas, and Wardrip-Fruin 2016a).[4] The result of the entire procedure, then, is a complete character thought in surface natural language. While there will always be an optimal path through the grammar—one whose nonterminals sum to the highest cumulative association score—it is not feasible to search the whole space, given its size (noted above). By using a greedy procedure, we sacrifice optimality for tractability, with the best options being chosen at each local decision point. This peculiar usage of CFGs makes our approach more akin to search-based methods for procedural content generation (Togelius et al. 2011) than constructive ones (Shaker et al. 2015).

Thought Effects

Once a complete thought has been generated, the system compiles all of the nonterminal symbols that were expanded to produce it and collects all of the signals that were tagged to those symbols during authoring. The result is an array of signals that are associated with the generated thought. These signals are then used to update receptors and synapses in the ways described above: new receptors are formed, as needed (see Figure 1); each corresponding receptor has its voltage increased by a base amount (see Figures 1 and 2); new synapses are formed, as needed (according to all pairwise combinations of the signals; see Figure 1); and each corresponding synapse has its weight increased by a base amount (see Figures 1 and 2). As such, thinking thoughts, which a character cannot help but do as her subjective experience proceeds, fundamentally changes her (in terms of the composition of her neural machinery).

Character Arcs

Now that we have outlined our architecture, we will explain how we utilize it as authors to structure thought generation such that character trains of thought resemble minimal narrative arcs that naturally come to resolutions—in this case, decisions about whether or not to leave town.

Decision Signals

There are two special *decision signals* that we utilized to structure our authoring process; these are named *don't depart* and *do depart*. These signals pertain to an inclination toward either leaving town or staying, respectively, and only originate from thinking a thought that is associated with one

[4] While we employed middle-out expansion in another project, the added rub of heuristic expansion is unique to *Juke Joint*.

of them. Any generated thought that is associated with a *don't depart* signal will contain reasoning in support of the character staying in town (as seen in Figure 2), and likewise (though oppositely) for *do depart* signals (as in Figure 1).

Justification Signals

Additionally, we make heavy use of a series of *justification signals*: *my job*, *my love interest*, *my partner*, *new job elsewhere*, *no romance here*, and *this town*. These signals correspond to justifications for a character either leaving or staying—*e.g.*, staying because of commitment to one's job, as in Figure 2. Each justification signal is associated with generable thoughts supporting both possible decisions.

Feedback Loops

Decision and justification signals work nicely with our architecture to yield feedback loops that lead characters toward decisions (resolutions to their dilemmas) that are associated with particular justifications. This works by virtue of the simple receptor and synapse update procedures that we outlined above. Specifically, the more a character has thoughts associated with a decision signal like *do depart*, the higher the voltage for her corresponding receptor will become—we then take this voltage to encode her inclination toward, in this case, choosing to leave town. In turn, the more she has thoughts associated with a decision signal *and* one or more justification signals, the heavier the weights of the synapses connecting them will become. For instance, if a character repeatedly has thoughts building up a synapse between *don't depart* and *my job*—as the thought in Figure 2 does—her mind will actually encode this history of reasoning in the weight of the synapse connecting those receptors (in Figure 2, the 4.0 value). This elegantly allows the system to reason about decisions and their justifications without treating these signals in any special way (they update receptors and synapses exactly as all signals do).

Trains of Thought

Because the first thoughts characters think during a playthrough will make the signals associated with them more salient (by increasing the voltages of their receptors), characters will tend to have thoughts that are associated with the same signals as thoughts that they had before. This naturally produces *trains of thought* that progress with some amount of coherence, *e.g.*, a character returning frequently to deliberations about her love life. Of course, this could easily lead to repetition—to combat this, we make it less likely to expand nonterminal symbols that were expanded to generate earlier thoughts by the thinker. Moreover, the lyrics emanating from the jukebox will always provide the set of environmental signals that prompt and guide all character thoughts in the first place.

Dilemma Resolutions

Once a character's receptor voltage for a decision signal—either *do depart* or *don't depart*—exceeds a threshold that we have set, the character arrives at the corresponding resolution to her dilemma. This triggers a *summative thought* that expresses the character's decision, as well as her top justifications for making the decision, which are inferred by looking at the heaviest synapses connecting the receptor for the decision signal and ones for justification signals. More technically, this is all handled by preconditions specified in our Expressionist grammar: once the voltage threshold for a decision signal's receptor exceeds the threshold, only a summative thought can be generated. This simply happens when it is time for a character to have another thought, and the particular justifications that will be expressed in the thought are also handled by preconditions (that check the thinker's current synapse weights). Here is an example summative thought, with callbacks to three justifications that would have been included in earlier thoughts:

> *That's it, that's it. This song has made me realize what I need to do. I've got three good reasons for my decision. First of all, I have a great job lined up elsewhere. Second, I'm making a fool of myself chasing after Ida all the time. Third, I don't even like it here. I'm ditching this terrible burg!*

If the song ends without the character coming to a decision, then there will be no resolution for that person—perhaps the ghost should have played a different song.

Conclusion and Future Work

We have presented *Juke Joint*, a small work of interactive storytelling that demonstrates an extension to the *Talk of the Town* framework enabling characters to form thoughts that are expressed in generated natural language. This is made possible by an AI architecture that simulates the processing of environmental stimuli through dedicated neural machinery in the minds of characters—adaptable receptors and synapses, to be specific. Once a set of stimuli has been processed by such machinery, a content request is sent to an NLG module that attempts to produce a character thought that is most associated with the stimuli included in the request. Critically, the generated thought comes packaged with metadata that specifies how to update the thinker's receptors and synapses. This makes the act of thinking a thought affect how exactly future stimuli will elicit future thoughts, allowing for streams of consciousness that progress naturally, as well as something more exciting: characters are fundamentally altered by the thoughts that they think.

In future work, we plan to extend our architecture so that character thoughts can be elicited in response to diverse kinds of signals beyond lyrical themes. For instance, signals could correspond to things like events or generated artifacts, or aspects of characters or places, or even the literal entities themselves. In tandem with this kind of extension, we would like to tune receptor and synapse updates so that they may work better over longer durations of simulated time, such as a character's entire life span. That way, a character's mind after an extended period of game time would come to encode interesting aspects of her subjective experience, with people, places, and things that remind her of other entities and various abstract notions. Though we ideally would like to be able to generate a huge variety of thoughts in natural language, we will likely have to develop a lower-fidelity no-

tion of a thought that merely captures the signals that are associated with it. This is all that would be needed to drive the receptor and synapse update procedures that we have outlined above, which is what is required to simulate the character subjective phenomena that we have just outlined.

While we hope to demonstrate such future extensions in a dedicated experience, in the meantime we are polishing *Juke Joint* and will release it as a graphical browser-based game. In any event, we hope that this project inspires future work exploring the generation of character thoughts, and the modeling of agent subjective experience more broadly.

References

Bates, J.; Loyall, A. B.; and Reilly, W. S. 1992. An architecture for action, emotion, and social behavior. In *Proc. Modelling Autonomous Agents in a Multi-Agent World*.

Cook, M.; Eladhari, M. P.; Smith, A. M.; Smith, G.; Thompson, T.; and Togelius, J. 2015. AI-based games: Contrabot and What Did You Do? In *Proc. FDG*.

Diller, D. E.; Ferguson, W.; Leung, A. M.; Benyo, B.; and Foley, D. 2004. Behavior modeling in commercial games. *Behavior Representation in Modeling and Simulation*.

Eladhari, M. P.; Sullivan, A.; Smith, G.; and McCoy, J. 2011. AI-based game design: Enabling new playable experiences. Technical Report UCSC-SOE-11-27, UC Santa Cruz.

Ferber, J. 1999. *Multi-agent systems: an introduction to distributed artificial intelligence*. Addison-Wesley Reading.

Garbe, J.; Reed, A.; Dickinson, M.; Mateas, M.; and Wardrip-Fruin, N. 2014. Author assistance visualizations for Ice-Bound, a combinatorial narrative. *Proc. FDG*.

Harrell, D. F. 2005. Shades of computational evocation and meaning: The griot system and improvisational poetry generation. In *Proc. Digital Arts and Culture*.

Ho, W. C.; Dautenhahn, K.; and Nehaniv, C. L. 2008. Computational memory architectures for autobiographic agents interacting in a complex virtual environment: a working model. *Connection Science*.

Horswill, I. 1993. Polly: A vision-based artificial agent. In *Proc. AAAI*.

Horswill, I. 2014. Game design for classical AI. In *Proc. EXAG*.

Horswill, I. 2015. MKULTRA. In *Proc. AIIDE*.

Isla, D. 2013. Third eye crime: Building a stealth game around occupancy maps. In *Proc. AIIDE*.

Kolodner, J. L. 1983. Reconstructive memory: A computer model. *Cognitive Science*.

Leonard, T. 2003. Building an AI sensory system: Examining the design of Thief: The Dark Project. *Proc. Game Development Conference*.

Lessard, J. 2016. Designing natural-language game conversations. *Proc. DiGRA-FDG*.

Loyall, A. B., and Bates, J. 1997. Personality-rich believable agents that use language. In *Proc. Autonomous Agents*.

Malenka, R. C., and Nicoll, R. A. 1999. Long-term potentiation—a decade of progress? *Science*.

Mateas, M., and Stern, A. 2003. Façade: An experiment in building a fully-realized interactive drama. In *Proc. Game Developers Conference*.

Meehan, J. R. 1977. Tale-spin, an interactive program that writes stories. In *Proc. IJCAI*.

Mueller, E. T. 1990. *Daydreaming in humans and machines: a computer model of the stream of thought*. Intellect Books.

Nehaniv, C., and Dautenhahn, K. 1998. Embodiment and memories-algebras of time and history for autobiographic agents. *Cybernetics and Systems*.

Orkin, J. 2006. Three states and a plan: the AI of FEAR. In *Proc. Game Developers Conference*.

Pérez y Pérez, R.; Sosa, R.; and Lemaitre, C. 2007. A computer model for visual-daydreaming. In *Proc. INT*.

Reed, A. A.; Garbe, J.; Wardrip-Fruin, N.; and Mateas, M. 2014. Ice-Bound: Combining richly-realized story with expressive gameplay. In *Proc. FDG*.

Ryan, J. O.; Summerville, A.; Mateas, M.; and Wardrip-Fruin, N. 2015. Toward characters who observe, tell, misremember, and lie. In *Proc. Experimental AI in Games*.

Ryan, J.; Seither, E.; Mateas, M.; and Wardrip-Fruin, N. 2016. Expressionist: An authoring tool for in-game text generation. In *Proc. ICIDS*.

Ryan, J.; Mateas, M.; and Wardrip-Fruin, N. 2016a. Characters who speak their minds: Dialogue generation in Talk of the Town. In *Proc. AIIDE*.

Ryan, J.; Mateas, M.; and Wardrip-Fruin, N. 2016b. A simple method for evolving large character social networks. In *Proc. Social Believability in Games*.

Samuel, B.; Ryan, J.; Summerville, A.; Mateas, M.; and Wardrip-Fruin, N. 2016. Bad News: An experiment in computationally assisted performance. In *Proc. ICIDS*.

Scheier, C., and Lambrinos, D. 1996. Categorization in a real-world agent using haptic exploration and active perception. In *Proc. SAB*.

Shaker, N.; Liapis, A.; Togelius, J.; Lopes, R.; and Bidara, R. 2015. Constructive generation methods for dungeons and levels (draft). In *Procedural Content Generation in Games*.

Togelius, J.; Yannakakis, G. N.; Stanley, K. O.; and Browne, C. 2011. Search-based procedural content generation: A taxonomy and survey. *TCIAIG*.

Treanor, M.; Zook, A.; Eladhari, M. P.; Togelius, J.; Smith, G.; Cook, M.; Thompson, T.; Magerko, B.; Levine, J.; and Smith, A. 2015. AI-based game design patterns. In *Proc. FDG*.

Weyns, D.; Steegmans, E.; and Holvoet, T. 2004. Towards active perception in situated multi-agent systems. *Applied Artificial Intelligence*.

Zhu, J., and Harrell, D. F. 2008. Daydreaming with intention: Scalable blending-based imagining and agency in generative interactive narrative. In *Proc. Creative Intelligent Systems*.