Appointment in Samarra:

Pre-destination and Bi-camerality in Lightweight Story-Telling Systems

Tony Veale
School of Computer Science and Informatics
University College Dublin, Belfield D4, Ireland.
Tony.Veale@UCD.ie

Abstract

Stories are most able to sweep us up and carry us along when we design them to be journeys of the mind. This paper presents a unification of two journey-based story generation models, the character-development model of *The Flux Capacitor* and the plot development model of *Scéalextric*. This union of complementary approaches allows us to build stories with shape *and* directionality. Moreover, since it facilitates the generation of coherent stories by the most minimal of computing architectures, the memory-less state machine, this joint model proves to be ideally suited to the generation of stories by bots. To squeeze a full story-generator into the context-free grammars of Tracery, we give a practical form to two exotic ideas: predestination, and bicamerality of mind.

Journey into Mystery

Every story is a journey we willingly undertake, especially when in the company of relatable characters and an adroit guide. Most are forays into the unknown, as only an author can lead the way to our final destinations. Our stories lay down these paths to other lives by instantiating a metaphor schema Lakoff and Johnson (1980) call *Life is a Journey*, and what Yorke (2013) calls – using another metaphor that shapes many a tale – a journey *into the woods*. Campbell (1949) saw this journey as the monomythic basis of most heroic tales: when heeding the call to adventure, heroes must leave behind the world of the familiar to meet new challenges in strange new lands. Only when they have been changed by their experiences can heroes ever return home, to find themselves and their old lives utterly transformed.

We have good reason for talking of the *twists and turns* of a thrilling tale, for twisty tales arise from journeys along twisted tracks. Authors sometimes propel their characters along paths with unexpected destinations, for reasons that only become clear at the very end of a journey. Consider this tiny gem from the master of the short story, Somerset Maugham (1933). The entirety of the tale is given below:

"The speaker is Death.

There was a merchant in Baghdad who sent his servant to market to buy provisions and in a little while the servant came back, white and trembling, and said, Master, just now when I was in the marketplace I was jostled by a woman in the crowd and when I turned I saw it was Death that jostled me. She looked at me and made a threatening gesture. Now, lend me your horse, and I will ride away from this city and avoid my fate. I will go to Samarra and there Death will not find me. The merchant lent him his horse, and the servant mounted it, and he dug his spurs in its flanks and as fast as the horse could gallop he went. Then the merchant went down to the marketplace and he saw me standing in the crowd and he came to me and said, Why did you make a threatening gesture to my servant when you saw him this morning? That was not a threatening gesture, I said, it was only a start of surprise. I was astonished to see him in Baghdad, for I had an appointment with him tonight in Samarra."

As Scrooge tells us in *A Christmas Carol*, "Men's courses will foreshadow certain ends ... but if courses be departed from, the ends will change." Tales of predestination, such as Maugham's, subvert this logic with characters who rush headlong toward the inevitable as they run from their fates. In truth, all fictional characters are subject to the forces of predestination; what differs from tale to tale is the extent to which authors reveal the shape of the tracks on which their characters are forced to run, and whether or not characters have any self-knowledge of those tracks. Automated story-tellers are no less *natural* then in their use of rigid plotting and goal-driven planning than their human counterparts. In this paper we argue it makes sound computational sense to explicitly model this notion of character predestination. We will show how predestination can simplify the construction of dense narrative spaces to a point where coherent stories can be generated with the simplest context-free grammars.

Our goals here are more practical than empirical: we aim to simplify the mechanics of story-telling to a level where complex stories can be woven by a minimal state machine with no memory and no global executive. To this end we rehabilitate another somewhat exotic idea, Jayne's (1976) theory linking consciousness to the bicamerality of mind.

For Jayne, the flow of data between the hemispheres of the brain is an interior dialogue that only becomes an internal *monologue* when beings become conscious enough to take full ownership of both sides of the conversation. We do not set out here to tackle the grand challenge of consciousness, for as Jayne notes, it is not at all clear that consciousness is even needed for creativity. Yet Computer Science makes many bicameral divisions that are usefully blurred by AI, such as the line between code and data that is erased by the LISP and PROLOG languages, and we will show here how simple generative systems can weave stories by sustaining a back-and-forth dialogue between simpler bicameral parts.

We unite these strands in the following sections, starting with a discussion of related work and ideas in the next. Our purpose is to unify two complementary approaches to story creation that focus, respectively, on character development and plotting: the *Flux Capacitor* of Veale (2014) and the *Scéalextric* model of Veale (2017). We show here that the unification of both permits the construction of dense story spaces in which characters may wander, not idly or blindly, but with a sense of purpose and narrative momentum. As labeled directed graphs, these spaces are easily transformed into lightweight *Tracery* grammars (Compton *et al*., 2015), which can then be used to specify generative Twitterbots. The advantages of the context-free *Tracery* formalism outweigh its expressive limitations, and we show here how the idea of predestination proves to be a practical workaround to the need for long or short-term memory. To also obviate the need for top-down planning in story telling, we show how Jayne's bicameral divide finds a practical counterpart in the two-grammar approach to bot definition of George Buckenham's *CheapBotsDoneQuick.com*, a web platform that hosts Twitterbots specified as Tracery grammars. So we model story generation as a two-level process in which we first build generators of story *spaces*, and then specify context-free explorers of these spaces to generate novel stories as they race to their own appointments in Samarra.

Related Work and Ideas

The journey schema is so conducive to story-generation by a machine not just because it offers a productive metaphor for narratives; it is also a productive metaphor for AI itself, or at least AI in the classic search-oriented mold. Just as a hero searches for resolution on some Campbellian quest, or roams the narrative thicket of Yorke's woods, AI problem solvers purposefully explore a state-space of possibilities, backtracking here and advancing there, until a predefined objective is reached. Creative systems are free to alter their objective functions – their sense of *value* – as they wander, just as they might transform the space itself. In either case, the need for search persists. For a story-telling AI the space is a graph of branching narrative possibilities, and the story is a function of the path taken by the teller to its goal state. This story-path can be given an *a priori* rationale *post-hoc*, to justify the actions of a hero in terms of their end state, as though the hero planned the actions to reach that very state. Or this rationale can be specified *a priori*, so that a planner can then seek the most dramatic path to making it a reality. Riedl & Young (2010) thus use an explicit planner to give their heroes issues to resolve and the plans to resolve them, yet most story-generation AI systems, from Meehan (1981) and Turner (1994) to Pérez y Pérez & Sharples (2004) to Riedl and Young (2010) to Gervás (2013) and Gervás *et al.* (2016) string together causes and effects to construct plots that seem to imbue characters with plan-like intentionality.

We read intentionality into the way a character interacts with others. If A assists B to reach C then reaching C may have been A's goal all along. The bric a brac of a story are its ancillary figures, obstacles, signs, magic talismans, its helpers and hindrances on the road to its final destination. In exploiting the affordances of these narrative morphemes – what Propp (1928) calls the *morphology of the tale* – a hero exhibits relatable drives and intentions. Propp applied his morphological analysis to Russian folktales, but authors such as Gervás *et al.* (2016) have applied his inventory of character types and functions to the generation of more modern narratives. Others focus on specific elements of the Proppian scheme. Veale (2014) sees the transformational role of stories – how they turn characters of type A into heroes or villains of type B – as the most fascinating aspect of story generation. Propp applied the label *transfiguration* to the transformation of a hero in a story, whilst Campbell dedicated several key stages of his hero's journey to the change, from the call to adventure and the crossing of the threshold to the midway ordeal and near-end resurrection.

Veale (2014) defined a Campbellesque annotation for use in the *Flux Capacitor* to label the actions we typically associate with people from different categories, from artists and scientists to priests and criminals. Every category can be viewed as a journey, with the *call to adventure* serving as its entry point, and the *ordeal* (after a trip to the *inmost cave*) serving as its point of egress. Actions of the first kind are annotated as level 0 when they initiate a person into a category; for instance, studying medicine is a level 0 action for doctors whilst renouncing religion is a level 0 action for atheists. Actions of the second kind are annotated as level 9 if they result in an erstwhile member breaking fully with a category; finding religion is a level 9 action for atheists, whilst losing religion is a level 9 action for believers. The labels 1 to 8 are reserved for actions that link the extremes, with 5 representing the high-water mark of a category, the point at which a person is fully operational as a member; for example, the act of evangelizing as a believer, treating illness as a doctor or spreading doubt as an atheist. Actions labeled with a 2, 3, 4 or 5 mark the growth of a character, while a 6, 7 or 8 document the character's gradual move to the exit. The *Flux Capacitor* generates its plots by linking an exit from one category with an entry into another, and pairs its categories so as to maximize affective dissonance. So, in this way, atheists become believers, heroes become tyrants, sinners become saints, billionaires become bums and cops turn into the crooks they most despise. As such, *Flux Capacitor* generates capsule tales with an ironic shape, mere plot outlines rather than fleshed-out narratives.

The *Scéalextric* model of Veale (2017) focuses more on the bread-and-butter issues of plot design: given an action

V by character A toward character B, with what action is B likely to respond? Given a suitable response V', a system can now determine how A might respond with V'', and so on, until a terminating action V* is performed by A or B. A causal graph of actions and reactions was first constructed by looking for pairs of annotated actions in *Flux Capacitor* with sequential labels, such as 0,1 or 6,7, and by linking these actions into a labeled directed graph. When the first action's label is in {0…5} and the second's is in {6…9} then the connecting arc is labeled "but" in the causal graph; it is labeled "then" in all other cases. This initial graph is manually edited to transform many "then" labels into "so" labels when the connection is a strongly causal one. At this stage additional arcs are also added to create a dense story graph in which 820 different action "verbs" are interlinked. To generate a story, a generator picks a verb at random and initiates a random walk in the forest of causal connections. For every action in the graph, a piece of text is defined to serve as a scene-setter for a story opening with that action. A short text is likewise defined for every action to serve as a moral summation should a story terminate at that action. Also associated with each action is a set of one or more idiomatic templates, to allow each to be rendered in fluent natural language. Any random walk in the causal graph can thus be framed as a complete narrative, with a motivating introduction and a summarizing conclusion bookending a locally-coherent journey along causally-connected actions.

When plotting is reduced to a random walk in the causal woods, characterization fulfills an ever more vital function. Characters may follow a plot as it winds through the graph, but readers will only follow those characters if they seem to know what they are doing. To achieve an integration of character and plot, a system must either choose its actions to suit a character, or it must at least render those actions to reflect what readers already know about the characters. The experiments of Veale & Valitutti (2017) evaluate the latter. Using *Scéalextric* to generate a range of plots, they render the plots as textual narratives using two alternate strategies. In the first, character labels are chosen at random from a pool of stock animals, such as koala, monkey and snake, and plots are rendered by inserting these labels (e.g. "*the koala*") into the slots in *Scéalextric*'s idiomatic templates. In the second, familiar characters are plucked from a large inventory of famous faces, fictional and historical, called the *NOC list* (Veale, 2015). This knowledge-base describes its characters in generous detail, providing for each a list of positive and negative qualities, a set of categories, a list of domains, typical activities, weapons, vehicles and clothing, known opponents and mates, political leanings, and so on. Characters are chosen at random, but in pairs, for each tale, so that the protagonist and antagonist are well-matched and perhaps thematically-related too. Steve Jobs might thus be paired with Leonardo Da Vinci or Bill Gates. When actions involving NOC characters are rendered, the system tries to shoehorn specific knowledge from their NOC entries into the text; for example, if A attacks B, the weapon of choice for A is used; when B flees from A, the vehicle of choice for B is used, as is an associated location to hide in.

Evaluating the outputs of each strategy on 6 dimensions – *laughter*, *entertainment*, *imagination*, *vividness*, *drama* and *silliness* – using the crowd-sourcing site CrowdFlower, Veale & Valitutti reported significant improvements for all dimensions when plots are rendered with NOC characters as opposed to generic animals. Strikingly, this applies just as much to *drama* – the dimension that is, most obviously, the product of plot-level decisions – as it does to any other. In the next section we take the road not followed by Veale & Valitutti, to explore the other approach to the integration of characterization and plot: picking (as opposed to merely rendering) a story's actions to suit the characters involved.

Lost in Narrative Space

The *Flux Capacitor* maps actions to the kinds of characters that perform them, while *Scéalextric* maps actions to each other, to yield a narrative model of cause and effect. Since character influences actions and actions shape character, it makes sense to unify these complementary approaches. To put plot at the service of character, we can use *Scéalextric* to search for the shortest sequence of actions that produces, and explains, any change proposed by the *Flux Capacitor*. Conversely, to use character to drive plot, we can use the *Flux Capacitor* to specify the first and last actions of a plot and use *Scéalextric* to trace out the intermediate journey.

Scéalextric assumes that each of its stories involves just two principal characters, a protagonist A and antagonist B, so its various structures and templates have slots to house the character choices that are ultimately made for A and B. The *Flux Capacitor* makes similar assumptions about arity: categories are associated with actions that comprise a verb and another category, such as *heal:illness* and *debate:idea*. When the other category denotes a kind of person, the verb may denote an interpersonal relationship, such as *criticize* or *debate_with*, that is also defined for *Scéalextric*. In those cases we can map the categories connected by the verb into two roles, the protagonist (A) and antagonist (B). Any verb linking A and B that is annotated with a 0, 1 or 2 can now be used as the opening action of a story involving A and B, while a connecting verb annotated with an 8 or a 9 (not every category has a level 0 verb or a level 9 verb) can be used as the closing action for the same story. Consider the example of *theorist* and *critic*, which are linked by the verbs *disagree_with* (level 1) and *denounce* (level 8). The level 0 action for theorist, *develop:theory*, is not one that can be exploited by *Scéalextric*, so we must settle for one annotated as level 1. Likewise, the denunciation of a critic does not usher a person out of the theorist category, so this action is annotated as level 8 rather than level 9. However, *denounce* is a verb that is also defined for *Scéalextric*, so it makes a suitable destination for any story about a theorist. Using *Scéalextric* to trace out a path from *disagree_with* to *denounce,* the following sequence of actions is proposed:

disagree_with → *are_debated_by* → *are_roused_by* → *fall_in_love_with* → *confess_to* → *are_betrayed_by* → *are_arrested_for_killing* → *denounce*

This shows precisely what *Scéalextric* brings to the union

of both systems that *Flux Capacitor* cannot provide alone: its journey through the causal graph pushes the relationship between theorist and critic into the realm of romance, with a dark turn into betrayal and retribution. This is just one of many pathways between disagreement and denunciation in *Scéalextric's* causal graph, and other plots can be derived from the same start and end points. These can be rendered with the roles of A and B filled with "the theorist" and "the critic" respectively, or the NOC can be used to suggest some appropriate names to attach to these categories, such as *Rush Limbaugh* as critic and *Charles Darwin* as theorist.

As presented in Veale (2017), all *Scéalextric* stories start and end at arbitrary points in the causal graph. The paths proposed by *Flux Capacitor* yield more interesting stories because they reflect the journeys taken by people through their chosen categories in life. This category-journey gives each narrative a satisfying shape, and directly instantiates Lakoff & Johnson's *Life is a Journey* schema. Taking its cues from *Flux Capacitor*'s annotations, the joint system generates 12,000 stories that start at a category-entry point and terminate at the brink of category-departure. We could generate far more or far less, but this is an ample sample. We then fold these 12,000 pathways into a single directed graph S that will serve as our story space. Each vertex V in S is an action verb that links to the next actions in a story with arcs labeled *so*, *then* or *but*. Unlike the causal graph used by *Scéalextric*, a subset of vertices are marked as start or end nodes for stories; a well-formed story can start at a vertex designated *start* and conclude at one designated *end*. Since the original 12,000 stories are merged, any single vertex leads directly to any of the subsequent actions from any story that contains it. In this way the graph S gives rise to story possibilities that are not in the original sample.

These possibilities include a potential for the story-teller to get lost in the woods, to wander aimlessly in the graph S until it finds a vertex, any vertex, designated *end*. For the teller to explore S with a sense of purpose, every vertex V must act as a signpost, not just to the very next vertices but to the end of the story too, otherwise the shape imposed on those stories by the *Flux Capacitor* will have been lost. To give vertices a sense of predestination, they must encode not just an action itself, but the final action of the story too. Here is our *theorist:critic* plot again, in this new encoding:

disagree_with/denounce → are_debated_by/denounce → are_roused_by/denounce →fall_in_love_with/denounce → confess_to/denounce → are_betrayed_by/denounce → are_arrested_for_killing/denounce → denounce/denounce

When our sample of 12000 stories is folded into S with this encoding, every vertex V/E in S carries with it a sense of narrative momentum. A vertex V/E represents the action V in a tale terminating with the action E, so that V/E can only be connected to other vertices V_1/E, V_2/E, …, V_n/E. Thus, any vertex in a story ending with betrayal can lead only to other vertices from tales of betrayal. So from the very start of a story, the teller knows how the tale will end, even if it does not yet know how that end will ultimately be reached.

Release the Bots

When a story graph S encodes long-distance directionality into every vertex V/E, an explorer of S no longer needs its own sense of direction. The territory becomes its own map *and* compass, so an explorer need keep no record of where it has been or where it is going. We can thus turn this map into a formal device that lacks all memory, such as a finite-state-machine. Since the graph S already resembles such a machine, with certain states/vertices marked as permissible start states and others marked as allowable end states, we can translate S directly into the corresponding Chomskyan grammar. Our choice of formalism is *Tracery* (Compton *et al*., 2015), a JSON-based format for context-free grammars that is widely-used for procedural content generation. The resulting Tracery grammar can be directly given to CBDQ (*CheapBotsDoneQuick*) to create a story-telling Twitterbot.

A Tracery grammar is a set of rewrite rules in which a non-terminal on the left-hand side is replaced by a random choice of expansions from the right-hand-side, as in:

"color": ["red", "blue", "green", "orange", "black"],

An expansion on the right may recursively mention a non-terminal (in hashes) that is then further expanded, as in:

"toy": ["#color# ball", "#color# bike", "#color# doll"],

The following Tracery rule is used by a Trump parody bot, *@trumpScuttleBot*, to tweet satirical *roses are red* poems:

"poem": ["#red_thing# are red, #blue_thing# are blue,
my #fan# #affirmation#, and #blue_rhyme#"],

When other non-terminals such as *red_thing* are defined, our grammar tweets (via CBDQ) the following short poem:

Plastic roses are red,
Sailors' curses are blue,
my human children will build my wall,
and pray that profits ensue

To generate a Tracery grammar from a story graph S, each vertex V/E is defined as a non-terminal with one expansion string for each adjacent next vertex in S. Each expansion string contains an idiomatic rendering (via *Scéalextric*) for its action verb, followed by a non-terminal reference to the set of possible next vertices on the path to a valid endpoint. The exception to this norm is the expansion string for any vertex of the form E/E, such as *denounce/denounce*: since this form indicates the last action in a story, the expansion contains the text "The End" in place of a non-terminal. The set of all vertices in S that can launch a story are gathered together as expansions for a single rule called "origin", the label Tracery reserves for the master rule of any grammar.

Since Tracery rules have no memory of prior expansions they cannot carry forward any context – such as names for the characters A and B – from one rule to the next. As a workaround, we can encode in the expansion of each V/E vertex the pair of categories that inspired a path through that vertex in S. In the following tweet these categories have been wrapped in quotes, and alternate across actions:

A 'master' was resented by a 'rival' and our 'master' overshadowed this 'rival' so our 'trailblazer' was copied by this 'imitator' but our 'tempter' misled this 'sinner' so our 'abbess' was dismissed by this 'bishop so our 'victimizer' begged forgiveness from this 'victim'

The quotes identify the categories as likely metaphors, yet no matter how relevant they may seem for specific actions, most metaphors are ambiguous and readers are easily disoriented as to who is who in this story. Is the *victim* that ends the tale the *master* that begins it, or is this *victim* the *rival*? To avoid confusion and foster narrative momentum, each action should be rendered with the same pair of characters. Yet since each is rendered independently of all others – this is what it means for a grammar to be context-free – we must rely on the sense of direction that is baked-in to each state and non-terminal. Predestination provides the answer: we associate a unique pair of characters (A & B) with each action E that can terminate a story with a vertex E/E. In our sample of 12,000 stories there are 220 distinct verbs that fit the bill, allowing 220 character pairs to be used by the grammar. Given the large inventory of name pairs that is harvested from the NOC list – we collect the first names of characters and their enemies or mates, such as *Woody* & *Mia* and *Sam* & *Diane* – we randomly assign these to the 220 termination actions. Suppose *Woody* & *Mia* is mapped to *beg_forgiveness_from*; all stories that end with this verb, and every action within those stories, will be rendered with A=*Woody* and B=*Mia*. In effect then, Woody is always destined to beg Mia for forgiveness, no matter how a story about them may begin.

While the number of terminating verbs is large, a reader may soon recognize the inevitability of tales with specific characters ending in foretold ways, so that e.g. Sam always marries Diane or Hillary always kills Bill. However, it is a simple matter to regularly regenerate S (once a week, say) and to randomly reassign characters to terminating verbs. Like the actors in a travelling repertory company, who may switch roles from one town or one production to another, the characters in our tales trade destinies with each other. When the new grammar that results from a new S is given to CBDQ, the bot's tales are given a new lease of life too.

Bicameral Bots

Our story above about a master and a rival barely squeaks under Twitter's newly enlarged 280-character tweet limit. To give a story room to breathe, a bot should ideally parcel it into an array of small episodes – say, one action apiece– and emit it as a threaded sequence of individual tweets, the way humans tend to use Twitter for fiction. But this kind of dismemberment would require planning and a global view of the story, and if a Tracery grammar lacks the memory to pass context between non-terminals, it certainly lacks the ability to pass control from one tweet to the next. However, CBDQ makes an interesting bicameral distinction in its use of grammars that offers bot-builders a nonobvious solution.

Twitter is more than a broadcast medium for the sharing of opinionated content; it is also a platform for interaction in which people relate to each other by replying to, and by commenting upon, each other's tweets. Our bots, likewise, are more than deaf generators. We often build these bots to respond to the provocations of others as much as to deliver automated provocations of their own. Buckenham's CBDQ thus allows *two* grammars to be specified for a Twitterbot: a core Tracery grammar, which, as we have seen, generates a bot's outputs on an agreed schedule, free of all influence from the outside world; and a simpler response grammar that allows a bot to reply directly to any mentions of its @ handle in the tweets of others. While also expressed in a JSON format, this second grammar is a not a fully-fledged piece of Tracery. Rather, it amounts to an ordered list of *stimulus:response* pairs: the stimulus is a literal string that any @ mention must contain before the response – a single Tracery expansion string, which may refer to non-terminals in the core Tracery grammar – is used to generate a reply. Suppose, for instance, that we want our Trump parody bot to produce poetry on demand. In response to a color from a user, the bot generates a poem around that color. Consider:

"[g|G]old": "#red_thing# are red, #gold_thing# are gold,
my #fan# #affirmation#, and #gold_rhyme#",

This response rule ensures that tweets to *@trumpScuttleBot* containing 'gold' or 'Gold' receive a response such as this:

Self-inflicted wounds are red,
Goldfinger's ladies are gold,
my local milk people are TREMENDOUS,
and are my special interests (I'm Sold!)

The bicameral parts of a bot can talk to each other in ways that are both simple and roundabout. An incoming tweet is matched to a stimulus in the response grammar, which then *talks* to the core grammar by invoking its non-terminals in the construction of its response. But the core grammar can only *talk* to the response grammar if it addresses its tweets to itself, by appending a mention of its own Twitter handle. Such mentions bring the outputs of the core grammar to the attention of the response grammar, which can then respond in kind, perhaps also appending a self-reference to ensure that the conversation between bicameral halves continues. While this conversation is carried on between its grammars the bot is basically, but quite productively, talking to itself.

The job of an automated story-teller is to spin a tale by talking to itself. To exploit CBDQ's basic bicamerality, we partition and reshape the story-telling grammar as follows. The core grammar again has a non-terminal/rule for every state V/E or E/E in the story graph S, but each right-hand-side expansion no longer contains recursive non-terminals. Instead, each expansion ends with the bot's Twitter handle. The main rule, *origin*, is responsible for generating just the first tweet of the story, a single action with a title, as in:

The 'Nanny' & The 'Child'
The story of how Lois supervised Kal's every effort
@BestOfBotWorlds

The trailing self-reference is later picked up by the bot's response grammar, as CBDQ is attuned to mentions of its bots (even by themselves) on Twitter. The tweet identifies the action verb *supervise* but does not explicitly identify the state in S, *supervise/are_arrested_for_killing*, to which it fully corresponds here. However, recall that the choice of characters Lois and Kal is a function of the very last action, so from their presence in this tweet the response grammar can recover this latent state in S. Here is the response rule:

"Lois supervised Kal": "#a# #*supervise/arrest_f_kill*#",

The non-terminal *a* is a shorthand that expands to the bot's own Twitter handle, which will prepend any bot response. The non-terminal *supervise/arrest_f_kill* is also defined in the core Tracery grammar, with a rule that ties together all the narrative consequences for the corresponding state in S. This would be laborious work if the grammars were hand-generated, as is the case for most Tracery/CBDQ bots, but these grammars are machine-generated. The tale continues and ends with the following self-addressed tweets:

@BestOfBotWorlds But Lois knowingly told lies for Kal

@BestOfBotWorlds Then Kal threatened to expose Lois's darkest secrets

@BestOfBotWorlds But Lois made a heartful appeal to Kal

@BestOfBotWorlds But Kal's insults struck Lois like poisoned darts

@BestOfBotWorlds And the police arrested Lois for her brutal attack on Kal
The End.

A trailing *The End* is provided by the rule for the end-state *are_arrested_for_killing/are_arrested_for_killing*, an end to which the response grammar knowingly does not reply.

The Blackboard Jungle

The generative grammar of a Tracery/CBDQ bot can only talk to itself by using Twitter as an intermediary, for only by posting tweets into its own public timeline can it pass those messages to the response part of its bot persona. This bicameral conversation uses Twitter as a *blackboard* onto which the bot reads and writes its data (Hayes-Roth, 1985; Veale & Cunningham, 1991). But Twitter is a very public blackboard, as well-suited to cooperation between relative strangers as it is between different parts of the same bot. A bot may thus delegate tasks or provide inspiration to others by sharing and appropriately addressing its ideas in public.

Consider a CBDQ bot, named *@MovieDreamBot*, which scours the category hierarchy of *DBpedia.org* to find ideas for its tweets. The bot targets fictional categories of films and books, exploiting the linguistic form of each to extract the key ideas underpinning a specific work. For example, *Blade Runner* (1982) is listed with the following categories at dbpedia.org/page/Blade_Runner: *flying_cars_in_fiction, climate_change_in_fiction*, *films_about_altered_memories* and *genetic_engineering_in_fiction*. If we now strip away the syntactic sugar, we are left with the themes *altered memories*, *genetic engineering*, *climate change* and *flying cars*. The bot chooses two themes to package into every tweet, of which the following is a representative example:

> *Influenced by the film 'Blade Runner,' I dreamt of amnesiacs who lose altered memories and drive flying cars*, @MovieDreamBot.

The bot combines fictional themes from *DBpedia.org* with propositional content from *Flux Capacitor*, to reason that even flying cars need chauffeurs, just as the memories that are lost by amnesiacs may have been altered in some way. Note how the bot addresses itself with an @ self-reference; this allows its response grammar to engage with each tweet and add some further creative value in the process. In this case the response grammar is designed to use a theme from the tweet as the basis for an automated story. Keying off of the term *amnesiacs*, the response grammar responds with:

> Hey @BestOfBotWorlds, spin us a yarn about how our amnesiac remembered this particular friend.

So the response grammar does not pass the ball back to its generative partner, but passes it onward to another bot, our story-telling CBDQ bot *@BestOfBotWorlds*. As it does so it shifts the emphasis from amnesiac to the *Scéalextric* verb *remember*, allowing the story-teller to reply with this tale:

The 'Freedom Fighter' & The 'Martyr'
The story of how Rick held on to memories of Ilsa
@BestOfBotWorlds

@BestOfBotWorlds And Ilsa filled Rick with inspiration

@BestOfBotWorlds So Rick heard wedding bells when looking at Ilsa

@BestOfBotWorlds But Rick made Ilsa sick to her stomach

@BestOfBotWorlds So Ilsa kicked Rick out into the cold

@BestOfBotWorlds So Rick whispered rumours behind Ilsa's back

@BestOfBotWorlds Then Rick hurled cruel taunts at Ilsa

@BestOfBotWorlds Then Ilsa rose up against Rick

@BestOfBotWorlds Then Ilsa toppled Rick from the top of the heap

@BestOfBotWorlds Yet Rick became a shining inspiration for Ilsa

@BestOfBotWorlds But Rick crucially underestimated Ilsa

@BestOfBotWorlds And Ilsa knew just how to manipulate Rick

@BestOfBotWorlds So Rick caught a bullet to save Ilsa
The End.

We humans throw ideas about on social media as though they were balls to be volleyed with great force and sliced with spin, and our bots should be able to do the same. For a story emerges from several distinct layers of interaction: the interplay of words and ideas, the interplay of teller and audience, and the interplay of fictional characters. Our bots and their grammars can interject themselves into each kind of interaction, to collectively create the Twitter equivalent of what Minsky (1986) called *the society of mind*. For even if each mindlessly executes a tiny task of its own, our bots can cumulatively give rise to surprisingly creative results.

West of Eden

Predestination is a recurring fictional trope that is found in movies, novels, TV shows and games. By comparison, the bicamerality of mind has remained the stuff of esoterica, at least until now. *Westworld*, a recent HBO television series, has put both ideas side-by-side in the popular imagination. Like the 1973 movie of the same name on which the series is based, *Westworld* is set in a Western-styled theme park where lifelike robotic "hosts" – in the guise of cowboys, barmaids, lawmen, thieves, farmers and cathouse madams – entertain paying guests with their highly scripted antics. *Westworld* is a technological marvel, overseen by operators who are as much story-tellers as roboticists or bureaucrats. Yet no matter how lifelike and conscious a host may seem, each is predestined to traverse the same narrative "loop" to reach its appointed fate. While each is given some latitude for improvisation within its loop, every host is always fated to be on time for its appointment in Samarra. The hosts are dancers that collectively glide through a highly-structured story-space, one that is regularly regenerated and rebooted with new loops, new fates, and new roles for old hosts.

In episode 3, season 1, the park's chief designers discuss the bicameral basis of the host's mental architectures. The younger designer sums up, and dismisses, Jayne's theory thusly: "the idea that primitive man believed his thoughts to be the voice of the gods, but I thought it was debunked," to which the older replies "as the theory for understanding the human mind perhaps, but *not* as a blueprint for building an artificial one." The distinction, so well articulated in this work of modern fiction, is one that has always been present in the field of Computational Creativity. Cognitive theories offer valuable insights into the working of creative systems but they need not always hold water for human cognition to be of practical value to the builders of artificial systems. In many ways the practitioners of CC are guided as much by stories as by theories. While theories come and go, a good story always retains its ability to inspire and to guide.

A subtle philosophical thread that is woven through the *Westworld* series is the possibility that the park's human operators are no more the possessors of a conscious mind and a soul than their robotic creations. Some hosts seem to be more human that their creators, while some guests, and some creators too, are stuck in loops of their own making. So to what extent is any CC system stuck in a loop of its designer's making, and what capacity does such a system have to transcend its programming? We all travel in ruts of society's making, improvising locally within loops that we cannot always see. Shouldn't our CC systems do likewise?

Our engagement with such themes in this paper has been largely superficial, focusing as we have on practical issues in the lightweight design of distributed CC systems. Yet our treatment of practical issues is still usefully informed by a consideration of more profound questions. Our model of automated story-generation aims to reconcile the loops with the improvisations to yield predestination with choice. As builders of these CC systems we become *meta*-tellers; for we build the story-spaces in which our hosts wander on unseen tracks, telling stories as they move around in loops. When a space has exhausted its potential to surprise, we regenerate it, with new paths and new character destinies.

Authorship in a Bottle

We have presented a number of new resources in this paper that are available for download by researchers. Bicameral grammars for our story-telling bots *@MovieDreamBot* and *@BestOfBotWorlds*, as well as for *@TrumpScuttleBot*, can be accessed via links on their Twitter pages or downloaded from the CC repository *github.com/prosecconetwork*. Code and data for the generation of these grammars, as well as knowledge representations for *Scéalextric* and the NOC list are also available from this repository. A forthcoming book on CC Twitterbots (Veale & Cook, 2018) offers greater detail on how these resources may be used by bot-builders.

Squeezing a fully functional story-teller into the expressive confines of a finite-state-machine, or even into the context-free grammars of *Tracery*, is the equivalent of squeezing a seaworthy ship into a bottle. Our focus in this paper has not been on improving the quality of the stories generated by *Scéalextric* or *Flux Capacitor*, even if we have shown how these two approaches can be unified to imbue stories with a greater sense of shape and completeness. Rather, we have focused on replicating the coherence and richness of stories from (Veale, 2017; Veale & Valitutti, 2017) with a more streamlined representation and a much reduced algorithmic complexity. We have shown how bots can use Twitter as a blackboard to distribute creative effort, and how we can squeeze the most from tools such as Tracery & CBDQ by automating the construction of grammars. The philosopher Daniel Dennett (2007:95) once remarked that "we have a soul, but its made of lots of tiny robots." We have set out to model nothing so grand as the soul here, except perhaps the soul of a new story-telling machine, made of tiny bots.

References

Campbell, J. (1949). *The Hero with a Thousand Faces.* Princeton: Princeton University Press.

Compton, K., Kybartas, B. & Mateas, M. (2015). *Tracery: An*

Author-Focused Generative Text Tool. In Proc. of International Conference on Interactive Digital Storytelling, Denmark:154-161.

Cunningham, P. & Veale, T. (1991). Organizational issues arising from the integration of the lexicon and concept network in a text understanding system. In Proc. of the 12th Int. Joint Conference on Artificial intelligence (IJCAI), Sydney, Australia, 981-986.

Dennett, D.. (2007). My body as a mind of its own. In D. Ross (Ed.), *Distributed cognition and the will: individual volition and social context*. Cambridge, MA: MIT Press.

Gervás, P. (2013). Propp's Morphology of the Folk Tale as a Grammar for Generation. In *Proceedings of the 2013 Workshop on Computational Models of Narrative*, Dagstuhl, Germany.

Gervás, P., Hervás, R., León, C. & Gale, C.V. (2016). Annotating Musical Theatre Plots on Narrative Structure and Emotional Content. In *Proc.* of the *7th International Workshop on Computational Models of Narrative*, Krakow, Poland.

Hayes-Roth. B. (1985). A blackboard architecture for control. *Artificial Intelligence*. **26** (3): 251–321.

Jaynes, J. (1976). The origins of consciousness in the breakdown of bicameral mind. Middlesex, UK: Penguin books.

McKee, R. (2010). *Story: Style, Structure, Substance, and the Principles of Screenwriting*. New York: Harper-Collins.

Meehan, J. (1981). *TALE-SPIN*. In Shank, R. C. and Riesbeck, C. K., (eds.), *Inside Computer Understanding: Five Programs plus Miniatures*. Hillsdale, NJ: Lawrence Erlbaum.

Minsky, M. (1986). *The Society of Mind*. Cambridge MA: MIT Press.

Pérez y Pérez, R. & Sharples, M (2004). Three Computer-Based Models of Storytelling: BRUTUS, MINSTREL and MEXICA. *Knowledge Based Systems Journal,* 17(1):15-29.

Propp, V. (1928/1968). *Morphology of the Folktale*. University of Texas Press (2nd edition; English translation by Laurence Scott).

Riedl, M. O. and Young, R. M. (2010). Narrative planning: balancing plot and character. *Journal of Artificial Intelligence Research* 39.1, 217-268.

Schank, R. & Abelson, R. (1977). *Scripts, Plans, Goals and Understanding*. Psychology Press: New York, NY.

Turner, S.R. (1994). *The Creative Process: A Computer Model of Storytelling*, Hillsdale, NJ: Lawrence Erlbaum.

Veale, T. (2014). Coming Good and Breaking Bad: Generating Transformative Character Arcs For Use in Compelling Stories. In *Proceedings of ICCC-2014, the 5th International Conference on Computational Creativity, Ljubljana, Slovenia, June 2014*.

Veale, T. (2016a). Round Up The Usual Suspects: Knowledge-Based Metaphor Generation. In Proceedings of *the Meta4NLP Workshop on Metaphor at NAACL-2016, the annual meeting of the North American Assoc. for Comp. Ling. San Diego, CA*.

Veale, T. (2016b). A Rap on the Knuckles and a Twist in the Tale: From Tweeting Affective Metaphors to Generating Stories with a Moral. In *Proceedings of the AAAI Spring Symposium on Ethical and Moral Considerations in Non-Human Agents.*

Veale, T. (2017). Déjà Vu All Over Again: On the Creative Value of Familiar Elements in the Telling of Original Tales. In Proc. of *ICCC 2017, the 8th Int. Conf. on Computational Creativity, Atlanta, Georgia, June 19-23*.

Veale, T. & Valitutti, A. (2017). Tweet dreams are made of this: Appropriate incongruity in the dreamwork of language. *LINGUA 197, 141--153.*

Veale, T. & Cook, M. (2018). *Twitterbots: Making Machines that Make Meaning*. Cambridge, MA: MIT Press (in press).

Vogler, S. (1984/1998). *The Writer's Journey: Mythic Structure For Writers*. Studio City, CA: Michael Wiese Productions (a book treatment of Vogler's original 7-page memo from 1984).

Yorke J. (2013). *Into the Woods: A Five-Act Journey into Story*. London, UK: Penguin.