Automated Extension of Narrative Planning Domains with Antonymic Operators

Julie Porteous, Alan Lindsay, Jonathon Read, Mark Truran and Marc Cavazza
Teesside University, Middlesbrough, UK
{j.porteous,a.lindsay,j.read,m.o.cavazza}@tees.ac.uk
mark.a.truran@gmail.com

Categories and Subject Descriptors

H.5.1 [**Multimedia Information Systems**]: Artificial, Augmented and Virtual Realities

General Terms

Algorithms, Performance

Keywords

Virtual Agents; Interactive Storytelling; Narrative Modelling; Planning

ABSTRACT

AI Planning has been widely used for narrative generation and the control of virtual actors in interactive storytelling. Planning models for such dynamic environments must include alternative actions which enable deviation away from a baseline storyline in order to generate multiple story variants and to be able to respond to changes that might be made to the story world. However, the actual creation of these domain models has been a largely empirical process with a lack of principled approaches to the definition of alternative actions. Our work has addressed this problem and in the paper we present a novel automated method for the generation of interactive narrative domain models from existing non-interactive versions. Central to this is the use of actions that are contrary to those forming the baseline plot within a principled mechanism for their semi-automatic production. It is important that such newly created domain content should still be human-readable and to this end labels for new actions and predicates are generated automatically using antonyms selected from a range of on-line lexical resources. Our approach is fully implemented in a prototype system and its potential demonstrated via both formal experimental evaluation and user evaluation of the generated action labels.

1. INTRODUCTION

Interactive Narrative systems are dynamic environments within which virtual agents act under the control of system generated storylines but where real-time interference with the ongoing story can impact on subsequent narrative unfolding. AI planning has been shown to be an appropriate technology for the task of generating stories in such systems [2, 29]. However the creation of the domain models for these planning based story generators raises an important practical domain modelling problem: how to author them to ensure they contain sufficient narrative content in order to produce alternative narratives to the baseline plot. Such alternatives can be part of either narrative generation (i.e. the production of multiple story variants) or interactive narrative in its strict sense (i.e. the system responding to dynamic changes made to the story world).

To illustrate this consider the examples generated using a domain based on the Aladdin folk tale as shown in Fig. 1 and 2 (we use this domain as a benchmark throughout the paper due to its familiarity, previous use in narrative research [29, 33] and suitable scope for experimentation). An initial narrative generated with a domain modelled around a baseline plot might proceed as shown down the left hand side of Fig. 1 with characters falling in love, casting love spells and marrying. However, for the system to produce multiple story variants, additional actions are needed for storylines that deviate from the baseline (right hand side of Fig 1). In addition, to be able to respond to dynamic changes to the story world and to continue the story through to the intended ending, alternative actions are required to generate an alternative course of action from that point onwards (Fig. 2).

To date the authoring of such interactive narrative domain models has been handled manually, a common strategy being to build up the model via systematic consideration of alternatives around a baseline plot [26] and it should be noted that many an Interactive Narrative (from Façade with "Who's Afraid of Virginia Woolf?" [19] to The Merchant of Venice [27]) has sought inspiration from existing literary or filmic work. However, this manual creation is extremely challenging and hence we were motivated to explore the automation of this process. Our starting point in this endeavour was to consider whether missing actions could be filled by generating actions that are opposites of the baseline ones and to explore this aspect more fundamentally from a formalism perspective.

We have developed an automated approach to the identification and creation of such actions, one which can be seen as automating a manual strategy of considering opposite actions and predicates, thus enabling the automated creation of alternate narrative content that departs from a linear baseline narrative. This approach is fully implemented in a prototype system, named ANTON, which takes as input a planning domain model, identifies missing contrary elements (section 3.2), and uses this to generate new content which is output in an extended version of the domain model (section 5). Since it is important that generated content is human-readable (e.g. to trace stories during development) and given that contrary relationships form the basis our approach to action generation, automatic labeling of new content in ANTON is based on *antonyms*, sourced from a range of linguistic resources (section 4).

The key contribution of this paper is a complete method to create

Appears in: *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Bordini, Elkind, Weiss, Yolum (eds.), May, 4–8, 2015, Istanbul, Turkey.*

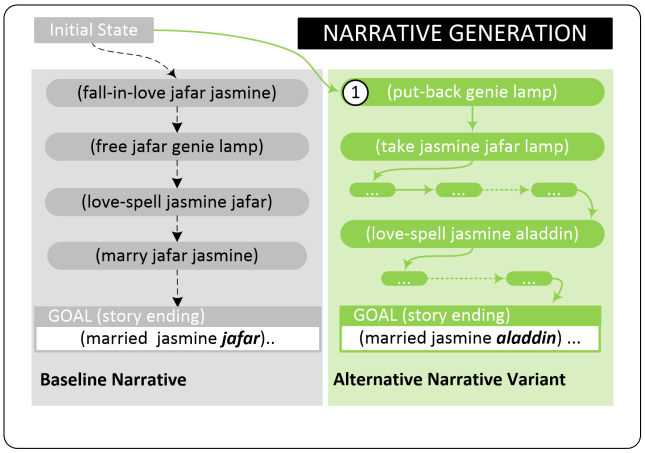

Figure 1: Example of Narrative Generation. The figure shows the process of narrative generation with an initial baseline narrative (down the left hand side) generated using a domain modelled around a baseline story. Also shown in the figure is the generation of an alternative narrative variant (down the right hand side) which demonstrates the need for additional narrative actions (here, this includes actions such as putting the genie back in the lamp, labelled (1), for a very different story ending with princess Jasmine married to Aladdin).

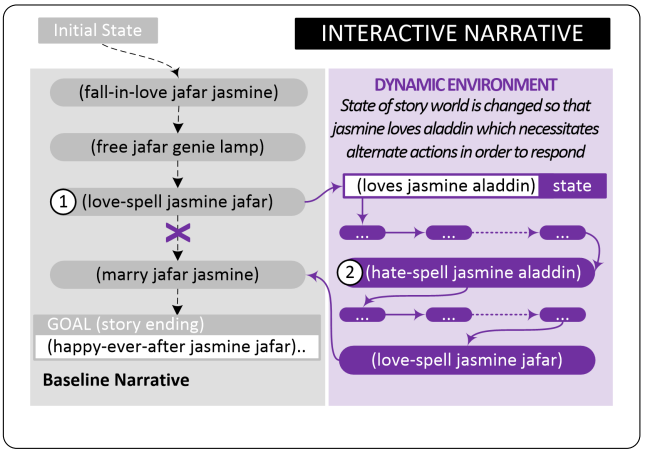

Figure 2: Example of Interactive Narrative. The figure illustrates the need for alternate actions in an interactive narrative in order to respond to dynamic changes made to the story world: here dynamic changes mean that the love spell (labelled (1) which should result in princess Jasmine falling in love with Jafar) goes wrong (with the result that Jasmine loves Aladdin) and consequently alternate actions are required to respond appropriately from that point onwards (such as casting a spell so that Jasmine thates Aladdin (2)).

alternative planning actions with automatically generated human-readable labels. We also include a user evaluation of the appropriateness of the labels and the believability of the actions, along with a quantitative evaluation of narrative generation.

2. RELATED WORK

Automated domain model creation is a topic of current interest in the AI planning community, in part due to increasing awareness that building domain models is challenging, time consuming and an obstacle to the further fielded application of planning technology [36]. Recent work in this area has been aimed at learning planner action models from correctly observed plan traces (e.g. [1, 5, 37]). However, this work has limited application for narrative domains which do not share the same consistency and alignment with real-world domains as do more traditional planning domains such as "logistics" or "rovers".

Similarly, in the area of narrative generation there has been increasing work aimed at automated creation of narrative content. A popular approach has been the gathering of story elements via crowdsourcing, an approach which can yield abundant content. For example, the SCHEHERAZADE system of [16] employ this approach to acquire typical story elements which can be assembled as plot graphs and used in a process of story generation. With SCENARIO-GEN [31] crowdsourcing was used to gather a database of scenarios of everyday activities and likely replacements for use within a serious game context. In [23] a crowdsourcing approach was used for the hand annotation of logs from user sessions with the *Restaurant Game* for subsequent use in automating character interactions with human participants in a dialogue-based narrative setting. An alternate to crowdsourcing aims to obtain narrative content through mining of weblogs and story corpora: the SAYANYTHING system of [32] selects narrative content on-the-fly from a corpora of weblogs in response to user text-based interaction; whilst the approach in [21] attempts to generate narratives using knowledge mined from story corpora for a particular genre. Our automated mechanism for diversification of narrative planning actions through antonymy differs from these approaches in that it aims to discover alternatives around baseline plots (which themselves could be drawn from existing linear narratives).

Another area of related work to our antonym-based approach is that aimed at the computational creation of content for use in applications requiring narrative generation. Examples of this include the use of conceptual blending in the creation of novel artefacts for use in narrative generation [10,17] and visual narrative generation [35].

3. DOMAIN ANALYSIS

We adopt a rather traditional framework, albeit rarely presented as such, in which a narrative domain is developed around a baseline plot often inspired from a well-known literary or filmic work. This model is used in an initial phase of domain analysis for which we assume a classical approach (i.e. in the tradition of STRIPS [7]) where actions in the model are represented using parameterised pre- and post-conditions: sets of facts describing properties (or partial states) of the story world. To illustrate the process of domain analysis we use an encoding of the Aladdin story and a selection of actions from this are shown in Fig 3 (this is based on the model in [29] modelled using PDDL2.1 augmented with object type hierarchy and parameterised operator schemas [20]).

Given this form of domain model, the narrative actions can be seen as specifying the properties that different types of domain objects can occupy and the ways in which these can be changed. The contrary is also true: the narrative actions specify those properties that cannot be changed and those transitions that cannot be reversed without compromising the narrative experience (this of course is strongly dependent on narrative genre). For example, in the Aladdin story (as specified in the actions in Fig. 3), genies can be trapped in a magic lamp (as opposed to an ordinary one) and can be either trapped or freed (by being summoned from the lamp).

```
(:action fall-in-love :parameters (?m - male ?p - princess ?l - location)
   :precondition (and (at ?m ?l) (at ?p ?l) (single ?m) (alive ?m)
      (alive ?p) (not (loves ?m ?p)) (not (loves ?p ?m)) (beautiful ?p))
   :effect (and (loves ?m ?p)))
(:action summon :parameters (?p - person ?g - genie ?t - thing)
   :precondition (and (confined ?g) (magic ?t) (has ?p ?t) (at ?p ?l))
   :effect (and (at ?g ?l) (controls ?p ?g) (not (confined ?g))))
(:action love-spell :parameters (?g - genie ?p1 ?p2 - person)
   :precondition (and (not (confined ?g)) (not (loves ?p1 ?p2))
      (alive ?g) (alive ?p1) (alive ?p2))
   :effect (and (loves ?p1 ?p2)))
(:action marry :parameters (?m - male ?p - princess ?l - location)
   :precondition (and (alive ?m) (alive ?p) (at ?m ?l) (at ?p ?l)
      (loves ?m ?p) (loves ?p ?m) (single ?m) (single ?p))
   :effect (and (married ?m ?p) (married ?p ?m)
      (not (single ?m)) (not (single ?p))))
(:action slay :parameters (?k - knight ?m - monster ?l - location)
   :precondition (and (at ?k ?l) (at ?m ?) (alive ?k) (alive ?m))
   :effect (and (not (alive ?m))))
```

```
(:types  agent location thing   - object
         person monster         - agent
         princess male          - person
         knight king            - male
         genie dragon           - monster
```

Figure 3: Selected actions and types from the Aladdin Domain used as illustration through the paper (adapted from [29]). Actions specify ways that the story world can change e.g. a male can fall-in-love with a princess as long as certain things are true in the story world at that point, such as her being beautiful. Types form a hierarchy which plays a role in the propagation of conditions e.g. a knight can slay a monster (genie or dragon) but only genies can be summoned when confined in a lamp.

Princesses are beautiful and can be either single or married. However different variants of the Aladdin story require contrary properties and transitions to allow for such things as ugly princesses becoming beautiful, married princesses becoming single and genies being put back into magic lamps.

These contrary properties and transitions are important since they may be required in order to continue the presentation of an ongoing narrative in a dynamic environment, as illustrated in the example in Fig. 1. Hence our approach to content creation seeks to identify core narrative elements in the input domain model whose negation impacts story progression (section 3.1) and then to use these as candidate transitions for content creation (section 3.2).

3.1 State Transition Analysis

Starting with an input domain model a set of state transition rules are constructed which represent the partial state transitions that are possible for each of the different types of objects in the model. This is related to the identification of Finite State Machines and transition rules with TIM [8] and analysis of Domain Transition Graphs in FAST DOWNWARD [11]. Building on [8], the transition rules are specified in terms of *properties*, where a property is a predicate subscripted by a number between 1 and the arity of the predicate, so that every predicate of arity n defines n properties. For typed PDDL domains we extend this to refer to the type of a property: for a property with subscript i its type corresponds to the type of the argument in predicate position i. For example, *(controls ?p - person ?g - genie)*, defines the properties $controls_1$ and $controls_2$, of type *person* and *genie*.

Type	Action	State Transition Rules: $E \Rightarrow S \rightarrow F$					
person	love-spell	$alive_1$	$\Rightarrow$	$\neg loves_1$	$\rightarrow$	$loves_1$	1
		$alive_2$	$\Rightarrow$	$\neg loves_2$	$\rightarrow$	$loves_2$	2
	summon	has_2 at_1	$\Rightarrow$	$\neg controls_1$	$\rightarrow$	$controls_1$	3
monster	slay	at_1	$\Rightarrow$	$alive_1$	$\rightarrow$	$\neg alive_1$	4
male	marry	$loves_1$, $loves_2$, ..	$\Rightarrow$	$single_1$	$\rightarrow$	$married_1$ $married_2$	5
	fall-in-love	$alive_1$	$\Rightarrow$	$\neg loves_1$	$\rightarrow$	$loves_1$	6
princess	marry	$loves_2$ $loves_1$...	$\Rightarrow$	$single_1$	$\rightarrow$	$married_2$ $married_1$	7
	fall-in-love	$beautiful_1$ at_1 $alive_1$	$\Rightarrow$	$\neg loves_2$	$\rightarrow$	$loves_2$	8
genie	summon	$alive_1$	$\Rightarrow$	in_1 $confined_1$	$\rightarrow$	$controls_2$ at_1	9

Figure 4: Example State Transition Rules for the Aladdin Domain. Rules represent how the effects of narrative actions can change the partial states of different types of objects. Rules take the form $E \Rightarrow S \rightarrow F$ where: E is a set of properties that enable the transition: S are the properties given up by the transition; and F are properties acquired by the transition. The absence of a property x is shown as $\neg x$. As an example, for the type *person* the action *summon* requires that the person initially *has* (a lamp in which the genie is confined) and is *at* (a location). Note that this transition is possible for all sub-types of type *person* (i.e. the types *knight, king* and *princess*).

Transition rules take the form $E \Rightarrow S \rightarrow F$ which is read *"E enables the transition from S to F"* and where the three components are bags of zero or more properties built for each action in the domain model and for each type of object specified in the parameters of that action. The rules are constructed as follows:

E (enablers)	those pre-condition properties that remain *unchanged* by the action
S (start)	those pre-condition properties that are *deleted* (in other words *lost*) by the action
F (finish)	properties that are *achieved (gained)* by the action

Note that rules can contain empty components, however any transitions where both *S* and *F* are empty are ignored since they don't describe any change of state for that type of object.

As illustration, consider the Aladdin domain (Fig 3), and the identification of transition rules for the action *love-spell*. The action parameters ?g - genie ?p1 ?p2 - person are considered as follows:

?g The action contains only the property $confined_1$ of type *genie*. This is unchanged by the action and is added to *E*. There are no other properties hence *S* and *F* are empty. Since this transition rule is empty (does not describe a state change as described above), it is discarded.

?p1 The properties $alive_1$, $\neg loves_1$ and $loves_1$ are of type *person*. The property $alive_1$ is required and not changed by the action and is added to *E*, $\neg loves_1$ is deleted by the action and added to *S*, and $loves_1$ is achieved by the transition so is added to *F*. The resulting rule is:

$$alive_1 \Rightarrow \neg loves_1 \rightarrow loves_1$$

(this rule is shown numbered 1 in Fig. 4)

?p2 The properties $alive_2$ (enables), $\neg loves_2$ (deleted) and $loves_2$ (achieved) are identified and added to *E*, *S* and *F*. The resulting rule is output:

$$alive_2 \Rightarrow \neg loves_2 \rightarrow loves_2$$

(this rule is shown numbered 2 in Fig. 4)

Hence the analysis of the action *love-spell* yields 2 transition rules for type *person* and none for type *genie* (these rules are numbered 1 and 2 in Fig. 4 which lists all the transition rules constructed for the selected actions shown in Fig. 3).

3.2 Identification of Core Transitions

Once constructed the transition rules are searched to identify contrary transitions and properties which suggest core candidate transitions for action generation. We consider each of these in turn.

3.2.1 Candidate Actions from Contrary Transitions

For any type of object for which a transition is specified in the rules we postulate that the contrary transition has a natural interpretation in the domain and that we would also expect to find it in the set of transitions. Hence, any transition whose contrary does not appear in the set of rules is proposed as a candidate for alternative action generation.

As an example from the Aladdin domain, consider transition rules 1 and 2 for action *love-spell* and rule 6 for *fall-in-love* shown in Fig. 4. For any *person* (or sub-type: *king, knight, princess*) the transition $\neg loves \rightarrow loves$[1] can be made via *love-spell* and in addition for any *male* (or sub-type *king, knight*) the transition can also be made via the action *fall-in-love*. However for all these types of objects the contrary transition *loves* $\rightarrow \neg$*loves* is missing, i.e. there is no way for a love spell to be undone or for someone to fall out of love. Hence the actions *fall-in-love* and *love-spell* are flagged as candidates for alternative action generation.

All of the transition rules are traversed and the names of any actions for which contrary transitions are missing are added to the set of candidates for generation. For the rules in Fig. 4 the set of candidates after this analysis is:

{love-spell, summon, slay, marry, fall-in-love}.

3.2.2 Candidate Actions from Contrary Properties

The rationale for identifying these candidates is based on consideration of possible causes of failure of narrative plans when intended action execution is affected by changes in the environment. From the transition rules shown in Fig 4 we observe that some properties are only ever required as enablers and that such properties cannot be changed by the effects of narrative actions (examples of this are beautiful_1 for type princess and alive_1 for person). For such properties, we postulate that their contrary has a natural interpretation in the domain and hence propose the gaining and losing of such properties as candidate actions. The natural validity of such transitions becomes more apparent if one considers typical fairy tale situations in which characters are killed or forced into eternal sleep, or their appearance is changed.

For the rules in Fig. 4, the properties $alive_1$ and $beautiful_1$ are identified. Hence the set of candidates added after this analysis is:

{gain(alive), lose(alive), gain(beautiful), lose(beautiful)}.

4. ANTONYMIC LABELING

In order to ensure human readability of the automatically generated PDDL structures we generate labels for new domain content based on antonyms of the names of actions and properties in the original domain. The use of antonyms is justified since candidate actions specify transitions between partial states that represent opposition such as, (*married, single*) or (*beautiful*, ¬*beautiful*) and these are used as part of a heuristic approach to action generation which requires transition to a partial state to include the opposite state as a pre-condition (this is discussed further in section 5).

[1]For brevity we omit the numeric subscript on the property, however this example applies equally to both $loves_1$ and $loves_2$.

Linguistic Resources	Antonyms for "marry"		
	dissociate	**divorce**	separate
Merriam-Webster	1	**2**	1
BigHuge Thesaurus	0	**0**	0
Power Thesaurus	1	**3**	1
Totals	2	**5**	2

Figure 5: Antonym selection for the action marry (only those returned by multiple providers are shown). The weights for each antonym are listed below it. They are summed and the highest ranked is selected. In this case *divorce* is selected.

Labels are generated using antonyms drawn from a range of publicly available lexical resources (as in [18, 34]): multiple resources are used since robustness has been shown to be improved when using this strategy [22]. In our experiments we used the following online lexical resources: WordNet (WN) [6] for synset expansion of queried words, Merriam-Webster (MW), BigHuge Thesaurus (BT), and PowerThesaurus (PT)[2].

Antonyms for Single Words

To generate antonyms, ANTON sends strings, such as a verb representing an action name, to each of the linguistic resources. The output of these differs slightly: WN organises words into synonym groups, called Synsets, which ANTON uses initially to expand the queried word; MW and BT are organised by definition and return an HTML page with tagged content which ANTON parses, using the tags, to extract the antonyms for each definition of the expanded query; PT provides a page specially for the antonyms of words and lists these with an associated score based on the frequency of use of the antonym in texts and is updated to reflect user opinion.

ANTON maps the response of each provider into a list of candidate antonyms with associated integer values. This value signifies the likelihood of the candidate being a desired antonym. For MW, BT and WN this value is based on the number of definitions that are antonymic with the candidate. For PT the value is a rating of the general use of the candidate in texts. The weights of the returned candidates are summed to obtain a single weight for each candidate, ignoring sources that did not list it (a voting strategy) to get a measure of the likelihood that the word is a strong candidate. The candidate(s) with the highest weight is selected to be used for the label. Any ties are broken by ordering the candidates alphabetically and selecting the first to ensure deterministic selection. As an illustration, Fig. 5 shows the process of generating a label for the contrary action to *marry*, which in this case is *divorce*. We note that divorce is only the contrary of marry if the pair are already married, otherwise the contrary would be something like "refuse to marry" or "not marry" (consider Beauty and the Beast). Our heuristic approach, where transition to a state requires the opposite state as a pre-condition, justifies the use of antonymy. Relaxing this assumption could form the basis of additional action creation but is beyond the scope of the work presented in this paper.

[2]These resources are available on-line as follows:
WordNet: `https://wordnet.princeton.edu`
Merriam-Webster: `http://www.dictionaryapi.com`
Big Huge Thesaurus: `http://words.bighugelabs.com`
Power Thesaurus: `http://www.powerthesaurus.org`

Antonymic naming with ANTON offers the possibility of replacing predicates that appear as negative pre-conditions with new predicates that exactly complement their positive use, as in [9]. For example, consider the negated predicate that appears in the pre-conditions of the action *become-beautiful* in Fig. 6. If this translation were to be applied to the domain then this would be replaced with the new predicate so that the action pre-condition becomes *(ugly ?p-princess)*. Also, the post-conditions require extension to handle the gaining and losing of the property, as follows:

(:action **become-beautiful** :parameters (?p - princess)
:precondition (and (ugly ?p))
:effect (and (beautiful ?p)) (not (ugly ?p))))

It may be that the addition of predicates via this translation could further enhance domain model surveyability.

Antonyms for Multiword Expressions

Most of the actions and properties to be renamed in the Aladdin domain are single words with the exception of *fall-in-love* and *love-spell* which have been named using multiword expressions. Dictionary providers perform poorly with multiword expressions [30] and hence these expressions require additional processing.

To find antonyms of multiword expressions we pair the individual words with their antonyms as predicted by the method described above (except for the case of prepositions, whose candidates we lookup in a small lexicon). For example, hate-spell yields [(love, hate), (spell, unspell)]. Candidate antonymic expressions are then generated by enumerating the combinations other than the original (e.g. love-unspell, hate-spell, hate-unspell). The most likely candidate is selected by estimating probabilities with an n-gram language model (BerkleyLM; [25]) trained on text extracted from weblogs. In this example the model estimates that hate-spell is most likely ($P = 3.1 \times 10^{-5}$) while love-unspell and hate-unspell are both least likely ($P = 1.0 \times 10^{-100}$). Hence *hate-spell* is returned as the antonym for the multiword expression *love-spell*.

5. GENERATING NEW ACTIONS

Actions from Contrary Transitions

For actions representing "missing" contrary transitions new actions are generated from the original action named in the set of candidates (as discussed in section 3.2.1). Here we illustrate this process with reference to the Aladdin action *marry* and the generation of its contrary, as shown in Fig. 6, which illustrates the process of extending the set of default actions to create potential for generating new stories. For the name of the new action the ANTON generated label is used: in this case the string *marry* yields the label *divorce*. The parameters for the new action are the same as for the original action. The rationale for this being that the same objects will participate in the action and the resulting transitions.

There are two aspects to the construction of the action pre-conditions. Firstly the set of predicates which are *achieved* by the original action are added to the pre-conditions of the new contrary action: recall from the discussion in section 4 that this heuristic approach justifies the use of antonymy. For the action *divorce* these are the predicates which require that the couple are already married, labelled (A) in Fig. 6. The second aspect of the pre-condition construction is the inclusion of any *enabling* conditions required for the transition, however the precise nature of these condition(s) is unclear: for example, are they the same or different to those for the original action? The Aladdin domain exhibits instances of both possibilities: the enabling condition for *divorce* is unlikely to be the same as for *marry* whereas it may well be the case that the same enablers might be required for *summon*-ing the genie from the lamp and putting it back in. Our solution, for the purposes of experimental evaluation and automation of our approach, was to introduce a generic enabling condition capturing the requirement of the necessary motivation to participate in the action formed by prefixing the ANTON generated action name with *motivated-to* and introducing an additional action to ensure the enabling condition can be achieved (as illustration see the predicate *motivated-to-divorce* in Fig. 6).

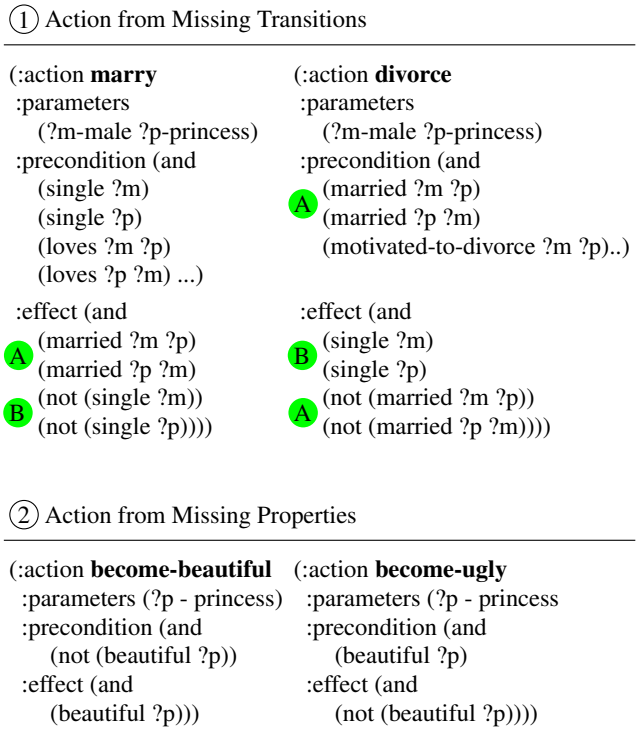

Figure 6: Example of Action Generation for candidate actions. ① For Missing Transitions new actions are generated using the original action as a template with original action (marry) and newly generated action (divorce); the name is generated using ANTON naming; the pre-conditions are those predicates achieved by the action A plus an enabling condition; the positive effects are the pre-conditions of the initial action B. ② For Missing Properties new actions are generated for: gaining the property, *become-beautiful*; and gaining the contrary property (i.e. losing the property), which is named using ANTON, in this case the action *become-ugly*. For more detail refer to text.

The post-conditions for the new action come directly from the original action: the positive post-conditions are the predicates that are *deleted* by the original action (e.g. *(single ?m)* is deleted by *marry* and achieved by *divorce*); and the negative post-conditions are the predicates that are *achieved* by the original (e.g. *(married ?m ?p)* is achieved by *marry* and hence deleted by *divorce*).

Actions from Contrary Properties

For candidate actions that represent the gaining or losing of a property then new actions are generated to enable these transitions. Here we illustrate this process for the candidate actions to gain and lose the property *beautiful* (discussed earlier in section 3.2.2).

For actions representing gaining a property the name is the property label prefixed with *become* and a single parameter variable of the type of the property. The action pre-condition is the negation of

Outline Algorithm: simulation of narrative continuation

Input: actions A, initial state I, goal G

```
1   current state S = I
2   N = generate_narrative(S, G)
3   if N = { } then exit(fail)
4     for i = 1...n do
5       current action a = N_i
6       if simulate_change then
            S = random_change(S, a)
7       if apply(S, a) = fail then
            goto line 2
8       S = apply(S, a)
9   exit(success)
```

Figure 7: Simulation to assess Narrative Continuation. Actions in generated narratives are considered in turn (lines 4-8), with non-deterministic state changes simulating dynamic environment (line 6). Termination with success at narrative end or fail if unable to continue after changes. See text for detail.

the property, since this action represents the transition from a partial state where the property is absent. Since the property is being gained, there is a single positive post-condition, the property itself. This is illustrated for action *become-beautiful* in Fig. 6.

For actions representing losing a property (or the gaining of the contrary property) the name is generated from the ANTON label prefixed with *become*, there is a single parameter variable of the appropriate type for the property, the pre-condition is the original property and there is a single negative post-condition representing the losing of the property (as illustration, action *become-ugly* is shown in Fig. 6).

As discussed earlier (section 4) the generation of labels for created content affords the possibility to replace predicates that appear as negative action pre-conditions with new predicates that exactly complement their use. For example, if the contrary predicate *(ugly ?p-princess)* was introduced to the domain model then this could replace the negative pre-condition *(not (beautiful ?p))* in the action *become-beautiful*. In addition the negative effects of the action would need to be extended to include *(not (ugly ?p))* reflecting the fact that this property is lost as a result of the transition.

6. EVALUATION

The objectives of the evaluation were to show that our method for domain content creation provided support for dynamic story environments and that generated content was surveyable and made sense. To this end a series of experiments were conducted on our original encoding of the Aladdin domain and an extended version of the domain model generated using our prototype, ANTON. Note that experiments using other example narrative domains, including Othello [3], Red Riding Hood [28] and Basketball [14], show consistent increase in model size and support our intuition that the approach is applicable across narrative domains. These results are not reported here due to space restrictions.

6.1 Support for Narrative Continuation

Increase in number of reachable states

Since the addition of contrary transitions by ANTON introduces the possibility of decisions being changed (e.g. putting the genie back in the lamp and therefore able to come under the control of another), we hypothesised that this would increase the number of

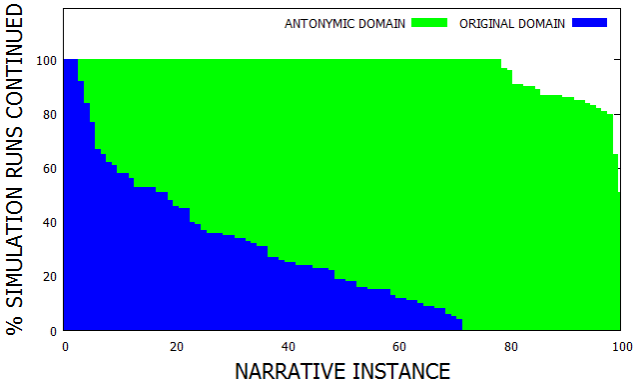

Figure 8: Results of Dynamic Environment Simulation: for 100 narrative planning instances (where a narrative could be generated using the original Aladdin domain) the figure plots the % of runs where the narrative could be continued through to the end after non-deterministic changes (y-axis), against each of the planning instances (x-axis). Results for the original domain ■ are plotted over the results for the antonymic domain ■ to enable visual comparison: with the antonymic domain all narratives are completed for 78% of instances in contrast to 2% for the original domain. Refer to text for further detail.

reachable states and hence open up new narrative possibilities. To assess the extent of this we carried out an analysis on the increase in number of reachable states. We used a canonical problem, based on the example in [29] and counted the number of reachable states in: (i) the original domain, $\mathbf{D}$; (ii) an extended domain $\mathbf{D_T}$ formed from $\mathbf{D}$ extended with actions generated on the basis of missing contrary transitions (section 5); and (iii) domain $\mathbf{D_{T+P}}$ formed from $\mathbf{D_T}$ extended with actions generated on the basis of missing contrary properties (section 5). Note that for domains $\mathbf{D_T}$ and $\mathbf{D_{T+P}}$ actions that achieved enabling conditions were excluded to avoid influencing the count. The counts for these models were:

Original Domain	Extended Domains	
(i) $\mathbf{D}$	(ii) $\mathbf{D_T}$	(iii) $\mathbf{D_{T+P}}$
9.22×10^4	9.44×10^7	3.26×10^{11}

As hypothesised these counts show that the size of the state space increases with the extension of the domain to include those elements identified as "missing". This results from the introduction of new actions via ANTON with the consequent increase in the number of narrative world states that can be reached: with the original domain model the absence of contrary elements made some states unreachable, for example because changes to the narrative world during an ongoing narrative could not be undone.

Experiment I: Dynamic Environment Simulation

To explore the ability of ANTON extended domain models to support the continuation of an ongoing story through to its desired ending in a dynamically changing environment we conducted a series of simulations. The simulation was designed to mirror the operation of a dynamic system where the current state of a story world can be changed at any point during the presentation of a narrative (e.g. with actions presented to the user as a text-based story as in [29]) with the frequency and positioning of these changes, and the resulting impact on the story world being determined at random.

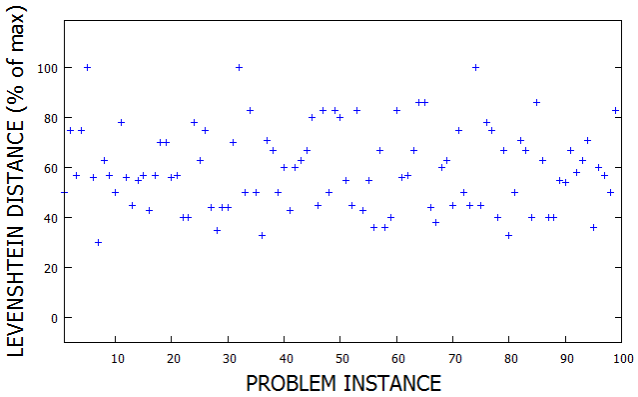

Figure 9: Narrative Diversity Experiment: for 100 narrative instances a narrative was generated with the original domain and the antonymic domain was used to generate a set of diverse narratives [4]. The plot shows the distance (as % of the maximum possible) between the original and the most diverse antonymic narrative: the distance between narratives was measured using Levenshtein distance[4]. An important result was that with the antonymic domain diverse narratives could be generated for all instances, with average at 59.58%, whereas it was not possible to generate any diverse narratives with the original domain model. For further detail see text.

The simulation was run 100 times for 100 randomly created narrative instances (for which narratives could be generated with the original domain). Each run of the simulation proceeds as shown in Fig. 7. For an input narrative instance an initial narrative is generated, using METRIC-FF[3] and each action in the narrative plan is considered in turn (line 4). If the current action a is selected (boolean $simulate\_change$), then the current state S is updated so a random selected set of pre-conditions of a are no longer satisfied ($random\_change$, line 6). For example, if the action required a person to be single, then the random change might be that they are already married or if a character was required to be at a particular location then this might be changed. Generation of the changes in this way, ensures they are filtered on the basis of the most likely affordances: the restriction to pre-conditions of planned narrative actions ensures that changes are via predicates attached to the states of narrative objects that are central to the current planning instance. The simulation continues and tries to apply the current action a in the current state S (line 7). If this fails (the case when random state changes have been made) the simulation restarts (line 2), regenerates the remainder of the narrative from the current state with the *original* narrative goal G. Whenever the system fails to generate a narrative the system reports failure and exits.

The results of the simulation are summarised in Fig. 8 and clearly show the improved performance of the ANTON domain model over the original domain with respect to the ability to continue a narrative towards its desired ending following dynamic changes. For the extended domain model, in 78 out of the 100 problem instances, it is possible to complete the original story for all 100 simulation runs. In addition, for those narrative instances where the narrative completion rate is less than 100%, the completion rate with the extended domain model still averages 60.52%. The figures for the original domain model are very different: in only 2 cases can it continue the narrative on all of the simulation runs and for the remaining 98% of runs the average is only 24.4% of narratives that can be continued through to the original goal.

[3]METRIC-FF: http://fai.cs.uni-saarland.de/hoffmann/metric-ff.html

Transitions		Properties	
Original	**Antonym**	**Original**	**Antonym**
give ✓	take	beautiful ✓	ugly
summon ✓	dismiss	alive ✓	dead
marry ✓	divorce	loyal ✓	disloyal
love-spell ✓	hate-spell		
fall-in-love ✓	fall-out-of-love		
slay ○	restore	magic ○	commonplace
pillage ○	protect		
order ○	disorder		
command ○	ask		
		scary ✗	bold

Figure 10: Automatically Generated Antonyms for the Aladdin domain. The names of antonyms for Transitions and Properties from the original domain were shown to users and they were asked to classify them as either correct ✓, partial ○ or incorrect ✗. From these results we conclude that the method generates appropriate labels for use as names for the ANTON generated narrative domain content.

When the simulation scores are averaged over the sample the number that complete to the story ending for the ANTON domain is 91.2% in comparison to 25.5% for the original. Given the limited extent to which the original Aladdin domain accepts interaction the support leveraged by the ANTON created content is considerable.

Experiment II: Narrative Diversity

Since the contrary transitions generated by ANTON introduce the possibility of decisions being changed (e.g. putting the genie back in the lamp and enabling another agent to take control), we hypothesised that this would increase the diversity of narrative generation i.e. for narrative instances where narratives could be generated using the original domain, the ANTON domain model would also be able to generate further *different* narratives. We note that the use of distance measures in this way as part of narrative evaluation is consistent with a growing trend in narrative research (e.g. [12]).

To demonstrate this we used a set of 100 random narrative planning instances for which a narrative could be generated using the original domain model and METRIC-FF[3]. For each instance a narrative was generated using the original domain model and a set of diverse narratives were generated with the ANTON extended domain adopting the approach of [4]. Then the original narrative was compared with the diverse set of narratives to find the ANTON narrative that was most different to the original. For comparison the difference between narratives was measured using Levenshtein Distance [13, 15] which counts the edit distance[4] between two strings. To obtain suitable strings for comparison, action names in the narratives were mapped to unique characters.

The results of the narrative comparisons are plotted in Fig. 9: with difference shown as a percent of the maximum possible between the original domain narrative and the most different ANTON domain narrative. The results clearly demonstrate that the extended domain increases the diversity of narratives that can be generated

[4]Levenshtein distance [15]: this is a count of the minimum number of editing operations needed to transform one string so that it is identical to another. This was selected since it is can be applied to measure the distance between strings of different length which is important for comparison of plans. The maximum possible distance between 2 strings is the length of shortest string.

since for all 100 test problems diverse sets of plans could be generated using the ANTON extended domain model, with an average distance of 59.59% of maximum.

Another important result is that *no* diverse plans could be generated with the original domain. This is important since the increased generativity resulting from use of the ANTON extended domain did not require the considerable effort of manual domain creation.

6.2 User Evaluation of Readability and Believability

In addition to support for narrative continuation we also conducted an evaluation of the created content to assess both the appropriateness of the antonymic labels that were attached to domain content and also the sense of the ANTON generated actions. This evaluation consisted of a series of questions which were put to a sample of 20 native English speaking adults (average age 37 $\pm$ 13.38) as discussed in the next sections.

Appropriateness of Antonyms

Our hypothesis was that antonyms would provide a means to generate appropriate names for actions. Fig. 10 lists the antonyms generated by ANTON for new actions in the extended Aladdin domain. To assess the appropriateness of the ANTON antonyms the sample of subjects participating in the study were asked to classify them as either correct, partially correct or incorrect (using the categories from [22]). Note that in the absence of any recognised method for classification and given there is less agreement on goodness of antonymy than other word categories [24] we use this classification for discussion only. Also, where subjects classified an antonym as partial or incorrect, they were asked to suggest a better antonym.

From the figure it can be seen that, of the 14 antonyms, the users judged 8 antonyms to be correct, 5 partially correct and 1 incorrect. Interestingly, for those which users classified as partial or incorrect they were either unable to suggest a better antonym or no clearly preferred antonym emerged. From this we conclude that our method generated appropriate names for the created content.

Believability of Generated Actions

To assess whether created content appears sensible, the same group of subjects were asked about a series of text translations of the ANTON created domain content. The text was generated from templates which were based on the post-conditions of the action, with suitable object instantiations of parameters.

For example, the action *(dismiss ?p-person ?g-genie ?t-thing)* with suitable instantiation of variables is translated as follows:

Action	*(dismiss ?p - person ?g - genie ?t - thing)*
Instantiation	*?p=Jafar, ?g= Genie, ?t=lamp*
Translation	*"Jafar put the Genie back into the lamp"*

and the action *(become-disloyal ?k - knight)* similarly as follows:

Action	*(become-disloyal ?k - knight)*
Instantiation	*?k=Aladdin*
Translation	*"Aladdin stopped being a loyal knight."*

In the study, translations of all of the ANTON generated actions were shown to the group of subjects and they were asked to rate how much "sense" they made and whether they were the sort of thing that could happen in Aladdin stories. Fig. 11 summarises the results (labels on the x-axis refer to the names of actions for which a text translation was shown). Our hypothesis was that ANTON generated actions have a natural interpretation in the narrative domain since they represent transitions and properties that exactly complement those already present. The results support this, with subjects ranking actions as making sense in 75% of cases.

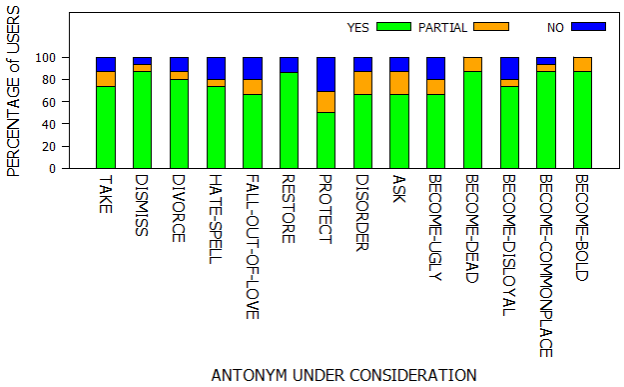

Figure 11: Results of user evaluation: users were asked how far the ANTON generated actions made sense to them and to rank this as either Yes ■, Partial ■ or No ■ (the labels on the x-axis are the names of actions for which users were shown a text-translation). The results are encouraging and indicate that content was largely judged to be sensible. See text for detail.

7. CONCLUSION

In this paper we have introduced a novel approach to extending a narrative domain model based on identification of missing transitions in the domain. In the evaluation we demonstrated that this approach can increase both the range and diversity of narratives that can be generated. We also presented the results of user evaluation of the generated content from which we conclude that this made sense to the users and also that the system generated labels were appropriate.

Overall these results clearly demonstrate the potential of the approach to increase the range and diversity of stories that can be generated whilst still maintaining the human-readability of the domain model. For interactive media applications this is an important result which represents considerable increase in generativity without the substantial effort required to achieve the same result by manual domain generation.

Acknowledgments

This work was funded in part by the European Commission through the FP7 Open FET "MUSE" Project (ICT-296703).

8. REFERENCES

[1] E. Amir and A. Chang. Learning partially observable deterministic action models. *Journal of Artificial Intelligence Research*, 33(1):349–402, Nov. 2008.

[2] R. Aylett, J. Dias, and A. Paiva. An Affectively Driven Planner for Synthetic Characters. In *Proceedings of 16th Int. Conference on Automated Planning and Scheduling (ICAPS)*, 2006.

[3] H.-M. Chang and V.-W. Soo. Planning-Based Narrative Generation in Simulated Game Universes. *IEEE Trans. Comput. Intellig. and Artificial Intelligence in Games*, 1(3):200–213, 2009.

[4] A. Coman and H. Muñoz-Avila. Generating Diverse Plans Using Quantitative and Qualitative Plan Distance Metrics. In

Proceedings of 25th AAAI Conference on Artificial Intelligence (AAAI), 2011.

[5] S. Cresswell and P. Gregory. Generalised Domain Model Acquisition from Action Traces. In *Proceedings of 21st Int. Conference on Automated Planning and Scheduling (ICAPS)*, 2011.

[6] C. Fellbaum. *WordNet: An Electronic Lexical Database*. Bradford Books, 1998.

[7] R. Fikes and N. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208, 1971.

[8] M. Fox and D. Long. The Automatic Inference of State Invariants in TIM. *Journal of Artificial Intelligence Research*, 9:367–421, 1998.

[9] B. C. Gazen and C. Knoblock. Combining the Expressiveness of UCPOP with the Efficiency of Graphplan. In *Proceedings of European Conference on Planning (ECP)*, 1997.

[10] D. F. Harrell, C. Gonzalez, H. Blumenthal, A. Chenzira, N. Powell, N. Piazza, and M. Best. A Cultural Computing Approach to Interactive Narrative: The Case of the Living Liberia Fabric. In *Proceedings of AAAI Fall Symposium on Computational Models of Narrative*, 2010.

[11] M. Helmert. The Fast Downward planning system. *Journal of Artificial Intelligence Research*, 26:191–246, 2006.

[12] J. K. Jones and C. L. Isbell. Story Similarity Measures for Drama Management with TTD-MDPs. In *Proceedings of 13th Int. Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 2014.

[13] N. Jones and P. Pevzner. *An Introduction to Bioinformatics Algorithms (Computational Molecular Biology)*. The MIT Press, 2004.

[14] B. Kartal, J. Koenig, and S. J. Guy. User-Driven Narrative Variation in Large Story Domains using Monte Carlo Tree Search. In *Proceedings of the 13th Int. Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 2014.

[15] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Cybernetics and Control Theory*, 10:707–710, 1966.

[16] B. Li, S. Lee-Urban, G. Johnston, and M. Riedl. Story Generation with Crowdsourced Plot Graphs. In *Proceedings of 27th AAAI Conference on Artificial Intelligence (AAAI)*, 2013.

[17] B. Li, A. Zook, N. Davis, and M. Riedl. Goal-driven conceptual blending: A computational approach for creativity. In *Proceedings of 3rd Int. Conference on Computational Creativity (ICCC)*, 2012.

[18] D. Lin, S. Zhao, L. Qin, and M. Zhou. Identifying synonyms among distributionally similar words. In *Proceedings of the 18th Int. Joint Conference on Artificial Intelligence*, 2003.

[19] M. Mateas and A. Stern. Structuring Content in the Façade Interactive Drama Architecture. In *Proceedings of 1st Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, 2005.

[20] D. McDermott, M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld, and D. Wilkins. PDDL - the Planning Domain Definition Language. Technical report, CVC TR-98-003/DCS TR-1165, Yale University, 1998.

[21] N. McIntyre and M. Lapata. Learning to Tell Tales: A Data-driven Approach to Story Generation. In *Proceedings of 47th Meeting of the Association for Computational Linguistics (ACL)*, 2009.

[22] R. Nazar and M. Janssen. Combining resources: Taxonomy extraction from multiple dictionaries. In *Proceedings of 7th Int. Conference on Language Resources and Evaluation (LREC)*, 2010.

[23] J. Orkin and D. K. Roy. Understanding Speech in Interactive Narratives with Crowdsourced Data. In *Proceedings of 8th Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, 2012.

[24] C. Paradis, C. Willners, and S. Jones. Good and bad opposites using textual and experimental techniques to measure antonym canonicity. *The Mental Lexicon*, 4(3):380–429, 2009.

[25] A. Pauls and D. Klein. Faster and smaller n-gram language models. In *Proceedings of 49th Meeting of the Association for Computational Linguistics (ACL)*, 2011.

[26] J. Porteous, M. Cavazza, and F. Charles. Applying Planning to Interactive Storytelling: Narrative Control using State Constraints. *ACM Transactions on Intelligent Systems and Technology (ACM TIST)*, 1(2):1–21, 2010.

[27] J. Porteous, M. Cavazza, and F. Charles. Narrative Generation through Characters' Point of View. In *Proceedings of 9th Int. Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 2010.

[28] M. Riedl. Incorporating Authorial Intent into GenerativeNarrative Systems. In *Proceedings of AAAI Spring Symposium on Intelligent Narrative Technologies*, 2009.

[29] M. O. Riedl and R. M. Young. Narrative Planning: Balancing Plot and Character. *Journal of Artificial Intelligence Research*, 39:217–267, 2010.

[30] I. Sag, T. Baldwin, F. Bond, A. Copestake, and D. Flickinger. Multiword Expressions: A Pain in the Neck for NLP. In *Proceedings of 3rd Int. Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*, 2002.

[31] S. Sina, A. Rosenfeld, and S. Kraus. Generating content for scenario-based seriousgames using CrowdSourcing. In *Proceedings of 28th AAAI Conference on Artificial Intelligence (AAAI)*, 2014.

[32] R. Swanson and A. S. Gordon. Say Anything: Using Textual Case-Based Reasoning to Enable Open-Domain Interactive Storytelling. *ACM Trans. Interact. Intell. Syst.*, 2(3), 2012.

[33] J. Teutenberg and J. Porteous. Efficient Intent-based Narrative Generation Using Multiple Planning Agents. In *Proceedings of 12th Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 2013.

[34] P. Turney. A uniform approach to analogies, synonyms, antonyms, and associations. In *Proceedings of the 22nd Int. Conference on Computational Linguistics*, 2008.

[35] R. P. y Pérez, N. Morales, and L. Rodríguez. Illustrating a Computer Generated Narrative. In *Proceedings of 3rd Int. Conference on Computational Creativity (ICCC)*, 2012.

[36] H. H. Zhuo and S. Kambhampati. Action-model acquisition from noisy plan traces. In *Proceedings of 23rd Int. Joint Conference on Artificial Intelligence (IJCAI)*, 2013.

[37] H. H. Zhuo, Q. Yang, D. H. Hu, and L. Li. Learning complex action models with quantifiers and logical implications. *Artificial Intelligence*, 174(18):1540–1569, 2010.