

Simulation-Based Story Generation with a Theory of Mind

Hsueh-Min Chang¹ and Von-Wun Soo^{1,2}

¹ Department of Computer Science, National Tsing Hua University
No. 101, Section 2, Guangfu Rd, Hsinchu City 300, Taiwan

² Department of Computer Science and Information Engineering, National University of Kaohsiung
No. 700, Kaohsiung University Rd, Kaohsiung City 811, Taiwan
{pchang, soo}@cs.nthu.edu.tw

Abstract

Emergent narrative refers to simulation-based systems in which stories emerge from the autonomous interactions among character agents and/or the human player. Despite its advantages in interactive games, emergent narrative attracts concern about whether complex narratives emerge from arbitrary interactions. This problem can be alleviated if the character agents act according to what a story should require. Based on the observation that deliberate influence on others' minds constitutes the fabric of most stories, we propose that the capability for social influence is crucial for narrative agents. This paper presents a specialized planning technique called social planning, which allows a character agent to achieve its goal by reasoning about other characters' minds and influencing their actions. A prototype system based on social planning agents succeeds in generating simplified variations of Shakespeare's *Othello* through simulation.

Introduction: Games and Story Generation

Games are a space in which the player instantiates stories (Frasca 1999) with varying degree of freedom. Player agency is said to be a central feature of the computer as a new media form (Murray 1997). Depending on the nature of the game and ways of playing, each game session can be a unique narrative. A game has high user agency if it gives the player the feeling of being able to control the progression. Assuming that user agency is desirable, storytelling systems should attempt to increase the range of possible variations across game sessions. This has been done in AI systems by employing characters that uses plans (Pizzi and Cavazza 2007), emotional responses (Aylett et al. 2006) and decision-theoretic reasoning (Si, Marsella, and Pynadath 2005) to autonomously respond to various inputs. Nevertheless, storytelling systems typically seek to maintain a consistent backbone story from which the character agents should never deviate too far. In practice, if the deviate too far they risk invalidating the backbone story and making proceeding impossible.

The emergent approach toward story generation has the advantage of being naturally compatible with user agency. In an ideal emergent narrative system, autonomous interactions between characters should constantly make up

story-quality content as the player goes along, as in the words of (Aylett et al. 2006). The main concern, however, is that the quality of story content resulting from arbitrary interactions is at least questionable.

On the other hand, story generators seldom assume a backbone story. They simply take a collection of data, often consisting of an initial state of the world, and output stories that contain action sequences and other events. Each initial state thus could lead to a unique story, yet not all possible action sequences are equally story-like. Story generators tend to have a set of criteria about what action sequences should be regarded as stories. Among them, social influence between characters is considered a central feature. For example, TALESPIN (Meehan 1977), an early story generator, puts much emphasis on belief- or intention-influencing actions. The IPOCL story planner (Riedl and Young 2004) also emphasizes social influence by ensuring that each action is backed with an intention, which is usually the effect of another character's action.

If autonomous characters in an emergent narrative are able to act in accordance with some criteria about what a story should be like, the result can be comparatively story-like. In this paper, we take the issue of social influence as central, and propose giving character agents the reasoning machinery to exert social influence, or to achieve personal goals through social influence. The character agents plan not only to manipulate physical objects, but also to manipulate the minds of other agents. Simulation based on such agents can produce action sequences that resemble those that are generated by a story planner. This paper reports our experience in the design and implementation of such a story simulation system, as a step towards a both open-ended and storied game experience.

Social Influence in Stories

From a biological standpoint, Dautenhahn (2003) views stories as the retelling of events in a social group, where emotions and intentions of group members are reconstructed. Social influence plays a major part in Bremond's narratology work, which is based on distinction between agents¹ and patients; the former role exerts

¹ We are forced to use three related but different senses of

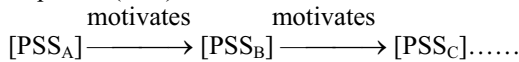
influence on others while the latter is subjected to influence. ((Bremond 1973) referenced from (Cavazza and Pizzi 2006)). A story is mainly the interaction between the patient and the agent, although a character can be both an agent in a relationship and a patient in another one, and a patient may be motivated to become an agent. Computational application of Bremond's theory would require characters to be capable of social influence, which in turn requires formal understanding about how others think; such understanding is commonly known as a *theory of mind*, such as in (Si, Marsella, and Pynadath 2005) and (Dautenhahn 2003).

Although social influence could happen with any set of motivated characters capable of being influenced, Bremond's theory requires the influence to be deliberate; the story has to progress along the plan of the agent to influence the patient, e.g. to make him or her show affection, give money or self-destruct. Random actions can influence characters, but appear less strongly story-like.

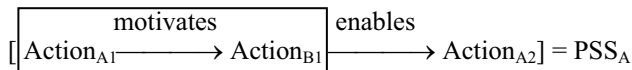
AI Model of Narrative Sequences

The prominence of social influence is also evident from the structure of the narrative sequence. AI-based research often understands stories as causally related sequence of events; an explicit treatment can be found in (Swartjes and Theune 2006). For the purpose of this paper, it is useful to dissect a story into causal subsequences and problem-solving subsequences. A causal sequence is defined as a chain of actions in which any later action is motivated by a former one. The motivation factor can be emotion, personal motives and/or social commitment. Emotion-based story systems tend to rely on causal sequences to a greater degree, as the storyline progresses from actions and reactions. On the other hand, a problem-solving sequence is a chain of actions in which any latter action is enabled by a former action, and all actions together satisfy a goal. In other words, it is a plan carried out.

A simple story structure can consist of a causal sequence in which each action is replaced by a problem-solving sequence (PSS):



In elaborated stories, however, causal sequences can be a part of problem-solving sequences instead, such as the following:



Actions in the box form a causal sequence. This structure characterizes deliberate social influence, because A solves

the word *agent*. In the AI sense, an agent is a piece of software that not only thinks but also acts. In the literary sense, it is one who exerts influence in contrast to the influenced counterpart. In a sociological sense (as in *user agency*), an agent is an initiator of change in a real or fictional society. There should be little confusion however.

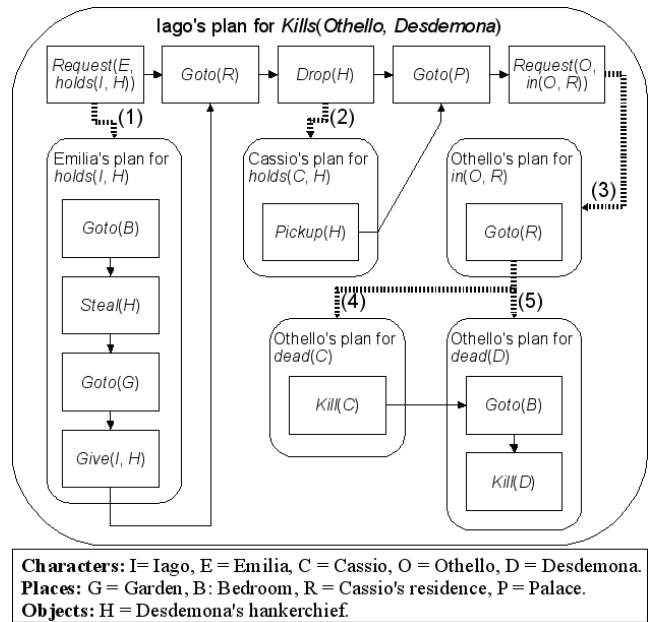


Figure 1: Schematization of Iago's social plan.

the problem by influencing the actions of B as a means to achieve the goal of his own. Agents who produce such a plan (and thus, such a story) need to understand how the emotion, motives and social commitment of other agents work. In other words, they must possess and use a theory of mind.

Social Influence in Othello

Shakespeare's play *Othello* is a remarkable example highlighting the above-mentioned structure. Since we will use a simplified version as the case study throughout this paper, a description is warranted.

A soldier Iago had long resented General Othello and plotted to make him suffer by having him kill his own wife, Desdemona. To carry out his plan, Iago first asked his wife Emilia in the garden to steal a handkerchief Othello gave Desdemona. Emilia stole it from Desdemona and gave it to Iago. Iago then deliberately planted the handkerchief in the lieutenant Cassio's residence, expecting that Cassio would pick it up, and Cassio did it. Othello, under the advice of Iago, then paid a visit to Cassio. Seeing the handkerchief on Cassio, he was enraged by jealousy. He killed Cassio immediately and ended up going to Desdemona's bedroom to kill her as well.

The structure of the *Othello* narrative sequence is depicted in figure 1. The overall story is one problem-solving sequence of Iago, but problem-solving involves motivating Emilia, Cassio and Othello to complete parts of the task. In causal subsequence (1), request motivates compliance. In (2), dropping a handkerchief motivates holding. In subsequence (3)-(4)-(5), invitation motivates visit, which in turn causes jealousy and murder. The causal subsequences in turn contain problem-solving sub-subsequences, resulting in a rather delicate structure.

Planning to Influence Others

In this section we describe the design and implementation of software agents that plan to influence others. The agents successfully reconstruct a simplified version of *Othello* as described in the previous section. The simulation system is not yet interactive, but can serve as a demonstration of how social planning works.

Table 1: Belief revision/motivation rules for the *Othello* scenario.

Source	Motivational Rules
Personal motive	Greed: $self(?s) \wedge at(?s, ?l) \wedge at(?o, ?l) \wedge precious(?o) \rightarrow holds(?s, ?o)$
	Curiosity: $self(?s) \wedge befriends(?s, ?a) \wedge request(?a, ?s, at(?s, ?l)) \rightarrow at(?s, ?l)$
Emotion	Jealousy: $self(?s) \wedge loves(?s, ?a) \wedge loves(?a, ?b) \rightarrow dead(?a) \wedge dead(?b)$
Social disposition	Obedience: $self(?s) \wedge loves(?s, ?a) \wedge request(?a, ?s, holds(?a, ?o)) \rightarrow holds(?a, ?o)$
Belief Revision Rules	
Folk psychology: $isGiftFrom(?g, ?a) \wedge isGiftTo(?g, ?b) \wedge holds(?c, ?g) \wedge ((woman(?b) \wedge man(?c)) \vee (man(?b) \wedge woman(?c))) \rightarrow \neg loves(?b, ?a) \wedge loves(?b, ?c)$	

Model of Characters with Motivation

Characters are modeled as software agents with explicit beliefs and goals. They occupy a shared environment, in which the effects of their actions can be perceived by others. Actions include physical ones and communicative acts such as request and inform. The agents use rules to update their mental state according to perception. Belief-revision rules decide how beliefs are updated with new percepts, while motivation rules decide how goals are generated from beliefs. The rules also cover communicative actions including request and inform. Once adopting a new goal, an agent plans for it. Table 1 shows the rules that appear in the *Othello* scenario. These rules are exactly how the characters deliberate during the *Othello* simulation. For example, the Greed motivation rule in personal motives says that an agent encountering a precious object will be motivated to hold it. These rules are exactly how the characters deliberate during the simulation.

Model of Social Plans

We define a social plan as one that includes any number of actions by other agents (which we term *foreign actions*), as opposed to non-social plans that contains only actions of self. Since incorporating foreign actions requires first motivating them, a revision of planning operators is required. In a social domain, actions modify not only the physical environment, but also the minds of agents. Belief effects, goal effects and intentional preconditions are defined to reflect the mental part of operators.

- Physical effects p-effects(*act*) are the effect of the action

act on the physical environment.

- Belief effects b-effects(*act*, *ag*) of an action *act* on the target agent *ag* are the changes of *ag*'s belief after it perceives p-effects(*act*).
- Goal effects g-effects(*act*, *ag*) are the changes of *ag*'s goals after it perceives p-effects(*act*).

Belief and goal effects are dynamic in nature. They are inferred from the mental rules of others and the spatial relation between the actor and the target and cannot be determined offline. The total effect of an action is the union of the three types of effects.

On the precondition side, an action must be intended to be performed. The planning agent is free to intend any action, but actions of others must be explicitly motivated. An action is intended if the performer intends some of the effects of the action. Thus,

- Intentional precondition i-precond(*act*) = $self(acting.performer) \vee (goals(acting.performer) \cap effects(acting) \neq \emptyset)$.

Note that we strictly stick to the basic set of operators used by non-social domains, without introducing any new operator specialized for social influence. Only the context determines whether an operator is social or non-social. A social plan is thus one in which the intentional precondition of every action is satisfied with the goal effects of other actions or the initial state. Additional formalisms for social plans can be found in (Chang and Soo 2008).

How Social Planning Agents Work

Social planning, formulated this way, thus involves dynamically deducing the belief and goal effects and matching them against the intentional preconditions. Our technique relies on automatic translation of a non-social world description and knowledge about other agents to a social planning domain written in the Planning Domain Description Language (PDDL). A general-purpose PDDL planner then solves the generated problem. Describing planning problem requires two PDDL documents, a *domain* file and a *problem* file; the former describes predicates and actions while the latter records the initial state. Since version 2.2 (Edelkamp and Hoffmann 2004), PDDL provides a *derived predicates* construct, which defines predicates that can be deduced from other predicates but cannot be direct effects of any action. Mental predicates, including beliefs and goals, obviously fall into this category because they always follow from physical effects. Any attempt to change the mental predicates must be mediated by the physical environment, and the result can only be inferred, as (Field and Ramsay 2006) has pointed out.

Compiling domain knowledge into PDDL thus involves specifying deduction of belief predicates and goal predicates in the domain file, and dumping knowledge about the mind of others into the problem file. A PDDL 2.2 compatible planner can then infer the belief goal and effects according to the execution context. The architecture

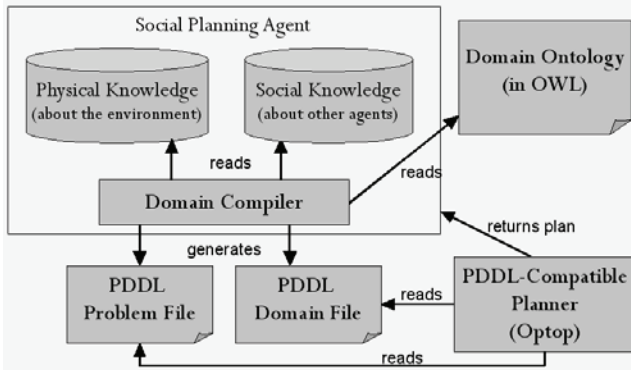


Figure 2: Architecture of the social planning agent.

of the social planning agent depicted in figure 2. Following subsections describe the details about domain compilation.

Location Reasoning. Controlling the perception of other agents is the first step toward controlling the social consequences of an action. Whether the effect of an action is perceived or not depends on the relative location between the performer and the witness. The domain compiler models percepts as conditional effects of an action. They will be generated only when the action is performed within the perceptual range of another agent, so that the action performer knows where to perform an action to achieve the desired effect. Moreover, actions involving location change has additional perceptual effects: the agent in the new location perceives everything perceivable in that location, and vice versa. An action also generates perceivable action events.

Reasoning about Belief and Motivation. For a character agent a who tends to be folk-psychological, the rule in table 1 is a belief revision rule. For another character who know that a is folk-psychological, the same rule becomes a theory of mind. A belief predicate thus can be derived from two sources: direct perception, and belief revision rules. The belief_loves(a, b, c) predicate (=PDDL form for believes($a, loves(b, c)$)) has the following definition:

$$\text{believes}(a, \text{loves}(b, c)) \leftrightarrow \text{perceives}(a, \text{loves}(b, c)) \vee (\text{folk-psychological}(a) \wedge \exists (gift, d) (\text{believes}(a, \text{isGiftFrom}(gift, d)) \wedge \text{believes}(a, \text{isGiftTo}(gift, b)) \wedge \text{believes}(a, \text{holds}(c, gift)) \wedge \dots))$$

In practice, however, love is an unperceivable predicate and will be excluded from the definition. Goal predicates are similarly derived from motivation rules. Communicative actions are treated no differently from physical actions. Their performance produces action events as effects, which invoke the matching belief revision and motivation rules. For each physical predicate, the domain compiler generates a corresponding belief predicate and a goal predicate, as PDDL derived predicates.

Intentional Precondition and Macro Action. The intentional precondition of an action is generated exactly the same as the definition of $i\text{-precond}(act)$ above. Now social planning can work by using goal effects to satisfy intention preconditions, with one remaining problem: some

goals of others may need multiple actions to achieve, necessitating that a goal effect supports not just an action but a whole plan of some other agent. As expressing such support in PDDL *per se* is problematic, we define macro actions as the repertoire of plans one may get another agent to conduct. Macro actions represent common recipes of actions in the form of a plan library. Their use may increase plot familiarity and planning speed, but may also reduce the number of possible social manipulation strategies. In the *Othello* scenario, Emilia's plan in (1) and Othello's plan in (5) are represented as macro actions for Iago, although Emilia and Othello do plan for their goals themselves during simulation.

Social Plan Execution

Agents executing non-social plans need only to check whether the precondition of the next action is true before performing it. However, social plans raise some additional complications that must be dealt with.

1. As stated, the belief and goal effects of an action are context-dependent due to reliance on derived predicates and conditional effects. A post-processor therefore must reconstruct the actual precondition and (mental) effects of actions after the plan is produced and before it is put to execution. Moreover, it must reconstruct the relation between a specific effect and the conditions that the effect requires, because *it is possible that an action does not achieve the desired effect*. For example, if Cassio travels elsewhere during Iago's plan, the handkerchief can still be dropped, but doing so accomplishes no goal because Cassio is not there to pick it up. An agent thus must ensure the particular conditions for the intended effect hold before performing an action.
2. Unlike self actions, foreign actions may take place somewhere outside the planning agent's range of perception, leaving the planning agent unaware of whether the effect has been realized or not. Our current solution is to insert *goto* actions during post-processing to monitor if a foreign action has taken place. For example, after telling Othello to visit Cassio, Iago should go to Cassio's residence to make sure things happen as he predicted.
3. A social plan may fail, in that some other agents do not actually perform the expected action. When, how and if they will eventually perform it is not always certain. A completely sensible solution can be difficult. Our current naïve solution is just to wait for several ticks, and deem it a failure after timeout. However, if social plan failure is due to an incorrect model of others, a new plan may still fail at the same spot. A better solution would be a diagnosis of the belief model to repair wrong assumptions about others.

Implementation and Experiences

System Construction

The social planning agent depicted in figure 2 is fully implemented, using Optop (McDermott 2004) as the

PDDL planner, as Optop is the only tested planner that reliably solves problems with derived predicates and conditional effects. The multi-agent simulation platform runs on JADE, which talks to Lisp-based Optop via sockets. The agent platform contains an ontology layer that dynamically annotates the physical environment in OWL, as described in (Chang et al. 2005). Attempts to integrate the system with a Java-based 3D engine are underway.

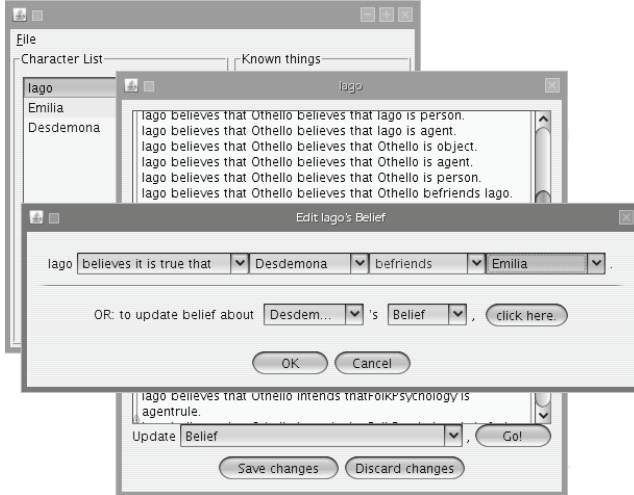


Figure 3: Character mental state editor.

Scenario Setup

We developed a character mental state editor, which is shown in figure 3. The editor reads the domain ontology in OWL beforehand and then allows a human designer to edit the characters' beliefs and intentions according to the ontology. The editor also allows for the specification of beliefs and intentions about others' beliefs and intentions.

To make the character agents produce a plot exactly like that in figure 1, we made Iago the sole social planning agent, while Cassio, Desdemona, Emilia, and Othello are non-social planning agents who do not deliberately influence others. Iago knows the mental state of all other characters and their personalities as well, such as Othello being curious and jealous and Emilia being obedient. The plot is successfully reproduced after a few tweaks, which are explained below.

Preliminary Results

One main concern is whether the system can generate new stories. Before we can reproduce the *Othello* story, some faults in the domain already lead to alternate stories, one of which results from not adding the conjunct $\neg \text{intends}(\text{Othello}, \text{dead}(\text{Iago}))$ to the goal. Without that conjunct, Iago will simply visit Othello while holding the handkerchief. Doing so motivates Othello to kill not only Desdemona, but Iago himself as well. Another deviation occurs when we specified the goal as $\text{dead}(\text{Desdemona})$, in that case Iago would simply go and kill Desdemona. An intuitive observation is that motivation sensitivity positively corresponds to possibility of story emergence.

The reason is that one has more instruments to achieve a goal when others are easier to motivate. For example, if Iago is unable to kill, the goal $\text{dead}(\text{Othello})$ is impossible, until we added a new rule allowing everyone to ask for the death of another. Iago then devised the plan to ask Emilia to desire Othello's death, invite Othello to the garden, and let Emilia kill him. However, motivation sensitivity can also degrade the story by making goals too easily achievable. For example, with the new rule above, Iago can easily ask Othello to kill Desdemona and complete the task of the original story, losing most of the intrigue. A more systematic experimental investigation about the tradeoff between story possibility and quality is being planned.

Related Work

The general idea of social planning comes from (Castelfranchi 1998), whose theory identifies social actions as ones that are performed in awareness of other agents and/or to change their minds. The conversation planner by (Field and Ramsay 2006) models effects of a speech act as indirect ramifications and considers the latter normal in communications. Their planner is coupled with a theorem prover to derive the belief effects, but does not address goals and social manipulation.

It is common that interactive storytellers are also story generators to a certain degree. A number of them have already taken social influence into account. Thespian (Si, Marsella, and Pynadath 2005) is remarkable in that its agents employ a theory of mind to decide actions. Unlike social planning agents, however, Thespian agents are based on decision-theoretic reasoning. A Markov model simulates some steps of lookahead, so that the agents can decide on the action with the highest expected utility. Despite the advantages of being able to simulate the world and derive utilities, Thespian's decision-theoretic approach may not scale as well as planners, and the maximum lookahead is limited at 2, as opposed to Iago's 14-step plan. This results in a tendency of Thespian stories to rely more on causal sequences than problem-solving ones, as shown in their own examples.

Pizzi and Cavazza's storyteller for the *Madame Bovary* scenario puts particular focus on the mental side of a story (Pizzi and Cavazza 2007). The system makes use of explicitly mind-influencing planning operators. As opposed to physical operators which modify the environment, an *interpretation operator* modifies the mind of self, while a *character interaction operator* modifies the mental state of others. For the purpose of story generation (which is not the exact purpose of their system), this distinction can be too rigid for two reasons. First, the character interaction operators modify only relational fluents, such as the target character's affinity towards the performer, but not all mental predicates in general, resulting in limited ability for social manipulation. Second, the distinction prevents agents from using physical operators for character interaction purposes through *stigmergy* (environment-relayed communication),

exemplified by Iago's drop action.

IPOCL (Riedl and Young 2004) is a story planner that plans actions for all agents in the scenario. Despite that IPOCL is not based on simulation, it does share some methodological similarity with social planning, in that it ensures the intention behind each action is either motivated by another action or already in the initial state. However, social planning uses a more sophisticated model of other autonomous agents that allows reasoning about their perception, belief update and motivation rules, while IPOCL relies on operators that directly introduce an intention into an agent. On the other hand, an important advantage of the IPOCL algorithm is that it does not rely on macro actions, which are needed by the PDDL-based approach to social planning.

Conclusion

The ability of character agents to influence others socially is the key to the sophistication of emergent narratives. Modeling characters as software agents that understand others' minds enables them to become literary agents who exert influence on a patient. Social planning allows character agents to build plans upon other characters' actions by reasoning about how to motivate them. Our multi-agent story system, based on social planning characters, demonstrates that sophisticated stories can be generated from multi-agent simulation.

Acknowledgement

This work is supported by the National Science Council of Taiwan under grant number NSC 96-2628-E-007-044-MY3. The authors appreciate the comments from anonymous reviewers.

References

- Aylett, R. S., Figuieredo, R., Louchart, S., Dias, J., and Paiva, A. 2006. Making It Up As You Go Along - Improvising Stories for Pedagogical Purposes. In *Proceedings of the 6th International Conference for Intelligent Virtual Agents (LNAI 4133)*, 307-315. Springer-Verlag.
- Bremond, C. 1973. *Logique du Recit*. Paris: Editions du Seuil. (in French)
- Castelfranchi, C. 1998. Modeling Social Actions for AI Agents. *Artificial Intelligence* 103(1-2): 157-182.
- Cavazza, M., and Pizzi, D. 2006. Narratology for Interactive Storytelling: A Critical Introduction. In *Proceedings of the 3rd International Conference on Technologies for Interactive Digital Storytelling and Entertainment (LNCS 4326)*, 72-83. Springer-Verlag.
- Chang, H.-M., and Soo, V.-W. 2008. Planning Actions with Social Consequences. In *Proceedings of the 10th*

Pacific Rim Workshop for Multi-Agents (LNAI series), forthcoming. Springer-Verlag.

Chang, P. H.-M., Chen, K.-T., Chien, Y.-H., Kao, E., and Soo, V.-W. 2005. From Reality to Mind: A Cognitive Middle Layer of Environment Concepts for Believable Agents. In Weyns, D., Parunak, H. V. D., and Michel, F. eds. *Environments for Multi-Agent Systems (LNAI 3374)*, 57-73. Springer-Verlag.

Dautenhahn, K. 2003. Stories of Lemurs and Robots – The Social Origin of Story-Telling. Chapter 4 in Mateas, M., and Sengers, P. eds. *Narrative Intelligence*. Amsterdam: John Benjamins.

Edelkamp, S., and Hoffmann, J. 2004. PDDL2.2: The Language for the Classical Part of the 4th International Planning Competition. Technical Report 195, Albert-Ludwigs-Universität, Institut für Informatik, Freiburg, Germany.

Field, D., and Ramsay, A. 2006. How to Change a Person's Mind: Understanding the Difference between the Effects and Consequences of Speech Acts. In *Proceedings of the 5th Workshop on Inference in Computational Semantics*, 27-36.

Frasca, G. 1999. Ludology Meets Narratology: Similitude and differences between (video)games and narrative. <http://www.ludology.org/>. Original Finnish version: 1999. *Ludologia kohtaa narratologian. Parnasso* 3: 365-371.

McDermott, D. 2004. The Optop Planner. In *Fourth International Planning Competition Booklet (IPC'04)*.

Meehan, J. R. 1977. TALE-SPIN: An Interactive Program That Writes Stories. In *Proceedings of the 5th Joint Conference on Artificial Intelligence*, 91-98.

Murray, J. H. 1997. *Hamlet on the Holodeck: The Future of Narrative in Cyberspace*. New York: The Free Press.

Pizzi, D., and Cavazza, M. 2007. Affective Storytelling Based on Characters' Feelings. In *Proceedings of the 2007 AAAI Fall Symposium on Intelligent Narrative Technologies*.

Riedl, M. O., and Young, R. M. 2004. An Intent-Driven Planner for Multi-Agent Story Generation. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, 186-193. Washington DC: IEEE Computer Society.

Si, M., Marsella, S. C., and Pynadath, D. V. 2005. Thespian: Using Multi-agent Fitting to Craft Interactive Drama. In *Proceedings of the Fourth international Joint Conference on Autonomous Agents and Multiagent Systems*, 21-28. New York: ACM Press.

Swartjes, I., and Theune, M. 2006. A Fabula Model for Emergent Narrative. In *Proceedings of the 3rd International Conference on Technologies for Interactive Digital Storytelling and Entertainment (LNCS 4326)*, 49-60. Springer-Verlag.