A Synthetic Mind Model for Virtual Actors in Interactive Storytelling Systems

Samir Araujo and **Luiz Chaimowicz**
Computer Science Department
Federal University of Minas Gerais

Abstract

Interactive Storytelling (IS) is a growing research area that has attracted a large number of researchers in recent years. However, there are few works exploring the perspective of creating IS systems that conduct emergent stories using autonomous intelligent agents as virtual actors. In this paper, we propose a model of a synthetic mind for virtual actors that can make them interpret a role of a given character in a specific narrative. We discuss the model architecture, implement a prototype, and present some proof-of-concept experiments to show its effectiveness.

Introduction

Virtual actors play an important role in games and other interactive entertainment applications. They are used to compose the screenplay, interacting with human users and driving them inside the main plot. With the increasing complexity of these applications, the use of less scripted and more autonomous virtual actors is becoming almost mandatory. This is specially true in Interactive Storytelling (IS) Systems, in which the story is not linear and can be dynamically changed according to the behaviors of players and characters.

Interactive Storytelling (IS) is a very exciting and complex research field that receives contributions from several areas such as Human-Computer Interaction (HCI), Artificial Intelligence (AI), Natural Language Processing (NLP), Narrative and Linguistics among others. In the last decade, several researches have become involved with IS and much work has been done. Generally, these works can be classified into two distinct categories: story-based and character-based (Crawford 2005). On story-based systems, virtual actors behave in a less autonomous way. Commonly, they follow a script made by the story author in order to guarantee the plot consistency. On the other hand, character-based systems give more autonomy to virtual agents, allowing them to plan and act more freely. A system can also use concepts of both categories, characterizing it as a hybrid system.

We believe that an IS system that gives more freedom to the autonomous agents is capable of conducting larger stories, besides making the narrative more versatile and attractive to the human user. Larger stories because autonomous agents can handle unforeseen situations, allowing the author to prepare a large number of events in the story that requires less details to be specified. Versatile because handling unforeseen situations can lead to consistent changes in the narrative, and attractive because variations in the narrative can create new stories at each play.

Unfortunately, creating an IS system composed of virtual actors with a high level of autonomy is not an easy task. It requires an artificial intelligence system capable of achieving complex goals in different kinds of virtual environments. Moreover, it must do this in a very efficient manner. In general, simple classical deductive systems cannot be used for planning goals for this kind of agent because, depending on the size of the knowledge base and the type of inference that must be done, it would take a very large amount of time to be computed, affecting the responsiveness of the system. Thus, another AI approach must be used to solve this problem.

Artificial General Intelligence (AGI) (Goertzel and Pennachin 2007) is a good solution for the problem mentioned above. Wang (1995) defined intelligence as "..the ability to achieve complex goals in a complex environment, using severely limited resources.". Thus, a system capable of achieve complex goals in this way can be considered a general purpose AI or simply an AGI.

The main objective of this work[1] is to propose a synthetic 'mind' model for an intelligent autonomous agent that can be used as a virtual actor, interpreting the role of a specific character in a IS system. This model allows agents to handle unforeseen situations, autonomously taking actions according to behaviors that emerge from the mind inference process, giving the IS system more versatility and flexibility. Given that an AGI is something that is natural conceived to handle general situations, we use several AGI concepts in our work, such as commonsense reasoning (Minsky 1985), situation recognition by using analogy with acquired knowledge (Minsky 2006), problem solving using customizable reasoning rules, etc. It is important to mention that the term 'mind' is used to refer to the mechanism created to control the functionalities of a specific IS virtual actor. In other words, this work has a very straight objective and has not the pretension of creating a generic artificial human-like intelligence.

[1]This work was partially supported by CNPq and Fapemig.

Related Work

The core of our work is mainly based on the *Frames Theory* (Minsky 1985). Minsky proposed a Frame system to represent knowledge. In general, a frame is a data-structure used for representing things and situations, structured in a network of nodes and relations. Mainly applied for commonsense reasoning (Minsky 2006), a set of frames can be used to directly store answers for any kind of question. So, the agent does not need to start a reasoning logical chain every time it needs to solve a problem. It just needs to retrieve that answer.

Other systems take different approaches. A system called FearNot! (Aylett et al. 2005) was created with the objective of experimenting narratives constructed with emergent behaviors. Several works were done using FearNot!. One of them is the "Double Appraisal Story Facilitator" (an extension of the original Story Facilitator), which uses a concept of a director agent (Aylett et al. 2007) named Story Facilitator (SF). SF is responsible to manage the storyline through an episode sequencing. An episode is a fragment of the story interpreted by the virtual actors. Each action executed by an agent is classified by its emotional impact on itself and on other agents. So, while planning actions, an agent takes into account its objectives, its motivations to do that, and the emotional impact of all the candidate actions on other agents. Each agent has some kind of mental model of the others, which is used to test the reactions of its actions before executing it. Then, the main line of the story can be changed due to the interaction among the agents and their emergent behaviors. Several other works were done by the same research group which developed FearNot!, exploring the emergent behavior approach in interactive narratives (Figueiredo, Paiva, and Aylett 2007).

Wang (2006) defined an inference system that does not require the use of axioms as basic inference rules. Non Axiomatic Reasoning System (NARS) is capable of answering questions and/or producing statements to improve its knowledge base, handling uncertainty and little information about the working context. Using probabilities to setup the confidence of all statements, NARS can infer complex and useful solutions for several types of problems. However, an open source version of NARS (OpenNARS[2]) is not recommended to be used in real-time systems which requires response times in milliseconds, as most logical inference systems which uses a NP-time algorithm.

Thus, differently from other approaches, we use commonsense reasoning and other AGI techniques to forge IS systems that conduct the narrative with a succession of emergent behaviors. As mentioned, this allows the IS system to work in a flexible and versatile way, but always respecting all hardware performance restrictions, to craft an useful model. It is important to mention that there are other tools and techniques commonly used to build intelligent agents such as Soar, HTN, STRIPS, etc. However, our architecture specifically takes into account a set of requirements to create agents that can be used as virtual actors in a IS application, as discussed in the next section.

[2]http://code.google.com/p/open-nars/

Model

In order to build intelligent agents that behave as virtual actors in an emergent Interactive Storytelling system, we need to use a robust yet flexible agent control system. Normally, the key requirements for this system are:

- Handle unforeseen situations.
- Be capable of achieving complex goals.
- Manage long term goals and short term goals to achieve its objectives.
- Handle "emotions" that can drive the agent to change its goals (short and long term).
- Allow the Story Builder (the person responsible for creating/organizing the narrative on the IS application) to model the personality and profile of each actor, changing the importance of its emotions and adding memory fragments.
- Use memory fragments to reason and decide what to do. This will lead the actor to do some expected actions when a predicted event occurs.

Generally, traditional Game AI techniques such as scripts and simple Rule Based Systems (Brachman and Levesque 2004) are not able to handle these requirements. Thus, the main purpose of our model is to create a mechanism to control autonomous virtual actors in an Interactive Storytelling context. This mechanism shall take into account several environment variables produced by the interaction among the story characters, the user and the virtual environment. These environment variables can modify the behavior of the virtual actors, changing their objectives and creating emergent and unforeseen new situations for them to deal with.

Obviously, the aforementioned requirements are not easy to be achieved and require a very complex system. However, we shall keep in mind that the modeled Artificial Intelligence should not try to imitate a real human. Far from this, the virtual actor must be a synthetic intelligent agent, capable of executing actions in a sequence such that it gives the impression of intelligence to the human user.

Considering this, our model is comprised of five different mind modules as shown in Figure 1. These modules work together to control the flow of information and process it into a continuous sequence of actions that can be interpreted by the user as behaviors. The next sections explain these modules in details.

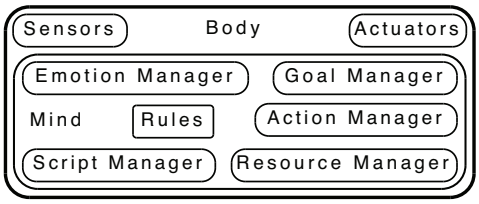

Figure 1: Agent components. The agent's body has a Mind, Sensors and Actuators. The former is composed by five mind modules and a rule set.

Resource Manager

The way memories and knowledge are represented inside the mind is one of the most important characteristics of an intelligent agent model. Almost all operations inside the artificial mind depend on memories. So, we should consider representations that facilitate the processes of writing and retrieving memory fragments from a database of memories. One of the best ways of representing knowledge into Artificial Intelligent systems is using Semantic Networks (Quillian 1968). A Semantic Network can be used to describe complex objects and situations in a straightforward manner. In this work, we use a kind of Semantic Network called Frames (Minsky 1985). A Frame is a structure very similar to classes and objects in the Object Oriented Paradigm (Brachman and Levesque 2004). Using frames we can represent from static objects and scenes to complex events or sequences of actions. In our model, there is one module, called Resource Manager, responsible for managing several structures built using frames.

This module is used by all other mind modules to store and retrieve several types of mind structures, called mind resources or simply resources. A resource is a common interface for these mind structures, which can be a memory fragment (represented by Frames), an emotion signal, a motivational signal, etc. Resource is a term extensively used by (Minsky 2006) to denote a unique and simple mind functionality. In our model, any kind of resource can relate to another, using a connection called Similarity Link. Each Similarity Link has a strength associated with it, used to determine the level of similarity between the two resources.

A memory fragment can be organized as a single Frame structure or hierarchically in a tree of Frames. A single Frame can describe complex objects using its slots. A Slot is like a property or a member of a Frame and each Frame can have an unlimited number of slots. Given that a slot can only be fulfilled by another Frame, they can be seen as mechanisms to build reference relations between two Frames (Brachman and Levesque 2004). A Frame can also relate to another Frame via inheritance. An inheritance relation between two Frames, which is similar to an Object Oriented inheritance, represents a "is a" relation. This relation, besides making descendants share the same type of their parents, make all the parent slots available to them. A group of resources, which relate to each other using slots, Similarity Links, and/or inheritance, can be seen as a Semantic Network. An example is shown in Figure 2.

When an agent is interacting with the environment, several types of signals are driven to its mind. These signals may come from outside, via sensors, can be emotions or any kind of proprioceptive sensors (physiological needs, for example, or other types of signals that motivates the agent to do something). Proprioceptive sensors are outside the scope of this work, even though they can be implemented as a mind resource, as mentioned before.

When a signal is recognized by the mind, one or more resources related to that signal emerge and become active for a while. Then, a signal is propagated from each active resource to other resources, following the Semantic Network relations, activating resources in a sequence until a threshold of maximum active resources is reached. An activation is described by a strength. The stronger the activation, the more important the resource will be at that specific moment. Note that this process avoids an explosion of resource activations, focusing only on those resources that matter most.

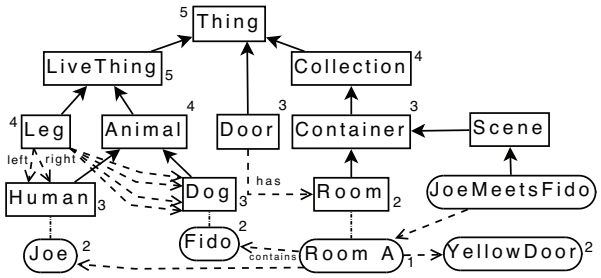

Figure 2: Continuous arrows indicates inheritance between Frames. Dashed arrows represents a reference between Frames created by a Slot on the target Frame. Rounded elements are instances of Frames. The numbers are the activation times that indicate a Frame propagation sequence.

There are two special types of Frames that play important roles in the model, Scenes and Stories Frames. A **Scene Frame** is a container that holds every element located at a specific place (real or imaginary) at a specific moment. It is like a snapshot taken from a room, a street or even a scene of a dream. A Scene also keeps the relations between elements: Joe is near Fido; Fido is "in front of" a Yellow Door, etc. Moreover, there are also some emotional information linked to a Scene Frame. When a Scene Frame is stored into the Resource Manager, the highest emotion signals (at that moment) are used to describe the emotional information of the given scene. When a scene emerges, its emotional information is used as input to the Emotion Manager module which, in turn, controls the emotional state of the agent. Another information linked to a Scene is its status situation, which can be Good, Neutral, Bad or Critical. When a Bad or Critical Scene emerges, the agent must try to figure out how to resolve that situation or, in the worst case, escape from it. For that, the agent needs to investigate some Story Frames associated with that scene.

When an action is executed by any kind of agent inside the virtual environment, changes in the current scene may occur. Thus, in order to keep track of any kind of event resulting from an action execution, a **Story Frame** can be used. Using a mixture of Trans/Story Frames concepts described in (Minsky 1985) and (Minsky 2006), our Story Frames are structures that store actions using a sequence of three basic components: Previous Scene (which describes a given environment), an Action (that was executed on the Previous Scene) and a Resulting Scene (that is the Previous Scene modified by the executed action). Note that in a Story Frame, a Resulting Scene can become the Previous Scene of another action. So, the final structure of a Story Frame is a sequence like: scene A, action 1, scene B, action 2, scene C, and so on.

Script Manager

Due to the dynamical behavior of resources, a script language can be very useful to help creating, destroying and retrieving resources. Furthermore, a script language can facilitate the access of resources to external parts of the mind, including sensors and actuators. So, our agent's mind model has a Script Manager module that registers each resource and makes them available to each other in a common environment, also accessible by the whole mind. That common environment is provided by the script language, which makes all resources available as variables that can have their interfaces called as little applications. This module makes the maintenance of all functions that are associated with frame slots easier, besides making the overall system a better tool for experimentation and creation of new functionalities without the need of compiling the system.

Goal Manager

This module is responsible for building action plans, a mechanism used by the agents to achieve their goals. There are two types of goals, long term and short term. A long term goal, in most cases, is a hard to achieve objective. Therefore, a long term goal can only be reached by achieving one or more short term goals. Basically, a short term goal is a simple action.

An action has preconditions and effects. Preconditions are represented by certain states that should be reached to allow the execution of the action. Effects are everything that will be modified after the execution of the action. For example, lets consider an action called *OpenDoor*. It can only be executed if the target door has its state set to closed. If the target door is closed and *OpenDoor* is executed, the resulting effect will be a new state in which the door is opened. Each State Frame has a list of actions attached to it. If one of these attached actions is executed, it is possible that this State Frame become active. Note that when an action is executed, there is no guarantee that it will succeed and the effect will be confirmed. An action can fail and if this is the case, its effect will not be fulfilled.

So, in summary, after the mind reasoning rules (discussed further) register a new goal on Goal Manager, this module tries to break that goal into easier subgoals and then converting them into simple actions, to be executed by the Action Manager module. The sequence of goals that must be achieved by the agent can be called a plan.

Action Manager

Executing an action means, in most cases, to use actuators to interact with the virtual environment. In general, each actuator could execute an independent action at any moment. But it would introduce an unnecessary complexity into the system. So, the Action Manager is responsible for receiving actions and scheduling them to be executed in a priority ordering. A scheduled action can be stopped, if in execution, or canceled, if it has not been sent to execution yet.

As mentioned before, when a short term goal has all its preconditions satisfied, its corresponding action is sent to the Action Manager. Thus, after the execution of that action, the short term goal in the Goal Manager is removed and marked as failed or achieved. Each attempt to execute an action is logged in a Story Frame as a history. After a failure trying to execute an action, this information can then be used to reason about alternative goals in the next mind processing cycle, given that the current plan did not work.

Emotion Manager

Reilly and Bates (1996) have detailed in their work an emotion system, based on OCC (Ortony, Clore, and Collins 1988), called *Em*. It defines two types of emotions, positive and negative. Positive emotions drive the agent to a good mood while negative emotions to a bad mood. *Em* is a straightforward and easy to develop/maintain system that was conceived by the authors to be a computer emotion system. So, we chose to use this approach to build the Emotion Manager. Each emotion is characterized by a signal inside the emotion module. In order to compute the intensity of each emotion signal we use a sigmoid based function proposed by (Picard 1997). The parameters of that function can be configured by the Story Builder to determine some aspects of the personality of the character. Besides simple emotions, the agent can have mood states. So, we used a logarithmic function, proposed by (Reilly and Bates 1996), to check if the agent is having a good or bad mood at a given mind cycle [3].

Sensors and Actuators

Sensors are responsible for collecting information from the environment and, with the help of the mind modules, creating short term memory fragments. Short term memories are resources that can be forgotten by the agent after a short period. These types of memories are very useful for planning short term goals. However, if a short term memory become active for a long period, it can be transformed into a long term memory, which is a kind of memory that is never excluded from the Resource Manager.

Each element collected by a sensor is classified using a strength of its perception. A Field of View (FOV) sensor, for example, will give more importance to objects positioned at its center. In that way, an object with greater strength of perception has greater chances of becoming a long term memory. Each perceived and active element will propagate its activation strength through the Resource Manager Semantic Network, activating other related memory fragments. So, that propagation directly influences the agent's plan, given that only active resources with greater strength will be used by the mind processes to reason about goals.

On the other hand, actuators are used by agents to execute actions in the virtual environment, like a grip that grab something after during the execution of the action 'Hold(target)'. As mentioned before, successfully executed actions change the state of some objects. These changes are made by the agent's actuators. So, after the execution of Hold(target) by an agent, the target object becomes "grabbed by the agent", "near the agent", "carried by the agent", etc.

[3] Look for Emotions and Moods for Animated Characters in (Picard 1997) for more details.

Reasoning Processes

At each system update cycle, the agent's mind executes several operations sequentially, as described in Algorithm 1. One of these operations is invoking the current Reasoning Mode. A Reasoning Mode is a fixed list of rules, encapsulated by a Strategy object (Strategy design pattern (Gamma et al. 1995)). It is the key point of the Mind. An agent can have more than one reasoning modes configured in its mind. If one does not produce any valuable goal, the agent can switch to another and try again.

The algorithm $ReasoningRules()$ executes the current active Reasoning Mode Strategy object. Even though this model has a customizable architecture, we present in Algorithm 2 a list of basic rules that comprise a default Reasoning Mode, used in our prototype to execute the experiments detailed in Section . Note that this Reasoning Mode will try to always keep the agent in good or neutral situations.

A resource may become active if it receives a signal from other resources by strength propagation or from emotion signals. So, to normalize all these strength sources and to compute the final strength, we must use Equation 1.

$$s = \begin{cases} A(((i * d) + (\sum_e i * e * f)/n)/2), & \text{if } n \geq 1; \\ A(i * d), & \text{otherwise.} \end{cases} \quad (1)$$

In this equation, s is the resulting activation strength; A is an amortization function used to keep the strength value inside a fixed range (in our prototype we used a sigmoid function); d is a distance factor which characterizes an intensity decay of the signal propagated from the source to other related resources. i is the input activation intensity given from a short term memory activation; e is the current emotional intensity of all emotions related to the given resource (in most of cases only Scene Frames will have emotions linked with); f is the emotion factor that affects the Scene Frame (influence of that emotion on the given Scene Frame); and n is the number of related emotions.

The Scene Frame Strength in Algorithm 2 is defined by the following equation:

$$s = \sum_{ir} ir/n, \quad (2)$$

where s is the final Scene Frame Strength; ir is the intensity of all active resources that are present in the given scene; and n is the number of elements of the scene.

A resource does not stay active forever. Just after becoming active, its strength starts to decrease by a constant factor. It is a raw simplification of the human brain forgetting process. So, if that resource does not receive a new strength signal in a future mind cycle, it will become inactive. A short term resource that has its activation value equals zero will be definitely forgotten (removed from Resource Manager). Alternatively, it will be transformed into a long term memory if it remains active for a long period (set by a threshold).

Algorithm 1: Mind Processes

```
foreach sensors do
    foreach perceived element by sensor do
        create/update short term memory for element;
foreach emotion signals do
    pass the current emotion signal to Res. Manager;
update resource manager;
compute the activation of the emergent resources;
collect emotion strengths of all active scenes and update
emotion manager;
create a temporary Scene Frame using all short term
memory elements;
ReasoningRules();
update Action Manager and fire actions;
```

Algorithm 2: Reasoning Rules

```
// f = scene identification factor
f ⟵ 0.5;
a ⟵ Equation 2(short term memory Scene) ;
foreach scene in active Scene Frames do
    b ⟵ Equation 2(scene) ;
    if b ≥ a * f then
        c ⟵ Story Frame which contains scene;
        if ∃ c then
            if scene ≡ bad situation then
                d ⟵ find good/neutral scene in c;
                if ∃ d then
                    create a goal using d;
            else if scene ≡ neutral situation then
                d ⟵ find good scene in c;
Update goal manager;
```

Experiments

In order to evaluate the functionality of the proposed model we have built a prototype. The agent's mind and all the five modules were implemented and integrated into an application that simulates the execution cycle of the mind. All input data has been directly inserted into the prototype to avoid the process of collecting data from the environment. We describe bellow some details about two important experiments executed using our prototype. Our intention in executing these experiments is to validate the capabilities of the synthetic mind in recognizing situations and creating simple action plans. To do that we created two semantic networks directly inside the agent's mind. The first network represents a simple room containing: a door, the agent and a dog between the agent and the door. That semantic network is conceptually presented in Figure 2. The second network is similar to the first but the elements, except the agent, have different names: *Joe, Pluto, RedDoor,Room B*. Also, we fixed at 0.6 the current signal intensity for emotion Fear, to check the influence of emotion variations on the reasoning processes. The distance propagation decay factor was set to 0.5.

The first experiment tests the ability to recognize situations. As mentioned before, situations are described by

Scene Frames. A Scene Frame contains several elements that describe the represented scene. Situation recognition is achieved by measuring the presence of active elements into a specific Scene Frame. To determine the strength of a Scene Frame we use Equation 2.

Firstly, the strength of the current scene present in the agent's short term memory, is computed using Equations 1 and 2, resulting in 0.73. Then, after the resource propagation process, the following resources become active: $Fido = 0.15$, $YellowDoor = 0.18$, $RoomB = 0.23$. As JoeMeetsFido is a Scene associated with emotion "Fear" with factor 0.8 and that emotion has a current value of 0.6, the final Scene Strength become: $0.31 + (0.31 * 0.8 * 0.6) = 0.46$. Thus, we can see that JoeMeetsFido is activated with strength of 0.46 and become available on the active resources list for other mind modules. Finally, given that 0.46 is greater than $0.37(0.73 * 0.5)$, and there is no other Scene with greater strength than that, JoeMeetsFido was recognized as the current Scene.

The second experiment shows the creation of new action plans. Basically, we will use the list of active resources identified in the scene recognition experiment to create a goal planning. We also inserted into the agent's mind a Story Frame, called FidoBitesJoe, with two Scene Frames, where Fido bites the agent. So, the agent will use the first Semantic Network (resources activated in the first experiment), composed by Pluto, Red Door, etc., to recognize the scene FidoBitesJoe. In the first scene the dog is in front of the agent. In the second scene the agent is near the dog and wounded, because Fido attacked the agent. The action "Bite" connects both scenes. The final scene was associated with "Fear" with a factor of 0.98 and was classified as "Critical". After starting the mind cycle, the agent recognizes the current scene, emerging the FidoBitesJoe Story Frame (with the Fear boost), and detected a critical situation. So it started searching for a Good or Neutral Scene Frame to pursue. As there aren't scenes with that situation, there is nothing to plan and the process stops here.

So, we added another Story Frame with the same first Scene Frame of the FidoBitesJoe, but we connected that Scene with another that has a "Neutral" situation, using action "RunTo (target = neighbor room)". Now, the second Scene has no dog near the agent. After executing again, the second Story Frame emerged and "RunTo" was added to Goal Manager as a short term goal to be pursued, once that action moved the agent from a "Critical" situation to a better one. At the next step the goal "RunTo" was added to the Action Manager to be executed, ending the experiment.

Conclusions

In this work we presented some of our efforts in creating a modular and very customizable synthetic artificial mind architecture for virtual actors in IS Systems. We have detailed each part of the model, explained how they work together and presented some experimental results obtained using a prototype implemented for that purpose.

In order to prepare a virtual actor to interpret a specific role in a real application, it is necessary to do some knowledge engineering to build a collection of memory fragments that will be used by the agent to reason and make decisions based on its personality and the past events of its 'life'. This collection must also contains basic commonsense memories, required by the agent to interact with elements of the virtual world and use them to achieve its goals.

We believe that the proposed model is a good solution to create autonomous agents that can be used as characters of interactive stories. Once the agent's personalities, memories and some goals were defined, the agents can be inserted into a dynamic environment to pursue its goals, to interact among them and with a user, to conduct different kinds of narratives. It is a very flexible and simple artificial mind model, which can be customized and used with several other tools and resources. For example, a director AI can be built to modify some environment attributes to conduct agents to a specific, and desired by Story Builder, goal.

The next step of this work, will be using a complete implementation of this module in a graphical environment to observe the behavior of the agents. We hope this work can become a relevant tool to create IS systems.

References

Aylett, R.; Louchart, S.; Dias, J.; Paiva, A.; and Vala, M. 2005. Fearnot! an experiment in emergent narrative. In *IVA*, 305–316.

Aylett, R.; Louchart, S.; Tychsen, A.; Hitchens, M.; Mata, C. D.; and Figueiredo, R. 2007. Managing emergent character-based narrative. In *INTETAIN '08*, 1–8. Brussels, Belgium: ICST.

Brachman, R., and Levesque, H. 2004. *Knowledge Representation and Reasoning*. Morgan Kaufmann.

Crawford, C. 2005. *On Interactive Storytelling*. New Riders, first edition.

Figueiredo, R.; Paiva, A.; and Aylett, R. 2007. Facilitating the emergence of educational stories. In *Workshop on Narrative Learning Environments - AIED*, 24–32.

Gamma, E.; Helm, R.; Johnson, R.; and Vlissides, J. 1995. *Design Patterns*. Addison-Wesley Professional.

Goertzel, B., and Pennachin, C. 2007. *Artificial General Intelligence*. Springer, first edition.

Minsky, M. 1985. *Society of Mind*. Simon & Schuster.

Minsky, M. 2006. *The Emotion Machine*. Simon & Schuster, first edition.

Ortony, A.; Clore, G. L.; and Collins, A. 1988. *The Cognitive Structure of Emotions*. Cambridge University Press.

Picard, R. W. 1997. *Affective Computing*. Cambridge, Massachusetts: The MIT Press.

Quillian, R. M. 1968. Semantic memory. In Minsky, M., ed., *Semantic Information Processing*. Cambridge, MA: MIT Press. 227–270.

Reilly, W. S. N., and Bates, J. 1996. *Believable Social and Emotional Agents*. Ph.D. Dissertation, CMU.

Wang, P. 1995. On the working definition of intelligence. Technical report, Temple University.

Wang, P. 2006. From nars to a thinking machine. *AGIRI Workshop*.