Scenario building based on formal methods and adaptative execution

R. Champagnat, A. Prigent and P. Estraillier

L3i, Université de La Rochelle
Pôle Science et Technologie
17 042 La Rochelle Cedex 1
France

Tel : +33 546 513 967
Fax : +33 546 458 289
{ronan.champagnat,armelle.prigent,pascal.estraillier}@univ-lr.fr

February 1, 2005

1 Introduction

The field of games has been subject of a detailed attention these last years, that it is in the field of data processing, whether of that of creation or the design. From this expansion, several classifications arise for video games according to historical, leading or narrative criteria's. Generally, games are classified in games of action, adventure, strategy or of sport.

[VN03] proposes a classification based on game mechanisms. First of all it distinguishes the case from the games "solo", for which the player only plays against the computer, of that of the games "multi". One then speak on the one hand about *cooperative game* where the players play together against the computer, and on the other hand of *competitive games* where the players face each other, alone or in a team. Each one of these classes is then divided into subclasses: puzzles, strategy games, action games and adventure games. If in a "solo" strategy game the player has the feeling that he controls its environment, in a "multi" strategy game each player influences the environment and thus the possible actions of the other players.

Interactive narrative: existing approaches. Jenkins [Jen03] divides narration for games depending on its importance on the gameplay:

- *Evoked narrative*, in which elements from a known linear narrative are included in the spatial design of the game (*e.g.* Star Wars Galaxies);
- *Enacted narratives*, organized around the player's movement through space (*e.g.* adventure games);
- *Embedded narratives*, in which narrative events (and their consequences) are embedded in a game space such that the player discovers a story as they progress through the game (*e.g.* Half-Life);

- *Emergent narrative*, game spaces enabling players to make their own stories (*e.g.* The Sims).

Drawbacks. Validating techniques differe depending on the choosen kind of narration. For *evoked narrative* and *enacted narratives*, the verification of scenario should be achieve before the specification phase. It implies that all the possibles events should be taken into account, and the validation will be limited by the state space explosion.

Embedded narratives and *emergent narrative* can be validating dynamically through the unfolding of the game. This is why we are focusing on this particular kind of narrative.

The *center for virtual environment* of Salford university works on analysing scripting either in the field of games, whereas in the field of pedagogical tools. There work is based on the control of the unfolding of a story. For instance, for pedagogical tools, the user is constrained in order to follow paths (in order to garantee that he has learn what he has to, and to be evaluated...). On the other hand, if the user has no freedom its interest for gaming will decrease (specially for gaming).

The problem is then to define a story that can be controled with offering freedom for user actions. *Emergent narrative* has been introduced to deals with it. This notion is based on an improvise model rather than a scripting model. *Emergent narrative* start for an initial state, characters (personality, role) and produces interaction (non scripting interactions) that make the story [AL04, LA04c].

The development of complex system are subject to a tight integration of the formal methodologies into the existing software development cycle, in order to increase design quality. As scenarii are getting more and more complicated their design become close to complex system's design. As a consequence the introduction of formal methods for game scenarii development is a way to improve the game quality.

Within the framework of this paper, we are interested in "solo" strategy games. In opposition to a fixed writing of the scenario, an emergent form of the narration stand in mechanisms of construction "on the fly" of the scenario. We propose to improve such mechanisms by formal approaches of modeling and validation to check basic properties of the game and the coherence of the derived scenario.

Contribution. In this paper, a method of adaptive execution of a strategy game is introduced.

Our aim is to define an agent system that analyse the former events of a script then analyse the future narratives and modify the script in order to remove the unexpected narratives.

- First, we present **a linear logic modeling for the validation** of concurrent interactions within the framework of a "solo" strategy game. We suggest a terminology for game analysis. And we introduce a new method for a linear logic model validation. It consists in translating this model into a Petri net in order to be able to generate the reachable sequences from a given scenario.
- In addition to the common properties checked for a system such as liveness, safety, we propose **a new class of properties** allowing to state on the relevance of the script.
- Finally, we are interested in **an dynamic construction of a scenario** during the unfolding of the game. Our approach is based on local analysis of the game and "on the fly" of the resources localization in this game.

Toy example. In order to illustrate our approach we will use, in the sequel, a toy example based on an exploration game.

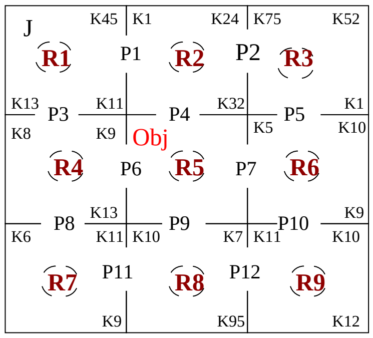

Figure 1: Map of the exploration game

Aim. It is a strategy and action solo game where a player should move around a world made up of 9 rooms and searching for a treasure. Figure 1 show the distribution of the 9 rooms and the localization of the keys (used by the player to move from one room to a adjacent one).

Rules. To achieve its quest, the player must find keys. These keys are used for opening doors between rooms. The game end when the player finds its goal (treasure).

Each key open a door or a set of doors. They are for single use, and the door is closed after the player goes through. Then a key can be used one and only one to cross a door.

A non-player character (named *ghost*) moves through the map. It can steale keys in order to block the human character (the player). But, it can also loose keys in order to give new paths for the player.

Game strategy. The human player issues are twice: he has to find keys to move around the map (and he should be fast enough in order to avoid ghost robbering); and find its goal.

Outline. After having described the existing approaches within the framework of the narration and the analysis of the scenarios, we propose a description of the terminology which we will use. Then, we define a classification of the verifiable properties for a scenario. The second section presents our approach of validation of the scenarios. After having carried out some recalls on linear logic, we present a first static method of analysis, then we describe an improvement of this approach consisting in validating the scenario during the unfolding of the game. We will finish on a presentation obtained results for the example of "solo" strategy game described previously.

2 Scripting and game properties

Before proposing our analysis method of the scenarios, we describe here the existing approaches within the framework of scripting and analysis of the games. We propose a terminology and a clasification of the properties to be validated on a scenario.

2.1 Previous approaches

2.1.1 Analysis of interactive narrative

The increasing complexity of scripting of the game allows editors to have interesting games (the players are the heroes of the game, they follow an increasingly complex scripting with various drama). Of this increasing complexity the need to validate the scenario to ensure the quality of the narrative screen proposed to the player. [VN03] underlines the absence of techniques of analysis within the framework of games. There we are interested in this kind of problem. Several teams face to interactive narration. We can quote the *"Liquid narration Group"* [You03a, YRB$^+$03, You03b, RY03], the *"Zero game studio"* [Ela02, Lin02], the university of Salford [AL04, LA04c, LA04a, LA04b], the university of Michigan [ML03], the university of Teesside [CLM$^+$03], the Oz project [Mat02]. All these works concern the difficulty in generating a coherent narration of game (compared to the rules of the game and the scripting) according to the player interactions.

2.1.2 Interactive drama

Emergent narrative means that it is the player that unfold the scenario and, then, create its own story. However, if the player is free to unfold the story, he may lead to unexpected end or go right through the end. In both cases its interest for gaming will decrease (because of its frustration with deadlock, or because the game is too easy). In addition to keep the player playing he has to feel emotions. This means that the drama of the story should be manage in order to modify the player emotion through the game (like aristotle's dramatic arc).

The *Artificial Intelligence Laboratory*, from Michigan university, works on the notion of interactive drama. It is defined as an atempt to use a computer to tell a story where the user is the main character. The deriving story should be affect by the user interactions.

They Interactive Drama Architecture is introduced in [ML03]. It consist in narrative games within the player interact and then modify the story. This architecture is based on the definition of a story director agent in charge of garanting the consistency of the story by:

- checking events by driving the story in order to avoid to compromise the future (respecting temporal constraints);
- using a predictive model to have detect inconsistent stories before they arise;
- modifing the behavior of characters in a way to force the user to return to one of the story (included in the narrative).

An other project is on interactive drama tension (ID tension project). It is based on managing the drama tension by using a narration agent. The aim is to generate interacting story [Szi99].

The interactive drama tension is made up:

- The story world: it contains the state of the world as it is currently described by the story. In an AI language, this is typically a base of facts.
- The narrative logic: it contains rules on how the world of the story can evolve.
- The narrator: it decides which action(s) should be proposed to the user.

- The user model: it is used by the narrator to choose the action according to its belief on user.
- The theatre: it manages the interaction with the user (display of actions, and capture of user actions).

We will base our approach on the concept of narrative agent. We will thus use an adaptative system that observes behavior of the player, analyse properties and compute an new valid scenario

2.1.3 Scenario analysis

Generally authors try to make coherent the unfolding of script execution. There is two ways for writing a scenario. The first one consist in anaylsing the scenario before game implementation. The second one is to perform "on the fly" analysis during the game execution. The former one is the choice of the *"Zero game studio"* team which uses causal graphs. The second, which consist in adding constraints which must remain valid during the game, such as, for example work, of the university of Teesside. All these works are generally based on Artificial Intelligence techniques and system point of view is never studied. It results from this that the concept of properties is used but no property is stated. Thus, the CNAM team [VN03, Véch] proposes the most recent approach for the game analysis. Their work concerns the video game analysis and specification by proposing a technique making it possible to guarantee that the actions carried out by the player remain coherent within the framework of the game. With this intention, they use techniques of subnet construction making it possible to guarantee that if each one of these subnet is coherent the whole remains coherent.

2.2 Vocabulary and concepts

The authors dealing with the subject of the games agree on one point: the vocabulary associated with the description of the narration in the games is not fixed and is still prone to discussion. We define here the various terms and concepts which we chose for our research. This terminology will be then applied within the framework of our illustrative example.

2.2.1 Terminology used

Action.

An action is an interruption caused by a reaction of the user on one of the input of the game (e.g. keyboard, mouse...), and of the adaptive system.

Event. Although it is often a question of sequence of events, this very important concept is never defined in a precise way. We choose the following definition:

An event is a non preemtive atomic element of a history.

We can distinguish several kinds of events:

- *player events* defined in the rules of the game which are proposed to him at the beginning (*e.g.* crossing a door).
- *adaptative system events* which arise within the framework of the control of the game unfolding (*e.g.* appearance of a dragon when the player enters a room).

Narrative. Starting from the event concept we can give the definition of a narrative. A narrative is comonly regarded as a history. But this concept of history is specified in various manners. For instance [LA04b] considers the theoretical history as being that which the author wishes to see, in opposition with the history (that which is held). [ML03], although, regards the history as being all the events being able to appear and the constraints between the events. That can also be seen like the narrative diagram. These remarks show that the narrative is in something which represents the whole of the possible stories (the narration for [Szi99], the world of the stories for [RY03]).

In existing work it has a confusion between the terms of narrative and history. This confusion results from the various existing manners to describe the possible stories. We then define the concept of narrative as:

A narrative is an ordered sequence of events.

As we are not focusing on the concepts of explanation of accounts, it is not necessary to model information on actions.

This first set of definition allows to describe the game story and its unfolding. The following definitions will deal with user interaction.

Resource.

A resource *can be* active *(with an associated behavior) or* passive *(its state remains constant throughout game). Independently it is* active *or* passive*, it can be* local *(it belongs to only one actor) or* shared *(this resource can be used by whole or part of the actors and is support of a interation).*

Actor.

An actor is an entity involved in events of the scenario insofar as it is characterized by its own events and prone to interact.

A particular kind of actor is the avatar representing the human player (called *player* simply). These actions will induce events and will have an influence on the unfolding of the story. Another kind of actor is that of the entities controled by the adaptive system, for instance *non player character* or NPC.

Interaction.

An interaction corresponds to an action carried out of an actor towards another actor and modifying the state of this last. An interaction uses a shared active resource. It can be synchronous or asynchronous.

Adaptative system. To implement a scenario, we define a system made up of three functional agents allowing to modify the actions of the non player charcater and the resources.

The adaptive system is the entity responsible of the game unfolding control

It is made up of:

- *Observant agent* which observes the player actions;
- *Analyzer agent* which analyzes the possible set of narratives;
- *Scripting agent* which modifies the possible events.

Scenario.

A scenario is the set of all ordered sequences of events. It is possible to take into account constraints relating to the actions.

An improvement of this terminology is to take into account in the concept of scenario, the concepts of resources and actor. Thus the sequences will depend on event occurence, the state of the actors and the availability of resources.

2.2.2 Toy example, terminology

We describe here, the application of this terminology within the framework of the exploration game.

- **Event**: four class of events are used in this game.
  - getting a key by the player or by the ghost;
  - crossing of a door;
  - achieving the player goal (to get its objective).
- **Narrative**: an example of narrative is "getting the key K_{12} by the player"; "getting the key K_{13} by the ghost"; "crossing the door D_2 by the player"; "achieving the player goal".
- **Ressources**:
  - *shared passive resources* correspond to keys;
  - objective is a *passive local resource*;
  - elements of decoration are *passive shared resources*;
  - the ghost is an *active shared resource.*
- **Actor**: this game reveals two actors (the player and the ghost).
- **Interaction**: there is no interaction in this game. However one could imagine an improve version where a possible interaction would be that the player can eliminate the ghost.
- **Scenario**: it is described by an initial state, the events which can appear during the game and a desired final state. The initial state here is characterized by the position of the player (room R_1), the keys localization (K_{32} located in room R_2...) and the objective localization (Obj in in room R_5).

We can notice that in this example the player is free to modify the world. In this example, the scripting agent determines the resources localisation (*e.g.* keys) allowing the player to carry out his objective. The scripting agent can modify this positioning through the unfolding game in order to increase the dramatic intensity of the game.

2.3 Game properties

We define here the set of properties that should be verified in the framework of the scenario validation before their implementaton. In order to build the the properties to be validated on the system, we use the traditional classes of properties such as safety or liveness.

However, the verification of a scenario requires to take into account another set of properties directly related to the relevance of this scenario. Indeed, we must on the one hand guarantee that the execution of the game will not carry out the player in an erroneous state or of deadlock (is the game correctly ended?, is it possible for the player to win?, is there exists a non-reachable room...) and in addition to establish some conclusion on the quality of the scenario (is the game not too *"easy"*...). It is then possible to distinguish two kinds of properties. One of it relates to the *playability* of the scenario (*i.e.* the game is running correctly), the other kind of properties allow us to evaluate the *relevance* of the scenario (*i.e.* the scenario is interesting). We describe here these two kind of properties.

playability properties The checking of the *playability* within the framework of the game is very strongly connected with the checking of classic complex systems. We check here properties of safety, liveness, reachability or deadlock free. We recall here, the definition of these properties.

- **Safety:** express that under some hypothesis, an event can never occur. For example, it should be guaranteed that the player cannot cross the door without holding the key.
- **Liveness**: express that under some hypothesis an event will occur. For example, the player will be enabled to take the objective.
- **Reachability**: express that a state of the system can be reached. For example, the player can reach the room where its objective is.
- **Deadlock free**: express that the system will be never in a situation where it cannot progress any more. For example, to guarantee that the player will never be blocked in a room.
- **Fairness**: express that, under some hypothesis events will infinitely often occured (or will not). For instance, the player can open a given door infinitely.

Relevance properties. One of the originality of our approach holds in the definition of a class of properties specific to the field of the game. Indeed, it is not enough that a scenario is playable to ensure the quality of it. One can consider that a game is useful (or valid) if it is ludic, sufficiently difficult so that the player is not bored, and if this scenario does not support one of the players.

- **Impartiality**: the players have equal chances (or almost) to win the game. This property can also take into account of the level of strategy requested from the player. Indeed, in the case of a game "solo" it is then possible to consider that a game respecting impartiality is a game where the human player does not win the game in 100% of the cases, but in 50% to 60% of the cases.
- **complexity**: the execution of the game does not contain stories which carry out to the victory too quickly.

- **Concurrence**: the game has a particular interest when the actors are in competition in the progression of the game (event) or in the division of the resources. For this kind of property one have to ensure that there exists some situations where the actors are in direct competition to take an object, or to cross a door. It is this type of situation which can possibly makes the game *ludic*. This same property can relates to competition between a human player and a non-player character. It is thus possible to satisfy this property by determining the events of the NPC so as to create a competition between him and the human player. question here, to be satisfied theory of large numbers, but to control with short expiry, the process of impartiality. Indeed, it is recognized that many plays arouse the interest of a player only during a given duration.

These properties (in particular relevance) are *relatives properties*. Indeed, it is necessary to determine before the verification step which is the driving number of executions minimal to the victory. Moreover, it is almost impossible to obtain a scenario for which the total impartiality is obtained. necessary to try to obtain for each player possible rate of success a most similar to 50 percent. The objective announced here is not only to validate a given scenario but to determine the best possible scenario, *i.e.* that which satisfies all the properties.

2.3.1 Toy example, properties

Within the framework of the game of exploration made up of nine rooms, we can define the following relevance properties:

- **Complexity** : The game is sufficiently complex if the player crosses between 4 and 7 scenes before finding his objective.
- **Concurrence** : The game presents a sufficient competition if the player and the NPC are at least 2 times in the same room at the same time.
- **Impartiality** : In each state of the system, the player has 60 % of chances to win the room.

We will show in thispaper how the actions of the ghosts must be generated. Moreover, we must guarantee that the actions of the ghost do not produces unexpected blockings in the game. With this intention, we base the analysis of the game on this whole of properties to be validated. We describe these concepts in the following section.

3 Scenarii analysis

This section presents the properties analysis methodology described previously. First, a linar logic based model of scenario is presented. Then the translation process form linear logic to Petri nets is described. Then paths can be identified as ordered sets of event from the scenario. As a consequence we deduce all the possible narratives. A static analysis can thous be performed.

This static analysis carry to the state space explosion problem. Thus a dynamic local analysis method is introduced. It is performed through the game processing and is based on the concepts of the static analysis.

Finally results on the toy example are discussed.

3.1 Design analysis

The scenario of the game is modelled by a sequent of the linear logic. Each event, such as door crossing and getting key, is modeled by an implication formula. The sequent is prooved by means of Petri net in order to generate the set of narratives for the given scenario.

3.1.1 Linear logic basics

Linear logic was introduced by J.Y. Girard [Gir87] as a restriction of classical logic. It is well suited for modeling dynamics system with resources. In linear logic formulas are treated as resources which are produced and consumed. Every formula should be used exactly once in the derivation.

Classical logic is commonly used to reasonning on eternal truth. On the contrary the advantage of using linear logic is not to say if an assertion is true or not but the way how the proof is achieved and the sequence of choices done.

Prooving a sequent consists in rewriting the sequent by replacing connector by their introduction rules until it remains only the meta-connector ",". A sequent is valid if all the finishing path of the proof tree are of identity rules.

A sequent will be written with the initial state of the game and expecting events in the left part and the obejctives in the right part.

3.1.2 Modeling

Six kind of atoms and four kind of events appears from the game modeling. J_{R_i}'s atom represent the player locate in room R_i. k_x represents the key for opening the set of doors x. k_{xJ} represents player J having the key for opening the set of doors x. Obj the objective of player one and END the end of the game.

The event *getting a key* allowing to open the set of doors x by the player is:

$$\text{Getting a key: } J_{R_i} \otimes k_x \multimap J_{R_i} \otimes k_{xJ}$$

Moving the player from room i to room z correspond to the event *crossing a door* of the set x using key k_{xJ} is:

$$\text{Crossing the door: } J_{R_i} \otimes k_{xJ} \multimap J_{R_z}$$

The event *achieving goal* is modeled by:

$$\text{Achieving goal: } J_{R_i} \otimes Obj \multimap END$$

A scenario corresponds to sequent:

$$J, Obj, K, E \vdash END$$

Where J is the player, Obj the objectives, K the set of key's atoms, E the set of events and END the end of the game.

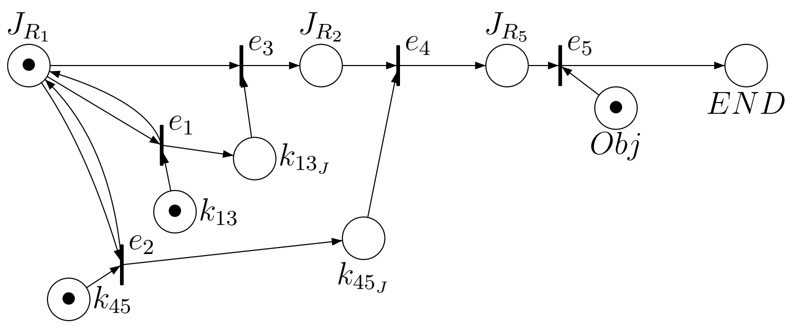

Figure 2: Petri net of the short scenario of the toy example

A scenario as a sequent gives all the possible narratives. In order to have a good game, we will define, in the sequel, all the properties a scenario must reach.

In the following section, we will present how a linear logic based analysis is used to chek properties.

It is based on sequent calculus. [Tam94] gives various strategies for deriving proof. We will follow the one given by P. Küngas [Kü01, Kü02]. As F. Girault show in [Girch], a sequent proof corresponds to a transition firing.

This methodology gives us the set of all the narratives for a scenario. The analysis is then based on this set.

3.1.3 Deriving Petri net: methodology

[PCVK99] propose a methodology for deriving Petri net from linear logic:

- Each atom correspond to a place. An atom true means a marked place.
- An implicative formula is a transition of the net.
- The left side of the sequent is made up implicative formulas and propositional formulas. It represents the set of transitions to be fired and the initial marking.

  The propositional formulas of the right side of the sequent is the final marking.

As an example we will consider the following model:

$$Obj, J_{R_1}, k_{13}, k_{45}, J_{R_1}\otimes k_{13} \multimap J_{R_1}\otimes k_{13_J}, J_{R_1}\otimes k_{45} \multimap J_{R_1}\otimes k_{45_J}, J_{R_1}\otimes k_{13_J} \multimap J_{R_2}, J_{R_2}\otimes k_{45_J} \multimap J_{R_5}, J_{R_5}\otimes Obj \multimap END \vdash END$$

The derived Petri net is given figure 2.

Proof of the sequent Proving the previous sequent corresponds to find a transitions firing involving transitions e_1, e_2, e_3, e_4, e_5 drawing from state places J_{R_1}, k_{45}, k_{13} and Obj marked to state where place END is marked.

A candidate sequence is $(e_1; e_2; e_3; e_4; e_5)$. It corresponds to the following proof:

$$\dfrac{\dfrac{\dfrac{}{J_{R_1} \vdash J_{R_1}}\,id \quad \dfrac{}{k_{13} \vdash k_{13}}\,id}{J_{R_1}, k_{13} \vdash J_{R_1} \otimes k_{13}}\,\otimes R \qquad B}{Obj, J_{R_1}, k_{13}, k_{45}, J_{R_1} \otimes k_{13} \multimap J_{R_1} \otimes k_{13_J}, J_{R_1} \otimes k_{45} \multimap J_{R_1} \otimes k_{45_J}, J_{R_1} \otimes k_{13_j} \multimap J_{R_2}, J_{R_2} \otimes k_{45_j} \multimap J_{R_5}, J_{R_5} \otimes Obj \multimap END \vdash END}\,\multimap L$$

Where B is:

$$\dfrac{C \qquad \dfrac{\dfrac{\dfrac{}{J_{R_1} \vdash J_{R_1}}\,id \quad \dfrac{}{k_{13_J} \vdash k_{13_J}}\,id}{J_{R_1}, k_{13_J} \vdash J_{R_1} \otimes k_{13_J}}\,\otimes R \qquad \dfrac{\dfrac{\dfrac{}{J_{R_2} \vdash J_{R_2}}\,id \quad \dfrac{}{k_{45_J} \vdash k_{45_J}}\,id}{J_{R_2}, k_{45_J} \vdash J_{R_2} \otimes k_{45_J}}\,\otimes R \qquad \dfrac{\dfrac{\dfrac{}{J_{R_5} \vdash J_{R_5}}\,id \quad \dfrac{}{Obj \vdash Obj}\,id}{J_{R_5}, Obj \vdash J_{R_5} \otimes Obj}\,\otimes R \quad \dfrac{}{END, \vdash END}\,id}{Obj, J_{R_5}, J_{R_5} \otimes Obj \multimap END \vdash END}\,\multimap L}{Obj, J_{R_2}, k_{45_J}, J_{R_2} \otimes k_{45_j} \multimap J_{R_5}, J_{R_5} \otimes Obj \multimap END \vdash END}\,\multimap L}{Obj, J_{R_1}, k_{13_J}, k_{45_J}, J_{R_1} \otimes k_{13_j} \multimap J_{R_2}, J_{R_2} \otimes k_{45_j} \multimap J_{R_5}, J_{R_5} \otimes Obj \multimap END \vdash END}\,\multimap L}{Obj, J_{R_1}, k_{13_J}, k_{45}, J_{R_1} \otimes k_{45} \multimap J_{R_1} \otimes k_{45_J}, J_{R_1} \otimes k_{13_j} \multimap J_{R_2}, J_{R_2} \otimes k_{45_j} \multimap J_{R_5}, J_{R_5} \otimes Obj \multimap END \vdash END}\,\multimap L$$

where C is:

$$\dfrac{\dfrac{}{J_{R_1} \vdash J_{R_1}}\,id \quad \dfrac{}{k_{45} \vdash k_{45}}\,id}{J_{R_1}, k_{45} \vdash J_{R_1} \otimes k_{45}}\,\otimes R$$

The corresponding narrative is: "getting key k_{13}" then "getting key k_{45}" then "crossing door D_1" then "crossing door D_4" then "achieve its objective".

3.1.4 Discussion

The state graph algorithm allows to generate all the possible sequence of events starting from a marking. But in order to reduce state space explotion, we will rather use a token player algorithm which will unfold only the desired events of the sequence.

It should be said that we have developped two tools: one for deriving Petri net from sequent of linear logic; and one for deriving all the ordered sequence of events for a given scenario.

Toy example verification. Let us validate the following scenario.

Required events:

- $$e_1 : J_{R_1} \otimes k_{13} \multimap J_{R_1} \otimes k_{13_J}$$

- $$e_2 : J_{R_1} \otimes k_{45} \multimap J_{R_1} \otimes k_{45_J}$$

- $$e_3 : J_{R_1} \otimes k_{13_J} \multimap J_{R_2}$$

- $$e_4 : J_{R_2} \otimes k_{45_J} \multimap J_{R_5}$$

- $$e_5 : J_{R_5} \otimes Obj \multimap END$$

Resources: $Obj, J_{R_1}, k_{13}, k_{45}$.

The sequent is given by the following sequent:

$$Obj, J_{R_1}, k_{13}, k_{45}, e_1, e_2, e_3, e_4, e_5 \vdash END$$

In order to have a more readable proof, implicative formulas are replaced by the name of there corresponding event.

$$\dfrac{\dots \quad \dfrac{\dots \quad \dfrac{\dots \quad \dfrac{\dots \quad \dfrac{\dots \quad \dfrac{}{END, \vdash END}\,id}{Obj, J_{R_5}, e_5 \vdash END}\multimap L}{Obj, J_{R_2}, k_{45_J}, e_4, e_5 \vdash END}\multimap L}{Obj, J_{R_1}, k_{13_J}, k_{45_J}, e_3, e_4, e_5 \vdash END}\multimap L}{Obj, J_{R_1}, k_{13_J}, k_{45}, e_2, e_3, e_4, e_5 \vdash END}\multimap L}{Obj, J_{R_1}, k_{13}, k_{45}, e_1, e_2, e_3, e_4, e_5 \vdash END}\multimap L$$

This proof corresponds to the following narrative: $(e_1; e_2; e_3; e_4; e_5)$.

From this proof tree, we only focus on the right side path. The following graph can then be deduced.

$$\begin{tikzcd}
 & & & & Obj, J_{R_5} \arrow[r, "e_5"] & END \\
 & & & Obj, J_{R_2}, k_{45_J} \arrow[ur, "e_4"] & & \\
 & & Obj, J_{R_1}, k_{13_J}, k_{45_J} \arrow[ur, "e_3"] & & & \\
 & Obj, J_{R_1}, k_{13_J}, k_{45} \arrow[ur, "e_2"] & & & & \\
Obj, J_{R_1}, k_{13}, k_{45} \arrow[ur, "e_1"] & & & & &
\end{tikzcd}$$

Figure 3: Narrative example

This narrative is not the only one that can be deduced from the scenario. Two other ones can be produced:

- $(e_1; e_3)$ which is not a valid one, and
- $(e_2; e_1; e_3; e_4; e_5)$.

As a consequence one can write the following graph (figure 4) that gives all the feasible narrative for a given scenario (the graph scenario).

This analysis process gives us the consequence of modifying events of an original scenario. Deduced narratives for this new scenario gives a new graph that can be analysed in the same way.

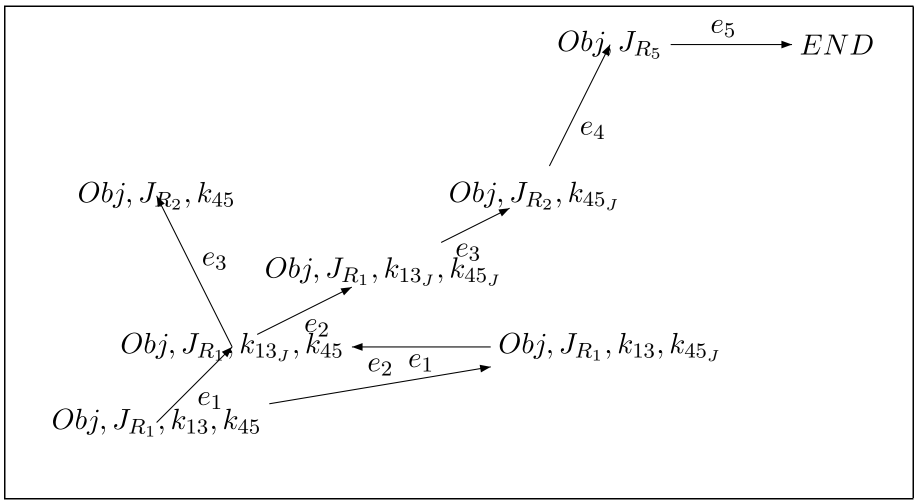
Obj, J_{R_5}
e_5
END
e_4
Obj, J_{R_2}, k_{45}
Obj, J_{R_2}, k_{45_J}
e_3
e_3
$Obj, J_{R_1}, k_{13_J}, k_{45_J}$
e_2
$Obj, J_{R_1}, k_{13_J}, k_{45}$
$Obj, J_{R_1}, k_{13}, k_{45_J}$
e_2
e_1
e_1
$Obj, J_{R_1}, k_{13}, k_{45}$

Figure 4: Scenario graph

The complete analysis of the toy example generate over than 46.000 narratives on about 30 hours. This is because the ordering of event algorithm is exponential.

As a consequence this method is not suited while playing (allthough it can be use to study the relevance of a scenario in a specification phase). A dynamic or local analysis is preferred when it is performed while the execution of the game.

3.2 An analysis method during game execution

We proposed a method for analyzing the quality of a scenario before its implementation. However, we showed that this approach is limited by the state space explosion problem related to the scheduling of the events. In order to solve this problem, we propose here second approach consisting in not generating the whole proof tree. Indeed, here the proof tree is generated "on the fly" during the analysis by considering a limited window of events.

We use, an adaptive system made up of a observant agent, a analyzer agent and a scenario agent. The analyzer agent performs local analyses (on a restricted window of event) during the execution of the game and the scenario agent modifies the ressources.

As it has been explained previously, the game is sufficiently complex if the player crosses between rooms R_4 and R_7 before finding his objective. Thus, we consider in this example, a window of size 4 (*i.e.* of 4 events).

The objective here is to determine the actions of the non-player character (*e.g.* the ghost) according to the stories which we wish to eliminate because they do not respect the properties described in section 2.3.1. Thus, we wish that the ghost "steal" some keys when the player is in the room where the key in question is.

3.2.1 Interative analysis method

The method of game interactive analysis consists in analyzing the possible stories during the the game execution. In order to propose the interest of this technique we use a tool for simulation of the course of the game. This last unrolls the execution directly starting from the Petri net derived from the model in linear logic. This simulator proposes to the player a continuation events, and this last carries out a choice. This method of analysis is described with the figure 5 and works in the following way:

1. the preliminary step consists in building the whole of the narratives made up of 4 events starting from the initial state of the game. The analysis of these narratives produces a set of ones leading on the one hand to blockings and on the other hand to a premature termination of the game.

2. the following step is to determine which are the resources responsible for these narratives erroneous. We then remove manually these resources. We simulate the intervention of the ghost here; that one will steal resources during the game. When these resources were removed, we obtain the new model in linear logic, the game can then start and the new Petri net corresponding to this model is built.

3. the simulator of the game unfolds the past fired transitions from this Petri net and proposes the events with the player. When he has carried out its choice, two cases can arise:

    - the selected player to get a key: in this case the execution of the game continues.
    - the selected player to cross a door: in this case, it is necessary to carry out a calculation of the possible accounts after the crossing of this this door. The events already carried out by the player are withdrawn from the model in linear logic (it is impossible that a player consumes a resource twice). Stage 1 is carried out again and the method is unrolled by regarding the state reached as the initial state of unfolding of the game.

3.2.2 An exemple of adaptative simulation for the exploration game

We describe here an example of simulation and analysis during the simulation of this game. We will show here a deadlock and a non-relevant scenario highlighted (and thus avoided) during the first two step of the adaptative simulation of the game.

Step 1: preliminary validation and the first simulation. Initially, we generate the whole of the possible narratives made up of 4 events, starting from the initial state of the game (*e.g.* the player in R_1). We obtain thus 577 possible narratives (length 4).

- **Analyzing complexity.** On the whole of these narratives, we can note that 33 narratives carries out the player to his objective. This result is not valid because the game is too easy and thus is a non-relevant case. These scenarios correspond in the crossing from room R_1 to the room R_2 which makes it possible to reach room R_5 (with the keys k_{45} and k_{24}) and thus to reach the objective. The solution is to require of the ghost to hold keys k_{45} and k_{24} located respectively in rooms R_1 and R_2.

- **Deadlock analysis.** In this case, no deadlock is detected among the stories of length 4.

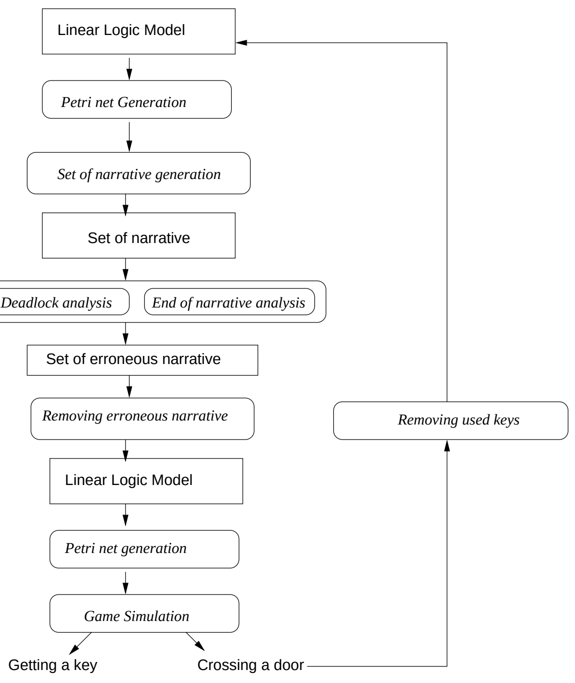

Figure 5: Dynamic analysis method

After removing keys k_{45} and k_{12} (in the final game, the order will be given to the ghost to remove these keys), we obtain a new model in linear logic. The new Petri net is generated and the game simulation is launched. The events suggested to the player are those represented in figure 6. In this figure, the choices carried out at the time of simulation are surrounded in dotted lines.

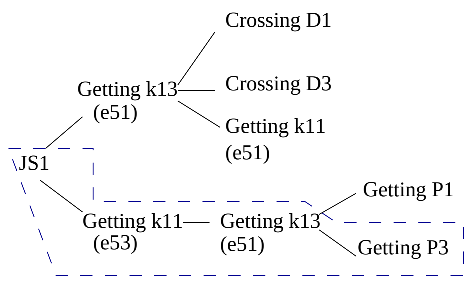

Figure 6: Possible choice to the player in the 1st step of the simulation of the game

Step 2: First door crossing. The player chooses to cross door D_3 (going in room R_4). The set of keys is computed another time, and deadlock analyzis produced the narrative described by the graph of the figure 7.

Consequently, before starting again simulation and proposing to the player to make the choices among the feasible events in the room F_4, we choose to remove the key k_{13} and the corresponding event. This event will be carried out by the ghost and visible by the player. The new linear logic model then makes it possible to generate the Petri net who will be carried out at the time of simulation. This simulation thus begins with an initial state: J_{R_4}.

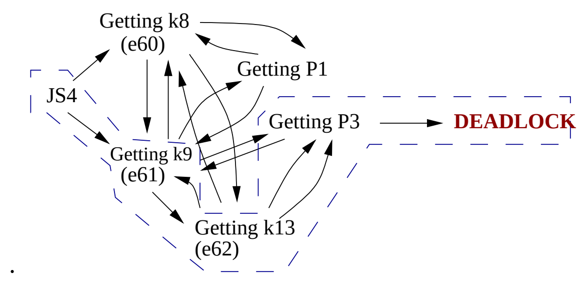

Figure 7: Possible deadlock when entering room 4

These steps are carried out until termination of the game.

4 Conclusion

This paper deals with the specification and analysis of a scenario in a game. It first underlines the needs for analysis and validate. Validation is required either for design purpose either to improve the playability of a game.

It allows to define in a formal way the properties of a scenario of a game. Then it is possible to derive automatic analysis and automatic improvement.

This paper is consider a toy example of a "solo" strategy game where the player is face to an adaptative system that manage the story by means of an observant agent, an analyzer agent and a scripting agent.

There are two main contributions. First a model for scenario in games based and linear logic. Second an analyze and validation of scenario based on linear logic modeling.

Initially we carried out a static analysis. The computing time of this one is of approximately 30h. The a dynamic approach makes it possible to avoid the state space explosion insofar as one never generates the whole of the possible narratives. The analysis carried out for a window of size 4 events during the simulation of the game lasts approximately 0,5 second. This computing time does not induce delay on the course of the game.

In this context we developped a tool set that allows us to produce a Petri net from a linear logic model; and deriving all the reachable narratives from this one.

In this approach we did not take into account of the bond between action and event. Indeed we consider that an action generates an event automatically and that an event is inevitably derived by an action. In a case where one would be interested in the model of execution, this assumption is not satisfactory. Actually it should be taken into account the fact that an action can generate several events and that an action can not generate any event. A possible prospect consists in being interested in the bond between action and event.

Moreover, it would be necessary to determine a method making it possible automatically to calculate the avoid events of the game (according to the erroneous stories). We would wish, also, to interest us in the plays of strategy using an economic model (standard *Pharaon*). Indeed, the scenario is less constrained what leaves more freedom with the agent scripting of the adaptive system. Lastly, the concept of impartiality makes it possible to manage the dramatic intensity. If one has a profile of the player (number of part played and frequency) one can decide to modify impartiality property according to the experiment of the player.

References

[AL04] R. Aylett and S. Louchart. Towards a narrative theory of virtual reality. *Virtual Reality, special issue on storytelling*, 7, 2004.

[Ale94] V. Alexiev. Applications of linear logic to computation: An overview. *Bull. of th IGPL*, 2(1):77–107, 1994.

[BBF$^+$01] Béatrice Bérard, Michel Bidoit, Alain Finkel, François Laroussinie, Antoine Petit, Laure Petrucci, and Philippe Schnoebelen. *Systems and Software Verification. Model-Checking Techniques and Tools.* Springer, 2001.

[Chach] R. Champagnat. Optimizing sequence of transition firings in t-time petri net. In Y. Dallery, J.C. Hennet, and P. Lopez, editors, *MOSIM'O3 (MOdélisation et SIMulation)*, pages 94–100, Toulouse (France), April 2003 (in French).

[CLM$^+$03] F. Charles, M. Lozano, S. J. Mead, A. F. Bisquerra, and M. Cavazza. Planning formalisms and authoring in interactive storytelling. In *1st international conference on technologies for interactive di gital storytelling and entertainment*, Darmstadt (Germany), 2003.

[Ela02] M. Eladhari. Object oriented story construction in story driven computer gam es, 2002.

[Gir87] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[Girch] F. Girault. *Using linear logic to formalize Petri nets.* Thesis, Université de Toulouse III, Toulouse (France), December 1997 (in French).

[GPNV02] V. Gal, C. Le Prado, S. Natkin, and L. Vega. Writing for video games. In *VRIC*, Laval (France), 2002.

[Jen03] H. Jenkins. Game design as narrative architecture. In *First person: new media as story, performance and game*, Cambridge, 2003.

[Kü01] J. Küngas. Using linear logic planning to make knowledge bases reactive. In *Seventh Symposium on Programming Languages and Software Tools*, pages 135–148, Szeged (Hungary), June 2001.

[Kü02] P. Küngas. Linear logic programming for ai planning, May 2002.

[LA04a] S. Louchart and R. Aylett. Emergent narrative, requirements and high-level architecture. In *SETN*, 2004.

[LA04b] S. Louchart and R. Aylett. The emergent narrative theoretical investigation. In *Narrative and learning environments conference NILE*, pages 24–33, Edinburgh (Scotland), 2004.

[LA04c] S. Louchart and R. Aylett. Narrative theories and emergent interactive narrative. *IJCEELL*, 2004.

[Lin02] C. Lindley. The gameplay gestalt, narrative, and interactive storytelling. In *Computer games and digital cultures*, Tampere (Finland), June 2002.

[Mat02] M. Mateas. *Interactive drama, art, and artificial intelligence.* Ph.d. thesis, Carnegie Mellon University (School of Computer Science), Pittsburg (USA), December 2002.

[ML03] B. Magerko and J. Laird. Building an interactive drama architecture. In *1st international conference on technologies for interactive di gital storytelling and entertainment*, Darmstadt (Germany), March 2003.

[PCVK99] B. Pradin-Chezalviel, R. Valette, and L.A. Künzle. Scenario durations characterization of t-timed petri nets using linear. In *PNPM'PP (Petri Nets Performance Modeling)*, Zaragoza (Espagne), September 1999.

[RY03] M.O. Riedl and R.M. Young. Character-focused narrative planning. Submitted to Virtual Reality 2003, 2003.

[Szi99] N. Szilas. Interactive drama on computer: beyond linear narrative. In *AAAI Fall symposium on narrative intelligence*, pages 150–156, 1999.

[Tam94] T. Tammet. Proof strategies in linear logic. *Journal of Automated Reasoning*, 12:273–304, 1994. Also available as Programming Methodology Group Report 70, Chalmers University, 1993.

[VN03] L. Vega and S. Natkin. A petri net model for the analysis of the ordering of actions in computer games. In *X*, 2003.

[Véch] L. Véga. *Petri net based model for analysing the event sequences in video games.* Thesis, CNAM, Paris (France), November 2004 (in French).

[You03a] R. Michael Young. Steps towards a computational theory of interactive narrative in virtual worlds. Unpublished document, 2003.

[You03b] R.M. Young. Story and discourse: a bipartite model of narrative generation in virtual worlds. Submitted to Virtual Reality 2003, 2003.

[YRB$^+$03] R.M. Young, M.O. Riedl, M. Branly, A. Jhala, R.J. Martin, and C.J. Saretto. An architecture for integrating plan-based behavior generation with interactive game environments. Submitted for Publication in Journal of Game Development, vanlent@icts.usc.edu and niles@charlesriver.com, 2003.