Using Existing Research for the Creation of a Scene-to-Adventure Game System: A Literature Review

Ross Berkland

May 30, 2009

Contents

1 Introduction

Research surrounding text-to-scene systems has become increasingly widespread. The capabilities of these systems have evolved from the ability to find several pictures relating to a piece of text to the generation of fully animated 3-dimensional scenes. Researchers are also discovering a multitude of applications for such systems extending out to areas such as the legal and law enforcement fields. This is partly due to the rapid growth of digital media as a means for communication and entertainment. Another fast growing area of computer science is that of computer games. As is discussed in later sections of the paper, game development is becoming increasingly popular as not only an educational tool but also a career choice.

This paper discusses key elements from both the text-to-scene and computer game fields and combines them to identify potential components for use in the development of a scene-to-adventure game system. This will be done by examining 3 main areas of research. The first deals with the applications which create a relationship between natural language and digital media such as Parts-of-Speech taggers and Text-to-Scene systems. Secondly, the paper discusses the structures that computer games and their narratives typically tend to assume. Finally, various aspects of game development and design are discussed. While these are significantly different areas of research, we join them in a logical and meaningful manner to define a structure for the Scene-to-Adventure Game system.

2 Natural Language and Digital Media

Digital media such as computer generated movies, digital art and video games have become increasingly popular. However, modelling a 3D scene on a computer is generally more time consuming than describing that same scene in words. This has led to the development of a new field in computer science known as text-to-scene conversion. By taking natural language as input, these systems generate a digital picture or scene with the aim of depicting the scene as accurately as possible. Understanding these systems may require basic knowledge about how they function which is described below. This section also details some of the existing text-to-scene systems and various applications of the concepts behind such systems.

2.1 Structure of Natural Language

2.1.1 Parts-of-Speech Taggers

These Text-to-Scene systems are possible because of the structured nature by which all languages are inevitably governed. Development of Parts of Speech (POS) taggers has been active for many years and papers on the subject can be found dating back to 1980 as can be seen in [21]. The basic concept behind these taggers is that of annotation of text. The tagger will parse the given text and on discovering significant words, will tag them with some form of markup language, typically XML [5]. These annotations are important in development of the scene-to-adventure game system as will be seen later. As mentioned above and confirmed by [5], the interpretation of written text is necessary for the creation of a text-to-scene system. As a first step towards creating an autonomous text-to-scene system, [5] offer an evaluation of several of the freely available parts-of-speech taggers. The research also involves combining existing taggers for increased accuracy. However, accuracy of these taggers are not within the scope of this paper so much as the mere fact that these systems exist. The idea of annotating text for the purposes of a text-to-scene system is extended by [7] in which they offer a method for speaker identification in fiction books with approximately 80% accuracy. The approach taken in developing the system involves making use of a collection of parts-of-speech taggers (as mentioned above) for the purposes of giving clues for speaker identification. This system could be largely helpful in the development of a scene-to-adventure game system due to the large quantity of narration and dialogue in typical fiction books.

2.1.2 Temporal Nature of Fiction Text

The narration style occurring in most books rarely assumes a linear sequence with regards to the logical progression of events. The majority of fiction books available will typically contain flashbacks and scenarios in which the speaker describes an event which occurred in the past. A method for dealing with this temporal nature of text is offered by [6] where they describe a method for creating a time-line of events occuring in fiction text. While this system is worth mentioning in the current context, the subject of its accuracy does not fall within the scope of this paper.

2.2 Text-to-Scene Systems

2.2.1 Existing Text-to-Scene Systems

Many text-to-scene systems have been developed to date, most of which differ in their methodology and output. Typically a text-to-scene system can be described as one which takes text as its input and generates a scene depicting that text. Those systems which adhere to this definition can be placed in one of three categories regarding their output. That is to say that the current text-to-scene systems typically output either pictures (text-to-picture)[25], still 3D scenes [2, 20] or animated 3D scenes [13, 8]. These systems make use of the above mentioned parts-of-speech taggers in order to annotate the text in meaningful way with some markup language.

Text-to-Picture

A system is introduced by [25] in which they attempt to augment communication through use of a text-to-picture system. The system functions by first selecting those key-phrases from the given text which are most *picturable*[1]. It then attempts to find pictures which best match the key-phrases and position them in a meaningful way. While the system specified in [25] is designed for augmented communication, its simplicity may have prevented it from reaching its goal. Using nothing but pictures to represent the idea described by a piece of text may remove and possibly distort the meaning of the text. This is where the true value of modern text-to-scene systems can be seen which is explained next.

Static Text-to-Scene

Systems such as Wordseye, detailed by [2], improve on the concept of the above mentioned text-to-picture systems by rendering 3D scenes. This gives systems like Wordseye the ability to not only illustrate the objects mentioned in text but also the spatial relation between them. This allows for a clearer and more accurate representation of the scene described by the text. Another important aspect of text-to-scene used in the Wordseye system is the use of 3D models as opposed to the pictures used by [25].

The Wordseye system makes use of an object database which contains all of the potential models to be used in the rendered scenes [2]. A database such as this one would be crucial to the implementation of a scene-to-game system if the process is to be fully automated. Additional functionality found in the Wordseye system is that of the inferred environment, an extension developed by [20]. By considering

[1]Ease with which an idea can be illustrated with pictures

the context of the scene described in the text [20] is able to infer information about the environment. The development of a scene-to-adventure game would require such functionality as large 3D open worlds are becoming more and more common as the backdrops for adventure games. However, it must be noted that the input used by many of the text-to-scene systems such as Wordseye takes in a specific storytelling structure dissimilar to natural-language text.

Animated Text-to-Scene

The third type of text-to-scene system is those which generate animated 3D scenes. The complexity and potential value of these systems is far greater than that of the simpler systems. Not only do these systems consider the spatial properties of objects but also their temporal nature. It is for this reason that these systems are highly applicable when considering the design of a scene-to-game system. Recently, [8] developed a system in which fiction text is converted into a 3D animated scene. The system works by annotating the text through use of the parts-of-speech taggers used in their earlier work [5]. Examples of annotated text can be seen in Figure 1. The annotated text is then used to create constraints which are solved to create trajectories and spatial relations for the objects [6, 8]. Finally, this information is used in created an animated scene.

Other similar systems for the creation of 3D animation from text including CarSim [9]which will be explained in the next section and the CONFUCIUS [13]system.

While several text-to-scene systems do exist, it is those which have the ability to recognize natural-language text which are of interest in the current context. The CarSim [9] and WordsEye [2] systems require text which is structured in a specific story form. Systems such as that created by Glass and Bangay [8] and the CONFUCIUS [13] system can use input text extracted directly from unmodified fiction books. This type of functionality would be a great asset to a scene-to-adventure game system in that there are countless fiction texts available with the potential to become fully-fledged adventure games. It should be noted that the constraints used by Glass and Bangay may be unnecessary in a scene-to-game system as we wish to grant the player the freedom to perform their own actions. However, The concept of player freedom and choice will be discussed further in Section 3.

2.2.2 Other Applications of the Text-to-Scene Concept

The systems mentioned above have valuable applications in various computer science fieldss. The technology found in such text-to-scene systems has been utilized to great effect elsewhere as well. The CarSim system [9] was developed for visualizing traffic

They had it on the top of a hill, in a sloping field that looked down into a sunny <setting>valley</setting>. <avatar>Anne</avatar> didn't very much like a big brown <object>cow</object> who <transition type='ARRIVE' subject='cow'>came</transition> up <relation type='near' subject='cow' object='her'>close<relation> and stared at her, but it <transition type='DEPART' subject='it'>went</transition> away when <avatar>Daddy</avatar> told it to.

Figure 1: ExampleText Example of text annotated with xml

accidents through a text based description of the event. This technology could prove to be useful in a variety of fields such as law, insurance and safety. Durupınar *et al.* [3] have created a system for the reconstruction of crime scene photographs. A system which makes use of text describing the contents of the photograph to create a 3-dimensional reconstruction of the scene.

While these applications may not be applicable to the development of a scene-to-adventure game system, it is interesting to note that the text-to-scene concept is very much in use and supported.

As we can see from the above discussions, the technology which exists in the text-to-scene area of research has progressed significantly. With the ability to render 3D scenes it can be seen how the functionality found in current text-to-scene systems can be adapted to allow for the creation of computer games. The next section details how various narrative structures can be used to create more realistic and enjoyable games.

3 Narrative and Game Structures

While text-to-scene systems have many components which could be useful in the develop of a scene-to-game system, the game design and narrative are also vital parts of creating a game. While the game narrative will obviously be well guided by the fiction text, the interactive nature of the game should allow for the narrative to evolve as the game progresses. This section details the various narrative structures used in interactive media along with the different structures which video games typically follow.

3.1 Interactive Narratives

The idea of interactive narratives in games is a very popular concept as increased opportunity for choice lends itself to greater realism. The reason being that human beings are not constrained by those elements which tend to limit a players choices in games. For example, a predefined story or small map which depicts the entire

world hinders the amount of freedom which a player has. However, while choice is desirable to a some extent, the lack of a narrative would greatly subtract from the quality of an adventure game. It is for this reason that a trade-off must be made between interactivity and narrative structure. [17] present this problem as being storytelling versus simulation or a trade-off between control and coherence. In this approach storytelling describes the predefined coherent narrative and simulation is the players ability to interact with the game and make decisions or their control. Riedl *et al.* [17] approach this problem through the creation of an automated story director which uses a high-level plot outline to guide the player. Louchart *et al.* [12] take a similar approach to the issue by offering guidelines for narrative authoring such as considering different actions which can be performed be the player and the repercussions thereof.

The idea of interactive narratives can be tied to that of the text-to-scene concept by looking at the Scene-Driver system [24]. The system, developed by Wolff *et al.* [24] in 2004 makes use of scenes from an existing childrenss television show called Tiny Planets. It is represented in the form of a game in which children are required to select dominoes which depict what will happen next in the story. This is another area where the text-to-scene system could potentially be used. As an alternative to re-using the Tiny Planets television show, existing "choose you own adventure" style stories could be used as the narrative while a text-to-scene system could be used in rendering the scene.

The terms interactive narrative and emergent narrative are often used interchangeably to describe those narratives giving the player choice and having alternate paths. However, we must distinguish between those narratives which have predefined alternate paths and those which have alternate paths as gateways to undefined story elements. These emergent narratives will be described in the next section.

3.2 Emergent Interactive Narratives

The concept of interactive narratives can be extended to that of emergent narratives which use less definitive story lines. Similarly to interactive narratives users are given the ability to make choices. However, in an emergent narrative those choices which the users make can alter the narrative in new undefined ways. Unlike basic interactive narratives which tend to coerce the player into conforming to some or other storyline, emergent narratives attempt to create new story-lines to accommodate the players decisions. Louchart *et al.* [11] define the author's role in an emergent narrative and subsequently give guidelines for fulfilling the role at design time.

Similarly to interactive narratives, emergent narratives can be linked to text-to-scene systems. Implementation of a scene-to-adventure game system would be the perfect case of this. The fiction text will be used in constructing a game world and emergent narrative in which the player will be able to explore alternative stroyline not mentioned in the text.

3.3 Game Structures

While narratives play an important role in creating a game with playable value, the game mechanics must also be considered. These can often work hand in hand with the narratives. Lindley [10] provides several ideas relating to the structure of games in terms of their goals and narratives. While the paper does not offer much in the way of research methodology, some interesting ideas are mentioned by Lindley relating to the structure of games. Game structure is highly dependent on the type of narrative used but the way in which the game world could be represented at any given point is also important. For example, the game could follow a tree-like structure whereby each leaf or node would represent some event or scenario [10]. Alternatively, the game could be statefull whereby attributes would represent all of the objects in the game world and the values of those attributes would determine the current state.

An important reason for selecting an appropriate game structure is that of goals. Pizzi *et al.* [16] offer a system in which the relation ship between game goals and the narrative is put to use. The system [16] takes advantage of the interactive narrative to create a new method for procedural content generation. However, this differs from traditional procedural content generation in that it does not generate terrain or objects. Rather, the system [16] generates all possible game goals and produces storyboard style output potentially aiding game design.

This could be of use in a scene-to-adventure game system whereby the generation of all possible game solutions could provide insight into the effectiveness of game goals. Also, incomplete narratives could be completed with the help of such a system [16] as the emergent nature of games could be controlled within reason.

As can be seen in the sections above, the stucture which a games narrative and architecture follow can greatly dictate the type of experience which players will have when playing the game. By implementing a scene-to-adventure game system, game developers could use existing fiction texts to create interesting new narratives and games.

The following section discusses the various methods available for game development and how they can be used in the creation of a scene-to-adventure game

system.

4 Game Development

With user created content becoming increasingly popular, more and more tools for game development have emerged. Numerous open source engines and game development tools have been made available. These ranging from entirely new languages to applications purpose built for the development of game creation. This section details the most relevant of these tools and their uses.

4.1 Manual Game Development

As mentioned above, a large jump in the availability of game development tools has caused the activity creating games to become increasingly popular. While projects are being undertaken relating to automatic game development, these systems would not be possible without some initial manual development.

4.1.1 Programming Based Development

While many of the game development tools currently available do not require any programming language, game development is still an area which has high educational value. It is no longer uncommon for universities to offer courses and perhaps even majors in the field of game development [4][1]. Schaefer and Warren [18] provide a guide for teaching students about geometric modelling and computer graphics through game design. Many open source games engines and languages exist such as the Glest engine[2] which makes a perfect candidate for use in the scene-to-adventure game system which will be discussed in Section 4.1.3.

4.1.2 Non Programming Based Development

The development of non-programming based game development tools was responsible for a large surge in the number of independent games currently available. These tools can also be used to speed up the game development process for existing game developers. Orrente *et al.* [23] provide an open source environment, the <e-Adventure3D>, for the creation of 3-dimensional educational adventure games. The application [23] simply consists of an authoring tool and game engine giving those users with no programming knowledge the ability to create games. A noticeable part of the system [23] is that the storyboard for the created games uses an XML-based

[2]http://www.glest.org/en/engine.php

language and modifying the game becomes as simple as editing certain variable values. This use of XML has potential to be largely useful in the development of a scene-to-adventure game system. However, that will be discussed further in the next section. Many other systems exist which allow users to create games without using any programming language such as GameMaker[3] and Adventure Game Studio[4]. It is these systems which prove that large parts of game development can be generalized which will be of great importance in developing a scene-to-game system

4.1.3 Markup Language in Games

As mentioned above the creation of non programmer based game development tools has become increasingly popular. Much effort has gone into creating systems which will allow the creation of both simple and advanced games with minimal effort. In order for this to be achieved, it was necessary for developers to create generic interfaces to make game engines accessible to those unfamiliar with programming. Sepcha *et al.* created a semi-automated game creation system [19] which creates tactile games for visually impaired children. While the scope of such a system is significantly smaller than that of the potential scene-to-adventure game system, it is interesting to observe the interface used as it makes use of generic properties which can be tweaked to alter the game.

One way of accomplishing this is through the use of a markup language such as XML. Simply filling in meta data is a much easier process than learning to program. Many of the systems mentioned above such as the <e-Adventure3D> [23] and the Glest engine [5] make use of XML in the same manner. An example of the typical use of XML in the Glest engine can be seen in Figure 2[6]. Moreno-Ger *et al.* [14] developed a system similar to <e-Adventure3D> [23] which also made use of a markup language for the story board and a processor (game engine) for the language.

These markup language interfaces could prove to be very useful in the development of a scene-to-adventure game system as XML is used to a large extent in both text-to-scene system and several modern game engines.

[3]http://www.yoyogames.com/gamemaker
[4]www.adventuregamestudio.co.uk
[5]http://www.glest.org/en/engine.php
[6]http://glest.wikia.com/wiki/GAE/Unit_XML

```
<?xml version="1.0" standalone="no" ?>

<unit>
    <parameters>
        <size value="#" />                          <!-- How many units per side a building uses -->
        <height value="#" />                        <!-- The height of a unit -->
        <max-hp value="#" regeneration="#"/>        <!-- Maximum hit points and rate of regeneration -->
        <max-ep value="#" regeneration="#"/>        <!-- Maximum energy points and rate of regeneration -->
        <armor value="#" />                         <!-- Armor value of unit -->
        <armor-type value="wood"/>                  <!-- Type of armor - some weapons are more effective against some
                                                         armor types; Editable in the Tech XML file -->
        <sight value="#" />                         <!-- Range of sight to react to other units -->
        <time value="#" />                          <!-- Time required to build unit. -->
        <multi-selection value="false" />           <!-- Is double-click on unit multi-select other unit of same type -->
        <cellmap value="true">                      <!-- For Buildings, which Cells in a building a unit can pass through
                                                         (0 can pass, 1 cannot). The number of rows and the number of
                                                         digits in a line must match the size given above! -->

            <row value="1011"/>
            <row value="1011"/>
            <row value="0000"/>
            <row value="1011"/>
        </cellmap>
```

Figure 2: Defining Properties for a custom unit in Glest

4.2 Automated Game Development

Automated game development is still a relatively small field. However, work is being done on creating systems which automate the design process for games. Nelson and Mateas [15] developed a prototype system which is aimed at the automatic creation of small "WarioWare" style games[7] such as clicking on a fast moving object in a small screen. The main tasks of which as specified by Nelson and Mateas [15] are building the system with the ability to make sense of the abstract rules for the games and determine some methods for visualizing these rules. Nelson and Mateas [15] also faced the problem of dealing with common sense as the system worked by allowing users to specify verbs or nouns to describe the game. Another attempt at automated game creation was made by Togelius and Schmidhuber [22] in which the initial game was generated and left to evolve. This involves the creation of neural networks to guide the evolution of the NPC (Non-player character) [8] controllers as well as the game rules [22]. However, the paper does not contain results and it is therefore difficult to imagine the possible current applications of such a system.

While these systems have potential in opening a new field of computer science, the methods which they use are only applicable to a very limited set of small games. The development of a scene-to-adventure game is a far more plausible concept with the currently available technology.

In this section we have shown that there is a clear link between the input of several of the available game development tools and the annotated text which modern text-to-scene systems generate in creating their output.

[7] Small and simple mini-games

[8] A computer controlled actor such as an opponent

5 Conclusion

These 3 significantly different research areas provide the resources needed to create a working Scene-to-Adventure game system. Parts-of-Speech taggers and Text-to-Scene systems have all of the functionality required for the initial step of annotating the fiction text efficiently and with a usable level of accuracy. Relating directly to this is the structure which many modern games are taking by creating generic interfaces for ease in game development. With this functionality available it can be seen how the annotations acquired from the first step can be used in providing the initial detail for the games. This can then be used in crafting enjoyable and playable games with rich environments and narratives based on popular fiction texts.

References

[1] Lawrence Argent, Bill Depper, Rafael Fajardo, Sarah Gjertson, Scott T. Leutenegger, Mario A. Lopez, and Jeff Rutenbeck. Building a game development program. *Computer*, 39(6):52–60, 2006.

[2] Bob Coyne and Richard Sproat. Wordseye: an automatic text-to-scene conversion system. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 487–496. ACM Press, 2001.

[3] Funda Durupinar, Umut Kahramankaptan, and Ilyas Cicekli. Intelligent indexing, querying and reconstruction of crime scene photographs. In *TAINN2004*, pages 297–306, 2004.

[4] David Finkel, Mark Claypool, Michael A. Gennert, Fred Bianchi, Dean ODonnell, and Patrick Quinn. Teaching game development: At the intersection of computer science and humanities & arts.

[5] Kevin Glass and Shaun Bangay. Evaluating parts-of-speech taggers for use in a text-to-scene conversion system. In Judith Bishop and Derrick Kourie, editors, *SAICSIT 2005 South African Institute of Computer Scientists and Information Technologists*, pages 20–28, White River, South Africa, September 2005.

[6] Kevin Glass and Shaun Bangay. Constraint-based conversion of fiction text to a time-based graphical representation. In *SAICSIT '07: Proceedings of the 2007 annual research conference of the South African institute of computer scientists and information technologists on IT research in developing countries*, pages 19–28, New York, NY, USA, October 2007. ACM Press. Best paper award.

[7] Kevin Glass and Shaun Bangay. A naive salience-based method for speaker identification in fiction books. In *PRASA 2007: Proceedings of the 18th Annual Symposium of the Pattern Recognition Association of South Africa*, pages 1–6, November 2007.

[8] Kevin Glass and Shaun Bangay. Automating the creation of 3D animation from annotated fiction text. In *Proceedings of the IADIS International Conference on Computer Graphics and Visualization 2008*, pages 3–10, July 2008.

[9] Richard Johansson, Anders Berglund, Magnus Danielsson, and Pierre Nugues. Automatic text-to-scene conversion in the traffic accident domain. In *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, pages 1073–1078, Edinburgh, Scotland, July 2005.

[10] Craig A. Lindley. *Story and Narrative Structures in Computer Games*. High Text, January 2005.

[11] S. Louchart, I. M. T. Swartjes, M. Kriegel, and R. S. Aylett. Purposeful authoring for emergent narrative. In *Proceedings of the First Joint International Conference on Interactive Digital Storytelling, Erfurt, Germany*, Berlin, September 2008. Springer Verlag.

[12] Sandy Louchart, Ruth Aylett, Michael Kriegel, João Dias, Rui Figueiredo, and Ana Paiva. Authoring emergent narrative-based games. *Journal of Game Development*, 3(1):19–37, March 2008.

[13] Minhua Eunice Ma. Confucius: An intelligent multimedia storytelling interpretation and presentation system. Technical report, School of Computing and Intelligent Systems, University of Ulster, Magee, September 2002.

[14] Pablo Moreno-Ger, José Luis Sierra, Iván Martínez-Ortiz, and Baltasar Fernández-Manjón. A documental approach to adventure game development. *Sci. Comput. Program.*, 67(1):3–31, 2007.

[15] Mark J. Nelson and Michael Mateas. Towards automated game design. In *AI*IA '07: Proceedings of the 10th Congress of the Italian Association for Artificial Intelligence on AI*IA 2007*, pages 626–637, Berlin, Heidelberg, 2007. Springer-Verlag.

[16] David Pizzi, Marc Cavazza, Alex Whittaker, and Jean-Luc Lugrin. Automatic generation of game level solutions as storyboards. In Christian Darken and Michael Mateas, editors, *AIIDE '08:*. The AAAI Press, 2008.

[17] Mark O. Riedl, Andrew Stern, and Don M. Dini. Mixing story and simulation in interactive narrative. In John E. Laird and Jonathan Schaeffer, editors, *AIIDE*, pages 149–150. The AAAI Press, 2006.

[18] Scott Schaefer and Joe Warren. Teaching computer game design and construction. Technical report, Worcester Polytechnic Institute, 2004.

[19] Alexis Sepchat, Nicolas Monmarché, Mohamed Slimane, and Dominique Archambault. Semi automatic generator of tactile video games for visually impaired children. In Klaus Miesenberger, Joachim Klaus, Wolfgang Zagler, and Arthur I. Karshmer, editors, *Proc. ICCHP 2006 (10^{th} International Conference on Computers Helping People with Special Needs)*, volume 4061 of *LNCS*, pages 372–379, Linz, Austria, July 2006. Springer.

[20] Richard Sproat. Inferring the environment in a text-to-scene conversion system. In *K-CAP 2001: Proceedings of the international conference on Knowledge capture*, pages 147–154. ACM Press, 2001.

[21] Jan Svartvik. Computer-aided grammatical tagging of spoken english. In *Proceedings of the 8th conference on Computational linguistics*, pages 29–31, Morristown, NJ, USA, 1980. Association for Computational Linguistics.

[22] Julian Togelius and Jürgen Schmidhuber. An experiment in automatic game design. In *Proceedings of the 2008 IEEE Symposium on Computational Intelligence in Games CIG-2008*, 2008.

[23] Javier Torrente, Angel del Blanco, Guillermo Canizal, Pablo Moreno-Ger, and Baltasar Fernandez-Manjon. e-adventure3d: an open source authoring environment for 3d adventure games in education. In *ACE '08: Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology*, pages 191–194, New York, NY, USA, 2008. ACM.

[24] Annika Wolff, Paul Mulholland, Zdenek Zdráhal, and Richard W. Joiner. Scenedriver: An interactive narrative environment using content from an animated children's television series. In Stefan Göbel, Ulrike Spierling, Anja Hoffmann, Ido Iurgel, Oliver Schneider, Johanna Dechau, and Axel Feix, editors, *TIDSE*, volume 3105 of *Lecture Notes in Computer Science*, pages 213–218. Springer, 2004.

[25] M. Eldawy C. R. Dyer X. Zhu, A. B. Goldberg and B. Strock. A text-to-picture synthesis system for augmenting communication. In *Proceedings of the 22nd Conference on Artificial Intelligence of the Association for the Advancement of Artificial Intelligence*, pages 1590–1595, 2007.