Dialog as a Game

Peter M. Border
School of Physics and Astronomy
University of Minnesota, Twin Cities
116 Church Street SE
Minneapolis, MN 55455
612-624-1020
border@mail.physics.umn.edu

ABSTRACT

We describe a technique to manage pre-written lines of dialog by treating a conversation as a game. Thinking of conversation as a game means structuring it as a series of moves, made according to rules, with some sort of score. Speakers in our system converse by participating in short dialog trees, and make "moves" by negotiating transitions between trees. Speakers have internal state variables which describe their standing in the conversation and their emotional state. Speakers try to manage the conversation so as to maximize a payoff function of their internal variables.

We believe that this technique will allow us to create lifelike NPC dialog, and allow our NPCs to play a more social role in game worlds. We also believe that games in general desperately need to have more socially coherent NPCs, and that improving dialog is a critical problem in game development.

Keywords

artificial intelligence, believable characters, dialog management, games

INTRODUCTION

Computer games developers have made tremendous progress in simulating the physical world, but they have almost ignored the world of social and human interaction. Modern games like Valve Software's *Half-Life 2* are set in a physical world that is simply stunning, with all sorts of "live" objects a player is free to pick up, toss at bad guys, hit with crowbars, stack, and climb. The emotional and social worlds, however, are essentially ignored; the non-player character's (NPCs') faces may be animated, but you certainly can't ask them about their world. Other games like BioWare's *NeverWinter Nights* do better, but the NPCs are still little more than animated billboards who exist solely to tell the player what to do next. This paper is about an attempt to fix this problem with a "dialog manager", a set of rules that allow NPCs to respond like socially and

Proceedings of DiGRA 2005 Conference: Changing Views – Worlds in Play.

emotionally complex characters.

The Problem

The usual technique to handle game dialog is with decision trees, which is fine for short sequences but becomes unmanageable as conversations get longer. Our new approach is to think of conversation as a game played by the speakers, which leads to a framework that is much more extensible than the dialog tree approach. In trials with index cards, our new system has had dozens of exchanges in a conversation happen in a fairly natural manner.

We believe the limitations of NPC dialog is a major reason why games have not caught on well outside their demographic of youngish males, and that a better way to handle dialog is crucial to the future of computer games. Dialog is a major part of the emotional and social immersion people experience when reading a novel, for example, and improving game dialog may lead to the same effect when playing a game.

Game dialog with the usual techniques is also prone to meta-gaming strategies which destroy the illusion of reality. Clever players know that a good strategy is to simply keep bumping into an NPC, restarting the conversation each time, and exploring different branches of the tree. NPCs repeat themselves when they reach the end of their dialog tree, so it is easy to tell when a particular branch has been exhausted. After a few trials, a player has learned everything an NPC has to say, and the NPC can be ignored thereafter. Gameplay using such meta-gaming strategies breaks the immersive quality of the game world completely. We propose to simulate real-world conversations much more realistically, which will hopefully lead to a much more socially and emotionally immersive game world.

Conversation as a Game

Our new approach is to think of conversation as a game played by the speakers. Speakers make their "moves" by talking about different topics. Thinking of conversation as a game has several advantages; games have rules, games have scores, and game players have a clear and rational way to decide what they wish to do next. Best of all, simulating a game with a computer is fairly straightforward once the rules of the game are known. Conversations clearly have rules; people who ignore the rules wind up saying some very odd things! Finding those rules and mimicking them in software is the goal of this project.

Previous Work

There has been a large amount of very good work done on believable software agents in the past. Janet Murray's book *Hamlet on the Holodeck* [4] is a classic, with a view of the future that is still sadly just a vision today. Barbara Hayes-Roth and the people at the Stanford Knowledge Lab wrote many papers on believable agents in the 1990's, and the Oz project at Carnegie-Mellon did some very interesting work on computer-managed theater. Some more recent work can be found in the proceedings of the Affective Dialogue Systems conference [1]. The book *Life-Like*

Proceedings of DiGRA 2005 Conference: Changing Views – Worlds in Play.

Characters [6] has some longer papers from various authors. Interactive fiction is still alive and healthy a decade after the end of Infocom, with a large online amateur fan base. Michael Mateas's thesis [3] describes his work on "*Facade*", a computer-directed interactive drama. *Facade* does have a dialog manager, but it is mainly concerned with advancing the state of the drama, not with managing an NPC in a game environment.

Despite all this effort, there does not seem to be a consistent set of ideas on how to manage dialog. Dialog is usually managed in a very ad hoc manner, with no clear set of rules. Dialog management is obviously an extremely important part of an NPC, but there does not seem to be a good general theory of how to do it.

Studying Plays

We began looking for rules of dialog by analyzing several short plays [2], and trying to discover what rules were being followed by the conversations in them. Plays are very useful for this sort of study- characters take turns speaking, they don't stutter or repeat themselves, they can hear each other clearly, and so on. After analyzing several plays with a spreadsheet, we concluded that dialog happens in a series of "chunks"- short sequences of lines with transitions between them. The dialog in a "chunk" is fairly predictable, and is best duplicated as a dialog tree (characters do not have many choices inside a chunk, so the branching combinatorics are not very complex). A new chunk is introduced when the old one ends, and the characters again have very little choice as long as the new chunk lasts. Occasionally a character will force a topic change in mid-chunk, but this causes repercussions in the feelings of the rest of the conversants.

The strategic, game-like part of a conversation comes in sequencing the chunks. For example, office-worker character Alf may start a "complaining chunk" by making some negative comments about his boss. Coworker character Betty has a choice whether to sympathize or reject the criticism, but, even if she wants to, she cannot start talking about a football game without making Alf feel resentful. After complaining for a while, the topic runs out of steam, and there is a pause. After the pause, Betty may introduce a new "chunk" and talk about the football game without paying a penalty- which gives Alf the choice of being positive or negative about the game. If he wants Betty to like him, he will respond positively, then either let her introduce the next chunk or start another chunk about football.

IMPLEMENTING RULES OF DIALOG

There are several parts to our new system. The following sections describe what the computer implementation we will eventually have will look like.

Player Interface

We envision a turn-based dialog system for a simple interface. An NPC says a pre-written line, after which the user is offered a choice of several lines for a response. The NPC responds appropriately from its list of choices, the user is shown new choices, and so forth (similar to the interface in BioWare's *NeverWinter Nights*, Fun.Com's *The Longest Journey*, and many other games). While a chunk is happening, the choices will be rather minimal (one or two responses), but there may be 5 or 6 responses allowed at chunk-transition time.

Ideally, the choices available to the player will always include what he wants to say next, and all the responses from the NPC will be believable and in character. We do not expect such a system to make any noticeable demands on CPU cycles, since it only requires a few table lookups every second or so.

Character Internal State Variables

Characters clearly have some sort of internal state which changes during the conversation. Their state can be simulated (as accurately as it needs to be, that is) with a limited set of variables, which change as the conversation progresses. The state variables need to include a summary of the characters emotional state, as well as some variables indicating the relationship of the characters. The definitive set of emotional variables are those developed by Ortony, Clore and Collins [5], but their system is rather unwieldy and we use a simplified system of happy/sad, afraid/confident, angry/peaceful and excited/calm. Most of our test scenarios take place in bars for some reason, so we include a drunk/sober internal state variable. Modeling a characters internal state does not seem to be as important as modeling the relationship variables.

Relationship variables

The variables reflecting the state of a relationship are very important, but less well-known. We studied relationship variables by typing short plays into a spreadsheet (one line per row) and commenting each line. Then we analyzed the comments and eventually came up with these relationship variables:

The Ball

Usually one character is driving a conversation, and has first choice on the next “chunk”. This is the “conversational ball”, which is passed from character to character as face is lost and gained. Only one character may have the ball at a time. Normally a character loses the ball if they lose face, but characters may seize the ball by interrupting. Interruptions cause a loss of closeness and will probably cause other characters to become irritated at the interruptor.

Closeness

Closeness is the feeling people have of being in contact with each other. Two strangers passing on the street have a closeness of zero. Greetings increase closeness to a “talking” level (close between 1 and 5). More interactions may increase the level to “interested” (5-10), “intimate” (10-20) and finally to “obsessive” (20+). Closeness is a very important variable, and it changes frequently.

Face

Saying something stupid or inappropriate causes one to lose face. Saying wise, intelligent, incisive things causes one to gain face. We include face as a variable distinct from closeness, though the two are highly correlated. Losing face normally causes a character to lose the conversational “ball”.

Liking/irritation

Being agreeable and telling jokes can get characters to like each other. Liking is a one-way feeling (A may like B without B liking A), so each character in a dialog has their own number, which indicates how much they like the other character. Similarly, being disagreeable can irritate others, so the liking number is allowed to become negative. Irritation and liking are related to the anger/peaceful internal state variable, but are directed specifically at other characters.

Trust/distrust

Characters may decide to trust or distrust each other in the course of a conversation. Trust may be required for some chunks. Finding contradictions and untruths in someone's statements causes a loss of trust, while discovering mutual acquaintances usually raise it.

Relationship variable levels decay with time, and, unless they are refreshed, will eventually return to 0. A rough guide is that relationship variables lose 10% of their intensity every turn (negative scores become less negative).

Payoff functions

Payoff functions define how a character evaluates the progress of a conversation. Different characters will evaluate conversations differently, and this is reflected in their payoff functions. A character who is looking to recruit helpers, for example, will want to impress her potential colleagues, so will be looking for high face and trust values. A character seeking a sexual encounter will also need high trust scores, but will need high like scores more than high face scores. A gossip will seek high trust and closeness, and a barkeeper will probably just want to avoid anger in his patrons and himself. Guards will definitely want to inspire fear in people. Payoff functions may depend on internal variables of either character, as well as the relationship variables. They may also depend on the progress of the conversation; a character who is trying to recruit someone will clearly be happier if the recruit responds favorably to his offer, for example.

Choosing the next chunk

Each chunk has some set of requirements that go with it, which must be satisfied before the chunk is offered as a choice. For example, a chunk called “he invites her to sit at his table” requires the two characters to have a large close value, and a positive liking/dislike value. A chunk called “she insults him” requires them to be talking, and her to dislike him.

Characters in the conversation game need some strategy to choose their next move, which may be very simple. In index-card trials of the system we have just tried to look a move ahead and see if the payoff function has improved. Automating such a strategy should not be difficult, and would probably mimic a real conversation well. NPCs who are skilled conversationalists (such as Dorothy Sayers' fictional Lord Peter Wimsey) may be allowed to look several moves ahead, and enjoy the corresponding rewards.

Communicating Facts

Conversations are not just about making artistic sequences of dialog. People do occasionally learn new things from talking to each other and a system of dialog management needs to have a way for characters to acquire new facts. The subsequent dialog choices may depend on what facts they know, and there must be some way for a characters' lines to be influenced by something he has just been told. So far, we have been using a very simple system of knowledge tracking, some chunks require other chunks to precede them.

This is an extremely simplified version of knowledge handling, and eventually we plan to give every character a hash table of "keyword=value" pairs to keep track of what he has been told. Each line of dialog may have some "keyword=value" pairs associated with it in the dialog database, and may also have an associated "test string" to determine whether it is usable, given what the character knows. The line will only be used if the test string evaluates to true.

For example, if a man and a woman are chatting at a bar, and she mentions her husband, the line about her husband has an associated knowledge string of "she_is_married=true", which is added to the man's hash table. Consequently, a host of subsequent lines for the man are cut off. If she then says a line about her children, with the associated knowledge string "she_has_kids=true", the man adds that string to his hash-table, and another set of dialog lines become possible. This scheme allows considerable control over the dialog without being too onerous. Test strings will be evaluated by a simple replace-and-evaluate strategy, and, if a line's associated test string does not evaluate to true, it will not be offered as a choice.

STATUS AND FUTURE WORK

This is still very much a work in progress. Our basic structure has been worked out in some detail, and we are in the process of working through some test scenarios. The most advanced test is a scenario due to Adam Momsen, called "The Fighter and the Barmaid", which is an encounter in a pseudo-medieval village between a fighter-adventurer and a barmaid. He tries to find out what she has overheard working at the bar, while she is trying to find a way out of her undesirable job. If things go well, he tries to recruit her for his band of adventurers. We started by having two students do some improvisational run-throughs of the scenario, and found about 50 chunks based on the recorded "improv" sessions. These chunks have all had their requirements and consequences roughed out, and been described on index cards. Sample lines of dialog are also on the cards, but, in a trial, the students frequently add their own improvisations. We are in the process of doing trials with these cards and evaluating the "naturalness" of the resulting dialog.

In the future we will implement a Java program to automate the system, and we will implement more interesting scenarios. Realistic dialog is likely to have many applications outside of gaming, online sales and help desks are obvious venues, and eventually tutoring.

We would like to thank Adam Momsen, Nathan Sanders and Kate Raach for their assistance. Part of this work was funded by a GRAVEL grant from the University of Minnesota.

REFERENCES

1. Andre, E., Dybkjaer, L., Minker, W. and Heisterkamp, P. (eds.). *Affective Dialogue Systems*. Springer-Verlag Berlin and Heidelberg, 2004
2. Harvey, A. *The Methuen Book of Duologues for Young Actors.* Methuen Press, London, 1995
3. Mateas, M. *Interactive Drama, Art and Artificial Intelligence.* Ph.D. Thesis. Technical Report CMU-CS-02-206, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA. December 2002
4. Murray, J. *Hamlet on the Holodeck.* MIT Press, Cambridge, MA, 1997
5. Ortony, A., Clore, G., Collins, A. *The Cognitive Structure of the Emotions.* Cambridge University Press, Cambridge, 1990
6. Prendinger, H. and Ishizuka, M. (eds.). *Life-Like Characters.* Springer-Verlag Berlin and Heidelberg, 2004

Proceedings of DiGRA 2005 Conference: Changing Views – Worlds in Play.