Episodic Memory: A Final Frontier (Abbreviated Version)

Kenneth D. Forbus, Sven Kuehne

Qualitative Reasoning Group, Northwestern University
2133 Sheridan Road, Evanston, IL, 60208, USA
Contact: forbus@northwestern.edu

Abstract

A major limitation of today's computer games is the shallowness of interactions with non-player characters. To build up relationships with players, NPCs should be able to remember shared experiences, including conversations, and shape their responses accordingly. We believe that progress in AI has already reached the point where research on using NLP and large KBs in games could lead to important new capabilities. We describe our *Listener Architecture* for conversational games, which has been implemented in a toolkit used to make short experimental games. Episodic memory plays a central role, using analogical reasoning over a library of previous conversations with the player. Examples and scale-up issues are discussed.

Introduction

While the game industry has been remarkably successful, the range of people playing computer games is far smaller than the range of people who enjoy reading novels. It is widely believed that improving the characters in games so that players could have sustained interesting interactions with them -- other than kill or be killed -- is one of the capabilities that will be needed to expand the range of players. Conversation, in particular, is a complex problem where little progress has been made.

We believe that research on how to use NLP and large KBs in games is needed to make these tools commonly available for game designers. This paper describes progress on three ideas in this area: (1) A key capability for NPCs is *episodic memory*. Episodic memory concerns the experiences of that particular agent, in contrast with knowledge of the world in general. (2) Episodic memories in NPCs need to be expressed in rich, relational vocabularies, instead of flags or feature lists, to express events that the NPC and player have experienced together, and to capture the content of broad conversations. (3) Analogical processing can provide human-like ways of manipulating episodic memory, such as retrieving relevant prior conversations and using a comparison of the current conversation with prior conversations to generate expectations and questions.

This paper describes the *Listener Architecture*, a new architecture for conversational systems we are developing to explore these ideas. Our first implementation exploits a large-scale (> 1M facts) knowledge base, human-like analogical processing techniques, and NLP in a toolkit for developing short conversation-based games.

We start by outlining some of the functions that episodic memory should provide for NPCs. We summarize the conversation-centric games that provide the context for this work, the Listener architecture, and its implementation. We illustrate its current strengths and weaknesses via examples, and discuss some of the practical issues that will be involved in deploying a knowledge-rich approach like this one in games. Finally, we discuss related work and plans for future work.

Why NPCs should have episodic memory

Shared experiences are a key factor in building relationships. Military officers know that shared pain improves unit cohesion, and everyone knows that familiarity breeds. If a player and an NPC go through something together, a natural thing to do is to talk about it afterwards. Conversations, too, are a form of experience. The stories that someone tells us reveal a great deal about their past, their predilections and their propensities. To better relate to players, an NPC needs to be able to have fairly open-ended conversations with them, to remember and learn about them based on what they said, and use this knowledge appropriately in future actions.

Developing NPCs with such abilities is a significant challenge. We focus here on episodic memory of conversations, for two reasons. (1) Conversation is relevant to many game genres, and needs the most drastic improvement. (2) It provides the simplest way to "close the loop" since the memories are of prior conversations.

Most conversational systems today use simple flags and state variables to provide NPCs with memory. While useful for imparting information, conversations in such systems quickly become repetitive and unsatisfying. Dialogue trees and other pre-scripted methods simply do not provide the expressive power needed to capture the content of conversations. Structured representations, where entities and relationships between them are explicitly represented, are required. This is a steep price compared to current technologies, but nobody ever said that revolutions were not without cost.

The blessing (for the player) and curse (for the developer) of conversations is that they can be wide-ranging. Sharing stories can promote engagement, but at the cost of needing more knowledge to represent the meaning of those stories, and better NLU capabilities to construct those representations from text. So, in addition to needing structured representations, NPCs must be able to tap into a shared knowledge base that provides the background that someone in the game world would have.

How can stories accumulated by NPC from a player be used? We see several ways. For example, stories can be used to generate expectations about how a player will behave. If the player talks about a shopping expedition with fondness, then proposing a joint shopping expedition with the player might be a way of getting them to go somewhere when needed. A simpler use of stories, and our focus here, is generating responses to new stories that move a conversation along. If the player describes a harrowing driving experience, for example, the NPC might ask a question or make a comment based on a previous story about driving that the player told them. These capabilities require the ability to retrieve relevant stories, compare them to the current situation, and use their similarities and differences to help decide what to do next.

To summarize, what we need in an episodic memory for NPCs is the ability to construct structured, relational representations via natural language processing. The representational vocabulary needs to be drawn from a large library of knowledge about the game world. Episodic memories encoded using this knowledge must be retrievable when relevant, and used to guide NPC actions.

Our Experimental Gaming Context

Our interface is a simplified IM chat window, which provides a running transcript of the conversation. Each conversational session is kept short, between 10 and 20 turns, since the scenario designs are focused on clear goals to prevent conversations from becoming too open-ended. Mini-games have been built around two scenarios so far: (1) In *Bomb Squad*, the player is talking a citizen (the NPC) through defusing a bomb. This requires calming them down first, so that the citizen doesn't panic and correctly executes their instructions. (2) In *Interstellar Trader*, the Wub [1], an alien species, has contacted us. Their representative (the NPC) wants our primitive cultural digital artifacts, which can be transmitted via hyperwave. The Wub have advanced medical technology. The player, negotiating for humanity, is trying to talk them out of cures for various cancers by offering them music, videos, and computer games. Six graduate students have successfully built single-session mini-games based on these two scenarios using this software, but without episodic memory. *Interstellar Trader* would become a lot more fun if player and alien built up a shared history, as good negotiators do. This is an example of the kind of gameplay that we are trying to support by incorporating analogy into a conversational architecture.

The Listener Architecture

Unlike chatbots, which use minimal processing of the text to simulate understanding, in the Listener architecture we want to understand what the player says as deeply as we can, even with errors. Our goals are: (1) *Exploit shared history appropriately*. Information gleaned about the player from prior conversations should be available to reason with and act upon appropriately. (2) *Breadth*. The architecture should support conversations on a range of topics, ideally anything that would make sense in the game world. (3) *Robustness*. When the system fails, it should do so in ways that maintain player engagement.

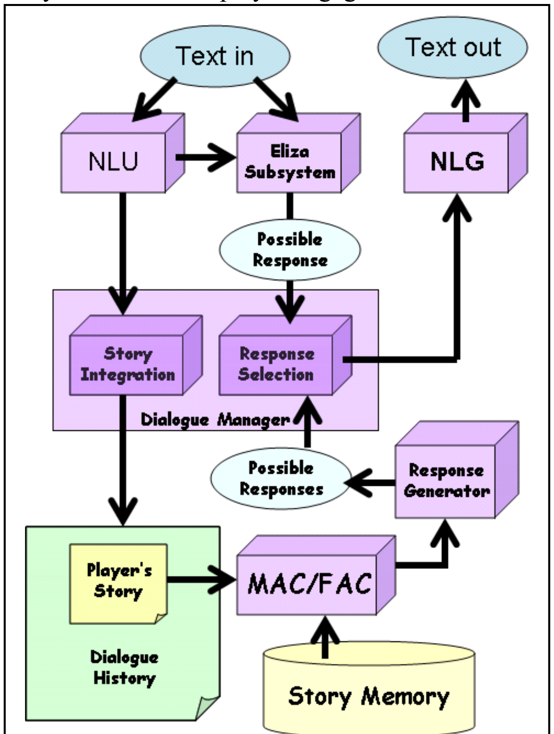

Figure 1: The Listener Architecture

The Listener Architecture is shown in Figure 1. Conversational interactions are divided into *sessions*. Each incoming sentence is handled by two systems in parallel. The NLU system translates it into predicate calculus as best it can. As the ongoing semantic interpretation is updated, relevant prior conversations are retrieved from a library of conversations with that player. Based on what the player said, what was retrieved, and the overall game context (including specialized, scripted knowledge in the finite-state machine tracking game state in the Dialogue Manager), possible responses are generated. In addition to analogy-based responses, an Eliza-style subsystem is also used to generate candidate responses, using a set of rules that depends on the state of the Dialogue Manager[1]. All potential responses are evaluated and if a response is chosen, a natural language version is generated for the player.

Many aspects of the Listener architecture are shared

[1] The Eliza subsystem was turned off for all of the examples in this paper, but we mention it because it has been used in games students have created with the system.

with other conversational architectures, so we focus on what is novel. We use a large-scale knowledge base (1.2 millions facts) to provide breadth, drawn from Cycorp's Cyc KB, plus extensions from our prior research. This includes lexical information mapping words to internal predicates and axiom schema (~149K facts), augmented by LDC's COMLEX lexicon (38K words) to fill in syntactic gaps in the KB's coverage.

The most novel feature of the architecture is the incorporation of analogical processing. The underlying theory is Gentner's [5] *structure-mapping theory*, which describes how the comparison process underlying human analogy and similarity works. Using a solid psychological theory is important for two reasons. First, people are the most robust reasoners we know of, so emulating them is a wise strategy. Second, we want what seems similar to us to seem similar to NPCs, as a means of maintaining immersion. We exploit two cognitive simulations of structure-mapping processes. The Structure-Mapping Engine (SME) [2] provides analogical matching. Given two structured representations, SME produces *mappings* consisting of a set of correspondences (i.e., what goes with what), a set of *candidate inferences* (i.e., what might be inferred on the basis of the mapping), and a *structural evaluation score* indicating overall match quality. MAC/FAC [4] provides retrieval. The first stage of MAC/FAC uses a special kind of feature vector, automatically constructed from structured representations, as a cheap means for extracting two or three best candidates from a large pool of memories. The second stage uses SME to compare the structured representations retrieved against the current situation. Both simulations have been used to model a variety of psychological results, and both have been used in performance systems. Using MAC/FAC provides two advantages. First, assuming reasonable representations, a Listener will be reminded of conversations like those that a person might be reminded of, given the same experience. For example, they will tend to involve similar kinds of entities and activities.

We believe analogical mapping plays several roles in conversational strategies. For example, candidate inferences provide expectations, in a more generative way than hand-crafted scripts do. Correspondences provide a means of calculating similarities and differences between two stories, which in turn provides grist for responses (e.g., commenting on a salient difference between them).

The story integrator in the Dialogue Manager adds assertions corresponding to what the NLU system understood from the latest sentence into the semantic representation of the player's utterances in the session. This includes hypothesizing resolutions for remaining ambiguities and converting pronoun references into internal tokens for the participants. Given the updated player's story, MAC/FAC is used to retrieve a conversation from prior sessions. The mappings computed by the operation of SME within MAC/FAC are searched for material that could be used in responses, using a set of rules. Another set of rules is used to calculate scores for each candidate response, based on factors such as how well the candidate inference is supported by the mapping and whether or not the content of the response has been said already by the Listener in response to a prior sentence.

Once a session is over, the understanding of what the player said during it is stored as a new case in the KB. The cases are organized into libraries, by player/Listener agent pairs. This enables the KB to be shared between multiple Listener agents. Currently retrievals are only carried out with respect to the case library for the current player/Listener pair, since that represents their shared history. Expanding retrievals to include conversations that the particular Listener has had with other players could provide useful data for generating gossip, and expanding retrievals to include conversations with the player that other Listeners have had, and which the current Listener could plausibly know, could provide useful data for detecting deceit or a change of heart in the player. These more complex strategies are left for future work.

An important concern is handling imperfections in NLP. We use the EANLU system [6] for understanding the player's input. EANLU was originally developed to explore how qualitative representations work in natural language semantics. EANLU has a number of grammar restrictions, including present tense, active voice, and only named entities (e.g., "Bill Clinton") for compound noun phrases. However, given these limitations, the Allen parser and initial stages of semantic interpretation are broad coverage. The semantic interpreter attempts to resolve ambiguities when possible, but propagates them forward otherwise, since responses can often be predicated on partial analyses. Even when there are multiple semantic interpretations, analogical processing still works, since it handles partial matches. (Similarly, the Eliza rules exploit information from fragmentary parses, reducing sensitivity to exact word choice and order.)

Our NLG capabilities are straightforward: Templates from the KB are used to construct English output from individual propositions. This is not the state of the art in NLG, but it has been sufficient for our experiments so far.

Examples

Here we use examples from our implemented prototype to illustrate how these ideas work. We use the shortest possible examples here for brevity, taken from the initial phase of a session, when chit-chat occurs as a prelude to serious negotiations. In all examples, Fred is the player and Wub is the NPC.

The first question is whether or not the system can generate appropriate responses based on prior stories. Given the prior session

```
Fred: Bill Gates authors software. Bill Gates is rich.
```

when the system in a later session gets

```
Fred: Richard Stallman writes software.
```

it produces

```
Wub: Richard Stallman is rich?
```

which is a reasonable question given the prior story. (While the KB knows about both of these individuals, it is silent as to their philosophies.) The candidate inference constructed from the retrieved story was

```
(personalWealthOwned RichardStallman
            (AnalogySkolemFn GreatWealth))
```

since `RichardStallman` was mapped to `BillGates`, but the new story doesn't mention wealth at all, so a new term is constructed (via `AnalogySkolemFn`) to represent that value. The suggestion rules use a heuristic that, for scalar values, replacing the skolem with the value from the original story is a plausible transformation, yielding the question asked. It is hard to imagine any simpler technique that could do this robustly.

To illustrate that the response crucially depends on the content of the prior stories, consider the response given to

`Fred: Yesterday I walked a dog.`

When the episodic memory includes

`Fred: I like dogs.`

the system's response is

`Wub: you like dog?`

but if the episodic memory includes instead

`Fred: I hate dogs.`

the system's response is

`Wub: you hate dog?`

This example also illustrates that NLU capabilities do not need to be perfect. Here both prior sentences were interpreted as concerning specific, unnamed individual dogs, rather than a generic affection or disaffection for the entire species. The same suggestion would have been generated if the prior sentences had been "I like/hate Lassie." The important thing is for the NLU system to be consistent in its mistakes. Having an alien as an NPC helps laugh off many blunders, of course.

Practicalities

Could NPCs based on the Listener architecture ever be used in commercially viable computer games? We examine three factors: CPU time, memory, and content.

CPU Time: For sentences between 4 and 7 words, average NLU time is 1.8s, with the worst case being 2.6 seconds, on a stock 2.8 Ghz desktop with 2GB of RAM. Average analogical retrieval and matching time is 2.7s, worst being 6.8s. MAC/FAC's processing time is dominated by the cost of SME, so larger libraries are not a problem. With suitable engineering, we think CPU time could be reduced by at least a factor of five. Reasoning and NLP can also be run as separate processes on other cores. In the meantime, scenario design can hide this: How fast is IM over hyperwave, anyway?

Memory: Performance is faster if the entire KB is in RAM. (The figures above ran the KB from disk!) Our current KB uses about 800MB, but is shared between NPCs, so this cost is independent of the number of NPCs.

Content: Mateas and Stern used a subset of the Cyc KB in Façade [7], providing evidence that relational knowledge can be used in building games.

We believe that the Listener architecture in the long run will become viable for commercial games, although the time-frame will vary substantially based on the genre: (1) Near-term: Conversation-centered games (e.g. [7]), where the number of NPCs tends to be small. (2) Medium-term: Adventure/Action/RPGs, where the number of NPCs is large but the number of them interacting at any time, at least conversationally, can be kept small. (3) Far-term: MMPG's, large-scale social simulators (e.g., political rally where 1K-100K NPCs are swayed, or not, by your rhetoric).

Discussion and Future Work

We have argued that incorporating rich episodic memories in NPCs is a crucial problem. Our prototype of the Listener architecture suggests that analogical processing over libraries of structured representations can provide useful new conversational abilities for NPCs. While many improvements are needed, we have been surprised by how useful and robust simple techniques can be. One need not wait until the technology is perfect.

We plan to expand the Listener toolkit's NL processing in a variety of ways, and expand the analogy-based conversation strategies, including the ability to suggest responses based on similarities and differences. We will continue to use this toolkit with students in game-oriented classes. Their experimental games provide the critical feedback we need to prioritize improvements. Hopefully this experience, in turn, will help some of them go out and create new types of conversation-based games.

References

1. Dick, P.K. 1952. Beyond lies the Wub. *Planet Stories*.
2. Falkenhainer, B., Forbus, K., Gentner, D. ``The Structure-Mapping Engine: Algorithm and examples'' Artificial Intelligence, 41, 1989, pp 1-63.
3. Forbus, K. 2001. Exploring analogy in the large. In Gentner, D., Holyoak, K. and Kokinov, B. (Eds) *The Analogical Mind: Perspectives from Cognitive Science.* Cambridge, MA: MIT Press.
4. Forbus, K., Gentner, D. and Law, K. 1995. MAC/FAC: A model of Similarity-based Retrieval. Cognitive Science, 19(2), April-June, pp 141-205.
5. Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, 7, pp 155-170.
6. Kuehne, S. and Forbus, K. 2004. Capturing QP-relevant information from natural language text. *Proceedings of the 18th International Qualitative Reasoning Workshop*, Evanston, Illinois, USA, August
7. Mateas, M. and Stern, A. 2003. Façade: An experiment in building a fully-realized interactive drama. *Proceedings of the 2003 Game Developer's Conference.*